i.MX 8M Mini Applications Processor Reference Manual

Document Number: IMX8MMRM
Rev. 0, 02/2019

Contents

Section number **Title** **Page**

Chapter 1
Introduction

Chapter 2
Memory Map

Chapter 3
Security

Chapter 4
ARM Platform and Debug

Chapter 5
Clocks and Power Management

Chapter 6
SNVS, Reset, Fuse, and Boot

Chapter 7
Interrupts and DMA

Chapter 8
Chip IO and Pinmux

Chapter 9
External Memory

Chapter 10
Mass Storage

Chapter 11
Connectivity

Chapter 12
Timers

Chapter 13
Multimedia

Chapter 14

Chapter 1
Introduction

1.1 Product Overview

This chapter introduces the architecture of the i.MX 8M Mini Applications Processor.

The i.MX 8M Mini is a family of products focused on delivering an excellent video and audio experience, combining media-specific features with high-performance processing optimized for low-power consumption.

1.2 Target Applications

The i.MX 8M Mini Media Applications Processor is built to achieve both high performance and low power consumption and relies on a powerful fully coherent core complex based on a quad Cortex-A53 cluster with video and graphics accelerators.

The i.MX 8M Family provides additional computing resources and peripherals:

- Advanced security modules for secure boot, cipher acceleration and DRM support
- General purpose Cortex-M4 processor for low power processing
- A wide range of audio interfaces including I2S, AC97, TDM and S/PDIF
- Large set of peripherals that are commonly used in consumer/industrial markets including USB 2.0, PCIe and Ethernet

1.3 Acronyms and Abbreviations

The table below contains acronyms and abbreviations used in this document.

Acronyms and Abbreviated Terms

Term	Meaning
ADC	Analog-to-Digital Converter
AHB	Advanced High-performance Bus
AIPS	Arm IP Bus
ALU	Arithmetic Logic Unit
AMBA	Advanced Microcontroller Bus Architecture
APB	Advanced Peripheral Bus
ASRC	Asynchronous Sample Rate Converter
AXI	Advanced eXtensible Interface
BIST	Built-In Self Test
CA/CM	Arm Cortex-A/Cortex-M
CAAM	Cryptographic Acceleration and Assurance Module
CAN	Controller Area Network
CPU	Central Processing Unit
CSI	CMOS Sensor Interface
CSU	Central Security Unit
CTI	Cross Trigger Interface
DAP	Debug Access Port
DDR	Double data rate
DMA	Direct memory access
DPLL	Digital phase-locked loop
DRAM	Dynamic random access memory
ECC	Error correcting codes
ECSPI	Enhanced Configurable SPI
LPSPI	Low-power SPI
EDMA	Enhanced Direct Memory Access
EIM	External Interface Module
ENET	Ethernet
EPIT	Enhanced Periodic Interrupt Timer
EPROM	Erasable Programmable Read-Only Memory
ETF	Embedded Trace FIFO
ETM	Embedded Trace Macrocell
FIFO	First-In-First-Out
GIC	General Interrupt Controller
GPC	General Power Controller
GPIO	General-Purpose I/O
GPR	General-Purpose Register
GPS	Global Positioning System
GPT	General-Purpose Timer
GPU	Graphics Processing Unit
GPV	Global Programmers View
HAB	High-Assurance Boot

Table continues on the next page...

Term	Meaning
I2C or I^2C	Inter-Integrated Circuit
IC	Integrated Circuit
IEEE	Institute of Electrical and Electronics Engineers
IOMUX	Input-Output Multiplexer
IP	Intellectual Property
IrDA	Infrared Data Association
JTAG	Joint Test Action Group (a serial bus protocol usually used for test purposes)
LCDIF	Liquid Crystal Display Interface
LDO	Low-Dropout
LIFO	Last-In-First-Out
LRU	Least-Recently Used
LSB	Least-Significant Byte
LUT	Look-Up Table
LVDS	Low Voltage Differential Signaling
MAC	Medium Access Control
MCM	Miscellaneous control Module
MMC	Multimedia Card
MSB	Most-Significant Byte
MT/s	Mega Transfers per second
OCRAM	On-Chip Random-Access Memory
OCOTP	On-Chip One-Time Programmable Controller
PCI	Peripheral Component Interconnect
PCIe	PCI enhanced
PCMCIA	Personal Computer Memory Card International Association
PGC	Power Gating Controller
PIC	Programmable Interrupt Controller
PMU	Power Management Unit
POR	Power-On Reset
PSRAM	Pseudo-Static Random Access Memory
PWM	Pulse Width Modulation
PXP	Pixel Pipeline
QoS	Quality of Service
R2D	Radians to Degrees
RISC	Reduced Instruction Set Computing
ROM	Read-Only Memory
ROMCP	ROM Controller with Patch
RTOS	Real-Time Operating System
Rx	Receive
SAI	Synchronous Audio Interface
SCU	Snoop Control Unit
SD	Secure Digital

Table continues on the next page...

Term	Meaning
SDIO	Secure Digital Input/Output
SDLC	Synchronous Data Link Control
SDMA	Smart DMA
SIM	Subscriber Identification Module
SNVS	Secure Non-Volatile Storage
SoC	System-on-Chip
SPBA	Shared Peripheral Bus Arbiter
SPDIF	Sony Phillips Digital Interface
SPI	Serial Peripheral Interface
SRAM	Static Random-Access Memory
SRC	System Reset Controller
TFT	Thin-Film Transistor
TPIU	Trace Port Interface
TSGEN	Time Stamp Generator
Tx	Transmit
TZASC	TrustZone Address Space Controller
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
USDHC	Ultra Secured Digital Host Controller
WDOG	Watchdog
WLAN	Wireless Local Area Network
WXGA	Wide Extended Graphics Array

1.4 Features

1.4.1 Arm Cortex-A53 MPCore™ Platform

The i.MX 8M Family Applications Processors are based on the Arm Cortex-A53 MPCore™ Platform, which has the following features:

- Quad symmetric Cortex-A53 processors, including:
  - 32 KB L1 Instruction Cache
  - 32 KB L1 Data Cache
  - Media Processing Engine (MPE) with NEON technology supporting the Advanced Single Instruction Multiple Data architecture
  - Floating Point Unit (FPU) with support of the VFPv4-D16 architecture
- Support of 64-bit Armv8-A architecture

- 512 KB unified L2 cache
- Target frequency of 1.8GHz

1.4.2 Arm Cortex-M4 Platform

Cortex-M4 Core Platform include the following:

- Low power microcontroller available for customer application:
  - Low power standby mode
  - IoT features including Weave
  - Manage IR or wireless remote
- Arm Cortex M4 CPU Processor, including:
  - 16 KB L1 Instruction Cache
  - 16 KB L1 Data Cache
  - 256 KB TCM

1.4.3 System Bus and Interconnect

System bus and interconnect include the following:

- Network interconnect (NoC) AXI arbiter
- Quality of service controller (QoSC) to configure priorities and limits of AXI transcations
- Performance monitor (PERFMON) to monitor AXI bus activity
- Debug monitor (DBGMON) to record AXI transactions preceding a system reset

1.4.4 Clocking and Resets

Clocking and resets include:

- Clock control module (CCM) provides centralized clock generation and control
  - Simplified clock tree structure
  - Unified clock programming model for each clock root
  - Multicore awareness for resource domains
- System reset controller (SRC) provides reset generation and distribution

1.4.5 Interrupts and DMA

Interrupts and DMA include:

- 128 shared peripheral interrupts routed to Cortex-A53 Global Interrupt Controller (GIC) and Cortex-M4 nested vector interrupt controller (NVIC) for flexible interrupt handling
- Three Smart direct memory access (SDMA) engines. Although these three engines are identical to each other, they are integrated into the processor to serve different peripherals.
  - SDMA-1 is a general-purpose DMA engine which can be used by low speed peripherals including UART, SPI and also others peripherals.
  - SDMA-2 and SMDA-3 is used for audio interface, including SAI-1/2/3/5/6, SPDIF and PDM audio input.

1.4.6 On-Chip Memory

The on-chip memory system consists of the following:

- Boot ROM (256KB)
- On-chip RAM (256KB + 32KB)

1.4.7 External Memory Interface

The external memory interfaces supported on this chip include:

- 16/32-bit DRAM Interface:
  - LPDDR4-3000
  - DDR4-2400
  - DDR3L-1600
- 8-bit NAND FLASH, including support for Raw MLC/SLC devices, BCH ECC up to 62-bit, and ONFi3.2 compliance (clock rates up to 100 MHz and data rates up to 200 MB/sec)
- eMMC 5.0 FLASH (2 interfaces)
- SPI NOR FLASH (3 interfaces)
- FlexSPI FLASH with support for XIP (for M4 in low-power mode) and parallel read mode of two identical FLASH devices

1.4.8 Timers

The timers on this chip include:

- One local generic timer integrated into each Cortex-A53 CPU
- Global system counter with timer bus interface to Cortex-A53 MPCore generic timers
- One local system timer (SysTick) integrated into the Cortex-M4 CPU
- Six general purpose timer (GPT) modules
- Three watchdog timer (WDOG) modules
- Four pulse width modulation (PWM) modules

1.4.9 Graphics Processing Unit (GPU)

The chip incorporates the following Graphics Processing Unit (GPU) features:

- 2D/3D acceleration
- Target frequency of 800 MHz
- Support OpenGL ES 1.1, 2.0, OpenVG 1.1
- TrustZone support using a local MMU to manage secure regions
- Support multi-source composition
- Support one-pass filter
- Support tile format

1.4.10 Video Processing Unit (VPU)

The chip incorporates the following Video Processing Unit (VPU) features:

- VP9 Profile 0, 2 (10 bit) decoder (VPU G2)
- HEVC/H.265 decoder (VPU G2)
- AVC/H.264 Baseline, Main, High decoder (VPU G1)
- VP8 decoder (VPU G1)
- AVC/H.264 encoder (VPU H1)
- VP8 encoder (VPU H1)
- TrustZone support

1.4.11 Display Interfaces

The chip has the following display support:

- LCDIF Display Controller:
  - Supports up to 2 layers of overlay
  - Support up to 1080p60 display through MIPI DSI
- MIPI Interface:
  - 4-lane MIPI CSI interface
  - 4-lane MIPI DSI interface
- CSI Interface:
  - CSI is a simple camera interface which is used to capture the MIPI CSI input and save the pixels into memory

1.4.12 Audio

Audio include the following:

- S/PDIF Input and Output, including a new Raw Capture input mode
- Five external SAI (synchronous audio interface) modules supporting I2S, AC97, TDM, codec/DSP and DSD interfaces, including one SAI with 8 TX and 8 RX lanes, one SAI with 4 TX and 4 RX lanes, two SAI with 2 TX and 2 RX lanes, and one SAI with 1 TX and 1 RX lanes. Supports over 20 channels of audio subject to I/O limitations.

1.4.13 General Connectivity Interfaces

The chip contains a rich set of general connectivity interfaces, including:

- One PCI Express (PCIe):
  - Single lane supporting PCIe Gen 2
  - Dual mode operation to function as root complex or endpoint
  - Integrated PHY interface
  - Supports L1 low power substate
- Two USB 2.0 OTG controllers with integrated PHY interface
  - Spread spectrum clock support
- Three Ultra Secure Digital Host Controller (uSDHC) interfaces
  - MMC 5.0 compliance with HS400 DDR signaling to support up to 400 MB/sec
  - SD/SDIO 3.01 compliance with 200 MHZ SDR signaling to support up to 100 MB/sec
  - Support for SDXC (extended capacity)
- One Gigabit Ethernet controller with support for EEE, Ethernet AVB and IEEE1588
- Four universal asynchronous receiver/transmitter (UART) modules

- Four I2C modules
- Three SPI modules

1.4.14 Security

Security functions are enabled and accelerated by the following hardware:

- RDC – Resource Domain Controller:
  - Supports 4 domains and up to 8 regions
- Arm TrustZone including the TZ architecture:
  - ARM Cortex-A53 MPCore TrustZone support
- On-chip RAM (OCRAM) secure region protection using OCRAM controller
- High Assurance Boot (HAB)
- Cryptographic Acceleration and Assurance Module (CAAM)
  - Support Widevine and PlayReady content protection
  - Public Key Cryptography (PKHA) with RSA and Elliptic Curve (ECC) algorithms
  - Real-time integrity checker (RTIC)
  - DRM support for RSA, AES, 3DES, DES
  - Side channel attack resistance
  - True random number generation (RNG)
  - Manufacturing protection support
- Secure Non-Volatile Storage (SNVS), including
  - Secure Real Time Clock (SRTC)
- Secure JTAG Controller (SJC)

1.4.15 Multicore Support

Multicore support contains:

- Resource domain controller (RDC) to support isolation and safe sharing of system resources
- Messaging unit (MU)
- Hardware Semaphore (SEMA42)
- Shared bus topology

1.4.16 GPIO and Pin Multiplexing

- General-purpose input/output (GPIO) modules with interrupt capability
- Input/output multiplexing controller (IOMUXC) to provide centralized pad control

1.4.17 Power Management

The power management unit consists of:

- Temperature sensor with programmable trip points
- Flexible power domain partitioning with internal power switches to support efficient power management

1.4.18 System Debug

The system debug features are:

- ARM CoreSight debug and trace architecture
- Trace Port Interface Unit (TPIU) to support off-chip real-time trace
- Embedded Trace FIFO (ETF) with 4 KB internal storage to provide trace buffering
- Unified trace capability for Quad Cortex-A53 and Cortex-M4 CPUs
- Cross Triggering Interface (CTI)
- Support for 5-pin (JTAG) debug interfaces

1.5 Architectural Overview

This section contains the i.MX 8M Mini architectural details.

1.5.1 Block Diagram

The high-level block diagram is shown in the figure below.

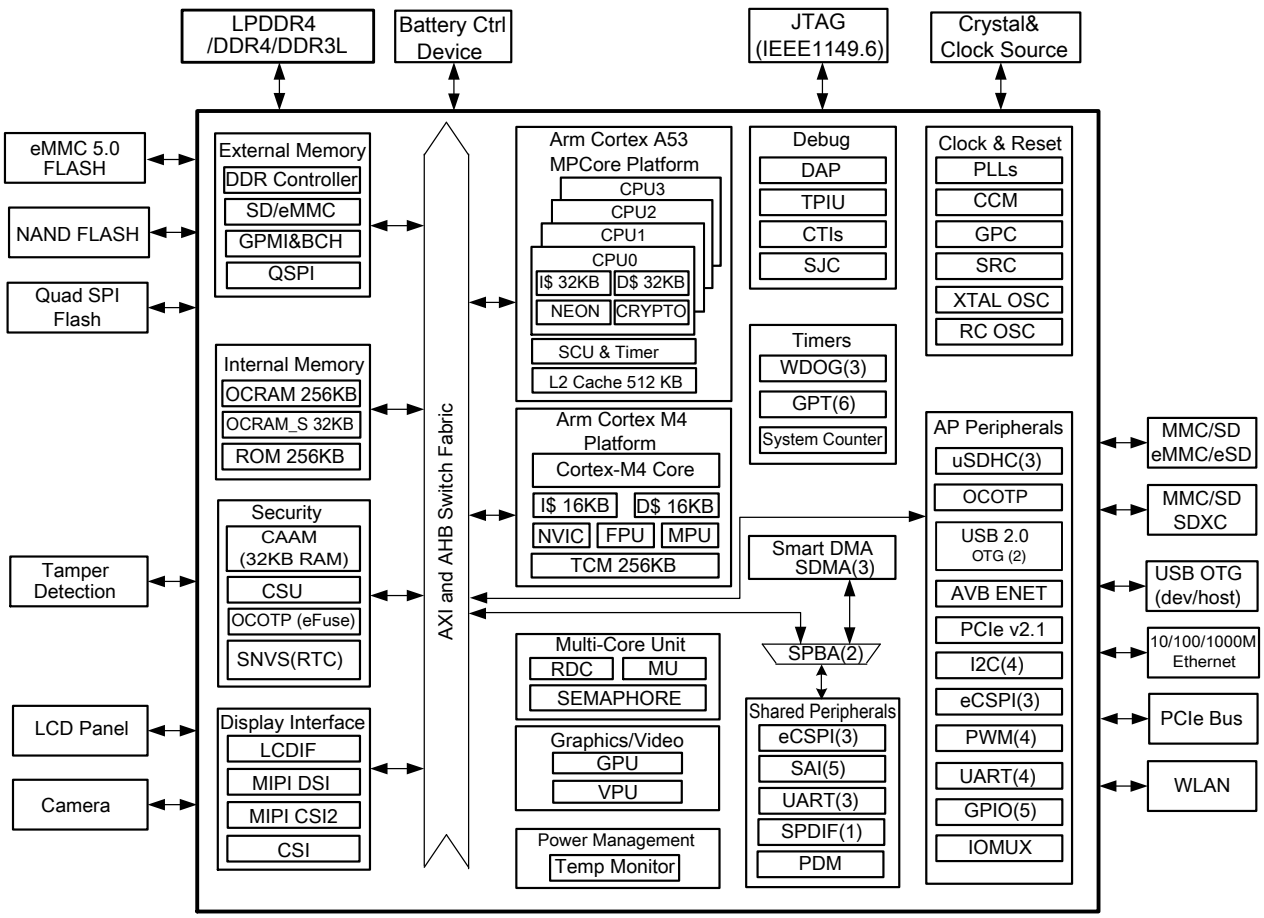


Figure 1-1. Block Diagram

1.6 Primary Boot Options

The i.MX 8M Mini supports the following boot devices:

- NAND FLASH (including SLC and MLC)
- SDIO / MMC / SDXC
- eSD 3.0/eMMC 5.0 (fast boot)
- SPI (serial FLASH)
- USB
- QSPI
- Ethernet (via plug-in mode)

The Quad-A53 core on i.MX 8M Mini is enabled during boot as the primary core to handle the entire secure boot flow. The chip will always boot from A53 core first, the M4 core will be held in reset during the A53 boot and won't run until it is enabled by the A53 core. The image for M4 cores will be loaded into memory and authenticated by the A53 core.

1.7 Endianness Support

This chip supports Little Endian mode only.

Chapter 2
Memory Map

2.1 Memory

This chapter introduces the memory architecture of the chip. The system memory high-level partition is defined below:

2.1.1 Memory system overview

2.1.1.1 On-chip L1, L2 caches, TCM

Cortex-A53 MPcore Platform

- Level 1 Cache (4x per Cortex-A53 Core)
  - Instruction (32 KB)
  - Data (32 KB)
- Level 2 Cache, shared by the four Cortex-A53 cores:
  - Unified instruction and data (512 KB)

Cortex M4 Platform

- Cache
  - Instruction (16 KB)
  - Data (16 KB)
- Tightly-Coupled-Memory
  - TCML on Code Bus (128KB)
  - TCMH on System Bus (128KB)

2.1.1.2 On-chip memories

- Boot ROM (256 KB)

- On-Chip RAM - OCRAM (256 KB)
- On-Chip RAM for State Retention - OCRAM_S (32 KB)

2.1.1.3 External L3 memories

The chip supports external memories, via the following memory interfaces / controllers:

- 16/32-bit DRAM Interface:
  - LPDDR4-3000, DDR4-2400, DDR3L-1600
- 8-bit NAND FLASH, including support for Raw MLC/SLC devices, BCH ECC up to 62-bit, and ONFi 2.x compliance (clock rates up to 100 MHz and data rates up to 200 MB/sec)
- eMMC 5.0 FLASH
- SPI NOR FLASH
- Quad SPI FLASH with support for XIP (for M4 in low-power mode) and parallel read mode of two identical FLASH devices

2.1.2 Cortex-A53 Memory Map

Start Address	End Address	Region	Size	Description
1_0000_0000	2_3FFF_FFFF	DDR Address	5120MB	DDR Memory (Quad-A53 only)
4000_0000	FFFF_FFFF	DDR Address	3072MB	DDR Memory (All modules)
3FC0_0000	3FFF_FFFF	Reserved	4MB	Reserved
3F80_0000	3FBF_FFFF	Reserved	4MB	Reserved
3F40_0000	3F7F_FFFF	Reserved	4MB	Reserved
3E00_0000	3F3F_FFFF	Reserved	20MB	Reserved
3DC0_0000	3DFF_FFFF	DDRC	4MB	Reserved
3D80_0000	3DBF_FFFF	DDRC	4MB	DDR PERF_MON
3D40_0000	3D7F_FFFF	DDRC	4MB	DDR CTL
3C00_0000	3D3F_FFFF	DDRC	20MB	DDR PHY
3890_0000	3BFF_FFFF	Reserved	55MB	Reserved
3880_0000	388F_FFFF	GIC	1MB	GIC REG
3870_0000	387F_FFFF	Reserved	1MB	Reserved
3850_0000	386F_FFFF	Reserved	2MB	Reserved
3830_0000	384F_FFFF	VPU	2MB	VPU
3820_0000	382F_FFFF	Reserved	1MB	Reserved
3810_0000	381F_FFFF	Reserved	1MB	Reserved
3801_0000	380F_FFFF	Reserved	960KB	Reserved
3800_0000	3800_FFFF	GPU REG	64KB	GPU REG

Table continues on the next page...

Start Address	End Address	Region	Size	Description
3600_0000	37FF_FFFF	QSPI RX Buffers	32MB	Reserved (QSPI2 RX Buffer)
3400_0000	35FF_FFFF		32MB	QSPI1 RX Buffer
33C0_0000	33FF_FFFF	PCIe REG	4MB	Reserved
3380_0000	33BF_FFFF		4MB	PCIe1 REG
3310_0000	337F_FFFF	Reserved	7MB	Reserved
3301_0000	330F_FFFF	Reserved	960KB	Reserved
3300_8000	3300_FFFF	QSPI TX Buffers	32KB	QSPI1 TX Buffer
3300_0000	3300_7FFF	APBH DMA	32KB	APBH DMA
32C0_0000	32FF_FFFF	Periph (AIPS)	4MB	AIPS4
3290_0000	32BF_FFFF	Reserved	3MB	Reserved
3280_0000	328F_FFFF	GPV_8	1MB	"HSIO" configuration
3270_0000	327F_FFFF	GPV_7	1MB	NoC configuration
3260_0000	326F_FFFF	GPV_6	1MB	"m4" configuration port
3250_0000	325F_FFFF	GPV_5	1MB	"display" configuration port
3240_0000	324F_FFFF	GPV_4	1MB	"enet" configuration port
3230_0000	323F_FFFF	GPV_3	1MB	"per_m" configuration port
3220_0000	322F_FFFF	GPV_2	1MB	"per_s" configuration port
3210_0000	321F_FFFF	GPV_1	1MB	"wakeup" configuration port
3200_0000	320F_FFFF	GPV_0	1MB	"main" configuration port
3100_0000	31FF_FFFF	Reserved	16MB	Reserved
30C0_0000	30FF_FFFF	Periph (AIPS)	4MB	Reserved
3080_0000	30BF_FFFF		4MB	AIPS-3, See IP listing on separate map.
3040_0000	307F_FFFF		4MB	AIPS-2, See IP listing on separate map.
3000_0000	303F_FFFF		4MB	AIPS-1, See IP listing on separate map.
2900_0000	2FFF_FFFF	Reserved	112MB	Reserved
2800_0000	28FF_FFFF	A53 / DAP	16MB	A53 / DAP
2000_0000	27FF_FFFF	Reserved	128MB	Reserved
1800_0000	1FFF_FFFF	PCIe-1	128MB	PCIe-1
0800_0000	17FF_FFFF	QSPI	256MB	QSPI
0400_0000	07FF_FFFF	Reserved	64MB	Reserved
0100_0000	03FF_FFFF	Reserved	48MB	Reserved
00C0_0000	00FF_FFFF	Reserved	4MB	Reserved
00B0_0000	00BF_FFFF	Reserved	1MB	Reserved
00A0_0000	00AF_FFFF	OCRAM	1MB	Reserved (OCRAM)
0094_8000	009F_FFFF		736KB	Reserved (OCRAM)
0094_0000	0094_7FFF		32KB	Reserved (OCRAM)
0092_0000	0093_FFFF		128KB	OCRAM 128KB
0090_0000	0091_FFFF		128KB	OCRAM 128KB
0084_0000	008F_FFFF	Reserved	768KB	Reserved
0082_0000	0083_FFFF	TCM	128KB	Reserved (TCM)
0080_0000	0081_FFFF		128KB	TCMU

Table continues on the next page...

Start Address	End Address	Region	Size	Description
007E_0000	007F_FFFF		128KB	TCML
007C_0000	007D_FFFF		128KB	Reserved (TCM)
0070_0000	007B_FFFF	Reserved	768KB	Reserved
0060_0000	006F_FFFF	Reserved	1MB	Reserved
0050_0000	005F_FFFF	Reserved	1MB	Reserved
0040_0000	004F_FFFF	Reserved	1MB	Reserved
0020_0000	003F_FFFF	Reserved	2MB	Reserved
0019_0000	001F_FFFF	Reserved	448KB	Reserved
0018_8000	0018_FFFF	OCRAM_S	32KB	Reserved (OCRAM_S)
0018_0000	0018_7FFF		32KB	OCRAM_S
0011_0000	0017_FFFF	Reserved	448KB	Reserved
0010_8000	0010_FFFF	CAAM	32KB	Reserved (CAAM)
0010_0000	0010_7FFF		32KB	CAAM (32K secure RAM)
0004_0000	000F_FFFF	Reserved	768KB	Reserved
0003_F000	0003_FFFF	Boot ROM	4KB	Boot ROM - Protected 4KB area
0000_0000	0003_EFFF		252KB	Boot ROM

2.1.3 Cortex-M4 Memory Map

Start Address	End Address	Region	Size	Allocation
E010_0000	FFFF_FFFF	Reserved	511MB	Reserved
E000_0000	E00F_FFFF	CM4 PPB	1MB	CM4 PPB
D800_0000	DFFF_FFFF	Reserved	128MB	Reserved
D000_0000	D7FF_FFFF	PCIe-1	128MB	PCIe-1
C000_0000	CFFF_FFFF	FLASH	256MB	QSPI
4000_0000	BFFF_FFFF	DDR Address	2048MB	DDR Memory
3FC0_0000	3FFF_FFFF	Reserved	4MB	Reserved
3F80_0000	3FBF_FFFF	Reserved	4MB	Reserved
3F40_0000	3F7F_FFFF	Reserved	4MB	Reserved
3E00_0000	3F3F_FFFF	Reserved	20MB	Reserved
3DC0_0000	3DFF_FFFF	DDRC	4MB	Reserved
3D80_0000	3DBF_FFFF	DDRC	4MB	DDR PERF_MON
3D40_0000	3D7F_FFFF	DDRC	4MB	DDR CTL
3C00_0000	3D3F_FFFF	DDRC	20MB	DDR PHY
3890_0000	3BFF_FFFF	Reserved	55MB	Reserved
3880_0000	388F_FFFF	GIC	1MB	GIC
3870_0000	387F_FFFF	Reserved	1MB	Reserved

Table continues on the next page...

Start Address	End Address	Region	Size	Allocation
3850_0000	386F_FFFF	Reserved	2MB	Reserved
3830_0000	384F_FFFF	VPU	2MB	VPU
3820_0000	382F_FFFF	Reserved	1MB	Reserved
3810_0000	381F_FFFF	Reserved	1MB	Reserved
3801_0000	380F_FFFF	Reserved	960KB	Reserved
3800_0000	3800_FFFF	GPU REG	64KB	GPU REG
3600_0000	37FF_FFFF	QSPI RX Buffers	32MB	Reserved (QSPI2 RX Buffer)
3400_0000	35FF_FFFF		32MB	QSPI1 RX Buffer
33C0_0000	33FF_FFFF	PCIe REG	4MB	Reserved
3380_0000	33BF_FFFF		4MB	PCIe1 REG
3310_0000	337F_FFFF	Reserved	7MB	Reserved
3301_0000	330F_FFFF	Reserved	960KB	Reserved
3300_8000	3300_FFFF	QSPI TX Buffers	32KB	QSPI1 TX Buffer
3300_0000	3300_7FFF	APBH DMA	32KB	APBH DMA
32C0_0000	32FF_FFFF	Periph (AIPS)	4MB	AIPS4
3290_0000	32BF_FFFF	Reserved	3MB	Reserved
3280_0000	328F_FFFF	GPV_8	1MB	"HSIO" configuration
3270_0000	327F_FFFF	GPV_7	1MB	NoC configuration
3260_0000	326F_FFFF	GPV_6	1MB	"m4" configuration port
3250_0000	325F_FFFF	GPV_5	1MB	"display" configuration port
3240_0000	324F_FFFF	GPV_4	1MB	"enet" configuration port
3230_0000	323F_FFFF	GPV_3	1MB	"per_m" configuration port
3220_0000	322F_FFFF	GPV_2	1MB	"per_s" configuration port
3210_0000	321F_FFFF	GPV_1	1MB	"wakeup" configuration port
3200_0000	320F_FFFF	GPV_0	1MB	"main" configuration port
3100_0000	31FF_FFFF	Reserved	16MB	Reserved
30C0_0000	30FF_FFFF	Pheriph (AIPS)	4MB	Reserved
3080_0000	30BF_FFFF		4MB	AIPS-3, See IP listing on separate map.
3040_0000	307F_FFFF		4MB	AIPS-2, See IP listing on separate map.
3000_0000	303F_FFFF		4MB	AIPS-1, See IP listing on separate map.
2900_0000	2FFF_FFFF	Reserved	112MB	Reserved
2800_0000	28FF_FFFF	A53/ DAP	16MB	A53/ DAP
2400_0000	27FF_FFFF	Reserved	64MB	Reserved
2200_0000	23FF_FFFF	Reserved	32MB	Reserved
2100_0000	21FF_FFFF	Reserved	16MB	Reserved
2040_0000	20FF_FFFF	Reserved	12MB	Reserved
2030_0000	203F_FFFF	CM4 ALIAS SYSTEM	1MB	Reserved (OCRAM)
2024_8000	202F_FFFF		736KB	Reserved (OCRAM)
2024_0000	2024_7FFF		32KB	Reserved (OCRAM)
2022_0000	2023_FFFF		128KB	OCRAM_128KB
2020_0000	2021_FFFF		128KB	OCRAM_128KB

Table continues on the next page...

Start Address	End Address	Region	Size	Allocation
2019_0000	201F_FFFF		448KB	Reserved
2018_8000	2018_FFFF		32KB	Reserved (OCRAM_S)
2018_0000	2018_7FFF		32KB	OCRAM_S
2011_0000	2017_FFFF		448KB	Reserved
2010_8000	2010_FFFF		32KB	CAAM (Reserved)
2010_0000	2010_7FFF		32KB	CAAM (32K secure RAM)
2006_0000	200F_FFFF	Reserved	640KB	Reserved
2002_0000	2005_FFFF	Boot ROM	256KB	Boot ROM (ROMCP)
2000_0000	2001_FFFF	TCM	128KB	TCMU
1FFE_0000	1FFF_FFFF		128KB	TCML
1000_0000	1FFD_FFFF	CM4 ALIAS CODE	262016KB	DDR Code alias
0800_0000	0FFF_FFFF		128MB	QSPI Code alias
0400_0000	07FF_FFFF		64MB	Reserved
0100_0000	03FF_FFFF	Reserved	48MB	Reserved
00C0_0000	00FF_FFFF	Reserved	4MB	Reserved
00B0_0000	00BF_FFFF	Reserved	1MB	Reserved
00A0_0000	00AF_FFFF	OCRAM	1MB	Reserved (OCRAM)
0094_8000	009F_FFFF		736KB	Reserved (OCRAM)
0094_0000	0094_7FFF		32KB	Reserved (OCRAM)
0092_0000	0093_FFFF		128KB	OCRAM_128KB
0090_0000	0091_FFFF		128KB	OCRAM_128KB
0081_0000	008F_FFFF	Reserved	960KB	Reserved
0080_8000	0080_FFFF	Reserved	32KB	Reserved
0080_0000	0080_7FFF	Reserved	32KB	Reserved
007F_8000	007F_FFFF	Reserved	32KB	Reserved
007F_0000	007F_7FFF	Reserved	32KB	Reserved
0070_0000	007E_FFFF	Reserved	960KB	Reserved
0060_0000	006F_FFFF	Reserved	1MB	Reserved
0050_0000	005F_FFFF	Reserved	1MB	Reserved
0040_0000	004F_FFFF	Reserved	1MB	Reserved
0020_0000	003F_FFFF	Reserved	2MB	Reserved
0019_0000	001F_FFFF	Reserved	448KB	Reserved
0018_8000	0018_FFFF	OCRAM_S	32KB	Reserved (OCRAM_S)
0018_0000	0018_7FFF		32KB	OCRAM_S
0011_0000	0017_FFFF	Reserved	448KB	Reserved
0010_8000	0010_FFFF	CAAM	32KB	Reserved (CAAM)
0010_0000	0010_7FFF		32KB	CAAM (32K secure RAM)
0002_0000	000F_FFFF	Reserved	896KB	Reserved
0000_0000	0001_FFFF	TCML	128KB	TCML alias

2.1.4 DMA memory maps

The Smart DMA memory maps are defined in the following tables.

Table 2-1. SDMA1 Peripheral Memory Map

Address	Peripheral
0xF000	SPBA
0xE000	Reserved for SDMA internal registers
0xD000	Reserved
0xC000	Reserved
0xB000	Reserved
0xA000	Reserved
0x9000	UART2
0x8000	UART3
0x7000	Reserved for SDMA internal registers
0x6000	UART1
0x5000	Reserved
0x4000	eCSPI3
0x3000	eCSPI2
0x2000	eCSPI1
0x1000	Reserved
0x0000	Reserved for SDMA internal memory

Table 2-2. SDMA2/3 Peripheral Memory Map

Address	Peripheral
0xF000	SPBA
0xE000	Reserved for SDMA internal registers
0xD000	Reserved
0xC000	Reserved
0xB000	Reserved
0xA000	Reserved
0x9000	SPDIF1
0x8000	MICFIL
0x7000	Reserved for SDMA internal registers
0x6000	SAI6
0x5000	SAI5
0x4000	Reserved
0x3000	SAI3
0x2000	SAI2

Table continues on the next page...

Table 2-2. SDMA2/3 Peripheral Memory Map (continued)

Address	Peripheral
0x1000	SAI1
0x0000	Reserved for SDMA internal memory

2.1.5 AIPS Memory Maps

Table 2-3. AIPS1 Memory Map

Start Address	End Address	Region	NIC Port	Size
303F_0000	303F_FFFF	AIPS-1 (s_b_0)	Reserved	64KB
303E_0000	303E_FFFF		CSU	64KB
303D_0000	303D_FFFF		RDC	64KB
303C_0000	303C_FFFF		SEMAPHORE2	64KB
303B_0000	303B_FFFF		SEMAPHORE1	64KB
303A_0000	303A_FFFF		GPC	64KB
3039_0000	3039_FFFF		SRC	64KB
3038_0000	3038_FFFF		CCM	64KB
3037_0000	3037_FFFF		SNVS_HP	64KB
3036_0000	3036_FFFF		ANA_PLL	64KB
3035_0000	3035_FFFF		OCOTP_CTRL	64KB
3034_0000	3034_FFFF		IOMUXC_GPR	64KB
3033_0000	3033_FFFF		IOMUXC	64KB
3032_0000	3032_FFFF		Reserved	64KB
3031_0000	3031_FFFF		ROMCP	64KB
3030_0000	3030_FFFF		Reserved	64KB
302F_0000	302F_FFFF		GPT3	64KB
302E_0000	302E_FFFF		GPT2	64KB
302D_0000	302D_FFFF		GPT1	64KB
302C_0000	302C_FFFF		SDMA2	64KB
302B_0000	302B_FFFF		SDMA3	64KB
302A_0000	302A_FFFF		WDOG3	64KB
3029_0000	3029_FFFF		WDOG2	64KB
3028_0000	3028_FFFF		WDOG1	64KB
3027_0000	3027_FFFF		ANA_OSC	64KB
3026_0000	3026_FFFF		ANA_TSENSOR	64KB
3025_0000	3025_FFFF		Reserved	64KB
3024_0000	3024_FFFF		GPIO5	64KB
3023_0000	3023_FFFF		GPIO4	64KB

Table continues on the next page...

Table 2-3. AIPS1 Memory Map (continued)

Start Address	End Address	Region	NIC Port	Size
3022_0000	3022_FFFF		GPIO3	64KB
3021_0000	3021_FFFF		GPIO2	64KB
3020_0000	3020_FFFF		GPIO1	64KB
301F_0000	301F_FFFF		AIPS1_Configuration	64KB
3014_0000	301E_FFFF	AIPS-1 Glob. Module Enable	Reserved	704KB
3010_0000	3013_FFFF		Reserved	256KB
300F_0000	300F_FFFF	AIPS-1 (s_b_1, via SPBA) Glob. Module Enable	SPBA2	64KB
300E_0000	300E_FFFF		Reserved for SDMA2 internal memory	64KB
300D_0000	300D_FFFF		Reserved	64KB
300C_0000	300C_FFFF		Reserved	64KB
300B_0000	300B_FFFF		Reserved	64KB
300A_0000	300A_FFFF		SPDIF2	64KB
3009_0000	3009_FFFF		SPDIF1	64KB
3008_0000	3008_FFFF		MICFIL	64KB
3007_0000	3007_FFFF		Reserved for SDMA2 internal memory	64KB
3006_0000	3006_FFFF		SAI6	64KB
3005_0000	3005_FFFF		SAI5	64KB
3004_0000	3004_FFFF		Reserved	64KB
3003_0000	3003_FFFF		SAI3	64KB
3002_0000	3002_FFFF		SAI2	64KB
3001_0000	3001_FFFF		SAI1	64KB
3000_0000	3000_FFFF		Reserved for SDMA2 internal memory	64KB

Table 2-4. AIPS2 Memory Map

Start Address	End Address	Region	NIC Port	Size
307F_0000	307F_FFFF	AIPS-2 (s_b_1)	QoSC	64KB
307E_0000	307E_FFFF		Reserved	64KB
307D_0000	307D_FFFF		PERFMON2	64KB
307C_0000	307C_FFFF		PERFMON1	64KB
307B_0000	307B_FFFF		Reserved	64KB
307A_0000	307A_FFFF		Reserved	64KB
3079_0000	3079_FFFF		Reserved	64KB
3078_0000	3078_FFFF		Reserved	64KB
3077_0000	3077_FFFF		Reserved	64KB
3076_0000	3076_FFFF		Reserved	64KB

Table continues on the next page...

Table 2-4. AIPS2 Memory Map (continued)

Start Address	End Address	Region	NIC Port	Size
3075_0000	3075_FFFF		Reserved	64KB
3074_0000	3074_FFFF		Reserved	64KB
3073_0000	3073_FFFF		Reserved	64KB
3072_0000	3072_FFFF		Reserved	64KB
3071_0000	3071_FFFF		Reserved	64KB
3070_0000	3070_FFFF		GPT4	64KB
306F_0000	306F_FFFF		GPT5	64KB
306E_0000	306E_FFFF		GPT6	64KB
306D_0000	306D_FFFF		Reserved	64KB
306C_0000	306C_FFFF		System_Counter_CTRL	64KB
306B_0000	306B_FFFF		System_Counter_CMP	64KB
306A_0000	306A_FFFF		System_Counter_RD	64KB
3069_0000	3069_FFFF		PWM4	64KB
3068_0000	3068_FFFF		PWM3	64KB
3067_0000	3067_FFFF		PWM2	64KB
3066_0000	3066_FFFF		PWM1	64KB
3065_0000	3065_FFFF		Reserved	64KB
3064_0000	3064_FFFF		Reserved	64KB
3063_0000	3063_FFFF		Reserved	64KB
3062_0000	3062_FFFF		Reserved	64KB
3061_0000	3061_FFFF		Reserved	64KB
3060_0000	3060_FFFF		Reserved	64KB
305F_0000	305F_FFFF		AIPS2_configuration	64KB
3050_0000	305E_FFFF	AIPS-2 Glob. Module Enable	Reserved	960KB
3040_0000	304F_FFFF		Reserved	1024KB

Table 2-5. AIPS3 Memory Map

Start Address	End Address	Region	NIC Port	Size
30BF_0000	30BF_FFFF	AIPS-3 (s_b_2)	Reserved	64KB
30BE_0000	30BE_FFFF		ENET1	64KB
30BD_0000	30BD_FFFF		SDMA1	64KB
30BC_0000	30BC_FFFF		Reserved	64KB
30BB_0000	30BB_FFFF		QSPI	64KB
30BA_0000	30BA_FFFF		Reserved	64KB
30B9_0000	30B9_FFFF		Reserved	64KB
30B8_0000	30B8_FFFF		Reserved	64KB
30B7_0000	30B7_FFFF		Reserved	64KB
30B6_0000	30B6_FFFF		uSDHC3	64KB

Table continues on the next page...

Table 2-5. AIPS3 Memory Map (continued)

Start Address	End Address	Region	NIC Port	Size
30B5_0000	30B5_FFFF		uSDHC2	64KB
30B4_0000	30B4_FFFF		uSDHC1	64KB
30B3_0000	30B3_FFFF		Reserved	64KB
30B2_0000	30B2_FFFF		Reserved	64KB
30B1_0000	30B1_FFFF		Reserved	64KB
30B0_0000	30B0_FFFF		Reserved	64KB
30AF_0000	30AF_FFFF		Reserved	64KB
30AE_0000	30AE_FFFF		Reserved	64KB
30AD_0000	30AD_FFFF		Reserved	64KB
30AC_0000	30AC_FFFF		SEMAPHORE_HS	64KB
30AB_0000	30AB_FFFF		MU_B	64KB
30AA_0000	30AA_FFFF		MU_A	64KB
30A9_0000	30A9_FFFF		Reserved	64KB
30A8_0000	30A8_FFFF		Reserved	64KB
30A7_0000	30A7_FFFF		Reserved	64KB
30A6_0000	30A6_FFFF		UART4	64KB
30A5_0000	30A5_FFFF		I2C4	64KB
30A4_0000	30A4_FFFF		I2C3	64KB
30A3_0000	30A3_FFFF		I2C2	64KB
30A2_0000	30A2_FFFF		I2C1	64KB
30A1_0000	30A1_FFFF		Reserved	64KB
30A0_0000	30A0_FFFF		Reserved	64KB
309F_0000	309F_FFFF		AIPS3_Configuration	64KB
3094_0000	309E_FFFF	AIPS-3 Glob. Module Enable	Reserved	704KB
3090_0000	3093_FFFF		CAAM	256KB
308F_0000	308F_FFFF	AIPS-3 (s_b_2, via SPBA) Glob. Module Enable	SPBA1	64KB
308E_0000	308E_FFFF		Reserved for SDMA internal registers	64KB
308D_0000	308D_FFFF		Reserved	64KB
308C_0000	308C_FFFF		Reserved	64KB
308B_0000	308B_FFFF		Reserved	64KB
308A_0000	308A_FFFF		Reserved	64KB
3089_0000	3089_FFFF		UART2	64KB
3088_0000	3088_FFFF		UART3	64KB
3087_0000	3087_FFFF		Reserved for SDMA internal registers	64KB
3086_0000	3086_FFFF		UART1	64KB
3085_0000	3085_FFFF		Reserved	64KB
3084_0000	3084_FFFF		eCSPI3	64KB
3083_0000	3083_FFFF		eCSPI2	64KB

Table continues on the next page...

Table 2-5. AIPS3 Memory Map (continued)

Start Address	End Address	Region	NIC Port	Size
3082_0000	3082_FFFF		eCSPI1	64KB
3081_0000	3081_FFFF		Reserved	64KB
3080_0000	3080_FFFF		Reserved for SDMA internal memory	64KB

Table 2-6. AIPS4 Memory Map

Start Address	End Address	Region	NIC Port	Size
32FF_0000	32FF_FFFF	AIPS-4 (s_h_10)	Reserved	64KB
32FE_0000	32FE_FFFF		PLATFORM_CTRL	64KB
32FD_0000	32FD_FFFF		Reserved	64KB
32FC_0000	32FC_FFFF		Reserved	64KB
32FB_0000	32FB_FFFF		Reserved	64KB
32FA_0000	32FA_FFFF		Reserved	64KB
32F9_0000	32F9_FFFF		Reserved	64KB
32F8_0000	32F8_FFFF		TZASC	64KB
32F7_0000	32F7_FFFF		Reserved	64KB
32F6_0000	32F6_FFFF		Reserved	64KB
32F5_0000	32F5_FFFF		Reserved	64KB
32F4_0000	32F4_FFFF		Reserved	64KB
32F3_0000	32F3_FFFF		Reserved	64KB
32F2_0000	32F2_FFFF		Reserved	64KB
32F1_0000	32F1_FFFF		Reserved	64KB
32F0_0000	32F0_FFFF		PCIE_PHY1	64KB
32EF_0000	32EF_FFFF		Reserved	64KB
32EE_0000	32EE_FFFF		Reserved	64KB
32ED_0000	32ED_FFFF		Reserved	64KB
32EC_0000	32EC_FFFF		Reserved	64KB
32EB_0000	32EB_FFFF		Reserved	64KB
32EA_0000	32EA_FFFF		Reserved	64KB
32E9_0000	32E9_FFFF		Reserved	64KB
32E8_0000	32E8_FFFF		Reserved	64KB
32E7_0000	32E7_FFFF		Reserved	64KB
32E6_0000	32E6_FFFF		Reserved	64KB
32E5_0000	32E5_FFFF		USB2	64KB
32E4_0000	32E4_FFFF		USB1	64KB
32E3_0000	32E3_FFFF		MIPI_CSI	64KB
32E2_0000	32E2_FFFF		CSI	64KB
32E1_0000	32E1_FFFF		MIPI_DSI	64KB
32E0_0000	32E0_FFFF		LCDIF	64KB

Table continues on the next page...

Table 2-6. AIPS4 Memory Map (continued)

Start Address	End Address	Region	NIC Port	Size
32DF_0000	32DF_FFFF		AIPS4_configuration	64KB
32D0_0000	32DE_FFFF	AIPS-4 Glob. Module Enable	Reserved	960KB
32C0_0000	32CF_FFFF		Reserved	1024KB

2.1.6 DAP Memory Map

Table 2-7. DAP Memory Map Table

Start Address	End Address	Size	Allocation
28C0_A000	28FF_FFFF	4056KB	Reserved
28C0_9000	28C0_9FFF	4KB	HUGO_CXCTI1
28C0_8000	28C0_8FFF	4KB	HUGO_CXCTI0
28C0_7000	28C0_7FFF	4KB	CXTPIU
28C0_6000	28C0_6FFF	4KB	CXTMC_ETR
28C0_5000	28C0_5FFF	4KB	ATB_REPLICATOR
28C0_4000	28C0_4FFF	4KB	CXTMC_ETB
28C0_3000	28C0_3FFF	4KB	HUGO_ATB_FUNNEL
28C0_2000	28C0_2FFF	4KB	CXTSGEN_READ
28C0_1000	28C0_1FFF	4KB	CXTSGEN_CTRL
28C0_0000	28C0_0FFF	4KB	HUGO ROM Table
28B5_0000	28BF_FFFF	704KB	Reserved
28B4_0000	28B4_FFFF	64KB	Reserved
28B3_0000	28B3_FFFF	64KB	Reserved
28B2_0000	28B2_FFFF	64KB	Reserved
28B1_0000	28B1_FFFF	64KB	Reserved
28A5_0000	28B0_FFFF	768KB	Reserved
28A4_0000	28A4_FFFF	64KB	Reserved
28A3_0000	28A3_FFFF	64KB	Reserved
28A2_0000	28A2_FFFF	64KB	Reserved
28A1_0000	28A1_FFFF	64KB	Reserved
2895_0000	28A0_FFFF	768KB	Reserved
2894_0000	2894_FFFF	64KB	Reserved
2893_0000	2893_FFFF	64KB	Reserved
2892_0000	2892_FFFF	64KB	Reserved
2891_0000	2891_FFFF	64KB	Reserved
2885_0000	2890_FFFF	768KB	Reserved
2884_0000	2884_FFFF	64KB	Reserved

Table continues on the next page...

Table 2-7. DAP Memory Map Table (continued)

Start Address	End Address	Size	Allocation
2883_0000	2883_FFFF	64KB	Reserved
2882_0000	2882_FFFF	64KB	Reserved
2881_0000	2881_FFFF	64KB	Reserved
2880_0000	2880_FFFF	64KB	Reserved
2875_0000	287F_FFFF	704KB	Reserved
2874_0000	2874_FFFF	64KB	MP4-CPU3 Trace
2873_0000	2873_FFFF	64KB	MP4-CPU3 PMU
2872_0000	2872_FFFF	64KB	MP4-CPU3 CTI
2871_0000	2871_FFFF	64KB	MP4-CPU3 Debug
2865_0000	2870_FFFF	768KB	Reserved
2864_0000	2864_FFFF	64KB	MP4-CPU2 Trace
2863_0000	2863_FFFF	64KB	MP4-CPU2 PMU
2862_0000	2862_FFFF	64KB	MP4-CPU2 CTI
2861_0000	2861_FFFF	64KB	MP4-CPU2 Debug
2855_0000	2860_FFFF	768KB	Reserved
2854_0000	2854_FFFF	64KB	MP4-CPU1 Trace
2853_0000	2853_FFFF	64KB	MP4-CPU1 PMU
2852_0000	2852_FFFF	64KB	MP4-CPU1 CTI
2851_0000	2851_FFFF	64KB	MP4-CPU1 Debug
2845_0000	2850_FFFF	768KB	Reserved
2844_0000	2844_FFFF	64KB	MP4-CPU0 Trace
2843_0000	2843_FFFF	64KB	MP4-CPU0 PMU
2842_0000	2842_FFFF	64KB	MP4-CPU0 CTI
2841_0000	2841_FFFF	64KB	MP4-CPU0 Debug
2840_0000	2840_FFFF	64KB	MP4 ROM Table
2801_0000	283F_FFFF	4032KB	Reserved
2800_0000	2800_FFFF	64KB	DAP ROM Table

Chapter 3
Security

3.1 System Security

3.1.1 Overview

The security system modules are described in the sections below.

3.1.2 Central Security Unit (CSU)

The chip uses the CSU to manage security for all masters/slaves that don’t directly support security like the CPU core.

3.1.3 Cryptographic Acceleration and Assurance Module (CAAM)

CAAM is the chip's cryptographic acceleration and assurance module, which serves as NXP's latest cryptographic acceleration and offloading hardware. It combines functions previously implemented in separate modules to create a modular and scalable acceleration and assurance engine. It also implements block encryption algorithms, stream cipher algorithms, hashing algorithms, public key algorithms, run-time integrity checking, a secure memory controller, and a hardware random number generator.

CAAM supports the following key features:

- PKHA block to support Public Key Cryptography with RSA 4096 and ECC algorithms
- 3 Job Rings
- RTIC (real time integrity checking)
- AES, DES, 3DES with Side channel attack resistance
- Widevine cipher text stealing (AES-CBC-CTS mode)
- PlayReady content protection

In order to provide better video content protection, CAAM also supports the domain based resource protection.

3.1.4 Secure Non-Volatile Storage (SNVS)

Secure Non-Volatile Storage (SNVS) is a companion logic block to the Cryptographic Acceleration and Assurance Module (CAAM). This block is the chip's central reporting point for security-relevant events, such as the success or failure of boot software validation and the detection of potential security compromises. This security event information allows the SNVS to determine whether the chip is in a trustworthy state.

3.1.5 On-Chip OTP Controller (OCOTP_CTRL)

The OCOPT_CTRL is used to control the 8K bit OTP fuse in the chip. It supports:

- Load fuse into shadow registers after power-on reset;
- Register interface to allow SW to read, override or program the fuse
- Flexible permission control to the fuse, including read-protect, override-protect, program-protect, lock and etc
- Programming sequence to allow single bit to be programmed individually, and also allow the non-programmed bit to be programmed later

For the 8K bits fuse, following bits are reserved for various IP requirements:

- Fuse control to disable each of the four Cortex A53 cores.
- Fuse control to disable individual IP modules such as MIPI, ENET, USB, PCIe and etc.
- Fuse control to disable the Dolby Vision logic inside the display controller block.
- Fuse for ROM patch, with a dedicated fuse lock bit for security purpose.

3.1.6 TrustZone

TrustZone security architecture is supported in the chip. For on-chip RAM, both OCRAM/OCRAM_S have TrustZone access control support through its TZ wrapper logic. One region with configurable address range of the OCRAM/OCRAM_S can be set to TZ access only. DRAM has a dedicated TZASC block that can support up to 16 configurable memory regions.

3.1.7 Resource Domain Controller (RDC)

The ichip uses domain based resource control architecture for the memory/peripheral resource sharing and isolation between the Quad-core Cortex A53 platform, Cortex M4 core, and other bus masters. RDC supports flexible configuration on IP's access permission, each individual IP can, for example, be configured as A-core only or M-core only. RDC also supports multiple regions with flexible access permission control for each memory space. The memory region access control is defined in the table below. The security and exclusive support for DRAM will be done by the TZASC and the DRAM controller itself.

NOTE

The RDC and TrustZone features are completely independent of each other but will be used together.

Table 3-1. Memory Region Access Control

Memory Space	Regions Supported	Granularity	Security Support	Exclusive Support
DRAM	8 regions	4K Bytes		
QSPI1	8 regions	4K Bytes		
PCIe1	8 regions	4K Bytes		
OCRAM	5 regions	128 Bytes		
OCRAM_S	5 regions	128 Bytes		
TCM	5 regions	128 Bytes		
GIC	4 regions	4K Bytes		
GPU	8 regions	4K Bytes		
VPU	4 regions	4K Bytes		
DEBUG(DAP)	4 regions	4K Bytes		
DDRC (REG)	5 regions	4K Bytes		

3.2 Resource Domain Controller (RDC)

3.2.1 Overview

The Resource Domain Controller (RDC) provides robust support for the isolation of destination memory mapped locations such as peripherals and memory to a single core, a bus master, or set of cores and bus masters.

Many of today's processors have multiple cores for increased performance and flexibility. In some cases, the cores serve different functions (e.g. user level applications versus real time machine control) and in such cases the software for each core may be developed by different providers.

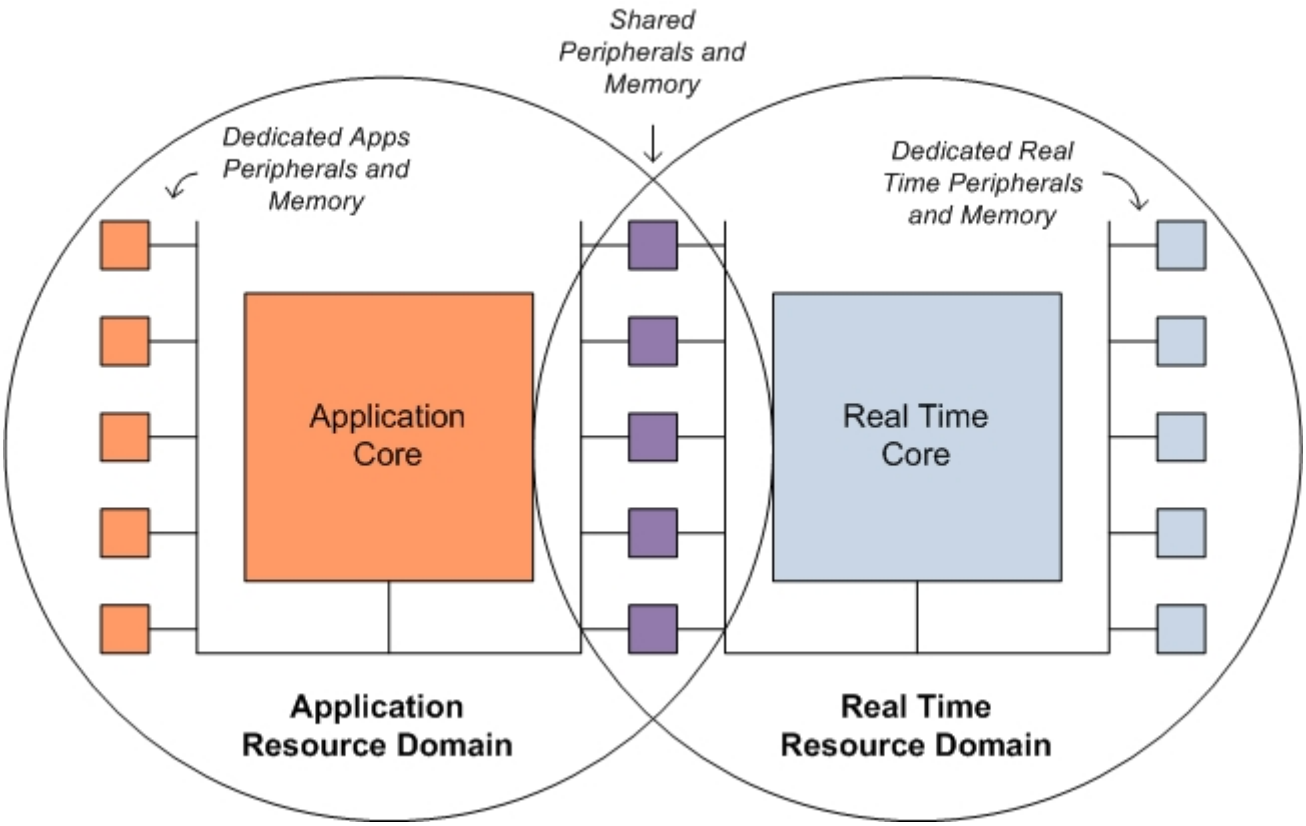


Figure 3-1. Dedicated and Shared Peripherals

For efficiency reasons the code on the cores may share chip resources such as peripherals and memory. The sharing of chip resources between the somewhat independent processing domains allows for the opportunity of data collisions where information stored in peripherals or memory by a process on one core is overwritten by software running on another core. Without careful collaboration between the two operating systems inadvertent malfunction or degradation in performance may result.

The RDC provides a mechanism to allow boot time configuration code to establish resource domains by assigning cores, bus masters, peripherals and memory regions to domain identifiers. Once configured, bus transactions are monitored to restrict accesses initiated by cores and bus masters to their respective peripherals and memory.

For shared peripherals, the RDC provides a semaphore-based locking mechanism to provide for temporary exclusivity while the domain software uses the peripheral. Once the software of one domain has finished the task and finished with the peripheral then it may release the semaphore making the peripheral available to the other domain.

3.2.1.1 Features

Resource domain subsystem has the following features:

- Assignment of cores, bus masters, peripherals, and memory regions to a resource domain
- Fixed memory resolution of 128 Bytes for small address spaces and 4 KB for large address spaces
- Four resource domain identifiers
- Memory read/write access controls for each resource domain and region
- Optional semaphore-based, hardware-enforced exclusive access of shared peripherals to a resource domain
- Prioritized access permissions for overlapping memory regions
- Automatic restoration of resource domain access permissions to memory regions in the power-down domain

3.2.2 Functional Description

The RDC is the central location for creation of isolated resource domains and for the enablement of semaphore-based access also known as “safe sharing”. Configuration software assigns one of four resource domain identifiers to each core and bus master, and allocates each memory region and peripheral to one or more resource domains.

Memory Read or Write access privileges for each resource domain are declared for each memory region. In addition, the software configuration determines which shared peripherals (those peripherals allocated to more than one domain) require safe sharing by setting the semaphore-required configuration for each peripheral.

The RDC configuration information is sent to the fabric ports, memories gaskets, semaphore controller, and peripherals to control access based on domain assignments. The fabric uses the domain identifier associated with each port to include this information along with the bus transaction. When the slave gasket encounters a bus transaction it makes a comparison of the transaction domain ID to the RDC-provided list of allowed domains. If the transaction domain ID is on the list then access may be permitted.

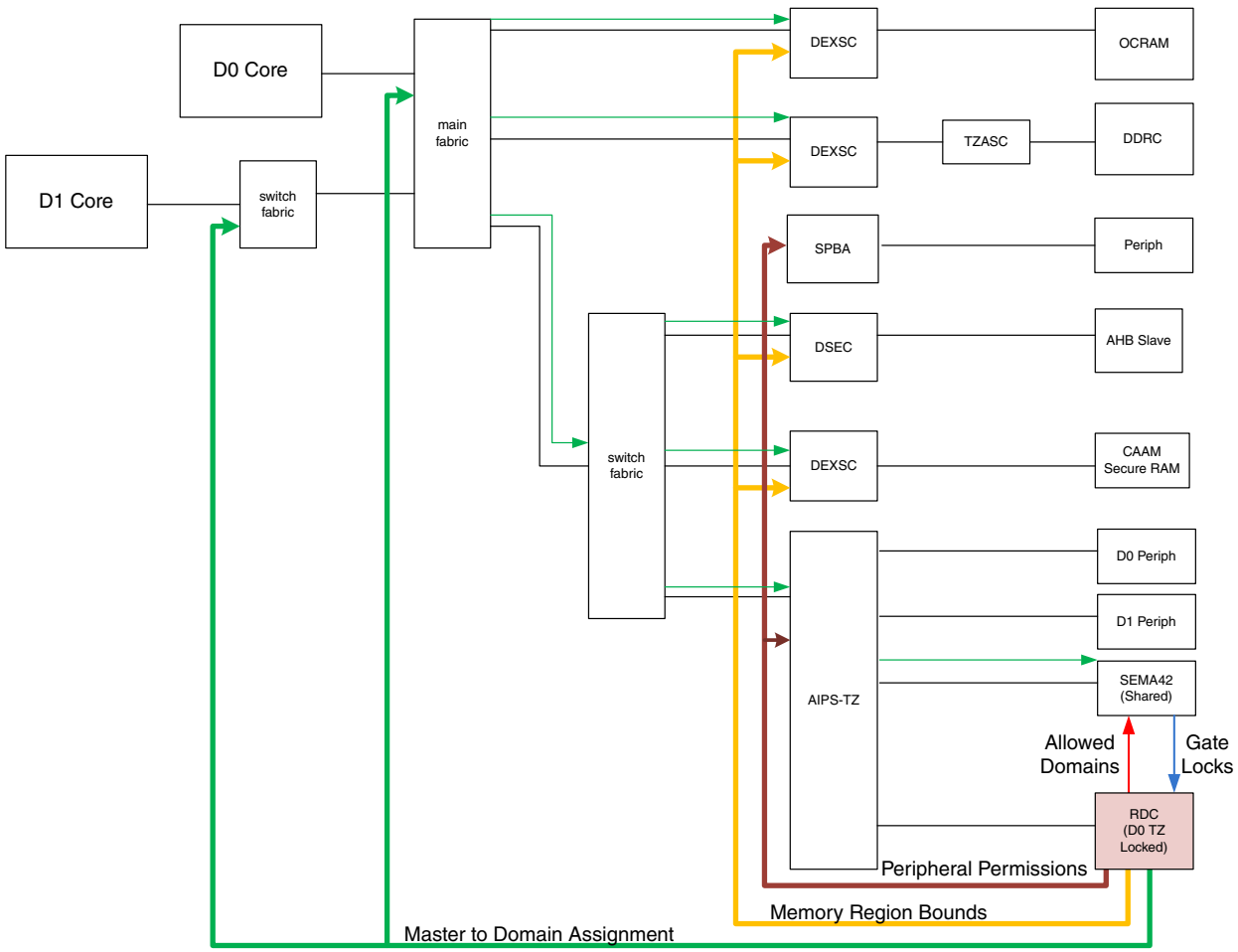


Figure 3-2. Example RDC Connections

For shared peripherals, RDC permits more than one domain access to a single peripheral. RDC also provides three ways to control synchronized use of shared peripherals. These methods include hardware-enforced synchronization, software-based semaphores, or no synchronization. The latter may be suitable for well-tuned multi-core operating systems that handle synchronization in the core platform, for instance.

For hardware-enforced synchronization, also known as "safe sharing", ownership of the peripheral must be claimed in the semaphore controller before access is allowed to the shared peripheral. The "semaphore required" bit (SREQ) is set in the PDAP register corresponding to the shared peripheral which causes the RDC to require that a semaphore is obtained by a domain before access by that domain to the shared peripheral is allowed. During the time that the domain has the semaphore in possession its bus masters have exclusive access to the peripheral.

When the semaphore is released then no domain masters have access until the semaphore is obtained again. When the SREQ is set, RDC does not allow masters to obtain semaphores of peripherals to which it is not allocated; the master must have designated access in the D-registers of the corresponding PDAP register (e.g. D3R bit set for Domain 3 access of the shared peripheral). There is a one-to-one mapping between the semaphore controller gate and the resource domain controller peripheral. The mapping of PDAP registers and peripherals can be found in the Peripheral Map section of the RDC chapter.

3.2.2.1 Domain ID

The RDC provides for an isolation of domain resources by use of a identifier called the Domain ID (DID). A core and its resources including memory, bus masters, and peripherals are all associated with a single DID. When software or a DMA attempts to access a peripheral or memory, the corresponding bus transaction includes the DID along with the other bus control information such as Read, Write, and privilege mode.

3.2.2.2 Resource Assignment

The RDC allows assignment of peripherals and memories to one or more domains while each bus master or core is placed in one of four domains. The masters are assigned a domain in the MDA register. A peripheral is given R/W access permissions to each domain in the PDAP register. Memory regions are bound by address space in start and end registers, the MRSA and MREA. Each memory region is assigned one or more allowed domains and R/W permissions in the MRC control register. Memory regions must be enabled before the permissions are active. Otherwise the permissions are not restricted.

The RDC itself should be isolated to ensure that only a trustworthy resource manager can configure the RDC registers. This process may either be present initially, during secure boot, or during the runtime in the secure world, for example. If the operating system does not support a runtime trusted execution then during the secure boot process the RDC configuration can be locked to prevent further modification after the operating systems are running.

NOTE

The CCM supports multicore awareness based on resource domain assignments programmed into the RDC. Refer to the CCM chapter regarding the relationship between core resource domains and their respective CCM resources. Failing to follow the proper sequence when updating the resource domain

assignments of the core can result in clocks being inadvertently gated.

3.2.2.3 Safe Sharing

For shared peripherals, the RDC can be configured to require a domain to obtain a semaphore lock before access to the peripheral is allowed. This feature helps prevent collisions from processes on separate cores that may want to use the same peripheral at the same time. The RDC sends a list of eligible domains to the semaphore module for each gate/peripheral. The eligible domains are those that are set in the peripheral domain access permissions (PDAP) registers. There is a one-to-one correspondence between semaphore gates and peripherals so each gate in the semaphore block represents a peripheral. The RDC receives semaphore locks from the hardware semaphore module (SEMA42). A semaphore lock is acquired when a core or bus master from a given domain requests a lock for a particular gate. The semaphore module compares the request's domain ID against the list of eligible domain IDs. If the domain ID is on the list and the lock is available then the lock is set and a signal is sent back to the RDC module indicating a lock has been acquired for a particular gate and to which domain ID the lock belongs. The RDC then restricts access to the corresponding peripheral to only transactions originating from the domain that has the lock. Another domain, though on the shared list to access the peripheral, must then wait until the lock is released before acquiring the lock and gaining access to the peripheral. To enable this feature of hardware enforcement for the semaphore locks, the SREQ bit is set in the RDC resource register.

If the SREQ is set, then when a process determines it needs a shared peripheral, it must first lock the resource in the semaphore module. Once the resource is locked, the semaphore module sends a signal to the RDC indicating the domain has access to the resource. The RDC will then set the access permissions to allow that domain access to the peripheral.

For a domain to acquire a lock on a peripheral, the domain must have been assigned to the peripheral in the RDC Peripheral Domain Access Permissions register (PDAP). The semaphore module only allows safe-sharing locks for those domains that are assigned to the peripheral. The semaphore module does not consider the access type (Read or Write) when allowing domains to acquire locks.

The SEMA42 module implements hardware-enforced semaphores as an IPS-mapped slave peripheral device. The feature set includes:

- Module definition supporting 64 hardware-enforced gates in a multi-processor configuration, where up to 15 processors can be supported; cp*X* is meant to represent core processor *X*
  - Gates appear as an *n*-entry byte-size array with read and write accesses (*n* = 16, 32, 64).
    - Processors lock gates by writing "Master_index" to the appropriate gate and must read back the gate value to verify the lock operation was successful. The Master_index value for the processors can be found in the Master Index Allocation table, which can be found in the AIPSTZ block.. Also note that after locking, the gate register contains the master_id value of the locking processor (in bits [3:0]), and also the value of the locking domain (in bits [5:4]).
    - Once locked, the gate is unlocked by a write of zeroes from the locking processor.
    - The number of implemented gates is specified by a hardware configuration define.
  - Each hardware gate appears as a 16-state, 4-bit state machine.
    - 16-state implementation

      if gate = 0x0, then state = unlocked
      if gate = 0x1, then state = locked by processor (master_index) 0
      if gate = 0x2, then state = locked by processor (master_index) 1
      …
      if gate = 0xF, then state = locked by processor (master_index) 14
    - Uses the logical bus master number (master_index) as a reference attribute plus the specified data patterns to validate all write operations.
    - Once locked, the gate can (and must) be unlocked by a write of zeroes from the locking processor.
  - Secure reset mechanisms are supported to clear the contents of individual gates, as well as a clear_all capability.
- Memory-mapped IPS slave peripheral platform module
  - Interface to the IPS bus for programming-model accesses

3.2.2.4 Resource Domain Control and Security Considerations

Conceptually, the RDC configuration is independent of the processor privilege mode and security domain. It is intended to allow for isolation between core processing environments to prevent collisions and increase reliability. Access between resource domains is mutually exclusive and each domain should be in control of its own privilege modes and access rights.

However, it is important to realize multi-core processors may have a multiple resource domains but only one overarching security domain. Chip security controls reside in one resource domain. In this configuration, a domain can affect at least one level of access privileges in the other domain. This may be acceptable but clarity and care is needed to ensure expected functionality.

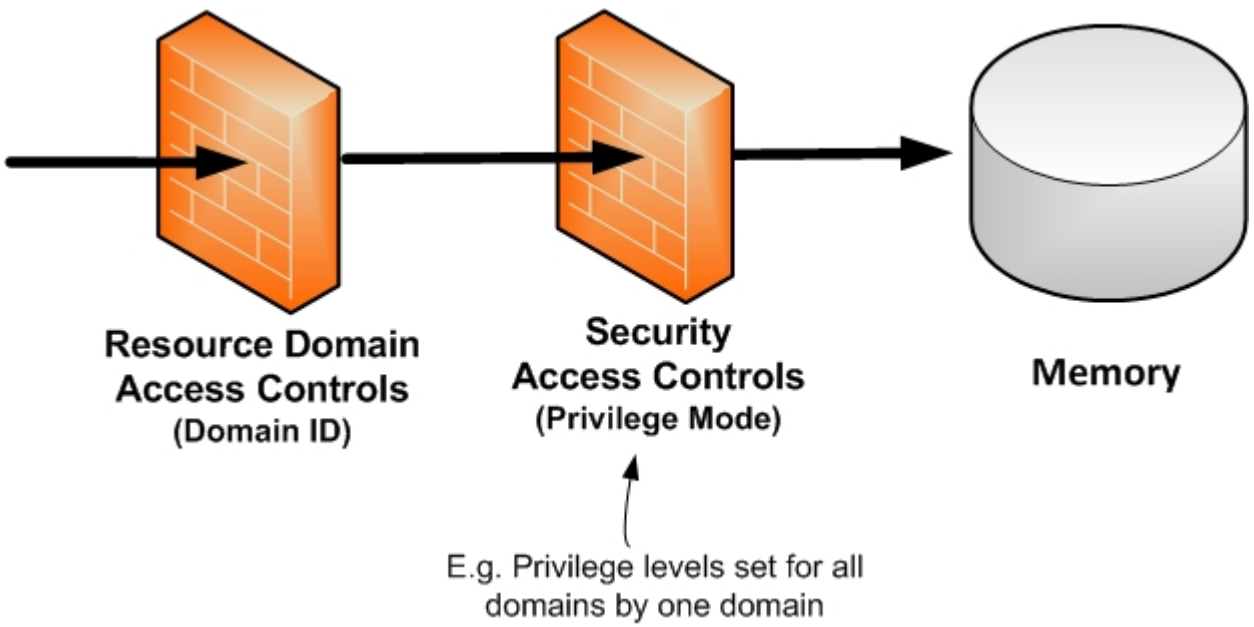


Figure 3-3. Access Control to Memory

Therefore, access to the security controls should be restricted to the most trustworthy operating mode of the core and privilege levels should be coordinated to ensure that shared peripherals and memory regions are accessible by both cores. For instance, if a memory region is designated for secure accesses then all domain masters that share that region must have secure privileges.

3.2.3 Modes of Operation

The RDC provides access controls to the resource domain subsystem. When the device is in a low power mode then some memory regions in the subsystem may be powered off. RDC responds to the impacted memory regions by automatically reconfiguring the memory regions once power returns and blocking access to those memory regions until the reconfiguration process is complete.

3.2.3.1 Low Power Modes

The RDC loads configuration information for memory regions (MRSA, MRSE, MRC) into access control mechanisms (gaskets) at the memory interface. The location of this configuration information may reside inside power domains that lose power during sleep modes for energy savings. To restore configuration information upon return from sleep mode, the RDC receives a global power control signal indicating power is restored. The RDC then automatically reconfigures the memory regions with the configuration information.

During reconfiguration, access is blocked to the previously powered down memories. When the RDC completes reconfiguration it issues an interrupt and allows access to the memory regions. Only the powered down regions are blocked during the reconfiguration. Memory regions in the "always-on" power domain (still powered during sleep mode) remain available according to the programmed access rights. If no memory regions were enabled then the powered down regions are available immediately when power is restored.

The figure below shows the Global Power Control signal which RDC uses to invalidate the configuration upon deassertion and to restore the configuration when re-asserted. The configuration is valid and bus transactions allowed once the memory regions have been restored.

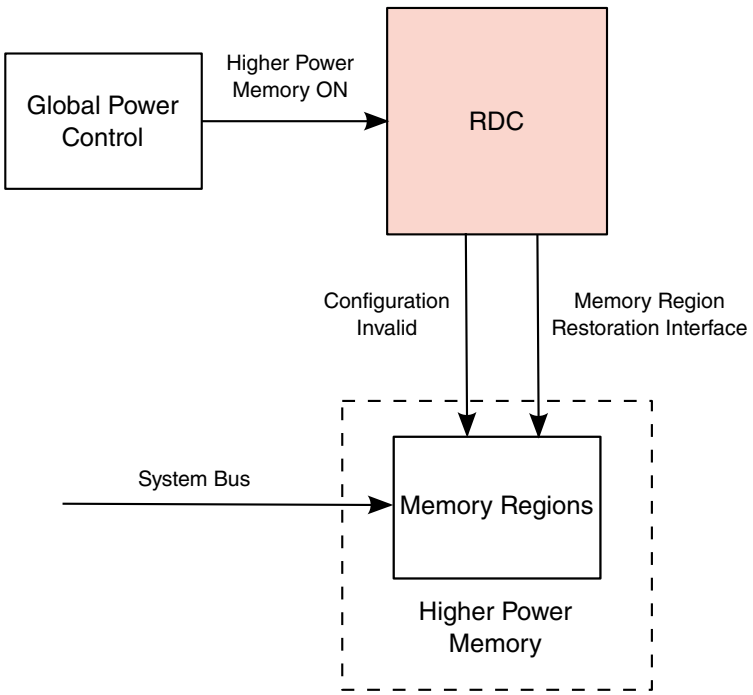


Figure 3-4. Memory Restoration Signaling

3.2.4 Programming Interface

This section provides product specific details describing the mapping of resources - peripherals, bus masters, and memory regions - to corresponding resource domain controls RDC registers.

The RDC and RDC_SEMA42 register maps are combined in this chapter. The base address for the one RDC map and two SEMA42 maps are each separated by 4KB. While there are two SEMA42 submodules and therefore two sets of SEMA42 registers, this chapter describes one. Please refer to the peripheral memory map for the base addresses of the RDC and SEMA42 modules.

3.2.4.1 Master Assignment Registers

3.2.4.2 Peripheral Mapping

Each peripheral has a corresponding resource domain assignment register in the RDC and semaphore lock register in the RDC_SEMA42 module. The following table shows allocation of the RDC PDAP and RDC_SEMA4 GATE registers for peripheral resource domain assignment.

NOTE

Access control of the RDC registers can be programmed using the respective PDAP register. The default setting of the PDAP register for the RDC allows access from all domains. Use caution when restricting access of the RDC registers to avoid conditions where access to the RDC registers is needed but no master is assigned to a domain with access rights to the RDC.

Table 3-2. RDC Peripheral Mapping

Peripheral	RDC PDAP register	RDC_SEMA42 block/gate register
GPIO1	RDC_PDAP00	SEMA42 B1 / G0
GPIO2	RDC_PDAP01	SEMA42 B1 / G1
GPIO3	RDC_PDAP02	SEMA42 B1 / G2
GPIO4	RDC_PDAP03	SEMA42 B1 / G3
GPIO5	RDC_PDAP04	SEMA42 B1 / G4
Reserved	RDC_PDAP05	SEMA42 B1 / G5
ANA_TSENSOR	RDC_PDAP06	SEMA42 B1 / G6
ANA_OSC	RDC_PDAP07	SEMA42 B1 / G7
WDOG1	RDC_PDAP08	SEMA42 B1 / G8
WDOG2	RDC_PDAP09	SEMA42 B1 / G9
WDOG3	RDC_PDAP10	SEMA42 B1 / G10
SDMA3	RDC_PDAP11	SEMA42 B1 / G11
SDMA2	RDC_PDAP12	SEMA42 B1 / G12
GPT1	RDC_PDAP13	SEMA42 B1 / G13
GPT2	RDC_PDAP14	SEMA42 B1 / G14
GPT3	RDC_PDAP15	SEMA42 B1 / G15
Reserved	RDC_PDAP16	SEMA42 B1 / G16
ROMCP	RDC_PDAP17	SEMA42 B1 / G17
Reserved	RDC_PDAP18	SEMA42 B1 / G18
IOMUXC	RDC_PDAP19	SEMA42 B1 / G19
IOMUXC_GPR	RDC_PDAP20	SEMA42 B1 / G20
OCOTP_CTRL	RDC_PDAP21	SEMA42 B1 / G21
ANA_PLL	RDC_PDAP22	SEMA42 B1 / G22
SNVS_HP	RDC_PDAP23	SEMA42 B1 / G23

Table continues on the next page...

Table 3-2. RDC Peripheral Mapping (continued)

Peripheral	RDC PDAP register	RDC_SEMA42 block/gate register
CCM	RDC_PDAP24	SEMA42 B1 / G24
SRC	RDC_PDAP25	SEMA42 B1 / G25
GPC	RDC_PDAP26	SEMA42 B1 / G26
SEMAPHORE1	RDC_PDAP27	SEMA42 B1 / G27
SEMAPHORE2	RDC_PDAP28	SEMA42 B1 / G28
RDC	RDC_PDAP29	SEMA42 B1 / G29
CSU	RDC_PDAP30	SEMA42 B1 / G30
Reserved	RDC_PDAP31	SEMA42 B1 / G31
LCDIF	RDC_PDAP32	SEMA42 B1 / G32
MIPI_DSI	RDC_PDAP33	SEMA42 B1 / G33
CSI	RDC_PDAP34	SEMA42 B1 / G34
MIPI_CSI	RDC_PDAP35	SEMA42 B1 / G35
USB1	RDC_PDAP36	SEMA42 B1 / G36
Reserved	RDC_PDAP37	SEMA42 B1 / G37
PWM1	RDC_PDAP38	SEMA42 B1 / G38
PWM2	RDC_PDAP39	SEMA42 B1 / G39
PWM3	RDC_PDAP40	SEMA42 B1 / G40
PWM4	RDC_PDAP41	SEMA42 B1 / G41
System_Counter_RD	RDC_PDAP42	SEMA42 B1 / G42
System_Counter_CMP	RDC_PDAP43	SEMA42 B1 / G43
System_Counter_CTRL	RDC_PDAP44	SEMA42 B1 / G44
Reserved	RDC_PDAP45	SEMA42 B1 / G45
GPT6	RDC_PDAP46	SEMA42 B1 / G46
GPT5	RDC_PDAP47	SEMA42 B1 / G47
GPT4	RDC_PDAP48	SEMA42 B1 / G48
Reserved	RDC_PDAP49	SEMA42 B1 / G49
Reserved	RDC_PDAP50	SEMA42 B1 / G50
Reserved	RDC_PDAP51	SEMA42 B1 / G51
Reserved	RDC_PDAP52	SEMA42 B1 / G52
Reserved	RDC_PDAP53	SEMA42 B1 / G53
Reserved	RDC_PDAP54	SEMA42 B1 / G54
Reserved	RDC_PDAP55	SEMA42 B1 / G55
TZASC	RDC_PDAP56	SEMA42 B1 / G56
Reserved	RDC_PDAP57	SEMA42 B1 / G57
Reserved	RDC_PDAP58	SEMA42 B1 / G58
USB2	RDC_PDAP59	SEMA42 B1 / G59
PERFMON1	RDC_PDAP60	SEMA42 B1 / G60
PERFMON2	RDC_PDAP61	SEMA42 B1 / G61
PLATFORM_CTRL	RDC_PDAP62	SEMA42 B1 / G62

Table continues on the next page...

Table 3-2. RDC Peripheral Mapping (continued)

Peripheral	RDC PDAP register	RDC_SEMA42 block/gate register
QoSC	RDC_PDAP63	SEMA42 B1 / G63
Reserved	RDC_PDAP64	SEMA42 B2 / G0
Reserved	RDC_PDAP65	SEMA42 B2 / G1
I2C1	RDC_PDAP66	SEMA42 B2 / G2
I2C2	RDC_PDAP67	SEMA42 B2 / G3
I2C3	RDC_PDAP68	SEMA42 B2 / G4
I2C4	RDC_PDAP69	SEMA42 B2 / G5
UART4	RDC_PDAP70	SEMA42 B2 / G6
Reserved	RDC_PDAP71	SEMA42 B2 / G7
Reserved	RDC_PDAP72	SEMA42 B2 / G8
Reserved	RDC_PDAP73	SEMA42 B2 / G9
MU_A	RDC_PDAP74	SEMA42 B2 / G10
MU_B	RDC_PDAP75	SEMA42 B2 / G11
SEMAPHORE_HS	RDC_PDAP76	SEMA42 B2 / G12
Reserved for SDMA2 internal memory	RDC_PDAP77	SEMA42 B2 / G13
SAI1	RDC_PDAP78	SEMA42 B2 / G14
SAI2	RDC_PDAP79	SEMA42 B2 / G15
SAI3	RDC_PDAP80	SEMA42 B2 / G16
Reserved	RDC_PDAP81	SEMA42 B2 / G17
SAI5	RDC_PDAP82	SEMA42 B2 / G18
SAI6	RDC_PDAP83	SEMA42 B2 / G19
uSDHC1	RDC_PDAP84	SEMA42 B2 / G20
uSDHC2	RDC_PDAP85	SEMA42 B2 / G21
uSDHC3	RDC_PDAP86	SEMA42 B2 / G22
Reserved	RDC_PDAP87	SEMA42 B2 / G23
PCIE_PHY1	RDC_PDAP88	SEMA42 B2 / G24
Reserved for SDMA2 internal memory	RDC_PDAP89	SEMA42 B2 / G25
SPBA2	RDC_PDAP90	SEMA42 B2 / G26
QSPI	RDC_PDAP91	SEMA42 B2 / G27
Reserved	RDC_PDAP92	SEMA42 B2 / G28
SDMA1	RDC_PDAP93	SEMA42 B2 / G29
ENET1	RDC_PDAP94	SEMA42 B2 / G30
Reserved	RDC_PDAP95	SEMA42 B2 / G31
Reserved for SDMA internal memory	RDC_PDAP96	SEMA42 B2 / G32
SPDIF1	RDC_PDAP97	SEMA42 B2 / G33
eCSPI1	RDC_PDAP98	SEMA42 B2 / G34
eCSPI2	RDC_PDAP99	SEMA42 B2 / G35
eCSPI3	RDC_PDAP100	SEMA42 B2 / G36
MICFIL	RDC_PDAP101	SEMA42 B2 / G37

Table continues on the next page...

Table 3-2. RDC Peripheral Mapping (continued)

Peripheral	RDC PDAP register	RDC_SEMA42 block/gate register
UART1	RDC_PDAP102	SEMA42 B2 / G38
Reserved for SDMA internal registers	RDC_PDAP103	SEMA42 B2 / G39
UART3	RDC_PDAP104	SEMA42 B2 / G40
UART2	RDC_PDAP105	SEMA42 B2 / G41
SPDIF2	RDC_PDAP106	SEMA42 B2 / G42
Reserved	RDC_PDAP108	SEMA42 B2 / G44
Reserved	RDC_PDAP109	SEMA42 B2 / G45
Reserved for SDMA internal registers	RDC_PDAP110	SEMA42 B2 / G46
SPBA1	RDC_PDAP111	SEMA42 B2 / G47
module_en_glbl[0]	RDC_PDAP112	SEMA42 B2 / G48
module_en_glbl[0]	RDC_PDAP113	SEMA42 B2 / G49
CAAM	RDC_PDAP114	SEMA42 B2 / G50

3.2.4.3 Memory Region Map

The number of memories with domain isolation support varies per device. The number of memory regions for a particular memory and the size of those regions varies per memory gasket. Each region of memory has a set of registers to define the boundaries of the region based on start and end addresses, a control register to set the domain access permissions and enable the region, and a status register to determin if access was denied to a region.

For this device, refer to the table below to determine the memories with domain support, the number of regions for each memory, the region resolution, the identifying numbers for the sets of memory region registers, and the addresses of the RDC registers to access the sets of Memory Region registers.

3.2.5 RDC Memory Map/Register Definition

RDC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303D_0000	Version Information (RDC_VIR)	32	R	0376_E204h	3.2.5.1/59
303D_0024	Status (RDC_STAT)	32	R/W	0000_0100h	3.2.5.2/60
303D_0028	Interrupt and Control (RDC_INTCTRL)	32	R/W	0000_0000h	3.2.5.3/61

Table continues on the next page...

RDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303D_002C	Interrupt Status (RDC_INTSTAT)	32	R/W	See section	3.2.5.4/61
303D_0200	Master Domain Assignment (RDC_MDA0)	32	R/W	0000_0000h	3.2.5.5/62
303D_0204	Master Domain Assignment (RDC_MDA1)	32	R/W	0000_0000h	3.2.5.5/62
303D_0208	Master Domain Assignment (RDC_MDA2)	32	R/W	0000_0000h	3.2.5.5/62
303D_020C	Master Domain Assignment (RDC_MDA3)	32	R/W	0000_0000h	3.2.5.5/62
303D_0210	Master Domain Assignment (RDC_MDA4)	32	R/W	0000_0000h	3.2.5.5/62
303D_0214	Master Domain Assignment (RDC_MDA5)	32	R/W	0000_0000h	3.2.5.5/62
303D_0218	Master Domain Assignment (RDC_MDA6)	32	R/W	0000_0000h	3.2.5.5/62
303D_021C	Master Domain Assignment (RDC_MDA7)	32	R/W	0000_0000h	3.2.5.5/62
303D_0220	Master Domain Assignment (RDC_MDA8)	32	R/W	0000_0000h	3.2.5.5/62
303D_0224	Master Domain Assignment (RDC_MDA9)	32	R/W	0000_0000h	3.2.5.5/62
303D_0228	Master Domain Assignment (RDC_MDA10)	32	R/W	0000_0000h	3.2.5.5/62
303D_022C	Master Domain Assignment (RDC_MDA11)	32	R/W	0000_0000h	3.2.5.5/62
303D_0230	Master Domain Assignment (RDC_MDA12)	32	R/W	0000_0000h	3.2.5.5/62
303D_0234	Master Domain Assignment (RDC_MDA13)	32	R/W	0000_0000h	3.2.5.5/62
303D_0238	Master Domain Assignment (RDC_MDA14)	32	R/W	0000_0000h	3.2.5.5/62
303D_023C	Master Domain Assignment (RDC_MDA15)	32	R/W	0000_0000h	3.2.5.5/62
303D_0240	Master Domain Assignment (RDC_MDA16)	32	R/W	0000_0000h	3.2.5.5/62
303D_0244	Master Domain Assignment (RDC_MDA17)	32	R/W	0000_0000h	3.2.5.5/62
303D_0248	Master Domain Assignment (RDC_MDA18)	32	R/W	0000_0000h	3.2.5.5/62
303D_024C	Master Domain Assignment (RDC_MDA19)	32	R/W	0000_0000h	3.2.5.5/62
303D_0250	Master Domain Assignment (RDC_MDA20)	32	R/W	0000_0000h	3.2.5.5/62
303D_0254	Master Domain Assignment (RDC_MDA21)	32	R/W	0000_0000h	3.2.5.5/62
303D_0258	Master Domain Assignment (RDC_MDA22)	32	R/W	0000_0000h	3.2.5.5/62
303D_025C	Master Domain Assignment (RDC_MDA23)	32	R/W	0000_0000h	3.2.5.5/62
303D_0260	Master Domain Assignment (RDC_MDA24)	32	R/W	0000_0000h	3.2.5.5/62
303D_0264	Master Domain Assignment (RDC_MDA25)	32	R/W	0000_0000h	3.2.5.5/62
303D_0268	Master Domain Assignment (RDC_MDA26)	32	R/W	0000_0000h	3.2.5.5/62
303D_0400	Peripheral Domain Access Permissions (RDC_PDAP0)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0404	Peripheral Domain Access Permissions (RDC_PDAP1)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0408	Peripheral Domain Access Permissions (RDC_PDAP2)	32	R/W	0000_00FFh	3.2.5.6/63
303D_040C	Peripheral Domain Access Permissions (RDC_PDAP3)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0410	Peripheral Domain Access Permissions (RDC_PDAP4)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0414	Peripheral Domain Access Permissions (RDC_PDAP5)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0418	Peripheral Domain Access Permissions (RDC_PDAP6)	32	R/W	0000_00FFh	3.2.5.6/63
303D_041C	Peripheral Domain Access Permissions (RDC_PDAP7)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0420	Peripheral Domain Access Permissions (RDC_PDAP8)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0424	Peripheral Domain Access Permissions (RDC_PDAP9)	32	R/W	0000_00FFh	3.2.5.6/63

Table continues on the next page...

RDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303D_0428	Peripheral Domain Access Permissions (RDC_PDAP10)	32	R/W	0000_00FFh	3.2.5.6/63
303D_042C	Peripheral Domain Access Permissions (RDC_PDAP11)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0430	Peripheral Domain Access Permissions (RDC_PDAP12)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0434	Peripheral Domain Access Permissions (RDC_PDAP13)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0438	Peripheral Domain Access Permissions (RDC_PDAP14)	32	R/W	0000_00FFh	3.2.5.6/63
303D_043C	Peripheral Domain Access Permissions (RDC_PDAP15)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0440	Peripheral Domain Access Permissions (RDC_PDAP16)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0444	Peripheral Domain Access Permissions (RDC_PDAP17)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0448	Peripheral Domain Access Permissions (RDC_PDAP18)	32	R/W	0000_00FFh	3.2.5.6/63
303D_044C	Peripheral Domain Access Permissions (RDC_PDAP19)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0450	Peripheral Domain Access Permissions (RDC_PDAP20)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0454	Peripheral Domain Access Permissions (RDC_PDAP21)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0458	Peripheral Domain Access Permissions (RDC_PDAP22)	32	R/W	0000_00FFh	3.2.5.6/63
303D_045C	Peripheral Domain Access Permissions (RDC_PDAP23)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0460	Peripheral Domain Access Permissions (RDC_PDAP24)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0464	Peripheral Domain Access Permissions (RDC_PDAP25)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0468	Peripheral Domain Access Permissions (RDC_PDAP26)	32	R/W	0000_00FFh	3.2.5.6/63
303D_046C	Peripheral Domain Access Permissions (RDC_PDAP27)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0470	Peripheral Domain Access Permissions (RDC_PDAP28)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0474	Peripheral Domain Access Permissions (RDC_PDAP29)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0478	Peripheral Domain Access Permissions (RDC_PDAP30)	32	R/W	0000_00FFh	3.2.5.6/63
303D_047C	Peripheral Domain Access Permissions (RDC_PDAP31)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0480	Peripheral Domain Access Permissions (RDC_PDAP32)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0484	Peripheral Domain Access Permissions (RDC_PDAP33)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0488	Peripheral Domain Access Permissions (RDC_PDAP34)	32	R/W	0000_00FFh	3.2.5.6/63
303D_048C	Peripheral Domain Access Permissions (RDC_PDAP35)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0490	Peripheral Domain Access Permissions (RDC_PDAP36)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0494	Peripheral Domain Access Permissions (RDC_PDAP37)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0498	Peripheral Domain Access Permissions (RDC_PDAP38)	32	R/W	0000_00FFh	3.2.5.6/63
303D_049C	Peripheral Domain Access Permissions (RDC_PDAP39)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04A0	Peripheral Domain Access Permissions (RDC_PDAP40)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04A4	Peripheral Domain Access Permissions (RDC_PDAP41)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04A8	Peripheral Domain Access Permissions (RDC_PDAP42)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04AC	Peripheral Domain Access Permissions (RDC_PDAP43)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04B0	Peripheral Domain Access Permissions (RDC_PDAP44)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04B4	Peripheral Domain Access Permissions (RDC_PDAP45)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04B8	Peripheral Domain Access Permissions (RDC_PDAP46)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04BC	Peripheral Domain Access Permissions (RDC_PDAP47)	32	R/W	0000_00FFh	3.2.5.6/63

Table continues on the next page...

RDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303D_04C0	Peripheral Domain Access Permissions (RDC_PDAP48)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04C4	Peripheral Domain Access Permissions (RDC_PDAP49)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04C8	Peripheral Domain Access Permissions (RDC_PDAP50)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04CC	Peripheral Domain Access Permissions (RDC_PDAP51)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04D0	Peripheral Domain Access Permissions (RDC_PDAP52)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04D4	Peripheral Domain Access Permissions (RDC_PDAP53)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04D8	Peripheral Domain Access Permissions (RDC_PDAP54)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04DC	Peripheral Domain Access Permissions (RDC_PDAP55)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04E0	Peripheral Domain Access Permissions (RDC_PDAP56)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04E4	Peripheral Domain Access Permissions (RDC_PDAP57)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04E8	Peripheral Domain Access Permissions (RDC_PDAP58)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04EC	Peripheral Domain Access Permissions (RDC_PDAP59)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04F0	Peripheral Domain Access Permissions (RDC_PDAP60)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04F4	Peripheral Domain Access Permissions (RDC_PDAP61)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04F8	Peripheral Domain Access Permissions (RDC_PDAP62)	32	R/W	0000_00FFh	3.2.5.6/63
303D_04FC	Peripheral Domain Access Permissions (RDC_PDAP63)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0500	Peripheral Domain Access Permissions (RDC_PDAP64)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0504	Peripheral Domain Access Permissions (RDC_PDAP65)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0508	Peripheral Domain Access Permissions (RDC_PDAP66)	32	R/W	0000_00FFh	3.2.5.6/63
303D_050C	Peripheral Domain Access Permissions (RDC_PDAP67)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0510	Peripheral Domain Access Permissions (RDC_PDAP68)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0514	Peripheral Domain Access Permissions (RDC_PDAP69)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0518	Peripheral Domain Access Permissions (RDC_PDAP70)	32	R/W	0000_00FFh	3.2.5.6/63
303D_051C	Peripheral Domain Access Permissions (RDC_PDAP71)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0520	Peripheral Domain Access Permissions (RDC_PDAP72)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0524	Peripheral Domain Access Permissions (RDC_PDAP73)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0528	Peripheral Domain Access Permissions (RDC_PDAP74)	32	R/W	0000_00FFh	3.2.5.6/63
303D_052C	Peripheral Domain Access Permissions (RDC_PDAP75)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0530	Peripheral Domain Access Permissions (RDC_PDAP76)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0534	Peripheral Domain Access Permissions (RDC_PDAP77)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0538	Peripheral Domain Access Permissions (RDC_PDAP78)	32	R/W	0000_00FFh	3.2.5.6/63
303D_053C	Peripheral Domain Access Permissions (RDC_PDAP79)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0540	Peripheral Domain Access Permissions (RDC_PDAP80)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0544	Peripheral Domain Access Permissions (RDC_PDAP81)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0548	Peripheral Domain Access Permissions (RDC_PDAP82)	32	R/W	0000_00FFh	3.2.5.6/63
303D_054C	Peripheral Domain Access Permissions (RDC_PDAP83)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0550	Peripheral Domain Access Permissions (RDC_PDAP84)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0554	Peripheral Domain Access Permissions (RDC_PDAP85)	32	R/W	0000_00FFh	3.2.5.6/63

Table continues on the next page...

RDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303D_0558	Peripheral Domain Access Permissions (RDC_PDAP86)	32	R/W	0000_00FFh	3.2.5.6/63
303D_055C	Peripheral Domain Access Permissions (RDC_PDAP87)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0560	Peripheral Domain Access Permissions (RDC_PDAP88)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0564	Peripheral Domain Access Permissions (RDC_PDAP89)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0568	Peripheral Domain Access Permissions (RDC_PDAP90)	32	R/W	0000_00FFh	3.2.5.6/63
303D_056C	Peripheral Domain Access Permissions (RDC_PDAP91)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0570	Peripheral Domain Access Permissions (RDC_PDAP92)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0574	Peripheral Domain Access Permissions (RDC_PDAP93)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0578	Peripheral Domain Access Permissions (RDC_PDAP94)	32	R/W	0000_00FFh	3.2.5.6/63
303D_057C	Peripheral Domain Access Permissions (RDC_PDAP95)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0580	Peripheral Domain Access Permissions (RDC_PDAP96)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0584	Peripheral Domain Access Permissions (RDC_PDAP97)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0588	Peripheral Domain Access Permissions (RDC_PDAP98)	32	R/W	0000_00FFh	3.2.5.6/63
303D_058C	Peripheral Domain Access Permissions (RDC_PDAP99)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0590	Peripheral Domain Access Permissions (RDC_PDAP100)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0594	Peripheral Domain Access Permissions (RDC_PDAP101)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0598	Peripheral Domain Access Permissions (RDC_PDAP102)	32	R/W	0000_00FFh	3.2.5.6/63
303D_059C	Peripheral Domain Access Permissions (RDC_PDAP103)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05A0	Peripheral Domain Access Permissions (RDC_PDAP104)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05A4	Peripheral Domain Access Permissions (RDC_PDAP105)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05A8	Peripheral Domain Access Permissions (RDC_PDAP106)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05AC	Peripheral Domain Access Permissions (RDC_PDAP107)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05B0	Peripheral Domain Access Permissions (RDC_PDAP108)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05B4	Peripheral Domain Access Permissions (RDC_PDAP109)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05B8	Peripheral Domain Access Permissions (RDC_PDAP110)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05BC	Peripheral Domain Access Permissions (RDC_PDAP111)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05C0	Peripheral Domain Access Permissions (RDC_PDAP112)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05C4	Peripheral Domain Access Permissions (RDC_PDAP113)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05C8	Peripheral Domain Access Permissions (RDC_PDAP114)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05CC	Peripheral Domain Access Permissions (RDC_PDAP115)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05D0	Peripheral Domain Access Permissions (RDC_PDAP116)	32	R/W	0000_00FFh	3.2.5.6/63
303D_05D4	Peripheral Domain Access Permissions (RDC_PDAP117)	32	R/W	0000_00FFh	3.2.5.6/63
303D_0800	Memory Region Start Address (RDC_MRSA0)	32	R/W	Undefined	3.2.5.7/64
303D_0804	Memory Region End Address (RDC_MREA0)	32	R/W	Undefined	3.2.5.8/66
303D_0808	Memory Region Control (RDC_MRC0)	32	R/W	0000_00FFh	3.2.5.9/67
303D_080C	Memory Region Violation Status (RDC_MRVS0)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0810	Memory Region Start Address (RDC_MRSA1)	32	R/W	Undefined	3.2.5.7/64

Table continues on the next page...

RDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303D_0814	Memory Region End Address (RDC_MREA1)	32	R/W	Undefined	3.2.5.8/66
303D_0818	Memory Region Control (RDC_MRC1)	32	R/W	0000_00FFh	3.2.5.9/67
303D_081C	Memory Region Violation Status (RDC_MRVS1)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0820	Memory Region Start Address (RDC_MRSA2)	32	R/W	Undefined	3.2.5.7/64
303D_0824	Memory Region End Address (RDC_MREA2)	32	R/W	Undefined	3.2.5.8/66
303D_0828	Memory Region Control (RDC_MRC2)	32	R/W	0000_00FFh	3.2.5.9/67
303D_082C	Memory Region Violation Status (RDC_MRVS2)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0830	Memory Region Start Address (RDC_MRSA3)	32	R/W	Undefined	3.2.5.7/64
303D_0834	Memory Region End Address (RDC_MREA3)	32	R/W	Undefined	3.2.5.8/66
303D_0838	Memory Region Control (RDC_MRC3)	32	R/W	0000_00FFh	3.2.5.9/67
303D_083C	Memory Region Violation Status (RDC_MRVS3)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0840	Memory Region Start Address (RDC_MRSA4)	32	R/W	Undefined	3.2.5.7/64
303D_0844	Memory Region End Address (RDC_MREA4)	32	R/W	Undefined	3.2.5.8/66
303D_0848	Memory Region Control (RDC_MRC4)	32	R/W	0000_00FFh	3.2.5.9/67
303D_084C	Memory Region Violation Status (RDC_MRVS4)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0850	Memory Region Start Address (RDC_MRSA5)	32	R/W	Undefined	3.2.5.7/64
303D_0854	Memory Region End Address (RDC_MREA5)	32	R/W	Undefined	3.2.5.8/66
303D_0858	Memory Region Control (RDC_MRC5)	32	R/W	0000_00FFh	3.2.5.9/67
303D_085C	Memory Region Violation Status (RDC_MRVS5)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0860	Memory Region Start Address (RDC_MRSA6)	32	R/W	Undefined	3.2.5.7/64
303D_0864	Memory Region End Address (RDC_MREA6)	32	R/W	Undefined	3.2.5.8/66
303D_0868	Memory Region Control (RDC_MRC6)	32	R/W	0000_00FFh	3.2.5.9/67
303D_086C	Memory Region Violation Status (RDC_MRVS6)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0870	Memory Region Start Address (RDC_MRSA7)	32	R/W	Undefined	3.2.5.7/64
303D_0874	Memory Region End Address (RDC_MREA7)	32	R/W	Undefined	3.2.5.8/66
303D_0878	Memory Region Control (RDC_MRC7)	32	R/W	0000_00FFh	3.2.5.9/67
303D_087C	Memory Region Violation Status (RDC_MRVS7)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0880	Memory Region Start Address (RDC_MRSA8)	32	R/W	Undefined	3.2.5.7/64
303D_0884	Memory Region End Address (RDC_MREA8)	32	R/W	Undefined	3.2.5.8/66
303D_0888	Memory Region Control (RDC_MRC8)	32	R/W	0000_00FFh	3.2.5.9/67
303D_088C	Memory Region Violation Status (RDC_MRVS8)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0890	Memory Region Start Address (RDC_MRSA9)	32	R/W	Undefined	3.2.5.7/64

Table continues on the next page...

RDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303D_0894	Memory Region End Address (RDC_MREA9)	32	R/W	Undefined	3.2.5.8/66
303D_0898	Memory Region Control (RDC_MRC9)	32	R/W	0000_00FFh	3.2.5.9/67
303D_089C	Memory Region Violation Status (RDC_MRVS9)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_08A0	Memory Region Start Address (RDC_MRSA10)	32	R/W	Undefined	3.2.5.7/64
303D_08A4	Memory Region End Address (RDC_MREA10)	32	R/W	Undefined	3.2.5.8/66
303D_08A8	Memory Region Control (RDC_MRC10)	32	R/W	0000_00FFh	3.2.5.9/67
303D_08AC	Memory Region Violation Status (RDC_MRVS10)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_08B0	Memory Region Start Address (RDC_MRSA11)	32	R/W	Undefined	3.2.5.7/64
303D_08B4	Memory Region End Address (RDC_MREA11)	32	R/W	Undefined	3.2.5.8/66
303D_08B8	Memory Region Control (RDC_MRC11)	32	R/W	0000_00FFh	3.2.5.9/67
303D_08BC	Memory Region Violation Status (RDC_MRVS11)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_08C0	Memory Region Start Address (RDC_MRSA12)	32	R/W	Undefined	3.2.5.7/64
303D_08C4	Memory Region End Address (RDC_MREA12)	32	R/W	Undefined	3.2.5.8/66
303D_08C8	Memory Region Control (RDC_MRC12)	32	R/W	0000_00FFh	3.2.5.9/67
303D_08CC	Memory Region Violation Status (RDC_MRVS12)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_08D0	Memory Region Start Address (RDC_MRSA13)	32	R/W	Undefined	3.2.5.7/64
303D_08D4	Memory Region End Address (RDC_MREA13)	32	R/W	Undefined	3.2.5.8/66
303D_08D8	Memory Region Control (RDC_MRC13)	32	R/W	0000_00FFh	3.2.5.9/67
303D_08DC	Memory Region Violation Status (RDC_MRVS13)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_08E0	Memory Region Start Address (RDC_MRSA14)	32	R/W	Undefined	3.2.5.7/64
303D_08E4	Memory Region End Address (RDC_MREA14)	32	R/W	Undefined	3.2.5.8/66
303D_08E8	Memory Region Control (RDC_MRC14)	32	R/W	0000_00FFh	3.2.5.9/67
303D_08EC	Memory Region Violation Status (RDC_MRVS14)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_08F0	Memory Region Start Address (RDC_MRSA15)	32	R/W	Undefined	3.2.5.7/64
303D_08F4	Memory Region End Address (RDC_MREA15)	32	R/W	Undefined	3.2.5.8/66
303D_08F8	Memory Region Control (RDC_MRC15)	32	R/W	0000_00FFh	3.2.5.9/67
303D_08FC	Memory Region Violation Status (RDC_MRVS15)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0900	Memory Region Start Address (RDC_MRSA16)	32	R/W	Undefined	3.2.5.7/64
303D_0904	Memory Region End Address (RDC_MREA16)	32	R/W	Undefined	3.2.5.8/66
303D_0908	Memory Region Control (RDC_MRC16)	32	R/W	0000_00FFh	3.2.5.9/67
303D_090C	Memory Region Violation Status (RDC_MRVS16)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0910	Memory Region Start Address (RDC_MRSA17)	32	R/W	Undefined	3.2.5.7/64

Table continues on the next page...

RDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303D_0914	Memory Region End Address (RDC_MREA17)	32	R/W	Undefined	3.2.5.8/66
303D_0918	Memory Region Control (RDC_MRC17)	32	R/W	0000_00FFh	3.2.5.9/67
303D_091C	Memory Region Violation Status (RDC_MRVS17)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0920	Memory Region Start Address (RDC_MRSA18)	32	R/W	Undefined	3.2.5.7/64
303D_0924	Memory Region End Address (RDC_MREA18)	32	R/W	Undefined	3.2.5.8/66
303D_0928	Memory Region Control (RDC_MRC18)	32	R/W	0000_00FFh	3.2.5.9/67
303D_092C	Memory Region Violation Status (RDC_MRVS18)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0930	Memory Region Start Address (RDC_MRSA19)	32	R/W	Undefined	3.2.5.7/64
303D_0934	Memory Region End Address (RDC_MREA19)	32	R/W	Undefined	3.2.5.8/66
303D_0938	Memory Region Control (RDC_MRC19)	32	R/W	0000_00FFh	3.2.5.9/67
303D_093C	Memory Region Violation Status (RDC_MRVS19)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0940	Memory Region Start Address (RDC_MRSA20)	32	R/W	Undefined	3.2.5.7/64
303D_0944	Memory Region End Address (RDC_MREA20)	32	R/W	Undefined	3.2.5.8/66
303D_0948	Memory Region Control (RDC_MRC20)	32	R/W	0000_00FFh	3.2.5.9/67
303D_094C	Memory Region Violation Status (RDC_MRVS20)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0950	Memory Region Start Address (RDC_MRSA21)	32	R/W	Undefined	3.2.5.7/64
303D_0954	Memory Region End Address (RDC_MREA21)	32	R/W	Undefined	3.2.5.8/66
303D_0958	Memory Region Control (RDC_MRC21)	32	R/W	0000_00FFh	3.2.5.9/67
303D_095C	Memory Region Violation Status (RDC_MRVS21)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0960	Memory Region Start Address (RDC_MRSA22)	32	R/W	Undefined	3.2.5.7/64
303D_0964	Memory Region End Address (RDC_MREA22)	32	R/W	Undefined	3.2.5.8/66
303D_0968	Memory Region Control (RDC_MRC22)	32	R/W	0000_00FFh	3.2.5.9/67
303D_096C	Memory Region Violation Status (RDC_MRVS22)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0970	Memory Region Start Address (RDC_MRSA23)	32	R/W	Undefined	3.2.5.7/64
303D_0974	Memory Region End Address (RDC_MREA23)	32	R/W	Undefined	3.2.5.8/66
303D_0978	Memory Region Control (RDC_MRC23)	32	R/W	0000_00FFh	3.2.5.9/67
303D_097C	Memory Region Violation Status (RDC_MRVS23)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0980	Memory Region Start Address (RDC_MRSA24)	32	R/W	Undefined	3.2.5.7/64
303D_0984	Memory Region End Address (RDC_MREA24)	32	R/W	Undefined	3.2.5.8/66
303D_0988	Memory Region Control (RDC_MRC24)	32	R/W	0000_00FFh	3.2.5.9/67
303D_098C	Memory Region Violation Status (RDC_MRVS24)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0990	Memory Region Start Address (RDC_MRSA25)	32	R/W	Undefined	3.2.5.7/64

Table continues on the next page...

RDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303D_0994	Memory Region End Address (RDC_MREA25)	32	R/W	Undefined	3.2.5.8/66
303D_0998	Memory Region Control (RDC_MRC25)	32	R/W	0000_00FFh	3.2.5.9/67
303D_099C	Memory Region Violation Status (RDC_MRVS25)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_09A0	Memory Region Start Address (RDC_MRSA26)	32	R/W	Undefined	3.2.5.7/64
303D_09A4	Memory Region End Address (RDC_MREA26)	32	R/W	Undefined	3.2.5.8/66
303D_09A8	Memory Region Control (RDC_MRC26)	32	R/W	0000_00FFh	3.2.5.9/67
303D_09AC	Memory Region Violation Status (RDC_MRVS26)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_09B0	Memory Region Start Address (RDC_MRSA27)	32	R/W	Undefined	3.2.5.7/64
303D_09B4	Memory Region End Address (RDC_MREA27)	32	R/W	Undefined	3.2.5.8/66
303D_09B8	Memory Region Control (RDC_MRC27)	32	R/W	0000_00FFh	3.2.5.9/67
303D_09BC	Memory Region Violation Status (RDC_MRVS27)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_09C0	Memory Region Start Address (RDC_MRSA28)	32	R/W	Undefined	3.2.5.7/64
303D_09C4	Memory Region End Address (RDC_MREA28)	32	R/W	Undefined	3.2.5.8/66
303D_09C8	Memory Region Control (RDC_MRC28)	32	R/W	0000_00FFh	3.2.5.9/67
303D_09CC	Memory Region Violation Status (RDC_MRVS28)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_09D0	Memory Region Start Address (RDC_MRSA29)	32	R/W	Undefined	3.2.5.7/64
303D_09D4	Memory Region End Address (RDC_MREA29)	32	R/W	Undefined	3.2.5.8/66
303D_09D8	Memory Region Control (RDC_MRC29)	32	R/W	0000_00FFh	3.2.5.9/67
303D_09DC	Memory Region Violation Status (RDC_MRVS29)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_09E0	Memory Region Start Address (RDC_MRSA30)	32	R/W	Undefined	3.2.5.7/64
303D_09E4	Memory Region End Address (RDC_MREA30)	32	R/W	Undefined	3.2.5.8/66
303D_09E8	Memory Region Control (RDC_MRC30)	32	R/W	0000_00FFh	3.2.5.9/67
303D_09EC	Memory Region Violation Status (RDC_MRVS30)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_09F0	Memory Region Start Address (RDC_MRSA31)	32	R/W	Undefined	3.2.5.7/64
303D_09F4	Memory Region End Address (RDC_MREA31)	32	R/W	Undefined	3.2.5.8/66
303D_09F8	Memory Region Control (RDC_MRC31)	32	R/W	0000_00FFh	3.2.5.9/67
303D_09FC	Memory Region Violation Status (RDC_MRVS31)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0A00	Memory Region Start Address (RDC_MRSA32)	32	R/W	Undefined	3.2.5.7/64
303D_0A04	Memory Region End Address (RDC_MREA32)	32	R/W	Undefined	3.2.5.8/66
303D_0A08	Memory Region Control (RDC_MRC32)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0A0C	Memory Region Violation Status (RDC_MRVS32)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0A10	Memory Region Start Address (RDC_MRSA33)	32	R/W	Undefined	3.2.5.7/64

Table continues on the next page...

RDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303D_0A14	Memory Region End Address (RDC_MREA33)	32	R/W	Undefined	3.2.5.8/66
303D_0A18	Memory Region Control (RDC_MRC33)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0A1C	Memory Region Violation Status (RDC_MRVS33)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0A20	Memory Region Start Address (RDC_MRSA34)	32	R/W	Undefined	3.2.5.7/64
303D_0A24	Memory Region End Address (RDC_MREA34)	32	R/W	Undefined	3.2.5.8/66
303D_0A28	Memory Region Control (RDC_MRC34)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0A2C	Memory Region Violation Status (RDC_MRVS34)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0A30	Memory Region Start Address (RDC_MRSA35)	32	R/W	Undefined	3.2.5.7/64
303D_0A34	Memory Region End Address (RDC_MREA35)	32	R/W	Undefined	3.2.5.8/66
303D_0A38	Memory Region Control (RDC_MRC35)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0A3C	Memory Region Violation Status (RDC_MRVS35)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0A40	Memory Region Start Address (RDC_MRSA36)	32	R/W	Undefined	3.2.5.7/64
303D_0A44	Memory Region End Address (RDC_MREA36)	32	R/W	Undefined	3.2.5.8/66
303D_0A48	Memory Region Control (RDC_MRC36)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0A4C	Memory Region Violation Status (RDC_MRVS36)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0A50	Memory Region Start Address (RDC_MRSA37)	32	R/W	Undefined	3.2.5.7/64
303D_0A54	Memory Region End Address (RDC_MREA37)	32	R/W	Undefined	3.2.5.8/66
303D_0A58	Memory Region Control (RDC_MRC37)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0A5C	Memory Region Violation Status (RDC_MRVS37)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0A60	Memory Region Start Address (RDC_MRSA38)	32	R/W	Undefined	3.2.5.7/64
303D_0A64	Memory Region End Address (RDC_MREA38)	32	R/W	Undefined	3.2.5.8/66
303D_0A68	Memory Region Control (RDC_MRC38)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0A6C	Memory Region Violation Status (RDC_MRVS38)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0A70	Memory Region Start Address (RDC_MRSA39)	32	R/W	Undefined	3.2.5.7/64
303D_0A74	Memory Region End Address (RDC_MREA39)	32	R/W	Undefined	3.2.5.8/66
303D_0A78	Memory Region Control (RDC_MRC39)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0A7C	Memory Region Violation Status (RDC_MRVS39)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0A80	Memory Region Start Address (RDC_MRSA40)	32	R/W	Undefined	3.2.5.7/64
303D_0A84	Memory Region End Address (RDC_MREA40)	32	R/W	Undefined	3.2.5.8/66
303D_0A88	Memory Region Control (RDC_MRC40)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0A8C	Memory Region Violation Status (RDC_MRVS40)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0A90	Memory Region Start Address (RDC_MRSA41)	32	R/W	Undefined	3.2.5.7/64

Table continues on the next page...

RDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303D_0A94	Memory Region End Address (RDC_MREA41)	32	R/W	Undefined	3.2.5.8/66
303D_0A98	Memory Region Control (RDC_MRC41)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0A9C	Memory Region Violation Status (RDC_MRVS41)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0AA0	Memory Region Start Address (RDC_MRSA42)	32	R/W	Undefined	3.2.5.7/64
303D_0AA4	Memory Region End Address (RDC_MREA42)	32	R/W	Undefined	3.2.5.8/66
303D_0AA8	Memory Region Control (RDC_MRC42)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0AAC	Memory Region Violation Status (RDC_MRVS42)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0AB0	Memory Region Start Address (RDC_MRSA43)	32	R/W	Undefined	3.2.5.7/64
303D_0AB4	Memory Region End Address (RDC_MREA43)	32	R/W	Undefined	3.2.5.8/66
303D_0AB8	Memory Region Control (RDC_MRC43)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0ABC	Memory Region Violation Status (RDC_MRVS43)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0AC0	Memory Region Start Address (RDC_MRSA44)	32	R/W	Undefined	3.2.5.7/64
303D_0AC4	Memory Region End Address (RDC_MREA44)	32	R/W	Undefined	3.2.5.8/66
303D_0AC8	Memory Region Control (RDC_MRC44)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0ACC	Memory Region Violation Status (RDC_MRVS44)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0AD0	Memory Region Start Address (RDC_MRSA45)	32	R/W	Undefined	3.2.5.7/64
303D_0AD4	Memory Region End Address (RDC_MREA45)	32	R/W	Undefined	3.2.5.8/66
303D_0AD8	Memory Region Control (RDC_MRC45)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0ADC	Memory Region Violation Status (RDC_MRVS45)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0AE0	Memory Region Start Address (RDC_MRSA46)	32	R/W	Undefined	3.2.5.7/64
303D_0AE4	Memory Region End Address (RDC_MREA46)	32	R/W	Undefined	3.2.5.8/66
303D_0AE8	Memory Region Control (RDC_MRC46)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0AEC	Memory Region Violation Status (RDC_MRVS46)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0AF0	Memory Region Start Address (RDC_MRSA47)	32	R/W	Undefined	3.2.5.7/64
303D_0AF4	Memory Region End Address (RDC_MREA47)	32	R/W	Undefined	3.2.5.8/66
303D_0AF8	Memory Region Control (RDC_MRC47)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0AFC	Memory Region Violation Status (RDC_MRVS47)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0B00	Memory Region Start Address (RDC_MRSA48)	32	R/W	Undefined	3.2.5.7/64
303D_0B04	Memory Region End Address (RDC_MREA48)	32	R/W	Undefined	3.2.5.8/66
303D_0B08	Memory Region Control (RDC_MRC48)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0B0C	Memory Region Violation Status (RDC_MRVS48)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0B10	Memory Region Start Address (RDC_MRSA49)	32	R/W	Undefined	3.2.5.7/64

Table continues on the next page...

RDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303D_0B14	Memory Region End Address (RDC_MREA49)	32	R/W	Undefined	3.2.5.8/66
303D_0B18	Memory Region Control (RDC_MRC49)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0B1C	Memory Region Violation Status (RDC_MRVS49)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0B20	Memory Region Start Address (RDC_MRSA50)	32	R/W	Undefined	3.2.5.7/64
303D_0B24	Memory Region End Address (RDC_MREA50)	32	R/W	Undefined	3.2.5.8/66
303D_0B28	Memory Region Control (RDC_MRC50)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0B2C	Memory Region Violation Status (RDC_MRVS50)	32	R/W	0000_0000h	3.2.5.10/ 68
303D_0B30	Memory Region Start Address (RDC_MRSA51)	32	R/W	Undefined	3.2.5.7/64
303D_0B34	Memory Region End Address (RDC_MREA51)	32	R/W	Undefined	3.2.5.8/66
303D_0B38	Memory Region Control (RDC_MRC51)	32	R/W	0000_00FFh	3.2.5.9/67
303D_0B3C	Memory Region Violation Status (RDC_MRVS51)	32	R/W	0000_0000h	3.2.5.10/ 68

3.2.5.1 Version Information (RDC_VIR)

The VIR provides version information including the number of domains, number of master slots, number of peripheral slots, and number of memory regions.

Address: 303D_0000h base + 0h offset = 303D_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				NRGN								NPER								NMSTR								NDID			
W																																
Reset	0	0	0	0	0	0	1	1	0	1	1	1	0	1	1	0	1	1	1	0	0	0	1	0	0	0	0	0	0	1	0	0

RDC_VIR field descriptions

Field	Description
31–28 Reserved	This field is reserved.
27–20 NRGN	Number of Memory Regions Indicates the number of memory regions in this instance of the RDC.
19–12 NPER	Number of Peripherals Indicates the number of peripherals that can be isolated or safe-shared
11–4 NMSTR	Number of Masters Indicates the number of masters supported by this instance of RDC.
NDID	Number of Domains

Table continues on the next page...

RDC_VIR field descriptions (continued)

Field	Description
	Indicates the number of domain ids supported by this instance of the RDC. Add one to the register value to get the actual number of domains.

3.2.5.2 Status (RDC_STAT)

Address: 303D_0000h base + 24h offset = 303D_0024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

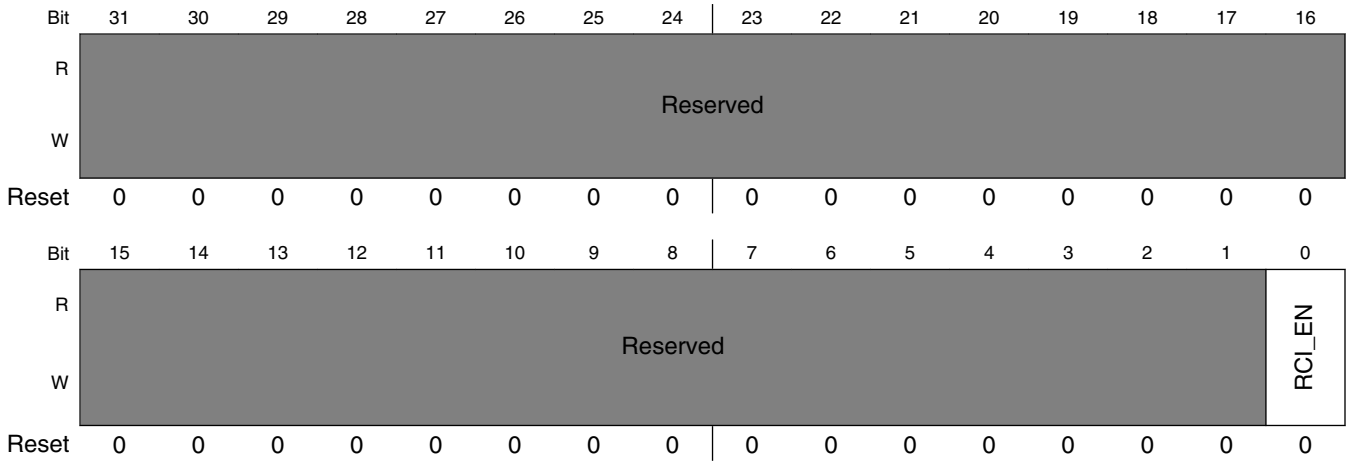
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PDS	Reserved				DID			
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

RDC_STAT field descriptions

Field	Description
31–9 Reserved	This field is reserved.
8 PDS	Power Domain Status Indicates if the "Power Down" memory regions are powered and available. Power Down memory regions are only those memory regions susceptible to power outage for power savings are unavailable if this is zero. "Always-On" memory regions remain available. Always On memory regions are those regions that are not powered down unless the entire SoC is powered down. This signal remains low until all access controls have been restored to the domain. 0 Power Down Domain is OFF 1 Power Down Domain is ON
7–4 Reserved	This field is reserved.
DID	Domain ID The Domain ID of the core or bus master that is reading this. The value is different for requests from different domains.

3.2.5.3 Interrupt and Control (RDC_INTCTRL)

Address: 303D_0000h base + 28h offset = 303D_0028h



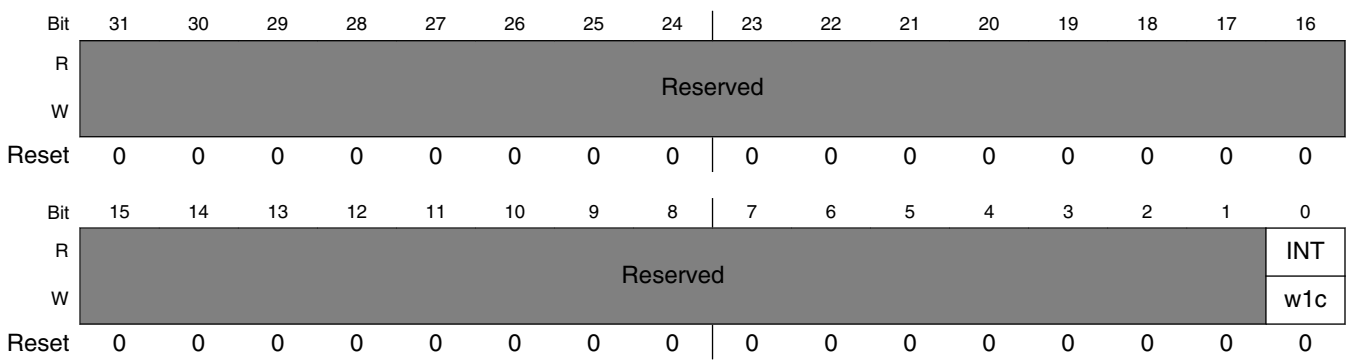
RDC_INTCTRL field descriptions

Field	Description
31–1 Reserved	This field is reserved.
0 RCI_EN	Restoration Complete Interrupt Interrupt generated when the RDC has completed restoring state to a recently re-powered memory regions. 0 Interrupt Disabled 1 Interrupt Enabled

3.2.5.4 Interrupt Status (RDC_INTSTAT)

Indication of Interrupt Pending for State Restoration

Address: 303D_0000h base + 2Ch offset = 303D_002Ch



RDC_INTSTAT field descriptions

Field	Description
31–1 Reserved	This field is reserved.
0 INT	Interrupt Status Indicates state of interrupt signal for state restoration. This is that status of the interrupt enabled in RDC_INTCTRL. Write one to interrupt status to clear it. 0 No Interrupt Pending 1 Interrupt Pending

3.2.5.5 Master Domain Assignment (RDC_MDA*n*)

Address: 303D_0000h base + 200h offset + (4d × i), where i=0d to 26d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LCK	Reserved														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DID	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RDC_MDA*n* field descriptions

Field	Description
31 LCK	0 Not Locked 1 Locked
30–2 Reserved	This field is reserved.
DID	Domain ID Indicates the domain to which the Master is assigned 00 Master assigned to Processing Domain 0 01 Master assigned to Processing Domain 1 10 Master assigned to Processing Domain 2 11 Master assigned to Processing Domain 3

3.2.5.6 Peripheral Domain Access Permissions (RDC_PDAP*n*)

Address: 303D_0000h base + 400h offset + (4d × i), where i=0d to 117d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LCK	SREQ	Reserved													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								D3R	D3W	D2R	D2W	D1R	D1W	D0R	D0W
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

RDC_PDAP*n* field descriptions

Field	Description
31 LCK	Peripheral Permissions Lock When set prevents further modification of the Peripheral Domain Access Permissions (sticky bit until reset) 0 Not Locked 1 Locked
30 SREQ	Semaphore Required When set the hardware semaphore state enforces the semaphore lock. If a domain has access permissions and a semaphore has locked a shared peripheral then only the domain holding the semaphore signal can access this peripheral. 0 Semaphores have no effect 1 Semaphores are enforced
29–8 Reserved	This field is reserved.
7 D3R	Domain 3 Read Access 0 No Read Access 1 Read Access Allowed
6 D3W	Domain 3 Write Access 0 No Write Access 1 Write Access Allowed
5 D2R	Domain 2 Read Access 0 No Read Access 1 Read Access Allowed
4 D2W	Domain 2 Write Access 0 No Write Access 1 Write Access Allowed
3 D1R	Domain 1 Read Access

Table continues on the next page...

RDC_PDAP*n* field descriptions (continued)

Field	Description
	0 No Read Access 1 Read Access Allowed
2 D1W	Domain 1 Write Access 0 No Write Access 1 Write Access Allowed
1 D0R	Domain 0 Read Access 0 No Read Access 1 Read Access Allowed
0 D0W	Domain 0 Write Access 0 No Write Access 1 Write Access Allowed

3.2.5.7 Memory Region Start Address (RDC_MRSA*n*)

NOTE

The DDR space is 33-bit width. The RDC memory region registers are 32-bit width. The RDC configuration is the most significant bits in the DDR address space (32:1). To set the start address for this configuration, the MRSA value should be shifted 1-bit and added to the DDR base address. The example below illustrates how to calculate the proper start and end address value. Please refer to the Memory Map for the actual DDR base address.

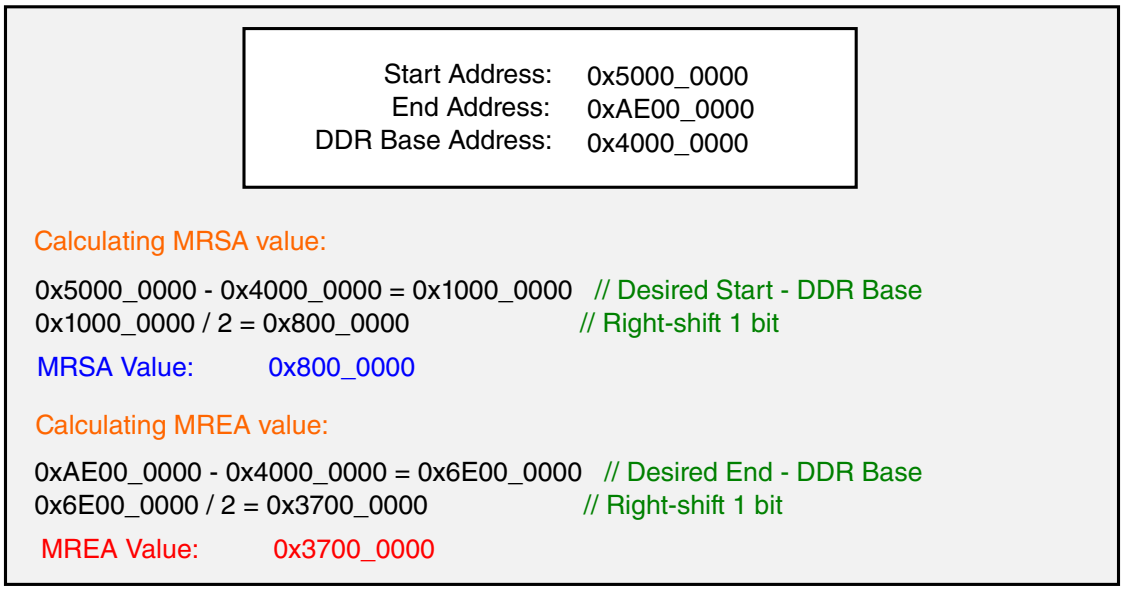


Figure 3-5. Calculating Address Value Examble

Address: 303D_0000h base + 800h offset + (16d × i), where i=0d to 51d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SADR																Reserved															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

RDC_MRSA*n* field descriptions

Field	Description
31–7 SADR	Start address for memory region Lower bound (inclusive) modulo the defined granularity byte size of a region. The region size (granularity) is defined for each Memory/Port in the Memory Region Map section. Region boundaries are aligned to the minimum possible region size for the Memory/Port.
Reserved	This field is reserved.

3.2.5.8 Memory Region End Address (RDC_MREA*n*)

NOTE

The DDR space is 33-bit width. The RDC memory region registers are 32-bit width. The RDC configuration is the most significant bits in the DDR address space (32:1). To set the start address for this configuration, the MRSA value should be shifted 1-bit and added to the DDR base address. The example below illustrates how to calculate the proper start and end address value. Please refer to the Memory Map for the actual DDR base address.

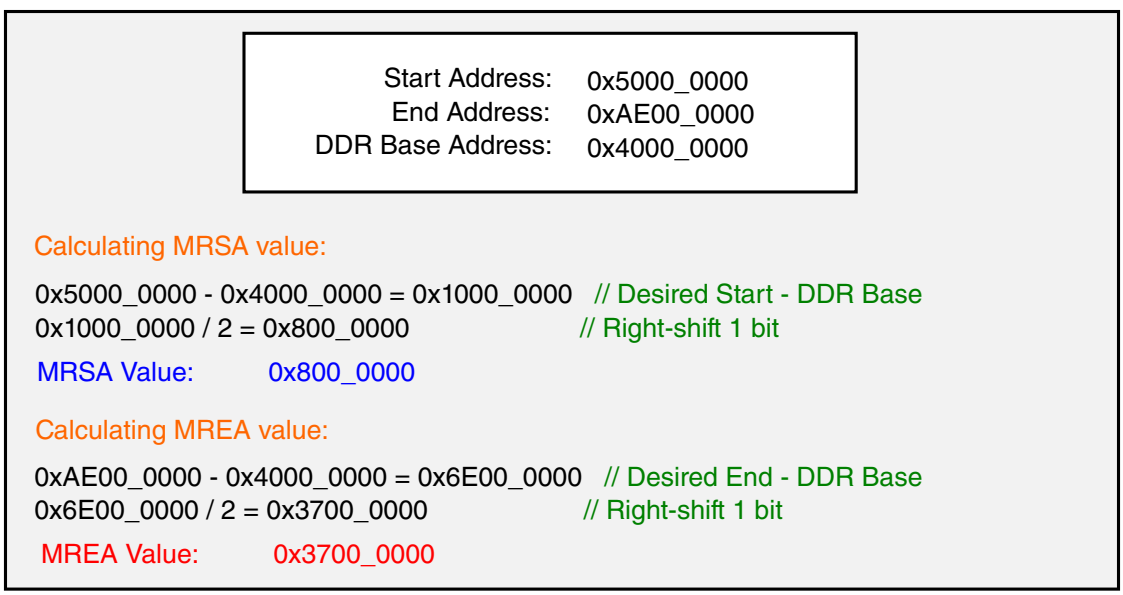


Figure 3-6. Calculating Address Value Examble

Address: 303D_0000h base + 804h offset + (16d × i), where i=0d to 51d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EADR																Reserved															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

RDC_MREA*n* field descriptions

Field	Description
31–7 EADR	Upper bound for memory region

Table continues on the next page...

RDC_MREA*n* field descriptions (continued)

Field	Description
	Upper bound (exclusive) modulo the defined granularity byte size of a region. The region size (granularity) is defined for each Memory/Port in the Memory Region Map section. Region boundaries are aligned to the minimum possible region size for the Memory/Port.
Reserved	This field is reserved.

3.2.5.9 Memory Region Control (RDC_MRC*n*)

Address: 303D_0000h base + 808h offset + (16d × i), where i=0d to 51d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LCK	ENA	Reserved													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								D3R	D3W	D2R	D2W	D1R	D1W	D0R	D0W
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

RDC_MRC*n* field descriptions

Field	Description
31 LCK	Region Lock Locks all region fields from further modification except ENA, which can be set but not reset after LCK is set. LCK is a sticky bit. 0 No Lock. All fields in this register may be modified. 1 Locked. No fields in this register may be modified except ENA, which may be set but not cleared.
30 ENA	Region Enable Activates the memory region. If the region is not activated then the permissions and address boundaries have not affect and the region will be fully accessible. 0 Memory region is not defined or restricted. 1 Memory boundaries, domain permissions and controls are in effect.
29–8 Reserved	This field is reserved.
7 D3R	Domain 3 Read Access to Region 0 Processing Domain 3 does not have Read access to the memory region 1 Processing Domain 3 has Read access to the memory region
6 D3W	Domain 3 Write Access to Region 0 Processing Domain 3 does not have Write access to the memory region 1 Processing Domain 3 has Read access to the memory region
5 D2R	Domain 2 Read Access to Region

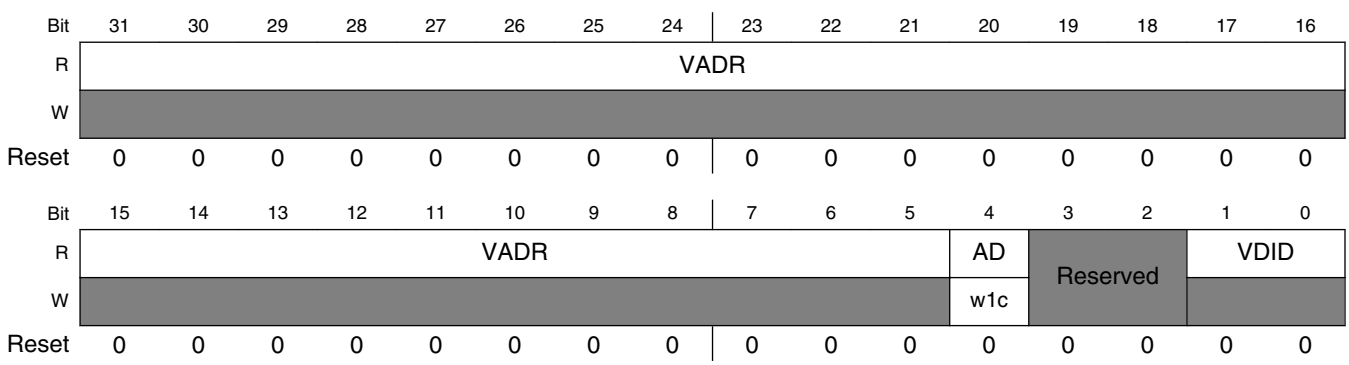
Table continues on the next page...

RDC_MRC*n* field descriptions (continued)

Field	Description
	0 Processing Domain 2 does not have Read access to the memory region 1 Processing Domain 2 has Read access to the memory region
4 D2W	Domain 2 Write Access to Region 0 Processing Domain 2 does not have Write access to the memory region 1 Processing Domain 2 has Write access to the memory region
3 D1R	Domain 1 Read Access to Region 0 Processing Domain 1 does not have Read access to the memory region 1 Processing Domain 1 has Read access to the memory region
2 D1W	Domain 1 Write Access to Region 0 Processing Domain 1 does not have Write access to the memory region 1 Processing Domain 1 has Write access to the memory region
1 D0R	Domain 0 Read Access to Region 0 Processing Domain 0 does not have Read access to the memory region 1 Processing Domain 0 has Read access to the memory region
0 D0W	Domain 0 Write Access to Region 0 Processing Domain 0 does not have Write access to the memory region 1 Processing Domain 0 has Write access to the memory region

3.2.5.10 Memory Region Violation Status (RDC_MRVS*n*)

Address: 303D_0000h base + 80Ch offset + (16d × i), where i=0d to 51d



RDC_MRVS*n* field descriptions

Field	Description
31–5 VADR	Violating Address The address of the denied access. The first access violation is captured. Subsequent violations are ignored until the status register is cleared. Contents are cleared upon reading the register. Clearing of contents occurs only when the status is read by the memory region's associated domain ID (s).

Table continues on the next page...

RDC_MRVS*n* field descriptions (continued)

Field	Description
4 AD	Access Denied Access to a memory region denied. This bit is cleared when this bit is written by one of the allowed domains.
3–2 Reserved	This field is reserved.
VDID	Violating Domain ID The domain ID of the denied access. The first access violation is captured. Subsequent violations are ignored until the status register is cleared. Contents are cleared upon reading the register. 00 Processing Domain 0 01 Processing Domain 1 10 Processing Domain 2 11 Processing Domain 3

3.2.6 RDC SEMA42 Memory Map/Register Definition

Only Supervisor Mode accesses are allowed on these registers. User accesses generate an error termination.

RDC_SEMAPHORE memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303B_0000	Gate Register (RDC_SEMAPHORE1_GATE0)	8	R/W	00h	3.2.6.1/73
303B_0001	Gate Register (RDC_SEMAPHORE1_GATE1)	8	R/W	00h	3.2.6.1/73
303B_0002	Gate Register (RDC_SEMAPHORE1_GATE2)	8	R/W	00h	3.2.6.1/73
303B_0003	Gate Register (RDC_SEMAPHORE1_GATE3)	8	R/W	00h	3.2.6.1/73
303B_0004	Gate Register (RDC_SEMAPHORE1_GATE4)	8	R/W	00h	3.2.6.1/73
303B_0005	Gate Register (RDC_SEMAPHORE1_GATE5)	8	R/W	00h	3.2.6.1/73
303B_0006	Gate Register (RDC_SEMAPHORE1_GATE6)	8	R/W	00h	3.2.6.1/73
303B_0007	Gate Register (RDC_SEMAPHORE1_GATE7)	8	R/W	00h	3.2.6.1/73
303B_0008	Gate Register (RDC_SEMAPHORE1_GATE8)	8	R/W	00h	3.2.6.1/73
303B_0009	Gate Register (RDC_SEMAPHORE1_GATE9)	8	R/W	00h	3.2.6.1/73
303B_000A	Gate Register (RDC_SEMAPHORE1_GATE10)	8	R/W	00h	3.2.6.1/73
303B_000B	Gate Register (RDC_SEMAPHORE1_GATE11)	8	R/W	00h	3.2.6.1/73
303B_000C	Gate Register (RDC_SEMAPHORE1_GATE12)	8	R/W	00h	3.2.6.1/73
303B_000D	Gate Register (RDC_SEMAPHORE1_GATE13)	8	R/W	00h	3.2.6.1/73
303B_000E	Gate Register (RDC_SEMAPHORE1_GATE14)	8	R/W	00h	3.2.6.1/73

Table continues on the next page...

RDC_SEMAPHORE memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303B_000F	Gate Register (RDC_SEMAPHORE1_GATE15)	8	R/W	00h	3.2.6.1/73
303B_0010	Gate Register (RDC_SEMAPHORE1_GATE16)	8	R/W	00h	3.2.6.1/73
303B_0011	Gate Register (RDC_SEMAPHORE1_GATE17)	8	R/W	00h	3.2.6.1/73
303B_0012	Gate Register (RDC_SEMAPHORE1_GATE18)	8	R/W	00h	3.2.6.1/73
303B_0013	Gate Register (RDC_SEMAPHORE1_GATE19)	8	R/W	00h	3.2.6.1/73
303B_0014	Gate Register (RDC_SEMAPHORE1_GATE20)	8	R/W	00h	3.2.6.1/73
303B_0015	Gate Register (RDC_SEMAPHORE1_GATE21)	8	R/W	00h	3.2.6.1/73
303B_0016	Gate Register (RDC_SEMAPHORE1_GATE22)	8	R/W	00h	3.2.6.1/73
303B_0017	Gate Register (RDC_SEMAPHORE1_GATE23)	8	R/W	00h	3.2.6.1/73
303B_0018	Gate Register (RDC_SEMAPHORE1_GATE24)	8	R/W	00h	3.2.6.1/73
303B_0019	Gate Register (RDC_SEMAPHORE1_GATE25)	8	R/W	00h	3.2.6.1/73
303B_001A	Gate Register (RDC_SEMAPHORE1_GATE26)	8	R/W	00h	3.2.6.1/73
303B_001B	Gate Register (RDC_SEMAPHORE1_GATE27)	8	R/W	00h	3.2.6.1/73
303B_001C	Gate Register (RDC_SEMAPHORE1_GATE28)	8	R/W	00h	3.2.6.1/73
303B_001D	Gate Register (RDC_SEMAPHORE1_GATE29)	8	R/W	00h	3.2.6.1/73
303B_001E	Gate Register (RDC_SEMAPHORE1_GATE30)	8	R/W	00h	3.2.6.1/73
303B_001F	Gate Register (RDC_SEMAPHORE1_GATE31)	8	R/W	00h	3.2.6.1/73
303B_0020	Gate Register (RDC_SEMAPHORE1_GATE32)	8	R/W	00h	3.2.6.1/73
303B_0021	Gate Register (RDC_SEMAPHORE1_GATE33)	8	R/W	00h	3.2.6.1/73
303B_0022	Gate Register (RDC_SEMAPHORE1_GATE34)	8	R/W	00h	3.2.6.1/73
303B_0023	Gate Register (RDC_SEMAPHORE1_GATE35)	8	R/W	00h	3.2.6.1/73
303B_0024	Gate Register (RDC_SEMAPHORE1_GATE36)	8	R/W	00h	3.2.6.1/73
303B_0025	Gate Register (RDC_SEMAPHORE1_GATE37)	8	R/W	00h	3.2.6.1/73
303B_0026	Gate Register (RDC_SEMAPHORE1_GATE38)	8	R/W	00h	3.2.6.1/73
303B_0027	Gate Register (RDC_SEMAPHORE1_GATE39)	8	R/W	00h	3.2.6.1/73
303B_0028	Gate Register (RDC_SEMAPHORE1_GATE40)	8	R/W	00h	3.2.6.1/73
303B_0029	Gate Register (RDC_SEMAPHORE1_GATE41)	8	R/W	00h	3.2.6.1/73
303B_002A	Gate Register (RDC_SEMAPHORE1_GATE42)	8	R/W	00h	3.2.6.1/73
303B_002B	Gate Register (RDC_SEMAPHORE1_GATE43)	8	R/W	00h	3.2.6.1/73
303B_002C	Gate Register (RDC_SEMAPHORE1_GATE44)	8	R/W	00h	3.2.6.1/73
303B_002D	Gate Register (RDC_SEMAPHORE1_GATE45)	8	R/W	00h	3.2.6.1/73
303B_002E	Gate Register (RDC_SEMAPHORE1_GATE46)	8	R/W	00h	3.2.6.1/73
303B_002F	Gate Register (RDC_SEMAPHORE1_GATE47)	8	R/W	00h	3.2.6.1/73
303B_0030	Gate Register (RDC_SEMAPHORE1_GATE48)	8	R/W	00h	3.2.6.1/73
303B_0031	Gate Register (RDC_SEMAPHORE1_GATE49)	8	R/W	00h	3.2.6.1/73
303B_0032	Gate Register (RDC_SEMAPHORE1_GATE50)	8	R/W	00h	3.2.6.1/73
303B_0033	Gate Register (RDC_SEMAPHORE1_GATE51)	8	R/W	00h	3.2.6.1/73
303B_0034	Gate Register (RDC_SEMAPHORE1_GATE52)	8	R/W	00h	3.2.6.1/73

Table continues on the next page...

RDC_SEMAPHORE memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303B_0035	Gate Register (RDC_SEMAPHORE1_GATE53)	8	R/W	00h	3.2.6.1/73
303B_0036	Gate Register (RDC_SEMAPHORE1_GATE54)	8	R/W	00h	3.2.6.1/73
303B_0037	Gate Register (RDC_SEMAPHORE1_GATE55)	8	R/W	00h	3.2.6.1/73
303B_0038	Gate Register (RDC_SEMAPHORE1_GATE56)	8	R/W	00h	3.2.6.1/73
303B_0039	Gate Register (RDC_SEMAPHORE1_GATE57)	8	R/W	00h	3.2.6.1/73
303B_003A	Gate Register (RDC_SEMAPHORE1_GATE58)	8	R/W	00h	3.2.6.1/73
303B_003B	Gate Register (RDC_SEMAPHORE1_GATE59)	8	R/W	00h	3.2.6.1/73
303B_003C	Gate Register (RDC_SEMAPHORE1_GATE60)	8	R/W	00h	3.2.6.1/73
303B_003D	Gate Register (RDC_SEMAPHORE1_GATE61)	8	R/W	00h	3.2.6.1/73
303B_003E	Gate Register (RDC_SEMAPHORE1_GATE62)	8	R/W	00h	3.2.6.1/73
303B_003F	Gate Register (RDC_SEMAPHORE1_GATE63)	8	R/W	00h	3.2.6.1/73
303B_0040	Reset Gate Write (RDC_SEMAPHORE1_RSTGT_W)	16	R/W	0000h	3.2.6.2/74
303B_0040	Reset Gate Read (RDC_SEMAPHORE1_RSTGT_R)	16	R/W	0000h	3.2.6.3/75
303C_0000	Gate Register (RDC_SEMAPHORE2_GATE0)	8	R/W	00h	3.2.6.1/73
303C_0001	Gate Register (RDC_SEMAPHORE2_GATE1)	8	R/W	00h	3.2.6.1/73
303C_0002	Gate Register (RDC_SEMAPHORE2_GATE2)	8	R/W	00h	3.2.6.1/73
303C_0003	Gate Register (RDC_SEMAPHORE2_GATE3)	8	R/W	00h	3.2.6.1/73
303C_0004	Gate Register (RDC_SEMAPHORE2_GATE4)	8	R/W	00h	3.2.6.1/73
303C_0005	Gate Register (RDC_SEMAPHORE2_GATE5)	8	R/W	00h	3.2.6.1/73
303C_0006	Gate Register (RDC_SEMAPHORE2_GATE6)	8	R/W	00h	3.2.6.1/73
303C_0007	Gate Register (RDC_SEMAPHORE2_GATE7)	8	R/W	00h	3.2.6.1/73
303C_0008	Gate Register (RDC_SEMAPHORE2_GATE8)	8	R/W	00h	3.2.6.1/73
303C_0009	Gate Register (RDC_SEMAPHORE2_GATE9)	8	R/W	00h	3.2.6.1/73
303C_000A	Gate Register (RDC_SEMAPHORE2_GATE10)	8	R/W	00h	3.2.6.1/73
303C_000B	Gate Register (RDC_SEMAPHORE2_GATE11)	8	R/W	00h	3.2.6.1/73
303C_000C	Gate Register (RDC_SEMAPHORE2_GATE12)	8	R/W	00h	3.2.6.1/73
303C_000D	Gate Register (RDC_SEMAPHORE2_GATE13)	8	R/W	00h	3.2.6.1/73
303C_000E	Gate Register (RDC_SEMAPHORE2_GATE14)	8	R/W	00h	3.2.6.1/73
303C_000F	Gate Register (RDC_SEMAPHORE2_GATE15)	8	R/W	00h	3.2.6.1/73
303C_0010	Gate Register (RDC_SEMAPHORE2_GATE16)	8	R/W	00h	3.2.6.1/73
303C_0011	Gate Register (RDC_SEMAPHORE2_GATE17)	8	R/W	00h	3.2.6.1/73
303C_0012	Gate Register (RDC_SEMAPHORE2_GATE18)	8	R/W	00h	3.2.6.1/73
303C_0013	Gate Register (RDC_SEMAPHORE2_GATE19)	8	R/W	00h	3.2.6.1/73
303C_0014	Gate Register (RDC_SEMAPHORE2_GATE20)	8	R/W	00h	3.2.6.1/73
303C_0015	Gate Register (RDC_SEMAPHORE2_GATE21)	8	R/W	00h	3.2.6.1/73
303C_0016	Gate Register (RDC_SEMAPHORE2_GATE22)	8	R/W	00h	3.2.6.1/73
303C_0017	Gate Register (RDC_SEMAPHORE2_GATE23)	8	R/W	00h	3.2.6.1/73
303C_0018	Gate Register (RDC_SEMAPHORE2_GATE24)	8	R/W	00h	3.2.6.1/73

Table continues on the next page...

RDC_SEMAPHORE memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303C_0019	Gate Register (RDC_SEMAPHORE2_GATE25)	8	R/W	00h	3.2.6.1/73
303C_001A	Gate Register (RDC_SEMAPHORE2_GATE26)	8	R/W	00h	3.2.6.1/73
303C_001B	Gate Register (RDC_SEMAPHORE2_GATE27)	8	R/W	00h	3.2.6.1/73
303C_001C	Gate Register (RDC_SEMAPHORE2_GATE28)	8	R/W	00h	3.2.6.1/73
303C_001D	Gate Register (RDC_SEMAPHORE2_GATE29)	8	R/W	00h	3.2.6.1/73
303C_001E	Gate Register (RDC_SEMAPHORE2_GATE30)	8	R/W	00h	3.2.6.1/73
303C_001F	Gate Register (RDC_SEMAPHORE2_GATE31)	8	R/W	00h	3.2.6.1/73
303C_0020	Gate Register (RDC_SEMAPHORE2_GATE32)	8	R/W	00h	3.2.6.1/73
303C_0021	Gate Register (RDC_SEMAPHORE2_GATE33)	8	R/W	00h	3.2.6.1/73
303C_0022	Gate Register (RDC_SEMAPHORE2_GATE34)	8	R/W	00h	3.2.6.1/73
303C_0023	Gate Register (RDC_SEMAPHORE2_GATE35)	8	R/W	00h	3.2.6.1/73
303C_0024	Gate Register (RDC_SEMAPHORE2_GATE36)	8	R/W	00h	3.2.6.1/73
303C_0025	Gate Register (RDC_SEMAPHORE2_GATE37)	8	R/W	00h	3.2.6.1/73
303C_0026	Gate Register (RDC_SEMAPHORE2_GATE38)	8	R/W	00h	3.2.6.1/73
303C_0027	Gate Register (RDC_SEMAPHORE2_GATE39)	8	R/W	00h	3.2.6.1/73
303C_0028	Gate Register (RDC_SEMAPHORE2_GATE40)	8	R/W	00h	3.2.6.1/73
303C_0029	Gate Register (RDC_SEMAPHORE2_GATE41)	8	R/W	00h	3.2.6.1/73
303C_002A	Gate Register (RDC_SEMAPHORE2_GATE42)	8	R/W	00h	3.2.6.1/73
303C_002B	Gate Register (RDC_SEMAPHORE2_GATE43)	8	R/W	00h	3.2.6.1/73
303C_002C	Gate Register (RDC_SEMAPHORE2_GATE44)	8	R/W	00h	3.2.6.1/73
303C_002D	Gate Register (RDC_SEMAPHORE2_GATE45)	8	R/W	00h	3.2.6.1/73
303C_002E	Gate Register (RDC_SEMAPHORE2_GATE46)	8	R/W	00h	3.2.6.1/73
303C_002F	Gate Register (RDC_SEMAPHORE2_GATE47)	8	R/W	00h	3.2.6.1/73
303C_0030	Gate Register (RDC_SEMAPHORE2_GATE48)	8	R/W	00h	3.2.6.1/73
303C_0031	Gate Register (RDC_SEMAPHORE2_GATE49)	8	R/W	00h	3.2.6.1/73
303C_0032	Gate Register (RDC_SEMAPHORE2_GATE50)	8	R/W	00h	3.2.6.1/73
303C_0033	Gate Register (RDC_SEMAPHORE2_GATE51)	8	R/W	00h	3.2.6.1/73
303C_0034	Gate Register (RDC_SEMAPHORE2_GATE52)	8	R/W	00h	3.2.6.1/73
303C_0035	Gate Register (RDC_SEMAPHORE2_GATE53)	8	R/W	00h	3.2.6.1/73
303C_0036	Gate Register (RDC_SEMAPHORE2_GATE54)	8	R/W	00h	3.2.6.1/73
303C_0037	Gate Register (RDC_SEMAPHORE2_GATE55)	8	R/W	00h	3.2.6.1/73
303C_0038	Gate Register (RDC_SEMAPHORE2_GATE56)	8	R/W	00h	3.2.6.1/73
303C_0039	Gate Register (RDC_SEMAPHORE2_GATE57)	8	R/W	00h	3.2.6.1/73
303C_003A	Gate Register (RDC_SEMAPHORE2_GATE58)	8	R/W	00h	3.2.6.1/73
303C_003B	Gate Register (RDC_SEMAPHORE2_GATE59)	8	R/W	00h	3.2.6.1/73
303C_003C	Gate Register (RDC_SEMAPHORE2_GATE60)	8	R/W	00h	3.2.6.1/73
303C_003D	Gate Register (RDC_SEMAPHORE2_GATE61)	8	R/W	00h	3.2.6.1/73
303C_003E	Gate Register (RDC_SEMAPHORE2_GATE62)	8	R/W	00h	3.2.6.1/73

Table continues on the next page...

RDC_SEMAPHORE memory map (continued)

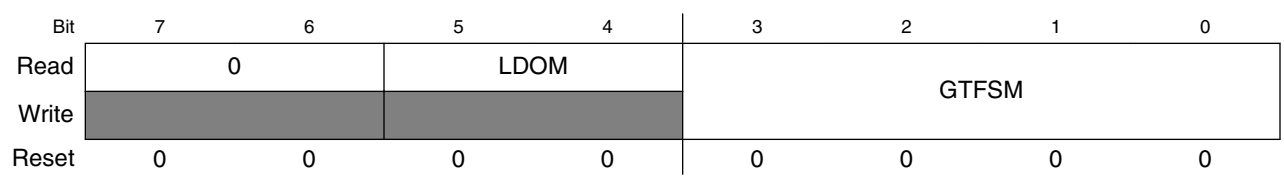
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303C_003F	Gate Register (RDC_SEMAPHORE2_GATE63)	8	R/W	00h	3.2.6.1/73
303C_0040	Reset Gate Write (RDC_SEMAPHORE2_RSTGT_W)	16	R/W	0000h	3.2.6.2/74
303C_0040	Reset Gate Read (RDC_SEMAPHORE2_RSTGT_R)	16	R/W	0000h	3.2.6.3/75

3.2.6.1 Gate Register (RDC_SEMAPHORE*x*_GATE*n*)

Each semaphore gate is implemented in a 4-bit finite state machine, right-justified in a byte data structure. The hardware uses the logical bus master number (master_index) in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. Attempted writes with a data value that is neither the unlock value nor the appropriate lock value (master_index + 1) are simply treated as "no operation" and do not affect any gate state. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes. Processor dex values can be found in AIPSTZ Memory Map/Register Definition.

Address: Base address + 0h offset + (1d × i), where i=0d to 63d



RDC_SEMAPHORE*x*_GATE*n* field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 LDOM	Read-only bits. They indicate which domain had currently locked the gate. 00 The gate is locked by domain 0. (True if bits [3:0] do not equal 0000.) 01 The gate has been locked by domain 1. 10 The gate has been locked by domain 2. 11 The gate has been locked by domain 3.
GTFSM	Gate Finite State Machine. The state of the gate reflects the last processor that locked it, which can be useful during system debug.

Table continues on the next page...

RDC_SEMAPHORE*x*_GATE*n* field descriptions (continued)

Field	Description
	The hardware gate is maintained in a 16-state implementation, defined as: 0000 The gate is unlocked (free). 0001 The gate has been locked by processor with master_index = 0. 0010 The gate has been locked by processor with master_index = 1. 0011 The gate has been locked by processor with master_index = 2. 0100 The gate has been locked by processor with master_index = 3. 0101 The gate has been locked by processor with master_index = 4. 0110 The gate has been locked by processor with master_index = 5. 0111 The gate has been locked by processor with master_index = 6. 1000 The gate has been locked by processor with master_index = 7. 1001 The gate has been locked by processor with master_index = 8. 1010 The gate has been locked by processor with master_index = 9. 1011 The gate has been locked by processor with master_index = 10. 1100 The gate has been locked by processor with master_index = 11. 1101 The gate has been locked by processor with master_index = 12. 1110 The gate has been locked by processor with master_index = 13. 1111 The gate has been locked by processor with master_index = 14.

3.2.6.2 Reset Gate Write (RDC_SEMAPHORE*x*_RSTGT_W)

Although the intent of the hardware gate implementation specifies a protocol where the locking processor must unlock the gate, it is recognized that system operation may require a reset function to re-initialize the state of any gate(s) without requiring a system-level reset.

To support this special gate reset requirement, the RDC Semaphores module implements a "secure" reset mechanism that allows a hardware gate (or all the gates) to be initialized by following a specific dual-write access pattern. Using a technique similar to that required for the servicing of a software watchdog timer, the secure gate reset requires two consecutive writes with predefined data patterns from the same processor to force the clearing of the specified gate(s). The required access pattern is:

1. A processor performs a 16-bit write to the RDC_SEMA42RSTGT memory location. The least significant byte (RDC_SEMA42RSTGT[RSTGDP]) must be 0xE2; the most significant byte is a "don't_care" for this reference.
2. The same processor then performs a second 16-bit write to the RDC_SEMA42RSTGT location. For this write, the lower byte (RDC_SEMA42RSTGT[RSTGDP]) is the logical complement of the first data pattern (0x1D) and the upper byte (RDC_SEMA42RSTGT[RSTGTN]) specifies the gate(s) to be reset. This gate field can specify a single gate be cleared, or else that all gates are to be cleared. If the same processor writes incorrect data on the second

access or another processor performs the second write access, the special gate reset sequence is aborted and no error signal will be asserted.

3. Reads of the RDC_SEMA42RSTGT location return information on the 2-bit state machine (RDC_SEMA42RSTGT[RSTGSM]) that implements this function, the bus master performing the reset (RDC_SEMA42RSTGT[RSTGMS]), and the gate number(s) last cleared (RDC_SEMA42RSTGT[RSTGTN]). Reads of the RDC_SEMA42RSTGT register do not affect the secure reset finite state machine in any manner.

Address: Base address + 40h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	RSTGTN								0							
Write									RSTGDP							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RDC_SEMAPHOREx_RSTGT_W field descriptions

Field	Description
15–8 RSTGTN	Reset Gate Number. This 8-bit field specifies the specific hardware gate to be reset. This field is updated by the second write. If RSTGTN < 64, then reset the single gate defined by RSTGTN, else reset all the gates.
RSTGDP	Reset Gate Data Pattern. This write-only field is accessed with the specified data patterns on the two consecutive writes to enable the gate reset mechanism. For the first write, RSTGDP = 0xE2 while the second write requires RSTGDP = 0x1D.

3.2.6.3 Reset Gate Read (RDC_SEMAPHOREx_RSTGT_R)

Although the intent of the hardware gate implementation specifies a protocol where the locking processor must unlock the gate, it is recognized that system operation may require a reset function to re-initialize the state of any gate(s) without requiring a system-level reset.

To support this special gate reset requirement, the RDC Semaphores module implements a "secure" reset mechanism that allows a hardware gate (or all the gates) to be initialized by following a specific dual-write access pattern. Using a technique similar to that required for the servicing of a software watchdog timer, the secure gate reset requires two consecutive writes with predefined data patterns from the same processor to force the clearing of the specified gate(s). The required access pattern is:

1. A processor performs a 16-bit write to the RDC_SEMA42RSTGT memory location. The least significant byte (RDC_SEMA42RSTGT[RSTGDP]) must be 0xE2; the most significant byte is a "don't_care" for this reference.
2. The same processor then performs a second 16-bit write to the RDC_SEMA42RSTGT location. For this write, the lower byte

(RDC_SEMA42RSTGT[RSTGDP]) is the logical complement of the first data pattern (0x1D) and the upper byte (RDC_SEMA42RSTGT[RSTGTN]) specifies the gate(s) to be reset. This gate field can specify a single gate be cleared, or else that all gates are to be cleared. If the same processor writes incorrect data on the second access or another processor performs the second write access, the special gate reset sequence is aborted and no error signal will be asserted.

3. Reads of the RDC_SEMA42RSTGT location return information on the 2-bit state machine (RDC_SEMA42RSTGT[RSTGSM]) that implements this function, the bus master performing the reset (RDC_SEMA42RSTGT[RSTGMS]), and the gate number(s) last cleared (RDC_SEMA42RSTGT[RSTGTN]). Reads of the RDC_SEMA42RSTGT register do not affect the secure reset finite state machine in any manner.

Address: Base address + 40h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	RSTGTN								0		RSTGSM		RSTGMS			
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RDC_SEMAPHOREx_RSTGT_R field descriptions

Field	Description
15–8 RSTGTN	Reset Gate Number. This 8-bit field specifies the specific hardware gate to be reset. This field is updated by the second write. If RSTGTN < 64, then reset the single gate defined by RSTGTN, else reset all the gates.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 RSTGSM	Reset Gate Finite State Machine. Reads of the RDC_SEMA42RSTGT register return the encoded state machine value. Note the RSTGSM = 10 state is valid for only a single machine cycle, so it is impossible for a read to return this value. The reset state machine is maintained in a 2-bit, 3-state implementation, defined as: 00 Idle, waiting for the first data pattern write. 01 Waiting for the second data pattern write. 10 The 2-write sequence has completed. Generate the specified gate reset(s). After the reset is performed, this machine returns to the idle (waiting for first data pattern write) state. The "01" state persists for only one clock cycle. Software will never be able to observe this state. 11 This state encoding is never used and therefore reserved.
RSTGMS	Reset Gate Bus Master. This 4-bit read-only field records the logical number of the bus master performing the gate reset function. The reset function requires that the two consecutive writes to this register must be initiated by the same bus master to succeed. This field is updated each time a write to this register occurs. The association between system bus master port numbers, the associated bus master device, and the logical processor number is SoC-specific. Consult the device reference manual for this information.

Chapter 4
ARM Platform and Debug

4.1 ARM Cortex A53 Platform (A53)

4.1.1 Overview

The Cortex-A53 cluster is a mid-range, low-power processor that implements the ARMv8-A architecture. The Cortex-A53 cluster has four cores, each with an L1 memory system and a single shared L2 cache that has a set of additional functions, which are included in a single APR region.

The core supports debug through real-time trace via ETM, and static debug via JTAG. The core platform supports static debug through the debug logic to SoC. This includes the capability of real-time trace via ARM's CoreSight ETM modules. The CTI and CTM modules allow cross-triggering of internal and external trigger sources.

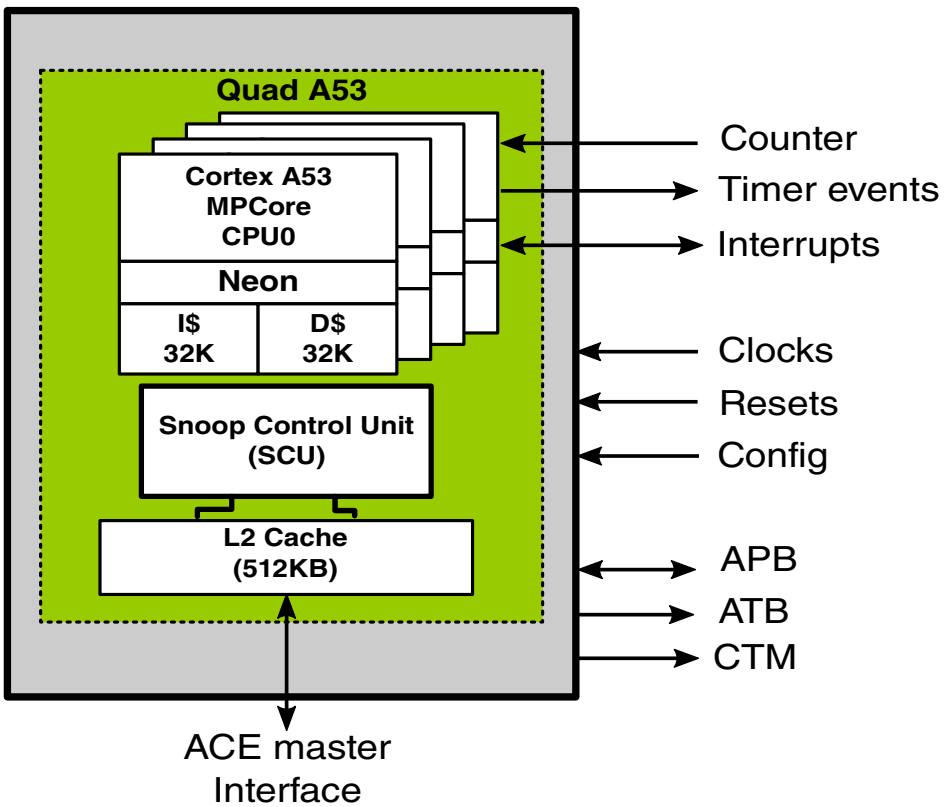


Figure 4-1. Cortex-A53 Block Diagram

4.1.1.1 Features

- 4x cores
- L1 instruction cache 32K with parity
- L1 data cache 32K with SECDED
- Advanced SIMD (NEON) per core
- Crypto extension per core
- CPU cache protection per core
- AMBA 4 ACE interface
- No ACP is included
- utilizes the ARMv8 debug map
- 512KB of L2 Cache
  - 1M with SECDED
  - Input latency 2 cycles
  - Output latency 2 cycles
  - SCU-L2 cache protection

4.1.2 Configuration

The configuration of the bus, cores and memory are detailed in the features and sections below. The core revision is r0p4-51rel0.

4.1.3 Performance

This section will discuss power and clocking performance of the Cortex-A53 Platform.

4.1.3.1 Power

There are several power states supported by the A53 Core Complex. Supported primary states are listed below:

- **Run** – At least one of 4 cores is running. The other cores may either be running, clock gated, or powered down.
- **L2 only coherent** – The L2 is powered up and servicing snoops. The cores are stopped (either powered down or clock gated). In this state the cache is retained coherent to the system.
- **L2 only non-coherent** – Both cores are stopped and powered down, the logic in the L2 controller are retaining the arrays only. In this state the L2 is not coherent, and as a result, the other AP cores must also be stopped.
- **Cluster Off** – The L2 has been flushed from the L2 only coherent state, using the HW flush mechanism (no core required). Then the cluster is fully powered off including the L2 arrays.

The power supply for the cluster is separated into two regions. The first is the control domain and second is for the remainder of the cluster (core and cluster domains). These regions are listed below:

- **Control domain** – The AON control domain contains the controls for powering up and down the rest of the core. The control domain is always powered on first and powered off last.
- **Core domains** – The core domains contains the whole core. The supply for the core domains is the same as the cluster domain, but require separate power down switches.
- **Cluster domain** – The cluster domain contains the rest of the cluster outside of the cores, which includes the shared logic of the L2 memory. The supply for the cluster power domain is the same as the core domains, but does not require power switches as it's shut down externally.

Isolation cells are required between each of these domains and is controlled by the signals on the control domain. These cells are necessary to force the output signal to be isolated to null values when the local power shuts down.

4.1.3.2 Clocking

The A53 platform is provided a main processor clock that supplies the component clocks to the cluster components, including CoreSight debug components. The maximum frequency targets are specified in the chip datasheet.

The cores are intended to support up to 1.5 GHz dependent on forward bias within the operating temperature range. Please see the datasheet for more information. The clocks are described in the table below:

Table 4-1. A53 Clocks

Clock Signal	Clock Name	Frequency	Description
Main Clock	CLKIN	Target	Main input clock.
APB Clock	PCLKENDBG	CLKIN/4	APB clock controls the timing on the debug port. Fixed frequency ratio to main frequency.
ACE Bus Clock	ACLKENM	CLKIN/2	ACE Interface bus clock. Controls the timing on the interface to CCI, but is not synchronous to CCI. Frequency is fixed ratio.
ATB Clock	ATCLKEN	CLKIN/4	ATB clock provides the clocks or outgoing trace. This clock needs to be reasonably high to enable sending samples out, but also needs to be the same as CNTCLKEN. Link to same signal as CNTCLKEN (1:4 core frequency fixed ratio).
Input Counter Clock	CNTCLKEN	CLKIN/4	Input counter clock. Timing is identical to debug port. The frequency is a fixed ratio (1:4) with the core clock. The same signal may be used for both inputs.

4.1.4 Platform sub-blocks

The sections below discuss the high-level overview of the ARM® Cortex®-A53 Platform components.

4.1.4.1 ARM Cortex-A53 MPCore Processor

The information presented in this section focuses on the design aspects of the ARM Cortex-A53 MPCore in the AP subsystem. The Cortex-A53 processor is a mid-range, low-power processor that implements the ARMv8-A architecture. The A53 complex has four cores, each with an L1 memory system and a single shared L2 cache.

The ARM Cortex-A53 MPCore processor consists of:

- Tightly-coupled L2 cache and an integrated Snoop Control Unit (SCU), connecting the four cores within the cluster providing cluster memory coherency, and a configurable coherent external interface supporting AMBA4 bus architecture.
- The Cortex-A53 implements the ARM Generic Interrupt Controller v3 (GICv3) architecture.

4.1.4.2 Advanced SIMD (Neon)

The Cortex-A53 processor supports the Advanced SIMD and Scalar Floating-point instructions in the A64 instruction set, and the Advanced SIMD and VFP instructions in the A32 and T32 instruction sets.

The ARMv8 architecture eliminates the concept of version numbers for Advanced SIMD and Floating-point in the AArch64 execution state.

4.1.4.3 Generic Interrupt Controller (GIC)

The Generic Interrupt Controller (GIC) supports and manages interrupts in the cluster. The GIC provides registers for managing:

- Interrupt sources
- Interrupt behavior
- Interrupt routing to one or more cores

The Cortex-A53 processor implements the GIC CPU interface as described in the Generic Interrupt Controller (GICv4) architecture. This interfaces with an external GICv3 or GICv4 interrupt distributor component within the system. The GICv4 architecture supports:

- Two security states
- Interrupt virtualization
- Software-generated Interrupts (SGIs)
- Message Based Interrupts
- System register access
- Memory-mapped register access
- Interrupt masking and prioritization

- Cluster environments
- Wake-up events in power management environments

The GIC includes interrupt grouping functionality that supports:

- Signaling interrupt groups to the target core using either the IRQ or the FIQ exception request, based on software configuration
- A unified scheme for handling the priority of Group 0 and Group 1 interrupts.

4.1.4.4 L1 Cache

The L1 instruction memory system has the following features:

- 32KB Instruction Cache
  - Instruction side cache line length of 64 bytes
  - 2-way set associative L1 Instruction cache
  - 128-bit read interface to the L2 memory system
- 32KB Data Cache
  - Data side cache line length of 64 bytes
  - 4-way set associative L1 Data cache
  - 256-bit write interface to the L2 memory system
  - 128-bit read interface to the L2 memory system
  - Read buffer that services the Data Cache Unit (DCU), the Instruction Fetch Unit (IFU) and the TLB
  - 64-bit read path from the data L1 memory system to the datapath
  - 128-bit write path from the datapath to the L1 memory system
  - Support for three outstanding data cache misses
  - Merging store buffer capability. This handles writes to:
    - Device memory
    - Normal Cacheable memory
    - Normal Non-cacheable memory
    - Data side prefetch engine

4.1.4.5 L2 Cache

The L2 cache consists of an integrated Snoop Control Unit (SCU), connecting four cores within the A53 cluster. The SCU also has duplicate copies of the L1 Data cache tags for coherency support. The L2 memory system interfaces to the external memory system through an AMBA 128-bit bus. The tightly-coupled L2 cache includes the following features:

- 1MB shared cache size
- AMBA 4 ACE Interface
- Fixed line length of 64 bytes

- Physically indexed and tagged cache
- 16-way set-associative cache structure
- No ACP support
- ECC/parity support

4.2 ARM Cortex M4 Platform (M4)

4.2.1 Overview

This block details the ARM Cortex-M4 core and platform. The Cortex-M4 implements the ARMv7- ME instruction set architecture (ISA) and adds significant capabilities with DSP and SIMD extensions. The ARM Cortex-M4 core provides additional general processing capability to the SoC with lower power and fast interrupt response time.

The Cortex-M4 also includes a single-precision floating-point unit (FPU) and two 32-bit system bus interfaces. The Cortex-M4 implementation includes two tightly coupled local memories, two cache memories connected to bus interfaces, 64-bit system bus interconnect, and supports a 32-byte cache line size.

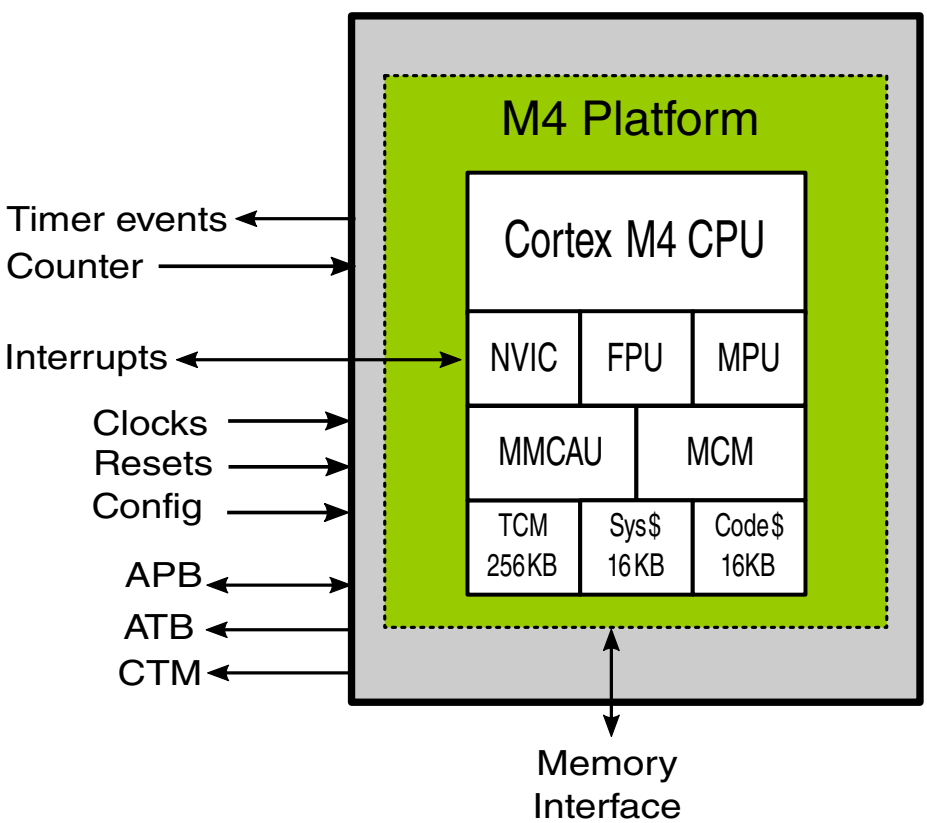


Figure 4-2. Cortex-M4 Block Diagram

4.2.1.1 Features

The features of the Cortex-M4 platform are detailed below:

- 1x Cortex-M4 processor
- AHB LMEM (Local Memory Controller) including controllers for TCM and cache memories
- 256 KB TCM (128 KB TCMU, 128 KB TCML)
- 16 KB Code Bus Cache
- 16 KB System Bus Cache
- ECC for TCM memories and parity for code and system caches
- Integrated Nested Vector Interrupt Controller (NVIC)
- Wakeup Interrupt Controller (WIC)
- FPU (Floating Point Unit)
- Core MPU (Memory Protection Unit)
- Support for exclusive access on the system bus
- MMCAU (Crypto Acceleration Unit)
- MCM (Miscellaneous Control Module)

4.2.2 Configuration

The configuration of the bus, core, and memory are detailed in the features and sections below.

Table 4-2. Cortex-M4 Configuration

Parameter	Description	Setting
NUM_IRQ	Number of Interrupts	128
LVL_WIDTH (translates to 2^n levels)	Number of interrupt priority bits	4
MPU_PRESENT	Exclude (0) or include (1) optional ARM MPU	1
DEBUG_LVL	Level of debug support	3
TRACE_LVL	Level of trace support 0 = No Trace 1 = ITM and DWT 2 = ITM, DWT, and ETM 3 = ITM, DWT, ETM, and HTM	2
RESET_ALL_REGS	Select whether all registers are reset (1)	1
JTAG_PRESENT	SWJ-DP (1) SW-DP (0)	1
CLKGATE_PRESENT	Disable (0) or enable (1) instantiation of architectural clock gates at all levels	1

Table continues on the next page...

Table 4-2. Cortex-M4 Configuration (continued)

WIC_PRESENT	Exclude (0) or include (1) optional ARM provided WIC	1
WIC_LINES	Select number of interrupts and/or events that WIC is sensitive to	NUM_IRQ+3
FPU_PRESENT	Exclude (0) or include (1) Floating Point Unit (FPU)	1
BB_PRESENT	Exclude (0) or include (1) Bit Banding logic	0
CONST_AHB_CTRL	Specifies whether the external AHB-Lite buses maintain control information during wait stated transfers	0

4.2.3 Performance

This section will discuss power and clocking performance of the Cortex-M4 Platform.

4.2.3.1 Power

There are several power states supported by the Cortex-M4. Supported primary core states are listed below:

- **Run** - Normal run state. CM4 Platform, TCM and cache memories on nominal voltage.
- **Sleep** – Wait power state. Depending on power state, CM4 Platform, TCM and cache memories can either be in nominal or low-voltage. State recovery through NVIC. NVIC clock is still running in this state.
- **Deep Sleep** – Stop power state. Depending on power state, CM4 Platform, TCM and cache memories can either be in nominal or low-voltage. State recovery through AWIC. Clocks can be completely stopped in this state.

4.2.3.2 Clocking

The M4 platform is provided a main processor clock that supplies the component clocks to the cluster components. The maximum frequency targets are specified in the chip datasheet. Please see the datasheet for more information. The clocks are described in the table below:

Table 4-3. CM4 Clocks

Clock Signal	Clock Name	Target Frequency	Description
Core gated clock	cpu_hclk	266 MHz	Cortex-M4 core clock.
Core Free-Running Clock	cpu_fclk	266 MHz	CM4 NVIC and timer clock

Table continues on the next page...

Table 4-3. CM4 Clocks (continued)

TCM Controller Clock	tcmc_hclk	266 MHz	CM4 platform TCM controller clock
Platform Clock	hclk	266 MHz	CM4 platform AXBS fabric and bus masters. Synchronous to the core clock.
Bus Clock	ipg_clk	133 MHz	Clock for bus slaves and peripherals. Synchronous to the system clock.

4.2.4 Platform sub-blocks

The sections below discuss the high-level overview of the ARM® Cortex®-M4 Platform components.

4.2.4.1 ARM Cortex-M4 Processor

The Cortex-M4 processor is a low-power processor that features low gate count, low interrupt latency, and low-cost debug. The Cortex-M4 includes floating point arithmetic functionality. The processor is intended for deeply embedded applications that require fast interrupt response features.

The ARM Cortex-M4 processor consists of:

- A processor core
- A Nested Vectored Interrupt Controller (NVIC) closely integrated with the processor core to achieve low-latency interrupt processing
- Multiple high-performance bus interfaces
- A low-cost debug solution with the optional ability to:
  - Implement breakpoints and code patches
  - Implement watchpoints, tracing, and system profiling
  - Support printf() style debugging
  - Bridge to a Trace Port Analyzer (TPA)
- Memory Protection Unit (MPU)
- Floating Point Unit (FPU)
- Miscellaneous Control Module (MCM)
- The Cortex-M4 implements the ARM the ARMv7-M architecture.

4.2.4.2 Nested Vectored Interrupt Controller (NVIC)

The M4 platform includes a Nested Vectored Interrupt Controller (NVIC) that is closely integrated with the processor core to achieve low-latency interrupt processing. The platform supports 16 priority levels for interrupts. The NVIC IPR registers will define 4 bits per IRQ.

NOTE

M4_NMI is Non-maskable Interrupt of M4 with fixed priority. It is available by selecting ALT1 mux_mode for pad GPIO1_IO05. Refer to ARM official document *Cortex™-M4 Devices Generic User Guide* for more details.

4.2.4.3 Floating Point Unit (FPU)

The Cortex-M4 platform supports Floating Point Unit (FPU). The FPU implements the single precision variant of the ARMv7-M Floating-Point Extension (FPv4-SP), which supports single-precision add, subtract, multiply, divide, multiply and accumulate, and square root operations. FPU also provides conversions between fixed-point and floating-point data formats, and floating-point constant instructions.

The FPU provides floating-point computation functionality that is compliant with the ANSI/IEEE Std 754-2008, IEEE Standard for Binary Floating-Point Arithmetic, referred to as the IEEE 754 standard. The FPU contains 32 single-precision extension registers, which you can also access as 16 doubleword registers for load, store, and move operations.

4.2.4.4 Memory Protection Unit (MPU)

The Cortex-M4 platform supports Memory Protection Unit (MPU). The MPU enforces privilege rules, separates processes, and enforces access rules to memory, and supports the standard ARMv7 Protected Memory System Architecture model. The MPU provides full support for:

- Protection regions
- Overlapping protection regions, with ascending region priority:
  - 7 = highest priority
  - 0 = lowest priority
- Access permissions
- Exporting memory attributes to the system

MPU mismatches and permission violations invoke the programmable-priority MemManage fault handler. You can use the MPU to:

- Enforce privilege rules
- Separate processes
- Enforce access rules

4.2.4.5 Tightly-Coupled Memory (TCM)

The Cortex-M4 uses Tightly-Coupled Memory (TCM) to perform low-latency memory operations including speculative read accesses. The TCM is split between upper and lower regions with each TCM interface operating independently of each other.

The Cortex-M4 supports 32-bit ECC error detection and correction for the TCM memories. TCM ECC can indicate to the processor that an access must be retried to return the corrected data.

Because AHB-Lite does not support write data strobes when accessing AHB-Lite slaves from an AXI master, care must be taken not to generate transactions that have partial strobes. Make sure to not have unaligned accessing to TCM from an AXI master. For example, when writing data to TCM from A53, ensure every write strobe address is 64bit aligned. When the MMU is enabled, the TCM memory range must have the MT_DEVICE_NGNRNE type attribute set. This will avoid A53 sparse writes to the TCM memory region.

4.2.5 Local Memory Controller (LMEM)

The Local Memory Controller provides the Arm®Cortex-M4™ processor with tightly-coupled processor-local memories and bus paths to all slave memory spaces.

4.2.5.1 LMEM Block Diagram

The Cortex-M4 processor has a modified 32-bit Harvard bus architecture. Using a 32-bit address space, low-order addresses (0x0000_0000 through 0x1FFF_FFFF) use the Processor Code (PC) bus, and high-order addresses (0x2000_0000 through 0xFFFF_FFFF) use the Processor System (PS) bus. As the bus names imply, normal operation has code accesses on the PC bus and data accesses on the PS bus.

This device has been augmented with tightly-coupled memories for the PC and PS buses. The memories include RAMs and caches. These local memories provide zero wait state access to RAM and cacheable address spaces.

The local memory controller includes four memory controllers and their attached memories:

- SRAM lower (SRAM_L) controller via the PC bus
- SRAM upper (SRAM_U) controller via the PS bus
- Cache memory controller via the PC bus
- Cache memory controller via the PS bus

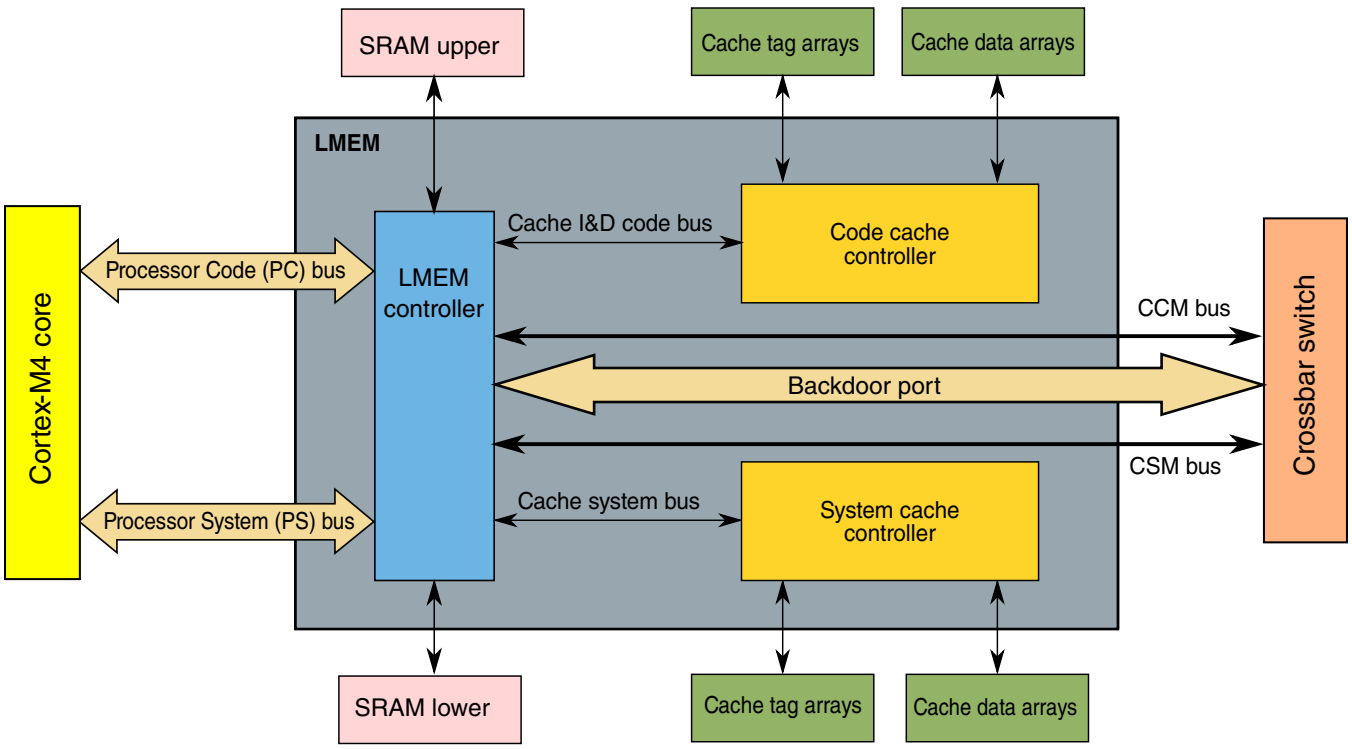


Figure 4-3. Local memory controller block diagram

NOTE

The SRAM and cache controllers reside within the LMEM, but the single-port synchronous RAM arrays used by these controllers are external.

The LMEM contains address decode logic for the PC and PS buses. This logic routes the core's accesses to the various system resources. The address spaces are device-specific and are specified in the device's Chip Configuration chapter.

4.2.5.2 Cache features

A cache is a block of high-speed memory locations containing address information (commonly known as a tag) and the associated data. The purpose is to decrease the average time of a memory access. Caches operate on two principles of locality:

- Spatial locality — An access to one location is likely to be followed by accesses from adjacent locations (for example, sequential instruction execution or usage of a data structure).
- Temporal locality — An access to an area of memory is likely to be repeated within a short time period (for example, execution of a code loop).

To minimize the quantity of control information stored, the spatial locality property is used to group several locations together under the same tag. This logical block is commonly known as a cache line.

When data is loaded into a cache, access times for subsequent loads and stores are reduced, resulting in overall performance benefits. An access to information already in a cache is known as a cache hit, and other accesses are called cache misses.

Normally, caches are self-managing, with the updates occurring automatically. Whenever the processor wants to access a cacheable location, the cache is checked. If the access is a cache hit, the access occurs immediately. Otherwise, a location is allocated and the cache line is loaded from memory. Different cache topologies and access policies are possible. However, they must comply with the memory coherency model of the underlying architecture.

Caches introduce a number of potential problems, mainly because of:

- memory accesses occurring at times other than when the programmer would normally expect them,
- the existence of multiple physical locations where a data item can be held.

The local memory controller supports three modes of operation:

1. Write-through — access to address spaces with this cache mode are cacheable.
    - A read miss on the input bus causes a line read on the output bus of a 32-byte-aligned memory address containing the desired address. This miss data is loaded into the cache and is marked as valid and not modified.
    - A write-through read hit to a valid cache location returns data from the cache with no output bus access.
    - A write-through write miss bypasses the cache and writes to the output bus (no allocate on write miss policy for write-through mode spaces).
    - A write-through write hit updates the cache hit data and writes to the output bus.
2. Write-back — access to address spaces with this cache mode are cacheable.
    - A write-back read miss on the input bus will cause a line read on the output bus of a 32-byte-aligned memory address containing the desired address. This miss data is loaded into the cache and marked as valid and not modified.

- A write-back read hit to a valid cache location will return data from the cache with no output bus access.
- A write-back write miss will do a "read-to-write" (allocate on write miss policy for write-back mode spaces). A line read on the output bus of a 16 byte aligned memory address containing the desired write address is performed. This miss data is loaded into the cache and marked as valid and modified; and the write data will then update the appropriate cache data locations.

3. Non-cacheable — access to address spaces with this cache mode are not cacheable. These accesses bypass the cache and access the output bus.

4.2.5.3 LMEM Function

The LMEM receives the following requests:

- Core master bus requests on the Processor Code (PC) bus,
- Core master bus requests on the Processor Space (PS) bus, and
- SRAM controller requests from all other bus masters on the backdoor port.

The LMEM address decode logic routes these accesses and also provides any crossbar switch slave target logic. Finally, the Local Memory controller provides the needed MPU connections for checking all SRAM controller and cacheable accesses.

The programming model for the Code and System Caches is accessed via the core's Private Peripheral Bus (PPB).

4.2.5.3.1 Processor Code accesses

Processor Code accesses are routed to the SRAM_L if they are mapped to that space. All other PC accesses are routed to the Code Cache Memory Controller. This controller then processes the cacheable accesses as needed, while bypassing the non-cacheable, cache write-through, cache miss, and cache maintenance accesses to the CCM bus and the crossbar switch using the Master0 port.

4.2.5.3.2 Processor Space accesses

Processor Space accesses are routed to the SRAM_U if they are mapped to that space. All other PS accesses are routed to the PS Cache Memory Controller. This controller then processes the cacheable accesses as needed, while bypassing the non-cacheable, cache write-through, cache miss, and cache maintenance accesses to the CCM bus and the crossbar switch using the Master1 port.

4.2.5.3.3 Backdoor port accesses

All LMEM backdoor port accesses are for the SRAM controller. These accesses go to the SRAM_L or the SRAM_U depending on their specific address.

4.2.5.3.4 SRAM Function

4.2.5.3.4.1 SRAM Configuration

The figure below shows how the SRAM controller is configured.

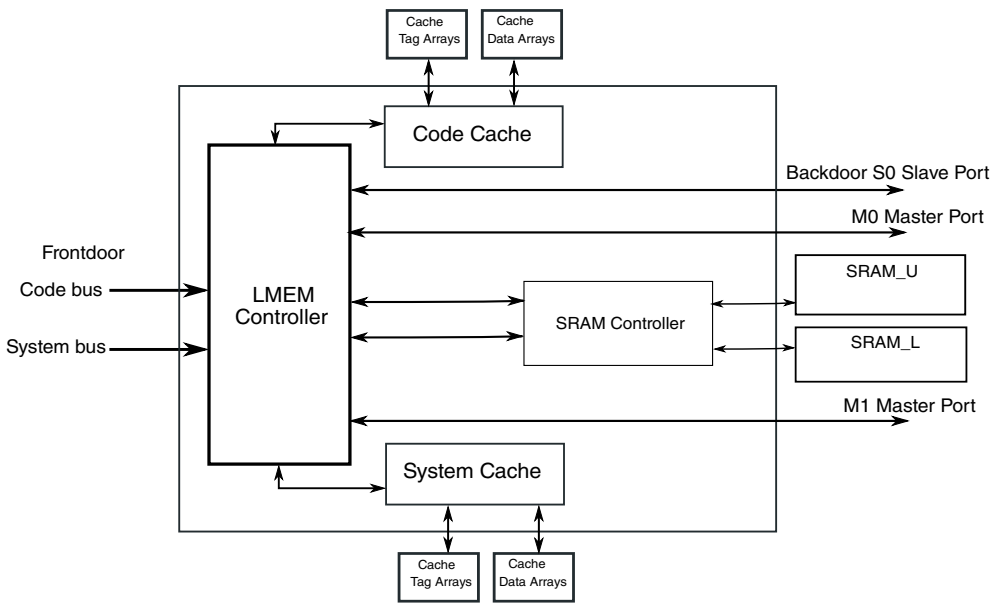


Figure 4-4. SRAM Configuration

4.2.5.3.4.2 SRAM Arrays

The on-chip SRAM is split into two logical arrays, SRAM_L and SRAM_U.

From equal-sized memories, valid address ranges for SRAM_L and SRAM_U are then defined as:

- SRAM_U = 0x2000_0000 - (0x2000_0000 + SRAM_size/2)

4.2.5.3.4.3 SRAM accesses

The SRAM is split into two logical arrays that are 64-bits wide:

- SRAM_L — Accessible by the code bus of the Cortex-M4 core and by the backdoor port.
- SRAM_U — Accessible by the system bus of the Cortex-M4 core and by the backdoor port.

The backdoor port makes the SRAM accessible to the non-core bus masters (such as DMA).

The figure below illustrates the SRAM accesses within the device.

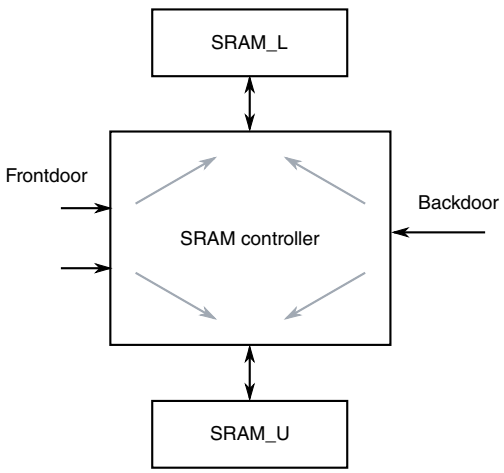


Figure 4-5. SRAM access diagram

The following simultaneous accesses can be made to different logical halves of the SRAM:

- Core code and core system
- Core code and non-core master
- Core system and non-core master

NOTE

Two non-core masters cannot access SRAM simultaneously. The required arbitration and serialization is provided by the crossbar switch. The SRAM_{L,U} arbitration is controlled by the SRAM controller based on the configuration bits in the MCM module.

4.2.5.3.5 Cache Function

The caches on this device are structured as follows. Both caches have a 2-way set-associative cache structure with a total size of 32 KBytes. The caches have a 32-bit address, 64-bit data paths and a 32-byte line size. The cache tags and data storage use single-port, synchronous RAMs.

For these 16-KByte caches, each cache TAG function uses two 256 x 22-bit RAM arrays and the cache DATA function uses two 1024 x 32-bit RAM arrays. The cache TAG entries store 20 bits of upper address as well as a modified and valid bit per cache line. The cache DATA entries store eight byes of code or data.

All normal cache accesses use physical addresses. This leads to the following cache address use:

CACHE - 16 KByte size = (256 sets) x (32-byte lines) x (2-way set associative)

TAG:

DATA

- address[31:13] not used
- address[12:5] used to select one of 256 sets
- address[4:2] used to select one of eight 32-bit words within a set
- address[1:0] used to select the byte within the 32-bit word

4.2.5.3.6 Cache Control

The Code and System Caches are disabled at reset. Cache tag and data arrays are not cleared at reset. Therefore, to enable the caches, cache commands must be done to clear and initialize the required tag array bits and to configure and enable the caches.

4.2.5.3.6.1 Cache set commands

The cache set commands may operate on:

- all of way 0,
- all of way 1, or
- all of both ways (complete cache).

Cache set commands are initiated using the upper bits in the CCR register. Cache set commands perform their operation on the cache independent of the cache enable bit, CCR[ENCACHE].

A cache set command is initiated by setting the CCR[GO] bit. This bit also acts as a busy bit for set commands. It stays set while the command is active and is cleared by the hardware when the set command completes.

Supported cache set commands are given in the table below. Set commands work as follows:

- Invalidate – Unconditionally clear valid and modify bits of a cache entry.

- Push – Push a cache entry if it is valid and modified, then clear the modify bit. If entry not valid or not modified, leave as is.
- Clear – Push a cache entry if it is valid and modified, then clear the valid and modify bits. If entry not valid or not modified, clear the valid bit.

Table 4-4. Cache Set Commands

CCR[27:24]				Command
PUSH W1	INVW1	PUSH W0	INVW0	
0	0	0	0	NOP
0	0	0	1	Invalidate all way 0
0	0	1	0	Push all way 0
0	0	1	1	Clear all way 0
0	1	0	0	Invalidate all way 1
0	1	0	1	Invalidate all way 1; invalidate all way 0 (invalidate cache)
0	1	1	0	Invalidate all way 1; push all way 0
0	1	1	1	Invalidate all way 1; clear all way 0
1	0	0	0	Push all way 1
1	0	0	1	Push all way 1; invalidate all way 0
1	0	1	0	Push all way 1; push all way 0 (push cache)
1	0	1	1	Push all way 1; clear all way 0
1	1	0	0	Clear all way 1
1	1	0	1	Clear all way 1; invalidate all way 0
1	1	1	0	Clear all way 1; push all way 0
1	1	1	1	Clear all way 1; clear all way 0 (clear cache)

After a reset, complete an invalidate cache command before using the cache. It is possible to combine the cache invalidate command with the cache enable. That is, setting CCR to 0x8500_0003 will invalidate the cache and enable the cache and write buffer.

4.2.5.3.6.2 Cache line commands

Cache line commands operate on a single line in the cache at a time. Cache line commands can be performed using a physical or cache address.

- A cache address consists of a set address and a way select. The line command acts on the specified cache line.
- Cache line commands with physical addresses first search both ways of the cache set specified by bits [11:4] of the physical address. If they hit, the commands perform their action on the hit way.

Cache line commands are specified using the upper bits in the CLCR register. Cache line commands perform their operation on the cache independent of the cache enable bit (CCR[ENCACHE]). Using a cache address, the command can be completely specified using the CLCR register. Using a physical address, the command must also use the CSAR register to specify the physical address.

A line cache command is initiated by setting the line command go bit (CLCR[LGO] or CSAR[LGO]). This bit also acts a a busy bit for line commands. It stays set while the command is active and is cleared by the hardware when the command completes.

The CLCR[27:24] bits select the line command as follows:

Table 4-5. Cache Line Commands

CLCR[27:24]			Command
LACC	LADSEL	LCMD	
0	0	00	Search by cache address and way
0	0	01	Invalidate by cache address and way
0	0	10	Push by cache address and way
0	0	11	Clear by cache address and way
0	1	00	Search by physical address
0	1	01	Invalidate by physical address
0	1	10	Push by physical address
0	1	11	Clear by physical address
1	0	00	Write by cache address and way
1	0	01	Reserved, NOP
1	0	10	Reserved, NOP
1	0	11	Reserved, NOP
1	1	*xx*	Reserved, NOP

4.2.5.3.6.2.1 Executing a series of line commands using cache addresses

A series of line commands with incremental cache addresses can be performed by just writing to the CLCR.

- Place the command in CLCR[27:24],
- Set the way (CLCR[WSEL]) and tag/data (CLCR[TDSEL]) controls as needed,
- Place the cache address in CLCR[CACHEADDR], and
- Set the line command go bit (CLCR[LGO]).

When one line command completes, initiate the next command by following these steps:

- Increment the cache address (at bit 2 to step through data or at bit 4 to step through lines), and
- Set the line command go bit (CLCR[LGO]).

4.2.5.3.6.2.2 Executing a series of line commands using physical addresses

Perform a series of line commands with incremental physical addresses using the following steps:

- Write to the CLCR.
  - Place the command in CLCR[27:24]
  - Set the tag/data (CLCR[TDSEL]) control
- Place the physical address in CSAR[PHYADDR] and set the line command go bit (CSAR[LGO]).

When one line command completes, initiate the next command by following these steps:

- Increment the physical address (at bit 2 to step through data or at bit 4 to step through lines), and
- Set the line command go bit (CSAR[LGO]).

The line command go bit is shared between the CLCR and CSAR registers, so that the above steps can be completed in a single write to the CSAR register.

4.2.5.3.6.2.3 Line command results

At completion of a line command, the CLCR register contains information on the initial state of the line targeted by the command. For line commands with cache addresses, this information is read before the line command action is performed from the targeted cache line. For line commands with physical addresses, this information is read on a hit before the line command action is performed from the hit cache line or has initial valid bit cleared if the command misses. In general, if the valid indicator (CLCR[LCIVB] is cleared, the targeted line was invalid at the start of the line command and no line operation was performed.

Table 4-6. Line command results

CLCR[22:20]			For cache address commands	For physical address commands
LCWAY	LCIMB	LCIVB		
0	0	0	Way 0 line was invalid	No hit
0	0	1	Way 0 valid, not modified	Way 0 valid, not modified
0	1	0	Way 0 line was invalid	No hit
0	1	1	Way 0 valid and modified	Way 0 valid and modified

Table continues on the next page...

Table 4-6. Line command results (continued)

CLCR[22:20]			For cache address commands	For physical address commands
LCWAY	LCIMB	LCIVB		
1	0	0	Way 1 line was invalid	No hit
1	0	1	Way 1 valid, not modified	Way 1 valid, not modified
1	1	0	Way 1 line was invalid	No hit
1	1	1	Way 1 valid and modified	Way 1 valid and modified

At completion of a line command other than a write, the CCVR (Cache R/W Value Register) contains information on the initial state of the line tag or data targeted by the command. For line commands, CLCR[TDSEL] selects between tag and data. If the line command used a physical address and missed, the data is don't care. For write commands, the CCVR holds the write data.

4.2.6 Miscellaneous Control Module (MCM)

The Miscellaneous Control Module (MCM) provides a myriad of miscellaneous control functions.

4.2.6.1 MCM features

The MCM includes the following features:

- Program-visible information on the platform configuration and revision

4.2.6.2 MCM Interrupts

The MCM generates the following interrupt requests:

- Normal interrupt

4.2.6.2.1 Normal interrupt

The MCM's normal interrupt is generated if any of the following is true:

- ISCR[ETBI] is set, when
  - The ETB counter is enabled, ETBCC[CNTEN] = 1
  - The ETB count expires
  - The response to counter expiration is a normal interrupt, ETBCC[RSPT] = 01

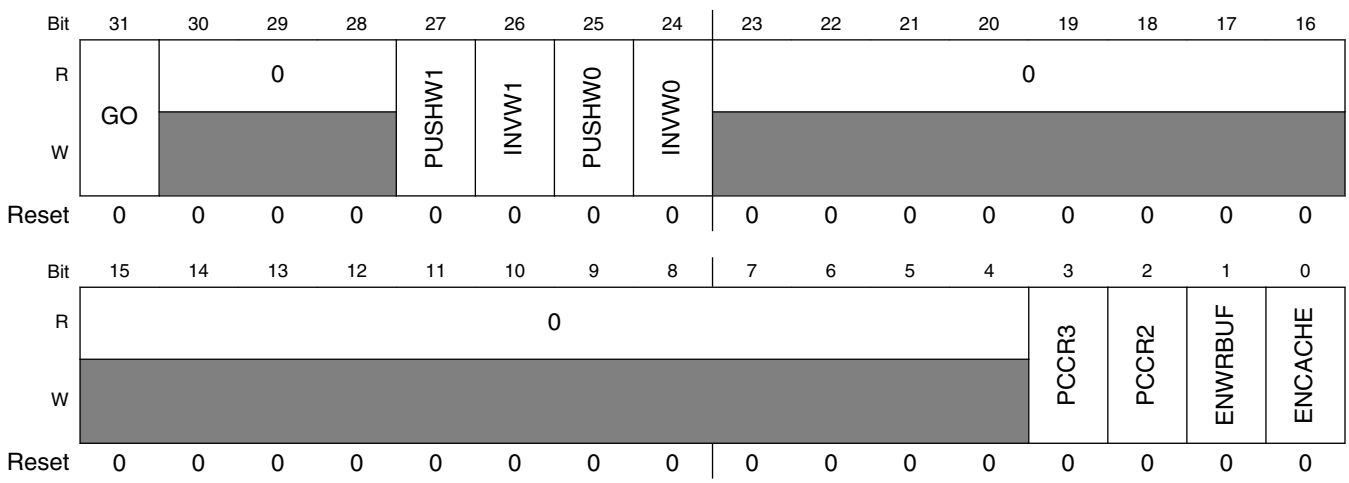
4.2.7 LMEM Memory Map/Register Definition

LMEM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
E008_2000	Cache control register (LMEM_PCCCR)	32	R/W	0000_0000h	4.2.7.1/99
E008_2004	Cache line control register (LMEM_PCCLCR)	32	R/W	0000_0000h	4.2.7.2/100
E008_2008	Cache search address register (LMEM_PCCSAR)	32	R/W	0000_0000h	4.2.7.3/103
E008_200C	Cache read/write value register (LMEM_PCCCVR)	32	R/W	0000_0000h	4.2.7.4/104
E008_2800	Cache control register (LMEM_PSCCR)	32	R/W	0000_0000h	4.2.7.5/104
E008_2804	Cache line control register (LMEM_PSCLCR)	32	R/W	0000_0000h	4.2.7.6/106
E008_2808	Cache search address register (LMEM_PSCSAR)	32	R/W	0000_0000h	4.2.7.7/108
E008_280C	Cache read/write value register (LMEM_PSCCVR)	32	R/W	0000_0000h	4.2.7.8/109

4.2.7.1 Cache control register (LMEM_PCCCR)

Address: E008_2000h base + 0h offset = E008_2000h



LMEM_PCCCR field descriptions

Field	Description
31 GO	Initiate Cache Command Setting this bit initiates the cache command indicated by bits 27-24. Reading this bit indicates if a command is active **NOTE:** This bit stays set until the command completes. Writing zero has no effect. 0 Write: no effect. Read: no cache command active. 1 Write: initiate command indicated by bits 27-24. Read: cache command active.

Table continues on the next page...

LMEM_PCCCR field descriptions (continued)

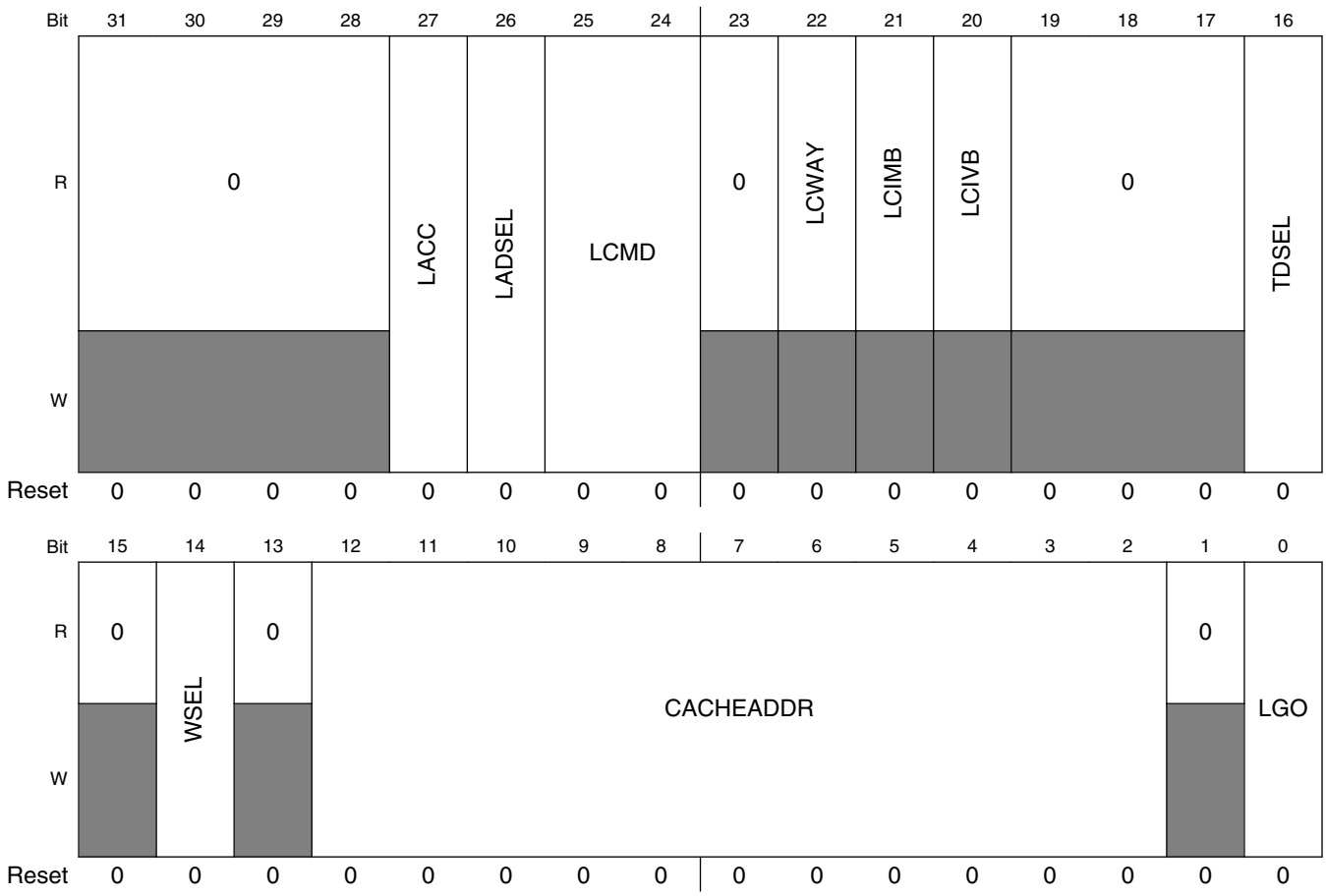
Field	Description
30–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27 PUSHW1	Push Way 1 0 No operation 1 When setting the GO bit, push all modified lines in way 1
26 INVW1	Invalidate Way 1 **NOTE:** If the PUSHW1 and INVW1 bits are set, then after setting the GO bit, push all modified lines in way 1 and invalidate all lines in way 1 (clear way 1). 0 No operation 1 When setting the GO bit, invalidate all lines in way 1
25 PUSHW0	Push Way 0 0 No operation 1 When setting the GO bit, push all modified lines in way 0
24 INVW0	Invalidate Way 0 **NOTE:** If the PUSHW0 and INVW0 bits are set, then after setting the GO bit, push all modified lines in way 0 and invalidate all lines in way 0 (clear way 0). 0 No operation 1 When setting the GO bit, invalidate all lines in way 0.
23–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 PCCR3	Forces no allocation on cache misses (must also have ACCR2 asserted)
2 PCCR2	Forces all cacheable spaces to write through
1 ENWRBUF	Enable Write Buffer 0 Write buffer disabled 1 Write buffer enabled
0 ENCACHE	Cache enable 0 Cache disabled 1 Cache enabled

4.2.7.2 Cache line control register (LMEM_PCCLCR)

This register defines specific line-sized cache operations to be performed using a specific cache line address or a physical address.

If a physical address is specified, both ways of the cache are searched, and the command is only performed on the way which hits.

Address: E008_2000h base + 4h offset = E008_2004h



LMEM_PCCLCR field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27 LACC	Line access type 0 Read 1 Write
26 LADSEL	Line Address Select When using the cache address, the way must also be specified in CLCR[WSEL]. When using the physical address, both ways are searched and the command is performed only if a hit. 0 Cache address 1 Physical address
25–24 LCMD	Line Command 00 Search and read or write 01 Invalidate 10 Push 11 Clear

Table continues on the next page...

LMEM_PCCLCR field descriptions (continued)

Field	Description
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 LCWAY	Line Command Way Indicates the way used by the line command.
21 LCIMB	Line Command Initial Modified Bit If command used cache address and way, then this bit shows the initial state of the modified bit If command used physical address and a hit, then this bit shows the initial state of the modified bit. If a miss, this bit reads zero.
20 LCIVB	Line Command Initial Valid Bit If command used cache address and way, then this bit shows the initial state of the valid bit If command used physical address and a hit, then this bit shows the initial state of the valid bit. If a miss, this bit reads zero.
19–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 TDSEL	Tag/Data Select Selects tag or data for search and read or write commands. 0 Data 1 Tag
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 WSEL	Way select Selects the way for line commands. 0 Way 0 1 Way 1
13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–2 CACHEADDR	Cache address CLCR[11:4] bits are used to access the tag arrays CLCR[11:2] bits are used to access the data arrays
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 LGO	Initiate Cache Line Command Setting this bit initiates the cache line command indicated by bits 27-24. Reading this bit indicates if a line command is active **NOTE:** This bit stays set until the command completes. Writing zero has no effect. **NOTE:** This bit is shared with CSAR[LGO] 0 Write: no effect. Read: no line command active. 1 Write: initiate line command indicated by bits 27-24. Read: line command active.

4.2.7.3 Cache search address register (LMEM_PCCSAR)

The CSAR register is used to define the explicit cache address or the physical address for line-sized commands specified in the CLCR[LADSEL] bit.

Address: E008_2000h base + 8h offset = E008_2008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PHYADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PHYADDR														0	LGO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

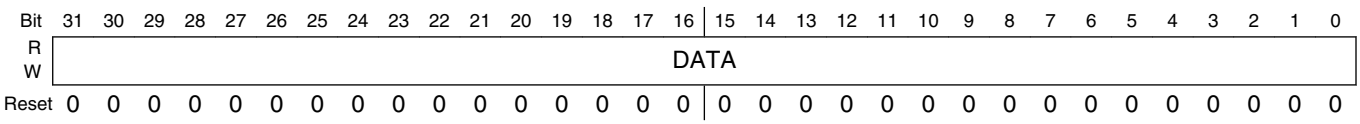
LMEM_PCCSAR field descriptions

Field	Description
31–2 PHYADDR	Physical Address PHYADDR represents bits [31:2] of the system address. CSAR[31:12] bits are used for tag compare CSAR[11:4] bits are used to access the tag arrays CSAR[11:2] bits are used to access the data arrays
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 LGO	Initiate Cache Line Command Setting this bit initiates the cache line command indicated by bits 27-24. Reading this bit indicates if a line command is active **NOTE:** This bit stays set until the command completes. Writing zero has no effect. **NOTE:** This bit is shared with CLCR[LGO] 0 Write: no effect. Read: no line command active. 1 Write: initiate line command indicated by bits CLCR[27:24]. Read: line command active.

4.2.7.4 Cache read/write value register (LMEM_PCCCVR)

The CCVR register is used to source write data or return read data for the commands specified in the CLCR register.

Address: E008_2000h base + Ch offset = E008_200Ch

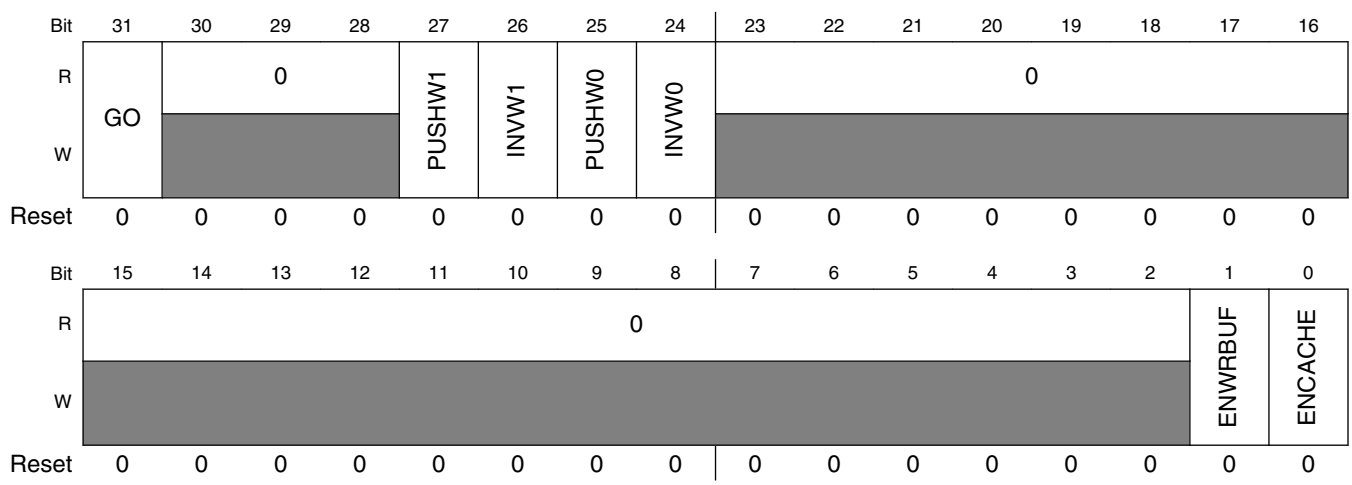


LMEM_PCCCVR field descriptions

Field	Description
DATA	Cache read/write Data For tag search, read or write: • CCVR[31:12] bits are used for tag array R/W value • CCVR[11:4] bits are used for tag set address on reads; unused on writes • CCVR[3:2] bits are reserved For data search, read or write: • CCVR[31:0] bits are used for data array R/W value

4.2.7.5 Cache control register (LMEM_PSCCR)

Address: E008_2000h base + 800h offset = E008_2800h



LMEM_PSCCR field descriptions

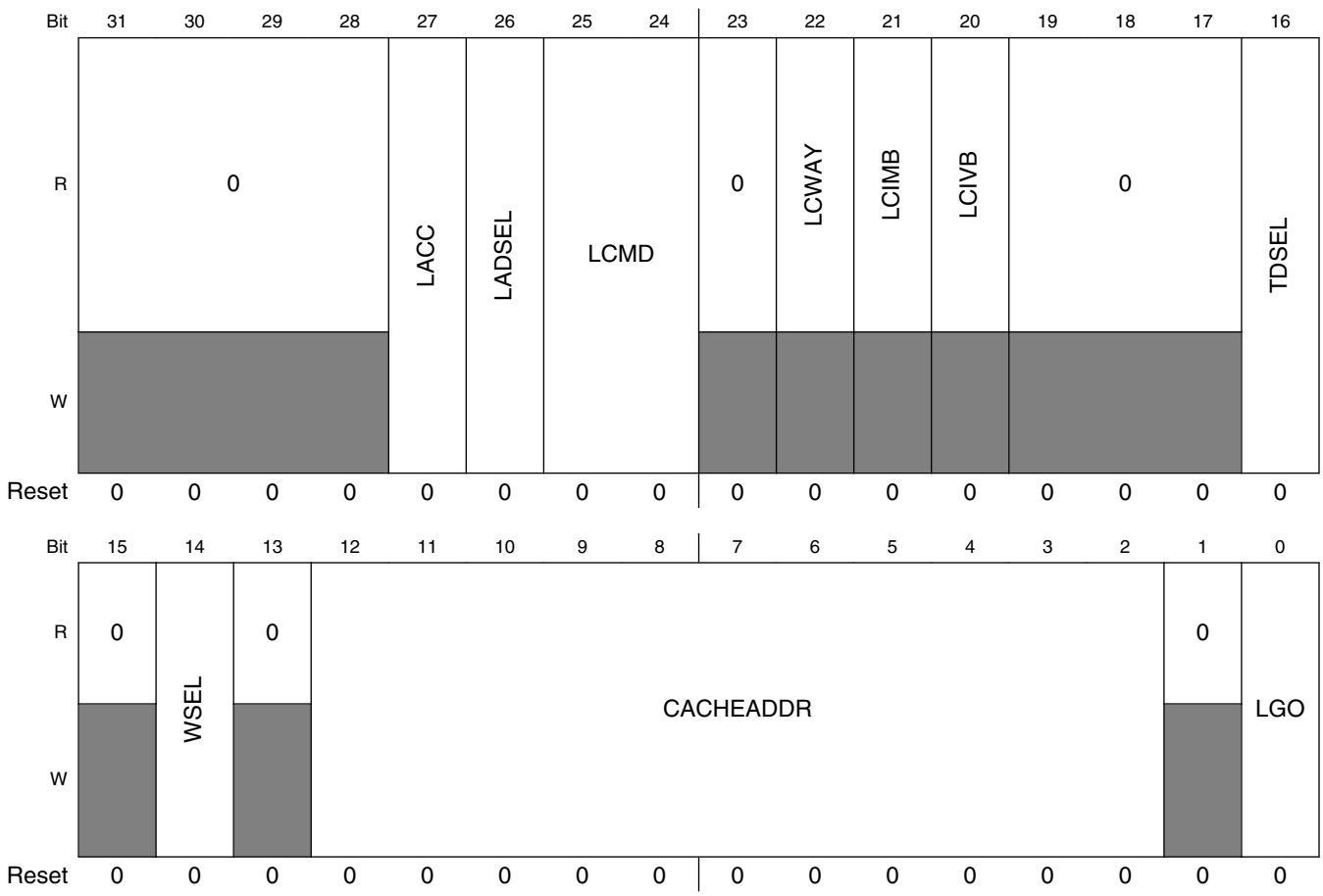
Field	Description
31 GO	Initiate Cache Command Setting this bit initiates the cache command indicated by bits 27-24. Reading this bit indicates if a command is active **NOTE:** This bit stays set until the command completes. Writing zero has no effect. 0 Write: no effect. Read: no cache command active. 1 Write: initiate command indicated by bits 27-24. Read: cache command active.
30–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27 PUSHW1	Push Way 1 0 No operation 1 When setting the GO bit, push all modified lines in way 1
26 INVW1	Invalidate Way 1 **NOTE:** If the PUSHW1 and INVW1 bits are set, then after setting the GO bit, push all modified lines in way 1 and invalidate all lines in way 1 (clear way 1). 0 No operation 1 When setting the GO bit, invalidate all lines in way 1
25 PUSHW0	Push Way 0 0 No operation 1 When setting the GO bit, push all modified lines in way 0
24 INVW0	Invalidate Way 0 **NOTE:** If the PUSHW0 and INVW0 bits are set, then after setting the GO bit, push all modified lines in way 0 and invalidate all lines in way 0 (clear way 0). 0 No operation 1 When setting the GO bit, invalidate all lines in way 0.
23–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 ENWRBUF	Enable Write Buffer 0 Write buffer disabled 1 Write buffer enabled
0 ENCACHE	Cache enable 0 Cache disabled 1 Cache enabled

4.2.7.6 Cache line control register (LMEM_PSCLCR)

This register defines specific line-sized cache operations to be performed using a specific cache line address or a physical address.

If a physical address is specified, both ways of the cache are searched, and the command is only performed on the way which hits.

Address: E008_2000h base + 804h offset = E008_2804h



LMEM_PSCLCR field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27 LACC	Line access type 0 Read 1 Write
26 LADSEL	Line Address Select When using the cache address, the way must also be specified in CLCR[WSEL].

Table continues on the next page...

LMEM_PSCLCR field descriptions (continued)

Field	Description
	When using the physical address, both ways are searched and the command is performed only if a hit. 0 Cache address 1 Physical address
25–24 LCMD	Line Command 00 Search and read or write 01 Invalidate 10 Push 11 Clear
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 LCWAY	Line Command Way Indicates the way used by the line command.
21 LCIMB	Line Command Initial Modified Bit If command used cache address and way, then this bit shows the initial state of the modified bit If command used physical address and a hit, then this bit shows the initial state of the modified bit. If a miss, this bit reads zero.
20 LCIVB	Line Command Initial Valid Bit If command used cache address and way, then this bit shows the initial state of the valid bit If command used physical address and a hit, then this bit shows the initial state of the valid bit. If a miss, this bit reads zero.
19–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 TDSEL	Tag/Data Select Selects tag or data for search and read or write commands. 0 Data 1 Tag
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 WSEL	Way select Selects the way for line commands. 0 Way 0 1 Way 1
13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–2 CACHEADDR	Cache address CLCR[11:4] bits are used to access the tag arrays CLCR[11:2] bits are used to access the data arrays
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

LMEM_PSCLCR field descriptions (continued)

Field	Description
0 LGO	Initiate Cache Line Command Setting this bit initiates the cache line command indicated by bits 27-24. Reading this bit indicates if a line command is active **NOTE:** This bit stays set until the command completes. Writing zero has no effect. **NOTE:** This bit is shared with CSAR[LGO] 0 Write: no effect. Read: no line command active. 1 Write: initiate line command indicated by bits 27-24. Read: line command active.

4.2.7.7 Cache search address register (LMEM_PSCSAR)

The CSAR register is used to define the explicit cache address or the physical address for line-sized commands specified in the CLCR[LADSEL] bit.

Address: E008_2000h base + 808h offset = E008_2808h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PHYADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PHYADDR														0	LGO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LMEM_PSCSAR field descriptions

Field	Description
31–2 PHYADDR	Physical Address PHYADDR represents bits [31:2] of the system address. CSAR[31:12] bits are used for tag compare CSAR[11:4] bits are used to access the tag arrays CSAR[11:2] bits are used to access the data arrays
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 LGO	Initiate Cache Line Command Setting this bit initiates the cache line command indicated by bits 27-24. Reading this bit indicates if a line command is active **NOTE:** This bit stays set until the command completes. Writing zero has no effect. **NOTE:** This bit is shared with CLCR[LGO]

Table continues on the next page...

LMEM_PSCSAR field descriptions (continued)

Field	Description
	0 Write: no effect. Read: no line command active. 1 Write: initiate line command indicated by bits CLCR[27:24]. Read: line command active.

4.2.7.8 Cache read/write value register (LMEM_PSCCVR)

The CCVR register is used to source write data or return read data for the commands specified in the CLCR register.

Address: E008_2000h base + 80Ch offset = E008_280Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LMEM_PSCCVR field descriptions

Field	Description
DATA	Cache read/write Data For tag search, read or write: • CCVR[31:12] bits are used for tag array R/W value • CCVR[11:4] bits are used for tag set address on reads; unused on writes • CCVR[3:2] bits are reserved For data search, read or write: • CCVR[31:0] bits are used for data array R/W value

4.2.8 MCM Memory Map/Register Definition

MCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
E008_0008	Crossbar Switch (AXBS) Slave Configuration (MCM_PLASC)	16	R	0002h	4.2.8.1/110
E008_000A	Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC)	16	R	0003h	4.2.8.2/110
E008_000C	Crossbar Switch (AXBS) Control Register (MCM_PLACR)	32	R/W	0000_0000h	4.2.8.3/111
E008_0020	Fault address register (MCM_FADR)	32	R	Undefined	4.2.8.4/111
E008_0024	Fault attributes register (MCM_FATR)	32	R	Undefined	4.2.8.5/112
E008_0028	Fault data register (MCM_FDR)	32	R	Undefined	4.2.8.6/114

4.2.8.1 Crossbar Switch (AXBS) Slave Configuration (MCM_PLASC)

PLASC is a 16-bit read-only register identifying the presence/absence of bus slave connections to the device's crossbar switch.

Address: E008_0000h base + 8h offset = E008_0008h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0								ASC							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

MCM_PLASC field descriptions

Field	Description
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ASC	Each bit in the ASC field indicates whether there is a corresponding connection to the crossbar switch's slave input port. 0 A bus slave connection to AXBS input port *n* is absent 1 A bus slave connection to AXBS input port *n* is present

4.2.8.2 Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC)

PLAMC is a 16-bit read-only register identifying the presence/absence of bus master connections to the device's crossbar switch.

Address: E008_0000h base + Ah offset = E008_000Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0								AMC							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

MCM_PLAMC field descriptions

Field	Description
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
AMC	Each bit in the AMC field indicates whether there is a corresponding connection to the AXBS master input port.

Table continues on the next page...

MCM_PLAMC field descriptions (continued)

Field	Description
	0 A bus master connection to AXBS input port *n* is absent 1 A bus master connection to AXBS input port *n* is present

4.2.8.3 Crossbar Switch (AXBS) Control Register (MCM_PLACR)

The PLACR register selects the arbitration policy for the crossbar masters.

Address: E008_0000h base + Ch offset = E008_000Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																Reserved															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MCM_PLACR field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
Reserved	This field is reserved.

4.2.8.4 Fault address register (MCM_FADR)

When a properly-enabled cache write buffer error interrupt event is detected, the faulting address is captured in the MCM_FADR register. The MCM logic supports capturing a single cache write buffer bus error event; if a subsequent error is detected before the captured error information has been read from the corresponding registers and the MCM_ISCR[CWBER] indicator cleared, the MCM_FATR[BEOVR] flag is set. However, no additional information is captured.

The bits in this register are set by hardware and signaled by the assertion of MCM_ISCR[CWBER]. Attempted writes to this location are terminated with an error.

Address: E008_0000h base + 20h offset = E008_0020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDRESS																															
W																																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

MCM_FADR field descriptions

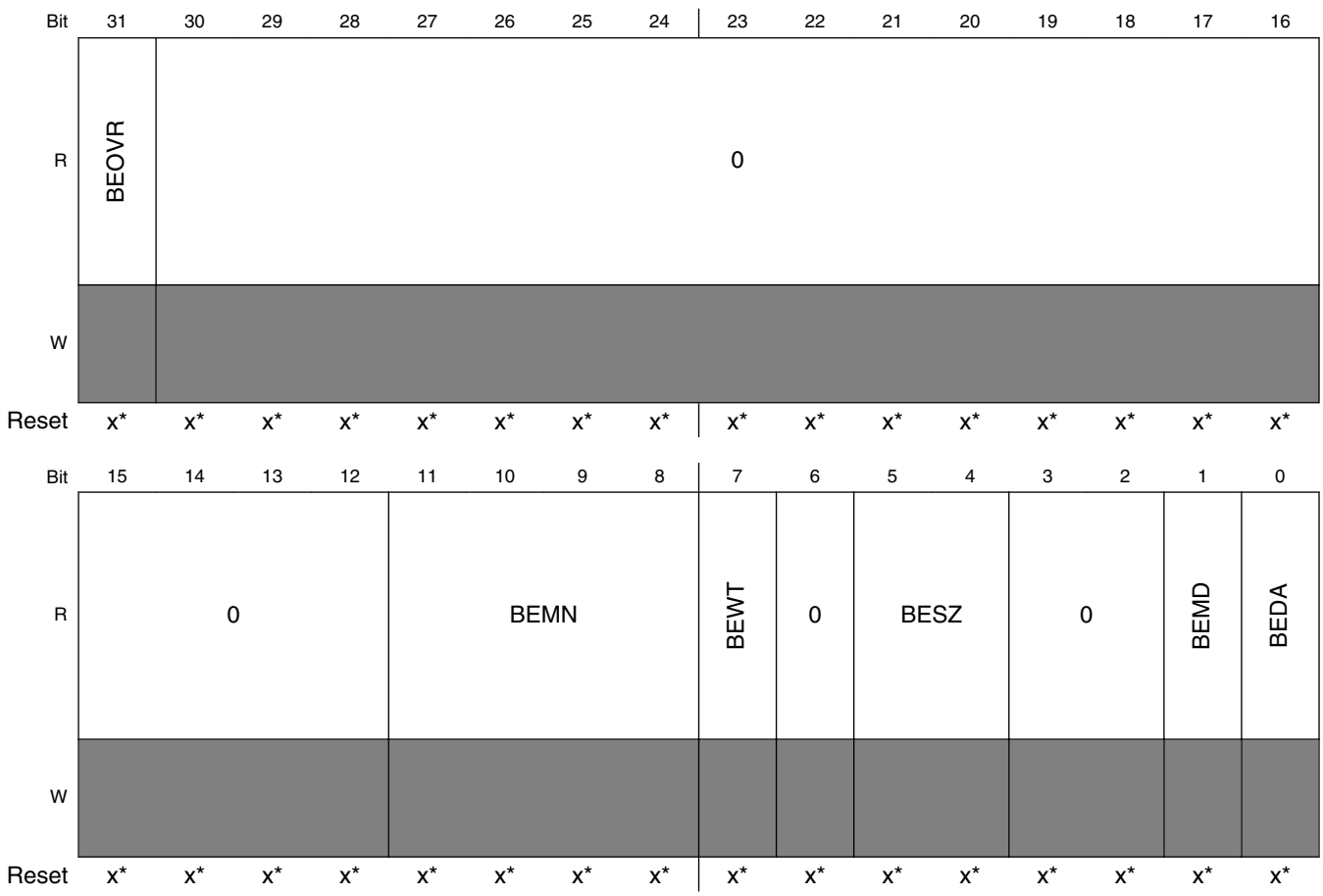
Field	Description
ADDRESS	Fault address

4.2.8.5 Fault attributes register (MCM_FATR)

When a properly-enabled cache write buffer error interrupt event is detected, the faulting attributes are captured in the MCM_FATR register.

The bits in this register are set by hardware and signaled by the assertion of MCM_ISCR[CWBER]. Attempted writes to this location are terminated with an error.

Address: E008_0000h base + 24h offset = E008_0024h



* Notes:
- x = Undefined at reset.

MCM_FATR field descriptions

Field	Description
31 BEOVR	Bus error overrun Indicates if anothercache write buffer bus error is detected before system software has retrieved all the error information from the original event, this overrun flag is set. The window of time is defined from the detection of the original cache write buffer error termination until the MCM_ISCR[CWBER] is written with a 1 to clear it and rearm the capture logic. This bit is set by the hardware and cleared whenever software writes a 1 to the CWBER bit. 0 No bus error overrun 1 Bus error overrun occurred. The FADR and FDR registers and the other FATR bits are not updated to reflect this new bus error.
30–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 BEMN	Bus error master number Crossbar switch bus master number of the captured cache write buffer bus error. For this device, this value is always 0x1.
7 BEWT	Bus error write Indicates the type of system bus access when the error was detected. Since this logic is monitoring data transfers from the cache write buffer, this bit is always a logical one, signaling a write operation. 0 Read access 1 Write access
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 BESZ	Bus error size Indicates the size of the cache write buffer access when the error was detected. 00 8-bit access 01 16-bit access 10 32-bit access 11 Reserved
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 BEMD	Bus error privilege level Indicates the privilege level of the cache write buffer access when the error was detected. 0 User mode 1 Supervisor/privileged mode
0 BEDA	Bus error access type Indicates the type of cache write buffer access when the error was detected. This attribute is always a logical one signaling a data reference. 0 Instruction 1 Data

4.2.8.6 Fault data register (MCM_FDR)

When a properly-enabled cache write buffer error interrupt event is detected, the faulting data is captured in the MCM_FDR register.

The bits in this register are set by hardware and signaled by the assertion of MCM_ISCR[CWBER]. For byte and halfword writes, only the accessed byte lanes contain valid data; the contents of the other bytes are undefined. Attempted writes to this location are terminated with an error.

Address: E008_0000h base + 28h offset = E008_0028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA																															
W																																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

MCM_FDR field descriptions

Field	Description
DATA	Fault data

4.3 Messaging Unit (MU)

4.3.1 Overview

The Messaging Unit module enables two processors within the SoC to communicate and coordinate by passing messages (e.g. data, status and control) through the MU interface. The MU also provides the ability for one processor to signal the other processor using interrupts.

Because the MU manages the messaging between processors, the MU uses different clocks (from each side of the different peripheral buses). Therefore, the MU must synchronize the accesses from one side to the other. The MU accomplishes synchronization using two sets of matching registers (Processor A-facing, Processor B-facing).

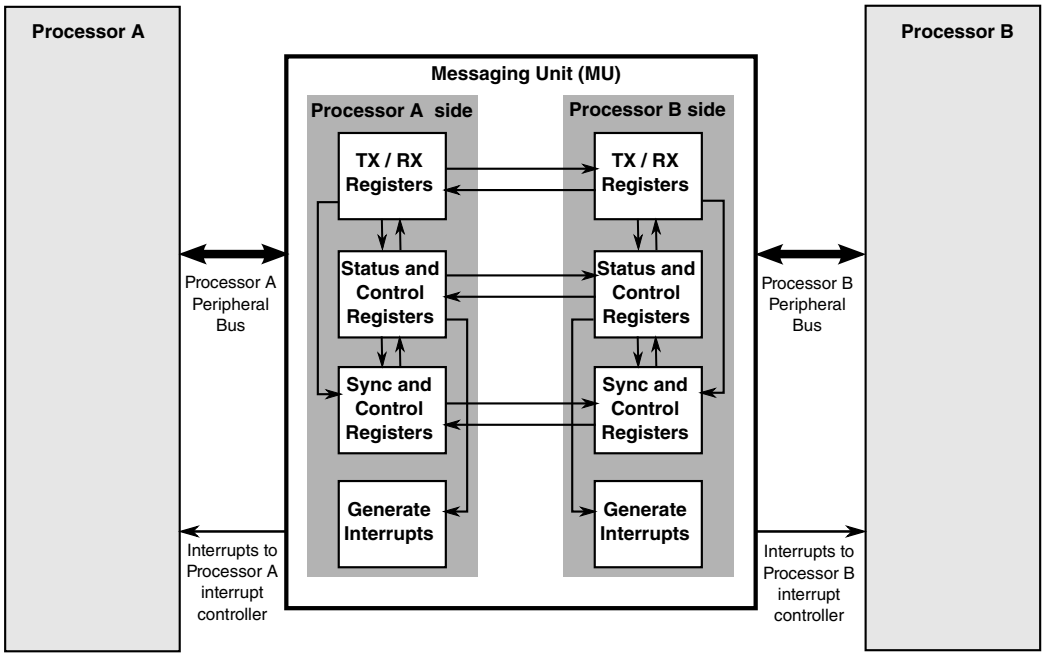


Figure 4-6. MU Block Diagram

4.3.1.1 Features

The MU includes the following features:

- Messaging control by interrupts or by polling
- The Processor B can take the Processor A out of low-power modes by asserting one of the above twelve interrupts to the Processor A and vice versa
- Symmetrical processor interfaces with each side supporting the following:
  - Four general-purpose interrupt requests reflected to the other side
  - Three general-purpose flags reflected to the other side
  - Four receive registers with maskable interrupt
  - Four transmit registers with maskable interrupt

4.3.1.2 Modes of Operation

The MU supports the modes described in the indicated sections:

- Operating Modes
- Low Power Modes

4.3.2 Functional Description

Table 4-7. Major Features of the MU

Major Feature	Description
Interprocessor Interrupts	• The MU has 12 interrupt sources on each side (Processor A-side, Processor B-side) that are used for signaling the other processor. The interrupts can be used for notification of RX/TX events and general-purpose signaling between the processors.
MU Reset	• The Processor A can issue a reset to the entire MU, using a control bit (MUR) in the Processor A Control Register (ACR). • The MUR bit is a self-clearing bit.

Processor B Boot Configuration	• The Boot Source for Processor B can be configured with the BBOOT bits in the ACR register.
	• Boot Source Options are:
	  • DMEM Base Address
	  • IMEM Base Address
	  • Address 0x00
	• The value at reset is loaded from Flash IFR

Processor B Reset Hold	• Processor B can be held in reset following any reset event. This is done by setting the BRSTH bit in the ACR register. • Processor B will be released from reset when this bit is cleared. • The value at reset is loaded from Flash IFR.
Processor A/B Clock Enable	• The Processor A/B platform clock can be enabled to continue running when Processor A/B enters Stop Mode, until Processor B/A also enters Stop Mode. This allows Processor B/A to continue accessing peripherals on Processor A/B's AIPS bus even when it has entered a Stop Mode.
Status and Control Communications between Cores	• The MU provides a way for the two cores to communicate using the status and control registers present on both the Processor B and Processor A sides of the MU. • The status register of one MU side reflects the status of the other MU side. • The control register is used for control operations, such as enabling an interrupt and sending an interrupt to the other processor.
Synchronized Message Transfers between Cores	• The transfer of data messages between cores uses transmit empty and receive full flags provided on both sides of the MU. • The update of these transmit and receive flags is accomplished using a synchronization mechanism. There is inherent latency between updating the flag on one side and reflecting its status on other side. For more about latency, see Event Update Timing
Accessing Shared Memory Directly and Avoiding Collisions	• For sending data or messages from one MU-side to the other MU-side, the MU provides 4 transmit registers and 4 receive registers on each side of the MU. • The Processor A or Processor B can access shared memory resources of the SoC directly. However, to avoid simultaneous access to shared memory by both cores, the MU provides a method (to prevent simultaneous access) using interrupts and transmit-receive registers for both processors.
Support for Different Clocks in the Two Cores	• The heart of the MU module is the event control mechanism, which synchronizes the access of one MU-side to the other MU-side, because these two MU-sides can operate using different clocks. • Formulated event update latency.
Memory-Mapped Registers	• The MU is connected as a peripheral under the Peripheral bus on both sides—on the Processor A-side, the Processor A Peripheral Bus, and on the Processor B-side, the Processor B Peripheral Bus.

4.3.2.1 Processor A Side Memory-Mapping

The messaging, control, and status registers of the Processor A-side for the MU are mapped to the Processor A memory as a regular peripheral. The Peripheral bus data bus is 32 bits wide inside the MU module.

4.3.2.2 Processor B Side Memory-Mapping

The messaging, control, and status registers of the Processor B-side for the MU are mapped to the Processor B memory as a regular peripheral. The Peripheral bus data bus is 32 bits wide inside the MU module.

4.3.2.3 MU Messaging

The MU provides 32-bit status and control registers to the Processor B and Processor A sides for control operations (such as interrupts and reset), and for status checking of the other MU-side.

For messaging, the MU has four, 32-bit write-only transmit registers and four, 32-bit read-only receive registers on the Processor B and Processor A-sides. These registers are used for sending messages to each other. These messages can be also be controlled using the 3 general purpose flags provided in the control and status registers of either MU-side.

4.3.2.3.1 Programmer Model

The messaging logic is used in conjunction with external memory. You have various messaging methods, which you can use to implement a messaging protocol. Some of these messages could mean “I have just written a message of N words, starting at offset X in the memory,” or “I have just finished reading the previous data block that was sent.” Having the messaging logic independent from the memory array does not restrict you to a predefined hardware protocol. On the other hand, the software needed to manage the messaging is short and straightforward.

Most of the messaging mechanisms are symmetric; they are duplicated and are available on both the Processor B-side and the Processor A-side. The messaging mechanisms are:

- Four, 32-bit write-only transmit registers, which are each reflected in four, read-only receive registers in the other processor’s side. You can use these registers to transfer 32-bit word messages or frame information of messages written to the shared memory (number of words, initial address, and message type code).

- A write to a transmit register on the transmitter side clears a “transmitter empty” bit in the Status Register on the transmitter side, and sets a “receiver full” bit in the Status Register on the receiver side. The setting of the bit at the receiver side can optionally trigger an interrupt at the receiver side (maskable receive interrupt).
- A read of one of the receive registers at the receiver side clears the “receiver full” bit in the Status Register at the receiver side, and sets the “transmitter empty” bit in the Status Register on the transmitter side. The setting of the “transmitter empty” bit can optionally trigger an interrupt at the transmitter side (maskable transmit interrupt).
- Four general purpose flags are reflected in the Status Register on the receiver side
- A read/write access to any reserved location and a write to a read-only register on the Processor A-side of the MU will generate a module transfer error acknowledge to the Processor A.
- A read/write access to any reserved location and write to a read-only register on the Processor B-side of the MU will generate a module transfer error acknowledge to the Processor B.

4.3.2.3.2 Messaging Examples

The following are messaging examples:

- **Passing short messages:** Transmit register(s) can be used to pass short messages from one to four words in length. For example, when a four-word message is desired, only one of the registers needs to have its corresponding interrupt enable bit set at the receiver side; the message’s first three words are written to the registers whose interrupt is masked, and the fourth word is written to the other register (which triggers an interrupt at the receiver side).
- **Passing frame information:** Transmit registers can be used to pass frame information for long messages written to the shared system (SDRAM and SyncFLASH). Such frame information normally includes a start address, number of words, and perhaps a message type code.
- **Passing event notices and requests:** Events and requests that do not include data words can be signaled from the Processor B to the Processor A using the general interrupts, such as acknowledging that a long message was read from the shared system memory.
- **Passing fixed length data:** Formatted data with a fixed length can be written in predetermined locations in the shared memory. A processor can use a general interrupt (Processor A or Processor B) to signal the other processor that the data is ready.
- **Passing announcements:** The three flags can be used by a processor to announce its current program state or other billboard messages to the other processor.

Figure 4-7 shows the MU registers schematic.

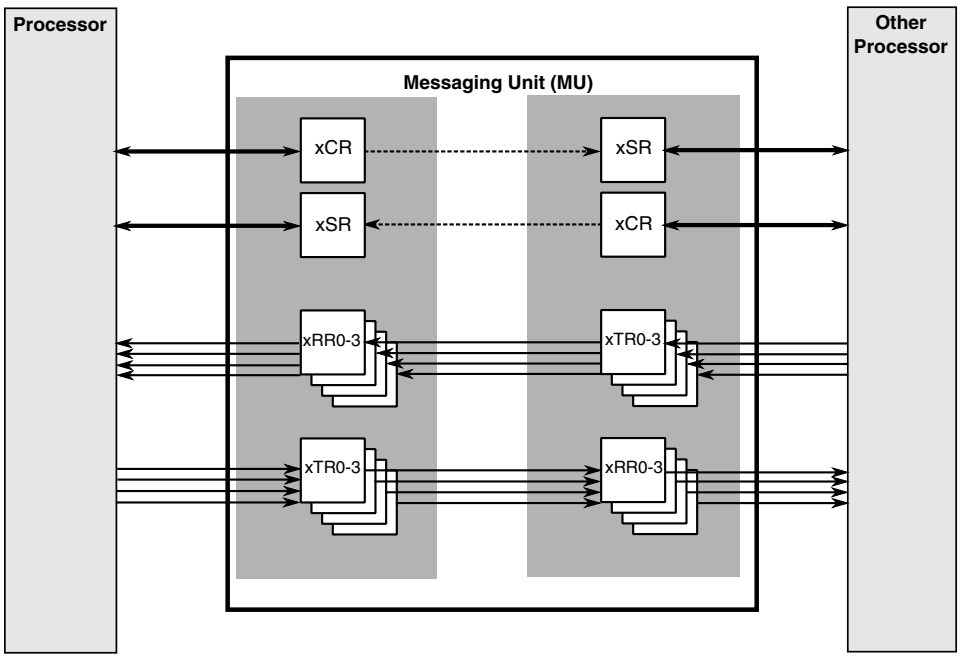


Figure 4-7. MU Registers

4.3.2.4 Operating Modes

This section describes all functional operation modes of the module.

4.3.2.5 Low Power Modes

This section describes the low power operating modes of the MU module.

4.3.2.5.1 Low Power Clocks and Synchronization

The Processor B and the Processor A clocks operate at different frequencies and from different sources. The MU design does not assume any frequency relationship between the Processor A and the Processor B clocks. Be aware, however, that the frequency relationship affects the MU's throughput performance.

- The data buffers and control logic of each MU-side operate with its corresponding clock.

4.3.2.5.2 Processor Low Power Modes

The Processors have four power modes:

- Run
- WAIT

- STOP
- DSM

The Processor can be awakened from a low-power mode by any enabled Processor side MU interrupt, as reflected in the xSR “status” register (RF0–3, TE0–3, GIP0–3 bits are set) and enabled in the xCR control register. Using these bits, the Processor can actively control when to wake the other Processor.

While the Processor is in STOP mode (such that the xSR register bits cannot be updated with events), special logic drives the enabled Processor interrupts directly from the other Processor-side (instead of from the xSR register).

While the Processor is in STOP mode, the asynchronous Processor interrupt will be asserted to wake the Processor:

- If any transmit data register of the other Processor-side is full, because of a write to it (transmit data register); that is, its “empty” bit in the xSR register is cleared while its corresponding receive interrupt is enabled on the Processor-side.
- If any receive data register of the other Processor-side is empty, because of a read on the other Processor -side; that is, its “full” bit in the xSR register is cleared while its corresponding transmit interrupt is enabled on the Processor-side.
- If any general purpose interrupt is set in the xCR register while the corresponding interrupt is enabled on the Processor-side.
- If the other Processor issues a non-maskable interrupt to the Processor.

The logic enables the other Processor to operate independently while the Processor is in any power mode (including STOP). However, the Processor power mode change protocol should be handled with care regarding:

- The interrupts that are enabled on the Processor-side
- The events that could be triggered by the other Processor-side
- The compatibility with the other Processor protocol of entering STOP mode

If the Processor is in STOP mode and an event on the other Processor is triggered, the EP bit (in the xSR register) will remain high until the Processor wakes up.

Before entering STOP mode, the Processor programmer should verify that the EP bit (in the xSR register) is cleared. This check is needed to ensure that all pending updates from the Processor, including the power mode change when STOP or WAIT is executed, will be updated in the xSR register.

- If the other Processor is in STOP mode or DSM mode, the EP bit (in the xSR register) may be stuck high; in this case, the Processor need not check the EP bit before entering STOP mode.

4.3.2.6 Event Update Timing

Each processor's MU messaging side (Processor B or Processor A) has a hardware mechanism to send "event update requests" to the other processor's side. An "event" is considered when any information change should be reflected at the Status Register of the receiving processor. The event update latency is the delay between the event being ready at one processor and the resulting update at the Status Register of the other processor.

- The minimum event latency is "1 clock of the sending side" + "2 1/2 clocks of the receiving side". The minimum case is if there is no event pending when the new event occurs.
- The maximum event latency is "6 clocks of the sending side" + "6 1/2 clocks of the receiving side." The maximum case is if the event occurred just after a previous event was sent to the other side. The event update latency will vary between the above-mentioned minimum and maximum latencies, depending on the time at which the subsequent event is triggered.

4.3.2.7 Interrupts

The MU controls the Processor B interrupt requests to the Processor A, and the Processor A interrupt requests to the Processor B. This section describes all the interrupts that the module generates.

4.3.2.7.1 Interrupts to the Processors

There are 12 interrupt sources from the MU to the Processors:

- Four receive interrupts (asserted when the Processors receive full bits are set and enabled in the xCR register) for each of the receive registers
- Four transmit interrupts (asserted when the Processor transmit empty bits are set and enabled in the xCR register) for each of the transmit registers
- Four general purpose interrupts (asserted when the GIP bits are set and enabled in the xCR register)

All the interrupts are maskable in the Processor Control Register (xCR). The MU does not assume any internal priority of these interrupts. Multiple interrupts (for example, Receive 0 and Receive 1 interrupts or any of the transmit and general purpose interrupts) can be asserted at one time. The priority of these interrupts should be resolved by the interrupt controller at the chip level.

The General Purpose Interrupt Pending bits (GIP0, GIP1, GIP2, and GIP3) should be cleared by the software (as part of the interrupt service routine) to de-assert the request to the interrupt controller.

4.3.2.7.2 General Purpose Interrupt Clearing Sequence

When a Processor writes to the general interrupt bit (GIR), the write event is synchronized to the other Processor clock to set the general interrupt request pending bit (GIP). When the GIP bit is set, and if the general purpose interrupt is enabled on the transmitting Processor side (GIE bit is set), then the receiving Processor general purpose interrupt is issued to the transmitting Processor. The transmitting Processor clears this interrupt by writing a “1” on the GIP bit. The interrupt is de-asserted as soon as the GIP bit is written. The write event of the GIP bit is synchronized to the other Processor clock. The synchronized signal clears the GIR bit. The software should not write the GIR bit again until the GIR bit is cleared.

4.3.2.8 Interrupt Messaging Protocols

4.3.2.8.1 Messaging Protocols using Interrupts

The example below describes a four-word messaging sequence sent by the Processor to the other Processor.

In this example, the first, second, and third receive interrupts are disabled, and the fourth receive interrupt is enabled. We write registers sequentially for n = 0, 1, 2, 3. For n = 0, 1, 2, the interrupts are disabled, therefore no interrupt will go to the other core (although interrupt conditions occur). For n = 3, the interrupt is enabled, and the last Receive Interrupt request is generated.

1. Write Sequence
   - The Processor writes the message information sequentially to its Transmit Registers 0, 1, 2.
   - When the write to the Transmit Register 3 occurs, the RF3 bit of the xSR is set after synchronization, and it immediately trigger the Receive 3 interrupt to the other Processor.
2. Read Sequence
   - The other Processor receives the Receive 3 interrupt and starts reading the message transferred from the receive registers.
   - After Receive Register 3 is read, the interrupt bit is cleared.

Figure 4-8 shows the programmer’s model of a messaging protocol using transmit and receive registers. Use Table 4-8 and Figure 4-8 to understand the generalized protocol sequence.

Table 4-8. Interrupt Messaging Protocol (Generalized)

Sequence	Action	Description
1	Processor A Data write	A data write to the ATRn register by Processor A is immediately reflected in the Processor B BRRn register.
2	Clear Tx Empty bit and Set Rx Full bit	The data write to the ATRn register • Clears the transmitter empty bit (TEn) in the Processor A Transmit Status Register • Sets the receiver full bit (RFn) in the Processor B Receive Status Register
3	Generate Receive Interrupt request	The setting of the receiver full bit (RFn) in the Receive Status Register generates a Receive Interrupt request to Processor B.
4	Processor B Data read	After receiving the Receive Interrupt request, Processor B performs a data read of the BRRn register.
5	Clear Rx Full bit and Set Tx Empty bit	Reading the data out of the BRRn register • Clears the receiver full bit (RFn) in the Processor B Receive Status Register • Sets the transmitter empty bit (TEn) in the Processor A Transmit Status Register
6	Generate Transmit Interrupt request	The setting of the transmitter empty bit (TEn) in the Transmit Status Register generates a Transmit Interrupt request to Processor A.

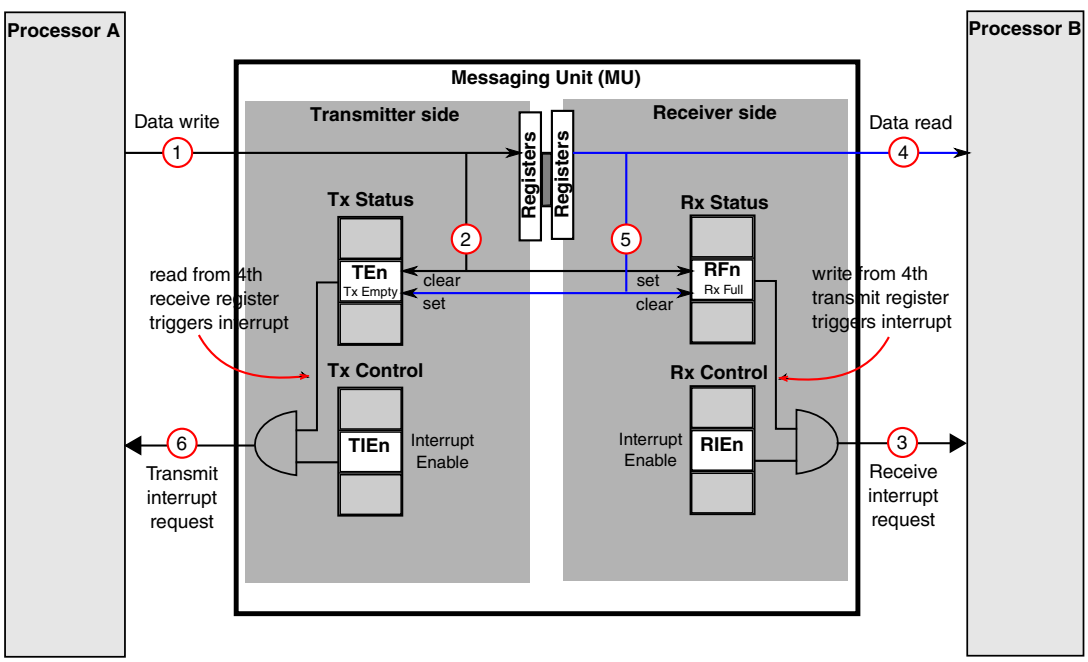


Figure 4-8. Messaging Model Using Transmit and Receive Registers

NOTE

The Transmit registers can be used to pass frame information on long messages written to the shared memory. Such frame

information would typically include an initial address, number of words, and perhaps a message type code.

The messaging hardware can be used by software to implement messaging protocols for a wide array of message types. Full support is given for both interrupt and polling management schemes.

4.3.2.8.2 Messaging Protocols using Event Interrupts

Events and requests that do not include data words can be signaled from the Processor B to the Processor A using the two general interrupts.

Formatted data with a fixed length can be written in predetermined locations in the shared memory. A processor can use a general purpose interrupt to signal the other processor that the data is ready.

The three flags can be used by a processor to announce to the other processor the program state it is currently in, or to announce similar messages.

Table 4-9 and Figure 4-9 describe the event sequence when the Processor triggers an interrupt.

Table 4-9. Interrupt Messaging Protocol (Generalized)

Sequence	Action	Description
1	Processor A sets General Interrupt request bit	Processor A sets its associated General Interrupt request bit (GIRn = 1) in the control register (ACR).
2	General Interrupt Request Pending status bit is set	The General Interrupt Request Pending status bit (GIPn) in the status register (BSR) is set to "1"
3	General Interrupt request to Processor B is generated	Setting the GIPn bit generates the General Interrupt request to Processor B (Interrupt Request Enable bit, GIEn, must be set for Processor B)
4	Processor B reads status register	The Processor B reads the GIPn bit in the BSR register.
5	Processor B services the interrupt	-
6	Processor B sets GIPn bit to clear interrupt	The Processor B writes "1" to the corresponding GIPn bit to clear the interrupt
7	GIRn bit is cleared	Setting the GIPn bit to "1" clears the General Interrupt request bit (GIRn) in the Processor A control register (ACR).

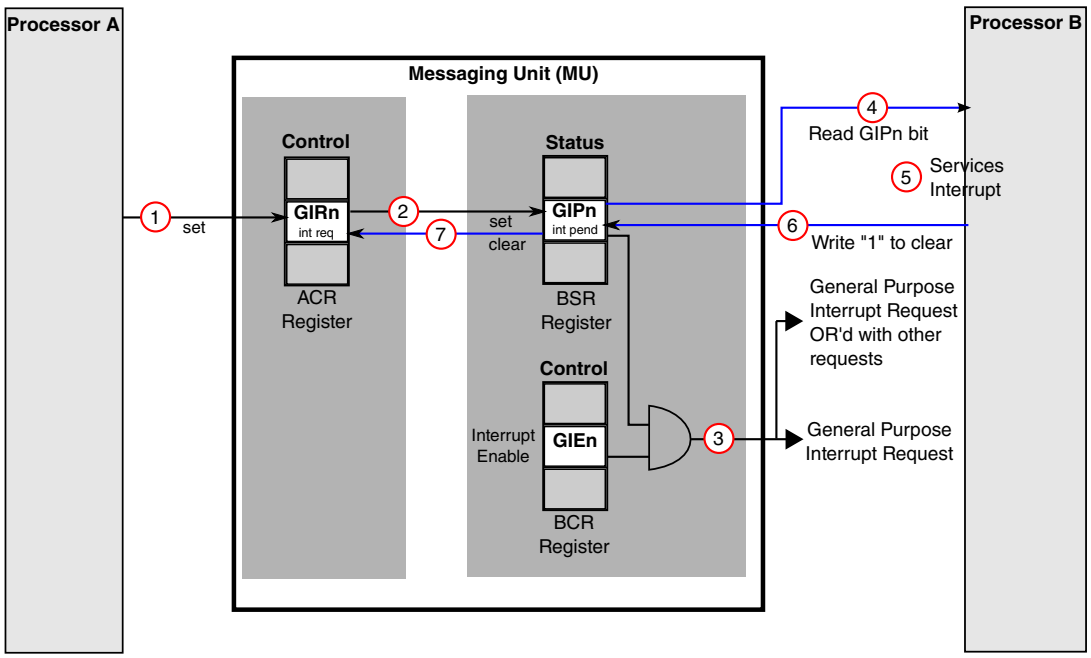


Figure 4-9. Messaging Model Using a General Purpose Interrupt

4.3.2.9 Exclusive Access to Shared Memory

You can use the MU to signal one processor about its current access to the shared memory, so that the data is not overwritten by the other processor during the exclusive memory access period.

The following tables describe the signaling protocol that the Processor A uses to inform the Processor B about its current access (write) to the shared memory, assuming that the set of bits and registers (GIR0 bit, BRR0 register, BTR0 register, GIR0 bit, ARR0 register, ATR0 register) are reserved to support exclusive access to the shared memory protocol.

Table 4-10. How the Processor A Performs an Exclusive Access to Shared Memory

Sequence	Action	Description
1	Processor A sends GIRn request to Processor B using Processor A control register	When the Processor A wants to perform an exclusive access to the shared memory, the Processor A sends an GIR0 request to the Processor B.
2	Processor A sends an exclusive-access request using a transmit data register (ATRn)	The Processor A will send an exclusive-access request (command, location, and length of target access) to Processor B using a selected transmit data register (ATR0).
3	Processor A waits for a dedicated interrupt from Processor B	The Processor A waits for a dedicated interrupt (as an acknowledgement) triggered by the Processor B before proceeding.

Table continues on the next page...

Table 4-10. How the Processor A Performs an Exclusive Access to Shared Memory (continued)

Sequence	Action	Description
4	Processor A accesses shared memory	After receiving a dedicated interrupt from the Processor B, Processor A proceeds.

Table 4-11. How the Processor B Scans for Transaction Information

Sequence	Action	Description
1	Processor B receives an interrupt from a receive data register (BRRn)	-
2	Processor B reads the receive data register (BRRn)	-
3	Processor B scans the receive data register contents	For transaction information (whether Processor A has requested an exclusive-access)

Table 4-12. How the Processor B Accepts Exclusive Access by Processor A

Sequence	Action	Description
1	Processor B triggers a dedicated interrupt	Processor B acknowledges the Processor A request by triggering a dedicated interrupt (ack) to the Processor A.
2	Processor B sends a code message to Processor A	Along with the acknowledge interrupt, the Processor B sends a code message to the Processor A through the selected transmit register (BTRn). The message informs the Processor A that it can exclusively access the shared memory.

Table 4-13. How the Processor B Rejects Exclusive Access by Processor A

Sequence	Action	Description
1	Processor B ignores Processor A request for exclusive access	If the Processor B does not want to give go-ahead permission to the Processor A, Processor B ignores the exclusive access request.

4.3.2.10 Packet Data Transfers

The following example describes the packet transfer sequence between the Processor B and Processor A subsystems:

Table 4-14. Packet Data Transfer Sequence

Action	Sequence	Description
Processor B requests DMA	1	The Processor B sends a DMA request to initiate the packet data transfer

Table continues on the next page...

Table 4-14. Packet Data Transfer Sequence (continued)

Action	Sequence	Description
DMA data transfer	2	DMA acknowledges.
	3	DMA starts transferring data from the specified Processor B location to the specified shared memory
	4	DMA interrupts the Processor B to signal that the packet transfer has finished.
Processor B informs Processor A that data is in shared memory	5	Using an MU Processor B-side transmit register, the Processor B sends a packet information message to the Processor A to inform the Processor A of the arrival of new packet data that is stored in shared memory . The message contains the command, location, and length of packet data information.
Processor A receives interrupt	6	The Processor A receives an interrupt (assuming its corresponding Processor A MU-side receive interrupt is enabled), and the pending processing task becomes active and processes packet data from memory.
Processor A reads data, writes data	7	The Processor A reads or processes packet data from shared memory.
	8	The Processor A writes the result from packet processing to a separate buffer.
Processor A informs Processor B that transfer is finished	9	After the processing of the packet data finishes, the Processor A informs the Processor B (using the MU Processor A-side transmit register, ATRn).
Processor A sends interrupt to Processor B (request for more data)	10	The Processor B receives the next interrupt from the Processor A, in which the Processor A requests more packet data.

4.3.2.11 MU Resets

The MU has two sources of reset, and each reset has a different function from the MU or system perspective.

- One asynchronous system that is connected to both sides of the MU interface.
- One programmable hardware reset (MUR bit) in the ACR register (on the Processor A-side).

Table 4-15. MU programmable resets

Reset	Description
Processor A MU reset	• Processor A MU Reset bit (MUR) of the ACR register • The MUR reset affects the messaging section on both the Processor A and the Processor B sides. The MUR reset causes all control and status registers to return to their default values and all internal states to be cleared. • It is up to the Processor A software to decide whether to use the MUR reset or not. • The instruction immediately following assertion of the MUR bit should not write to MU registers. Such a write may be overwritten by the reset sequence and the register will remain with the reset value. You should wait at least one instruction (after assertion of the MUR bit) before attempting a write to MU registers.

After issuing MUR bit reset events, the Processor A programmer can verify that the reset sequence on the Processor B-side has ended, by checking the RS bit in the ASR register.

NOTE

MUR bit assertion is a delicate operation because it affects the other side's registers asynchronously. MUR bit assertion may cause unpredictable behavior if, for example, the Processor B is concurrently testing an MU register bit (TE bit in Processor B SR register). Before asserting the MUR bit, you should verify that the Processor B is not presently engaged in an MU signalling activity.

4.3.3 Software Restrictions

This section describes certain software restrictions when accessing the MU.

4.3.3.1 General Restrictions

This section lists the restrictions that apply to both the sides (Processor A, Processor B) of the MU.

4.3.3.1.1 Write-After-Write to a Transmit Register

A write to a transmit register signals the receiver side that data is ready for retrieval.

- Writing to the transmit register again without verifying that the data was retrieved is prohibited, because the transmitter side has no way of knowing the exact time that the receiver will attempt to retrieve the data.
- Before attempting to write the transmit register again, the transmitter side should wait for a "Transmitter Empty" interrupt, or should poll the "Transmitter Empty" bit in the Status Register.
- Failure to follow this restriction may result in the wrong data being read on the receiver side of the MU.

4.3.3.1.2 Read-After-Read from a Receive Register

A read of a receive register signals the transmitter side that data can be written to that register. In the same way, the receiver processor should not read a receive register before receiving a "Receiver Full" interrupt or polling the "Receiver Full" bit in the Status Register.

- Reading the receive register again without verifying that the data was written is prohibited, because the receiver side has no way of knowing the exact time that the transmitter will attempt to write the data.
- Before attempting to read the receive register again, the receiver side should wait for a “Receiver Full” interrupt, or should poll the “Receiver Full” bit in the Status Register.
- Failure to follow this restriction may result in the wrong data being written on the transmitter side of the MU.

4.3.3.2 Processor Restrictions

This section lists the restrictions that apply each side of the processor in the MU.

4.3.3.2.1 Before Entering Low Power Mode

Before entering Low Power mode, the Processor should verify that the Processor Event Pending (EP) bit in the Status Register is cleared.

- If the Event Pending bit (EP) is still set to “1”, then the Processor should wait and poll the EP bit until it is cleared, before executing the LPM instruction.
- Note that if the other Processor is in Low Power mode (programmed for clock gating in CCM), the EP bit may be stuck high. In this case, the other Processor clock must be turned ON to get the EP bit cleared before the Processor can enter Low Power mode.
- To discover which power mode the other Processor is in, the Processor can check the PM bits in the xSR register.

4.3.3.2.2 Before Setting a General Interrupt Request Bit (GIR0–3)

Before setting a General Interrupt Request bit (GIR0–3), you must verify that the GIRn bit is cleared, which means that a general interrupt is not pending. Generally, setting the GIRn bit while the bit is set to “1” will be ignored, but in some cases it may issue a second interrupt. This restriction is meant to prevent this indeterministic behavior.

4.3.3.2.3 Reset Bit Restrictions

The reset bit (MUR, HR) restrictions are:

- Before asserting the MUR bit in the ACR register, verify that the Processor B-side is not engaged in some MU activity.
- Do not write to an MU register in the instruction immediately after the assertion of the MUR bit in the ACR register, because the written data can be overridden by the reset value.

4.3.4 MU Processor A-side Memory Map/Register Definition

This section contains the detailed register descriptions for the Processor A-side MU registers.

MUA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30AA_0000	Processor A Transmit Register 0 (MUA_ATR0)	32	R/W	0000_0000h	4.3.4.1/130
30AA_0004	Processor A Transmit Register 1 (MUA_ATR1)	32	R/W	0000_0000h	4.3.4.2/131
30AA_0008	Processor A Transmit Register 2 (MUA_ATR2)	32	R/W	0000_0000h	4.3.4.3/132
30AA_000C	Processor A Transmit Register 3 (MUA_ATR3)	32	R/W	0000_0000h	4.3.4.4/132
30AA_0010	Processor A Receive Register 0 (MUA_ARR0)	32	R	0000_0000h	4.3.4.5/133
30AA_0014	Processor A Receive Register 1 (MUA_ARR1)	32	R	0000_0000h	4.3.4.6/134
30AA_0018	Processor A Receive Register 2 (MUA_ARR2)	32	R	0000_0000h	4.3.4.7/134
30AA_001C	Processor A Receive Register 3 (MUA_ARR3)	32	R	0000_0000h	4.3.4.8/135
30AA_0020	Processor A Status Register (MUA_ASR)	32	R/W	00F0_0080h	4.3.4.9/136
30AA_0024	Processor A Control Register (MUA_ACR)	32	R/W	0000_0000h	4.3.4.10/ 139

4.3.4.1 Processor A Transmit Register 0 (MUA_ ATR0)

Use Processor A Transmit Register 0 (ATR0, 32-bit, write-only) to transmit a message or data to the Processor B.

- You can only write to the ATR0 register when the TE0 bit in ASR register is set to “1”.
- Reading the ATR0 register returns all zeros.

Address: 30AA_0000h base + 0h offset = 30AA_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ATR0																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MUA_ATR0 field descriptions

Field	Description
ATR0	Processor A Transmit Register 0. (Write-only)

MUA_ATR0 field descriptions (continued)

Field	Description
	• Data written to the ATR0 register is reflected on the Processor B-side in the Processor B Receive Register 0 (BRR0). The ATR0 and BRR0 registers are not double-buffered—a write to the ATR0 register overrides the data readable at the BRR0 register. • A write to the transmit register clears a “transmitter empty” bit (TE0) in the Processor A Status Register (ASR) on the transmitter side, and sets a “receiver full” bit (RF0) in the Processor B Status Register (BSR) on the receiver side (optionally triggering an interrupt 0 on the Processor B-side). • Any write to the ATR0 register will update all status information.

4.3.4.2 Processor A Transmit Register 1 (MUA_ATR1)

Use Processor A Transmit Register 1 (ATR1, 32-bit, write-only) to transmit a message or data to the Processor B.

- You can only write to the ATR1 register when the TE1 bit in ASR register is set to “1”.
- Reading the ATR1 register returns all zeros.

Address: 30AA_0000h base + 4h offset = 30AA_0004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ATR1																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MUA_ATR1 field descriptions

Field	Description
ATR1	Processor A Transmit Register 1. (Write-only) • Data written to the ATR1 register is reflected on the Processor B-side in the Processor B Receive Register 1 (BRR1). The ATR1 and BRR1 registers are not double-buffered—a write to the ATR1 register overrides the data readable at the BRR1 register. • A write to the transmit register clears a “transmitter empty” bit (TE1) in the Processor A Status Register (ASR) on the transmitter side, and sets a “receiver full” bit (RF1) in the Processor B Status Register (BSR) on the receiver side (optionally triggering an interrupt 1 on the Processor B-side). • Any write to the ATR1 register will update all status information.

4.3.4.3 Processor A Transmit Register 2 (MUA_ATR2)

Use Processor A Transmit Register 2 (ATR2, 32-bit, write-only) to transmit a message or data to the Processor B.

- You can only write to the ATR2 register when the TE2 bit in ASR register is set to "1".
- Reading the ATR2 register returns all zeros.

Address: 30AA_0000h base + 8h offset = 30AA_0008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ATR2																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MUA_ATR2 field descriptions

Field	Description
ATR2	Processor A Transmit Register 2. (Write-only) • Data written to the ATR2 register is reflected on the Processor B-side in the Processor B Receive Register 2 (BRR2). The ATR2 and BRR2 registers are not double-buffered—a write to the ATR2 register overrides the data readable at the BRR2 register. • A write to the transmit register clears a "transmitter empty" bit (TE2) in the Processor A Status Register (ASR) on the transmitter side, and sets a "receiver full" bit (RF2) in the Processor B Status Register (BSR) on the receiver side (optionally triggering an interrupt 2 on the Processor B-side). • Any write to the ATR2 register will update all status information.

4.3.4.4 Processor A Transmit Register 3 (MUA_ATR3)

Use Processor A Transmit Register 3 (ATR3, 32-bit, write-only) to transmit a message or data to the Processor B.

- You can only write to the ATR3 register when the TE3 bit in ASR register is set to "1".
- Reading the ATR3 register returns all zeros.

Address: 30AA_0000h base + Ch offset = 30AA_000Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ATR3																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MUA_ATR3 field descriptions

Field	Description
ATR3	Processor A Transmit Register 3. (Write-only) • Data written to the ATR3 register is reflected on the Processor B-side in the Processor B Receive Register 3 (BRR3). The ATR3 and BRR3 registers are not double-buffered—a write to the ATR3 register overrides the data readable at the BRR3 register. • A write to the transmit register clears a “transmitter empty” bit (TE3) in the Processor A Status Register (ASR) on the transmitter side, and sets a “receiver full” bit (RF3) in the Processor B Status Register (BSR) on the receiver side (optionally triggering an interrupt 3 on the Processor B-side). • Any write to the ATR3 register will update all status information.

4.3.4.5 Processor A Receive Register 0 (MUA_ARR0)

Use Processor A Receive Register 0 (ARR0, 32-bit, read-only) to receive a message or data from the Processor B.

- Data written to the BTR0 register is immediately reflected in the ARR0 register.
- You can only read the ARR0 register when the RF0 bit in the ASR register is set to “1”.
- Writing to the ARR0 register generates an error response to the Processor A.

Address: 30AA_0000h base + 10h offset = 30AA_0010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ARR0																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MUA_ARR0 field descriptions

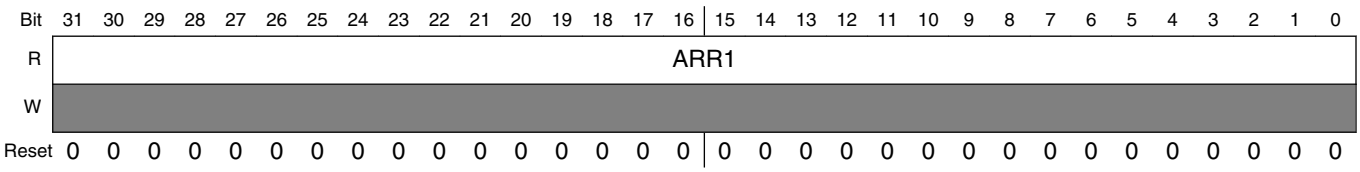
Field	Description
ARR0	Processor A Receive Register 0. (Read-only) • Reflects the data written to Processor B Transmit Register 0 (BTR0). • Reading the ARR0 register clears the “receiver full” bit (RF0) in the Processor A Status Register (ASR) on the receiver side, and sets the “transmitter empty” bit (TE0) in the Processor B Status Register on the transmitter side (optionally triggering a transmit interrupt 0 on the Processor B-side). • Any read of the ARR0 register will update all status information.

4.3.4.6 Processor A Receive Register 1 (MUA_ARR1)

Use Processor A Receive Register 1 (ARR1, 32-bit, read-only) to receive a message or data from the Processor B.

- Data written to the BTR1 register is immediately reflected in the ARR1 register.
- You can only read the ARR1 register when the RF1 bit in the ASR register is set to "1".
- Writing to the ARR1 register generates an error response to the Processor A.

Address: 30AA_0000h base + 14h offset = 30AA_0014h



MUA_ARR1 field descriptions

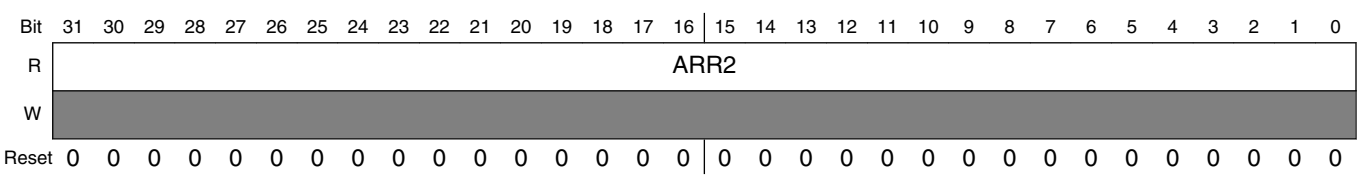
Field	Description
ARR1	Processor A Receive Register 1. (Read-only) • Reflects the data written to Processor B Transmit Register 1 (BTR1). • Reading the ARR1 register clears the "receiver full" bit (RF1) in the Processor A Status Register (ASR) on the receiver side, and sets the "transmitter empty" bit (TE1) in the Processor B Status Register on the transmitter side (optionally triggering a transmit interrupt 1 on the Processor B-side). • Any read of the ARR1 register will update all status information.

4.3.4.7 Processor A Receive Register 2 (MUA_ARR2)

Use Processor A Receive Register 2 (ARR2, 32-bit, read-only) to receive a message or data from the Processor B.

- Data written to the BTR2 register is immediately reflected in the ARR2 register.
- You can only read the ARR2 register when the RF2 bit in the ASR register is set to "1".
- Writing to the ARR2 register generates an error response to the Processor A.

Address: 30AA_0000h base + 18h offset = 30AA_0018h



MUA_ARR2 field descriptions

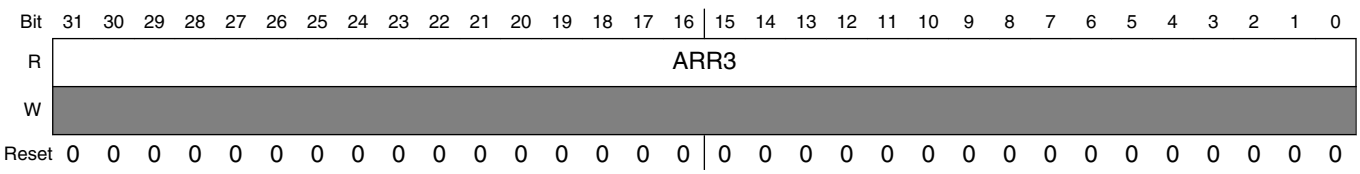
Field	Description
ARR2	Processor A Receive Register 2. (Read-only) • Reflects the data written to Processor B Transmit Register 1 (BTR2). • Reading the ARR2 register clears the “receiver full” bit (RF2) in the Processor A Status Register (ASR) on the receiver side, and sets the “transmitter empty” bit (TE2) in the Processor B Status Register on the transmitter side (optionally triggering a transmit interrupt 2 on the Processor B-side). • Any read of the ARR2 register will update all status information.

4.3.4.8 Processor A Receive Register 3 (MUA_ARR3)

Use Processor A Receive Register 3 (ARR3, 32-bit, read-only) to receive a message or data from the Processor B.

- Data written to the BTR3 register is immediately reflected in the ARR3 register.
- You can only read the ARR3 register when the RF3 bit in the ASR register is set to “1”.
- Writing to the ARR3 register generates an error response to the Processor A.

Address: 30AA_0000h base + 1Ch offset = 30AA_001Ch



MUA_ARR3 field descriptions

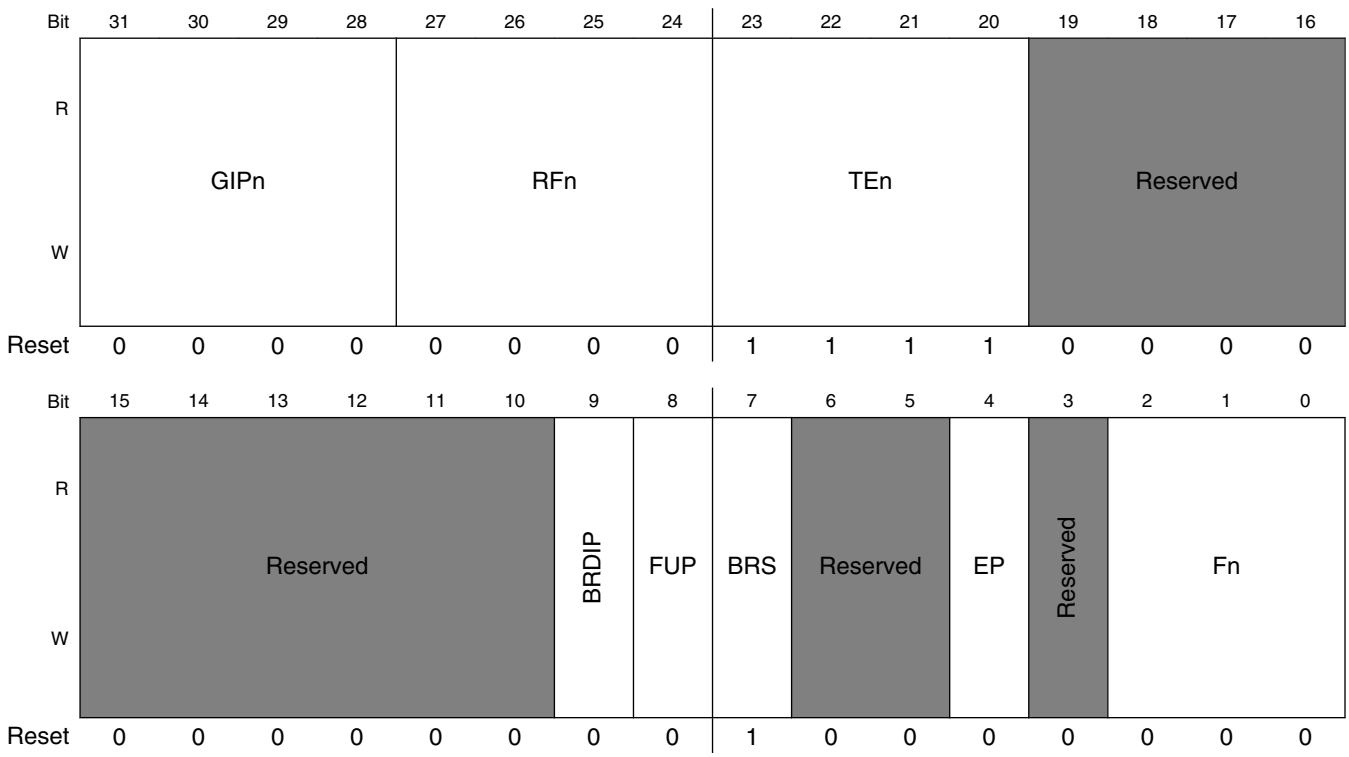
Field	Description
ARR3	Processor A Receive Register 3. (Read-only) • Reflects the data written to Processor B Transmit Register 3 (BTR3). • Reading the ARR3 register clears the “receiver full” bit (RF3) in the Processor A Status Register (ASR) on the receiver side, and sets the “transmitter empty” bit (TE3) in the Processor B Status Register on the transmitter side (optionally triggering a transmit interrupt 3 on the Processor B-side). • Any read of the ARR3 register will update all status information.

4.3.4.9 Processor A Status Register (MUA_ASR)

Use the Processor A Status Register (ASR, 32-bit, read-write) to show interrupt status from the Processor B, general purpose flags, and to set dual function control-status bits.

- Some dual-purpose bits are set by the MU logic, and cleared by the Processor A-side programmer
- Other dual-purpose bits are set by the Processor A-side programmer, and cleared by the MU logic.

Address: 30AA_0000h base + 20h offset = 30AA_0020h



MUA_ASR field descriptions

Field	Description
31–28 GIPn	For n = {0, 1, 2, 3} Processor A General Interrupt Request n Pending. (Read-Write) • GIPn bit signals the Processor A that the GIRn bit in the BCR register on the Processor B-side was set from “0” to “1”. If the GIEn bit in the ACR register is set to “1”, a General Interrupt n request is issued. • The GIPn bit is cleared by writing it back as “1”. Writing “0”, or writing “1” when the GIPn bit is cleared is ignored. Use this feature in the interrupt routine, where the GIPn bit is cleared in order to de-assert the interrupt request source at the interrupt controller. The proper bit clearing sequence is: clear an Processor A register, set the desired bit in it (Processor A register), and write it to the ASR register, thus clearing the GIPn bit. • GIPn bit is cleared when the MU is reset.

Table continues on the next page...

MUA_ASR field descriptions (continued)

Field	Description
	0 Processor A general purpose interrupt n is not pending. (default) 1 Processor A general purpose interrupt n is pending.
27–24 RFn	For n = {0, 1, 2, 3} Processor A Receive Register n Full. (Read-only) • The RFn bit is set to “1” when the BTRn register is written on the Processor B-side. • After the RFn bit is set to “1”, the RFn bit signals the Processor A-side that new data is ready to be read by the Processor A in the ARRn register, and a Receive n interrupt is issued on the Processor A-side (if the RIEn bit in the ACR register has been set to “1”). • RFn bit is cleared when the ARRn register is read, and when the MU is reset. 0 ARRn register is not full (default). 1 ARRn register has received data from BTRn register and is ready to be read by the Processor A.
23–20 TEn	For n = {0, 1, 2, 3} Processor A Transmit Register n Empty. (Read-only) • The TEn bit is set to “1” after the BRRn register is read on the Processor B-side. • After the TEn bit is set to “1”, the TEn bit signals the Processor A-side that the ATRn register is ready to be written on the Processor A-side, and a Transmit n interrupt is issued on the Processor A-side (if the TEn bit in the ACR register is set to “1”). • TEn bit is cleared after the ATRn register is written on the Processor A-side. • TEn bit is set to “1” when the MU is reset. 0 ATRn register is not empty. 1 ATRn register is empty (default).
19–10 -	This field is reserved. Reserved.
9 BRDIP	Processor B Reset De-asserted Interrupt Pending. (Read-Write) • BRDIP bit signals the Processor A-side that the Processor B-side has come out of reset. • BRDIP bit is set to “1” after the MU Processor B-side comes out of reset, after synchronization. The interrupt generated by a Processor B-side reset de-assertion is ORed with the Processor A general purpose interrupt 3. The Processor A general purpose interrupt 3 is issued when the Processor B-side comes out of reset (if the interrupt is enabled). • To clear the BRDIP bit, write “1”, which also clears general purpose interrupt 3. • When Processor A-side of MU comes out of reset BRDIP bit has value “0”(default).Then Processor A sees the status of Processor B-side and if Processor B-side has come out of reset then BRDIP bit goes high.This takes 5-6 clock cycles.And if you read BRDIP bit now you will see it as high although its reset value was "0". 0 The Processor A general purpose interrupt 3, because of a Processor B-side reset de-assertion, is cleared (default). 1 The Processor B-side is out of reset.
8 FUP	Processor A Flags Update Pending. (Read-only) • FUP bit is set to “1” when the Processor A-side sends a Flags Update request to the Processor B-side. • A Flags Update request is generated when the ABF[2:0] bits of the ACR register change. No flag update changes are allowed while the FUP bit is set to “1”. Any write to the ABF[2:0] bits, while the FUP bit is set to “1”, will not generate a Flags Update event, and the ABF[2:0] bits will stay unchanged. • FUP bit is cleared when this Flags Update request is internally acknowledged (that the flag is updated) from the MU Processor B-side, and during MU reset.

Table continues on the next page...

MUA_ASR field descriptions (continued)

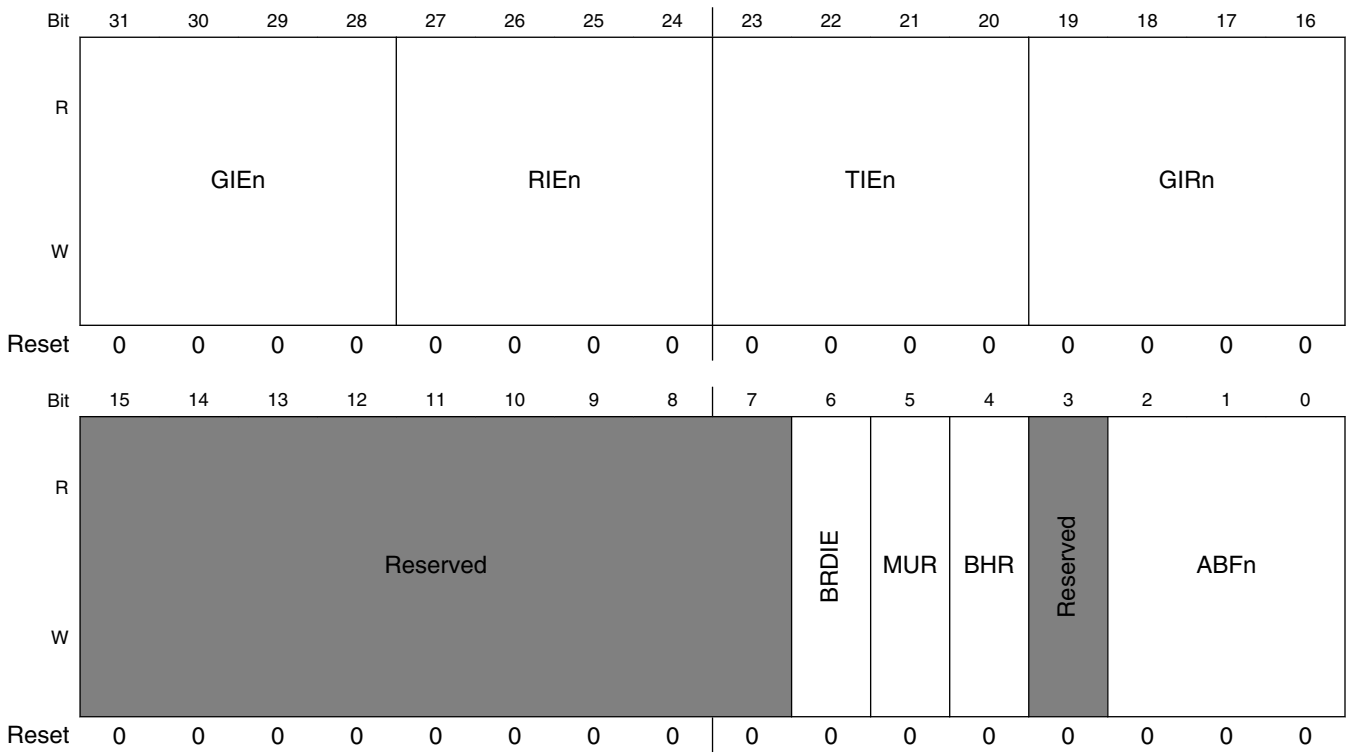
Field	Description
	0 No flags updated, initiated by the Processor A, in progress (default) 1 Processor A initiated flags update, processing
7 BRS	Processor B-side Reset State. (Read-only) • BRS bit indicates if the Processor B-side of the MU is in a reset state or not. • If the BRS bit is set to “1”, then the Processor B-side of the MU is still in the reset state. • If the BRS bit is cleared, then the Processor B-side of the MU are out of reset. • The BRS bit is set to “1” during: a Processor B system reset, or an MU reset (caused by setting the MUR bit at the ACR register). • The BRS bit is cleared when the reset sequence on the Processor B-side of the MU ends. After issuing any of the reset events mentioned previously, you should verify that the BRS bit is cleared before starting any accesses. • When Processor A side of MU comes out of reset BRS bit has value “1”(default).Then Processor A sees the status of Processor B-side and if Processor B-side has come out of reset then BRS bit goes low.This takes 5-6 clock cycles.And if you read BRS bit now you will see it as low although its reset value was "1" . 0 The Processor B-side of the MU is not in reset. 1 The Processor B-side of the MU is in reset.
6–5 -	This field is reserved. Reserved
4 EP	Processor A-Side Event Pending. (Read-only) • EP bit is set to “1” when the Processor A-side mechanism sends an event update request to the Processor B-side. • EP bit is cleared when the event update acknowledge is received. An “event” is any hardware message that is reflected in the BSR register on the Processor B-side (for example, “transmit register 0 written”). During normal operations, you do not have to deal with the state of the EP bit because the event update mechanism works automatically. • To ensure events have been posted to Processor B before entering STOP mode, you should verify that the EP bit is cleared. If EP bit is set to “1”, you should wait and continue to poll it (EP bit) before entering STOP mode. • Reading the ASR register (to check the EP bit) should be the last access to the MU that should be performed before entering STOPor WAIT modes; otherwise, the EP bit may be set by subsequent additional actions. • The EP bit is cleared when the MU resets. 0 The Processor A-side event is not pending (default). 1 The Processor A-side event is pending.
3 -	This field is reserved. Reserved.
Fn	For n = {0, 1, 2} Processor A-Side Flag n. (Read-only) • Fn bit is the Processor A-side flag that reflects the values written to the BAFn bit in the Processor B control register. • Every time that the BAFn bit is written, the BAFn bit write event updates the Fn bit after the event update latency, which is measured in terms of the number of clocks of the Processor B and the Processor A. 0 BAFn bit in BCR register is written 0 (default). 1 BAFn bit in BCR register is written 1.

4.3.4.10 Processor A Control Register (MUA_ACR)

Use the Processor A Control Register (ACR, 32-bit, read-write) to enable the MU interrupts on the Processor A-side, and trigger events and interrupts on the Processor B-side (general purpose interrupt, flag update).

For the fields GIEn, RIEn, TIEn and GIRn, n=0 corresponds to the high order bit and n=3 corresponds to the low order bit.

Address: 30AA_0000h base + 24h offset = 30AA_0024h



MUA_ACR field descriptions

Field	Description
31–28 GIEn	For n = {0, 1, 2, 3} Processor A General Purpose Interrupt Enable n. (Read-Write) When GIEn=0 corresponds to the high order bit and GIE3 corresponds to the low order bit. • GIEn bit enables Processor A General Interrupt n. • If GIEn bit is set to “1” (enabled), then a General Interrupt n request is issued when the GIPn bit in the ASR register is set to “1”. • If GIEn is cleared (disabled), then the value of the GIPn bit is ignored and no General Interrupt n request will be issued. • GIEn bit is cleared when the MU resets. 0 Disables Processor A General Interrupt n. (default) 1 Enables Processor A General Interrupt n.

Table continues on the next page...

MUA_ACR field descriptions (continued)

Field	Description
27–24 RIEn	For n = {0, 1, 2, 3} Processor A Receive Interrupt Enable n. (Read-Write) • RIEn bit enables Processor A Receive Interrupt n. • If RIEn bit is set to “1” (enabled), then an Processor A Receive Interrupt n request is issued when the RFn bit in the ASR register is set to “1”. • If RIEn bit is cleared (disabled), then the value of the RFn bit is ignored and no Processor A Receive Interrupt n request will be issued. • RIEn bit is cleared when the MU resets. 0 Disables Processor A Receive Interrupt n. (default) 1 Enables Processor A Receive Interrupt n.
23–20 TIEn	For n = {0, 1, 2, 3} Processor A Transmit Interrupt Enable n. (Read-Write) • TIEn bit enables Processor A Transmit Interrupt n. • If TIEn bit is set to “1” (enabled), then an Processor A Transmit Interrupt n request is issued when the TEn bit in the ASR register is set to “1”. • If TIEn bit is cleared (disabled), then the value of the TEn bit is ignored and no Processor A Transmit Interrupt n request will be issued. • TIEn bit is cleared when the MU resets. 0 Disables Processor A Transmit Interrupt n. (default) 1 Enables Processor A Transmit Interrupt n.
19–16 GIRn	For n = {0, 1, 2, 3} Processor A General Purpose Interrupt Request n. (Read-Write) • Writing “1” to the GIRn bit sets the GIPn bit in the BSR register on the Processor B-side. If the GIEn bit in the BCR register is set to “1” on the Processor B-side, a General Purpose Interrupt n request is triggered. • The GIRn bit is cleared if the GIPn bit (in the BSR register on the Processor B-side) is cleared by writing it (GIPn bit) as “1”, thereby signalling the Processor A that the interrupt was accepted (cleared by the software). The GIPn bit cannot be written as “0” on the Processor A-side. • To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit). • GIRn bit is cleared when the MU resets. 0 Processor A General Interrupt n is not requested to the Processor B (default). 1 Processor A General Interrupt n is requested to the Processor B.
15–7 -	This field is reserved. Reserved.
6 BRDIE	Processor B Reset De-assertion Interrupt Enable. (Read-Write) • BRDIE bit enables Processor A General Interrupt 3. • If BRDIE bit is set to “1”, then General Interrupt 3 request is issued to the Processor A when the BRDIP bit in the ASR register is set to “1”. • If BRDIE is cleared, then the value of the BRDIP bit is ignored and no General Interrupt 3 request will be issued. • The BRDIE bit is cleared when the MU resets. 0 Disables the Processor A General Purpose Interrupt 3 request due to the Processor B reset de-assertion to the Processor A. Processor B reset deassertion causes Processor B and MU-Processor B side to come out of reset thus setting BRDIP bit to “1”. 1 Enables Processor A General Purpose Interrupt 3 request due to the Processor B reset de-assertion to the Processor A.

Table continues on the next page...

MUA_ACR field descriptions (continued)

Field	Description
5 MUR	Processor A MU Reset. • Setting MUR bit to “1” resets both the Processor B and the Processor A sides of the MU module, forcing all control and status registers to return to their default values (except the BHR bit in the ACR register and BHRM bit in BCR register), and all internal states to be cleared. • Before setting the MUR bit to “1”, it is advisable to interrupt the Processor B, because setting the MUR bit may affect the ongoing Processor B program. • After setting the MUR bit, you should monitor the value of the BRS bit in the ASR register to know when the reset sequence on the Processor B-side has ended. • MUR bit can only be written as “1”. • MUR bit is always read as “0”. • MUR bit is cleared during the MU reset sequence. 0 N/A. Self clearing bit (default). 1 Asserts the Processor A MU reset.
4 BHR	Processor B Hardware Reset. (Read-Write) • BHR bit asserts and de-asserts the hardware reset of the Processor B. • Set BHR bit to “1” to start a hardware reset of the Processor B. • Clear the BHR bit to de-assert the Processor B hardware reset input. • Assert the BHR bit for a minimum of 3 clock cycles of network clock (sampling clock in SRC) clock. The BRS bit in MU_ASR register (b[7]) indicates the state of the Processor B. As soon as the Processor B goes into Reset (BRS bit is set to “1”), the BHR bit can be de-asserted. • Strobe-setting the BHR bit will not cause an internal MU reset but will be routed outside MU to Processor B domain reset logic • After clearing the BHR bit, monitor the value of the BRS bit at the ASR to know when the Processor B reset sequence has ended. • The BHR reset issued by the Processor A to the Processor B is maskable by the Processor B (according to the settings of the BHRM bit in the BCR register). If the BHRM bit (in the BCR register) is set to “1”, then the BHR reset is masked; if the BHRM bit (in the BCR register) is cleared (default), then the BHR reset is enabled. • The BHR bit does not return to the reset value during the software (MUR) reset. 0 De-assert Hardware reset to the Processor B. (default) 1 Assert Hardware reset to the Processor B.
3 -	This field is reserved. Reserved
ABFn	For n = {0, 1, 2} Processor A to Processor B Flag n. (Read-Write) • ABFn bit is a read-write flag that is reflected in Fn bit in the BSR register on the Processor B-side. • ABFn bit is cleared when the MU resets. 0 N/A. Self clearing bit (default). 1 Asserts the Processor A MU reset.

4.3.5 MU Processor B-side Memory Map/Register Definition

This section contains the detailed register descriptions for the Processor B-side MU registers.

MUB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30AB_0000	Processor B Transmit Register 0 (MUB_BTR0)	32	R/W	0000_0000h	4.3.5.1/142
30AB_0004	Processor B Transmit Register 1 (MUB_BTR1)	32	R/W	0000_0000h	4.3.5.2/143
30AB_0008	Processor B Transmit Register 2 (MUB_BTR2)	32	R/W	0000_0000h	4.3.5.3/144
30AB_000C	Processor B Transmit Register 3 (MUB_BTR3)	32	R/W	0000_0000h	4.3.5.4/144
30AB_0010	Processor B Receive Register 0 (MUB_BRR0)	32	R	0000_0000h	4.3.5.5/145
30AB_0014	Processor B Receive Register 1 (MUB_BRR1)	32	R	0000_0000h	4.3.5.6/146
30AB_0018	Processor B Receive Register 2 (MUB_BRR2)	32	R	0000_0000h	4.3.5.7/146
30AB_001C	Processor B Receive Register 3 (MUB_BRR3)	32	R	0000_0000h	4.3.5.8/147
30AB_0020	Processor B Status Register (MUB_BSR)	32	R/W	00F0_0080h	4.3.5.9/148
30AB_0024	Processor B Control Register (MUB_BCR)	32	R/W	0000_0000h	4.3.5.10/ 151

4.3.5.1 Processor B Transmit Register 0 (MUB_BTR0)

Use Processor B Transmit Register 0 (BTR0, 32-bit, write-only) to transmit a message or data to the Processor A.

- You can only write to the BTR0 register when the TE0 bit in BSR register is set to “1”.
- Reading the BTR0 register returns all zeros.

Address: 30AB_0000h base + 0h offset = 30AB_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BTR0																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MUB_BTR0 field descriptions

Field	Description
BTR0	Processor B Transmit Register 0. (Write-only)

MUB_BTR0 field descriptions (continued)

Field	Description
	• Data written to the BTR0 register is reflected on the Processor A-side in the Processor A Receive Register 0 (ARR0). The BTR0 and ARR0 registers are not double-buffered—a write to the BTR0 register overrides the data readable at the ARR0 register. • A write to the transmit register clears a “transmitter empty” bit (TE0) in the Processor B Status Register (BSR) on the transmitter side, and sets a “receiver full” bit (RF0) in the Processor A Status Register (ASR) on the receiver side (optionally triggering an interrupt 0 on the Processor A-side). • Any write to the BTR0 register will update all status information.

4.3.5.2 Processor B Transmit Register 1 (MUB_BTR1)

Use Processor B Transmit Register 1 (BTR1, 32-bit, write-only) to transmit a message or data to the Processor A.

- You can only write to the BTR1 register when the TE1 bit in BSR register is set to “1”.
- Reading the BTR1 register returns all zeros.

Address: 30AB_0000h base + 4h offset = 30AB_0004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BTR1																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MUB_BTR1 field descriptions

Field	Description
BTR1	Processor B Transmit Register 1. (Write-only) • Data written to the BTR1 register is reflected on the Processor A-side in the Processor A Receive Register 1 (ARR1). The BTR1 and ARR1 registers are not double-buffered—a write to the BTR1 register overrides the data readable at the ARR1 register. • A write to the transmit register clears a “transmitter empty” bit (TE1) in the Processor B Status Register (BSR) on the transmitter side, and sets a “receiver full” bit (RF1) in the Processor A Status Register (ASR) on the receiver side (optionally triggering an interrupt 1 on the Processor A-side). • Any write to the BTR1 register will update all status information.

4.3.5.3 Processor B Transmit Register 2 (MUB_BTR2)

Use Processor B Transmit Register 2 (BTR2, 32-bit, write-only) to transmit a message or data to the Processor A.

- You can only write to the BTR2 register when the TE2 bit in BSR register is set to "1".
- Reading the BTR2 register returns all zeros.

Address: 30AB_0000h base + 8h offset = 30AB_0008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BTR2																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MUB_BTR2 field descriptions

Field	Description
BTR2	Processor B Transmit Register 2. (Write-only) • Data written to the BTR2 register is reflected on the Processor A-side in the Processor A Receive Register 2 (ARR2). The BTR2 and ARR2 registers are not double-buffered—a write to the BTR2 register overrides the data readable at the ARR2 register. • A write to the transmit register clears a "transmitter empty" bit (TE2) in the Processor B Status Register (BSR) on the transmitter side, and sets a "receiver full" bit (RF2) in the Processor A Status Register (ASR) on the receiver side (optionally triggering an interrupt 2 on the Processor A-side). • Any write to the BTR2 register will update all status information.

4.3.5.4 Processor B Transmit Register 3 (MUB_BTR3)

Use Processor B Transmit Register 3 (BTR3, 32-bit, write-only) to transmit a message or data to the Processor A.

- You can only write to the BTR3 register when the TE3 bit in BSR register is set to "1".
- Reading the BTR3 register returns all zeros.

Address: 30AB_0000h base + Ch offset = 30AB_000Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BTR3																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MUB_BTR3 field descriptions

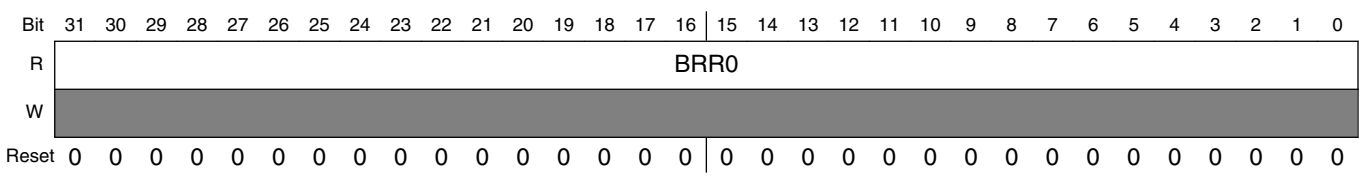
Field	Description
BTR3	Processor B Transmit Register 3. (Write-only) • Data written to the BTR3 register is reflected on the Processor A-side in the Processor A Receive Register 3 (ARR3). The BTR3 and ARR3 registers are not double-buffered—a write to the BTR3 register overrides the data readable at the ARR3 register. • A write to the transmit register clears a “transmitter empty” bit (TE3) in the Processor B Status Register (BSR) on the transmitter side, and sets a “receiver full” bit (RF3) in the Processor A Status Register (ASR) on the receiver side (optionally triggering an interrupt 3 on the Processor A-side). • Any write to the BTR3 register will update all status information.

4.3.5.5 Processor B Receive Register 0 (MUB_BRR0)

Use Processor B Receive Register 0 (BRR0, 32-bit, read-only) to receive a message or data from the Processor A.

- Data written to the ATR0 register is immediately reflected in the BRR0 register.
- You can only read the BRR0 register when the RF0 bit in the BSR register is set to “1”.
- Writing to the BRR0 register generates an error response to the Processor B.

Address: 30AB_0000h base + 10h offset = 30AB_0010h



MUB_BRR0 field descriptions

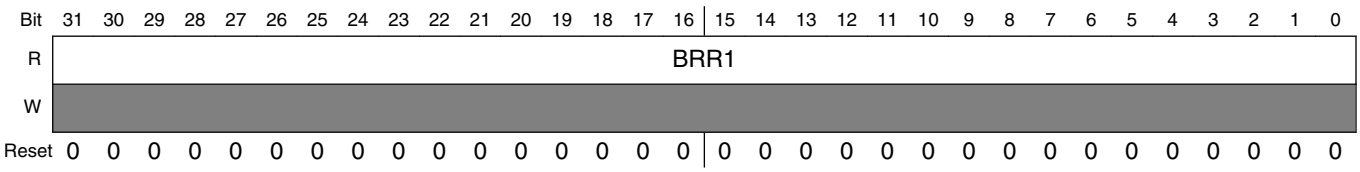
Field	Description
BRR0	Processor B Receive Register 0. (Read-only) • Reflects the data written to Processor A Transmit Register 0 (ATR0). • Reading the BRR0 register clears the “receiver full” bit (RF0) in the Processor B Status Register (BSR) on the receiver side, and sets the “transmitter empty” bit (TE0) in the Processor A Status Register on the transmitter side (optionally triggering a transmit interrupt 0 on the Processor A-side). • Any read of the BRR0 register will update all status information.

4.3.5.6 Processor B Receive Register 1 (MUB_BRR1)

Use Processor B Receive Register 1 (BRR1, 32-bit, read-only) to receive a message or data from the Processor A.

- Data written to the ATR1 register is immediately reflected in the BRR1 register.
- You can only read the BRR1 register when the RF1 bit in the BSR register is set to "1".
- Writing to the BRR1 register generates an error response to the Processor B.

Address: 30AB_0000h base + 14h offset = 30AB_0014h



MUB_BRR1 field descriptions

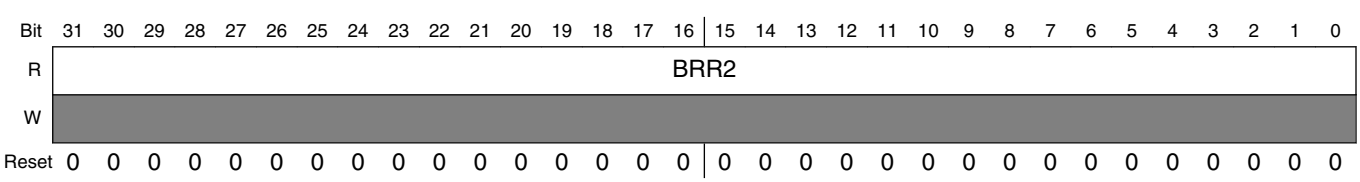
Field	Description
BRR1	Processor B Receive Register 1. (Read-only) • Reflects the data written to Processor A Transmit Register 1 (ATR1). • Reading the BRR1 register clears the "receiver full" bit (RF1) in the Processor B Status Register (BSR) on the receiver side, and sets the "transmitter empty" bit (TE1) in the Processor A Status Register on the transmitter side (optionally triggering a transmit interrupt 1 on the Processor A-side). • Any read of the BRR1 register will update all status information.

4.3.5.7 Processor B Receive Register 2 (MUB_BRR2)

Use Processor B Receive Register 2 (BRR2, 32-bit, read-only) to receive a message or data from the Processor A.

- Data written to the ATR2 register is immediately reflected in the BRR2 register.
- You can only read the BRR2 register when the RF2 bit in the BSR register is set to "1".
- Writing to the BRR2 register generates an error response to the Processor B.

Address: 30AB_0000h base + 18h offset = 30AB_0018h



MUB_BRR2 field descriptions

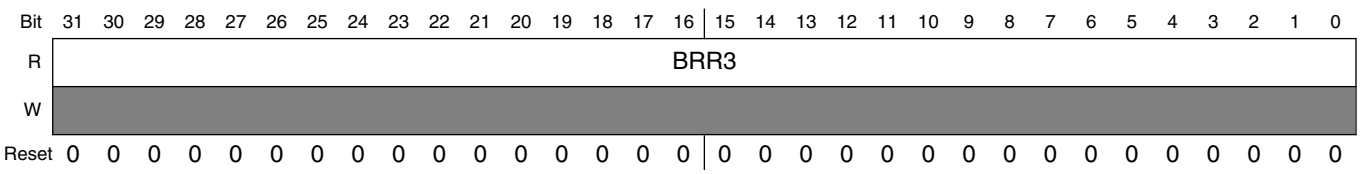
Field	Description
BRR2	Processor B Receive Register 2. (Read-only) • Reflects the data written to Processor A Transmit Register 1 (ATR2). • Reading the BRR2 register clears the “receiver full” bit (RF2) in the Processor B Status Register (BSR) on the receiver side, and sets the “transmitter empty” bit (TE2) in the Processor A Status Register on the transmitter side (optionally triggering a transmit interrupt 2 on the Processor A-side). • Any read of the BRR2 register will update all status information.

4.3.5.8 Processor B Receive Register 3 (MUB_BRR3)

Use Processor B Receive Register 3 (BRR3, 32-bit, read-only) to receive a message or data from the Processor A.

- Data written to the ATR3 register is immediately reflected in the BRR3 register.
- You can only read the BRR3 register when the RF3 bit in the BSR register is set to “1”.
- Writing to the BRR3 register generates an error response to the Processor B.

Address: 30AB_0000h base + 1Ch offset = 30AB_001Ch



MUB_BRR3 field descriptions

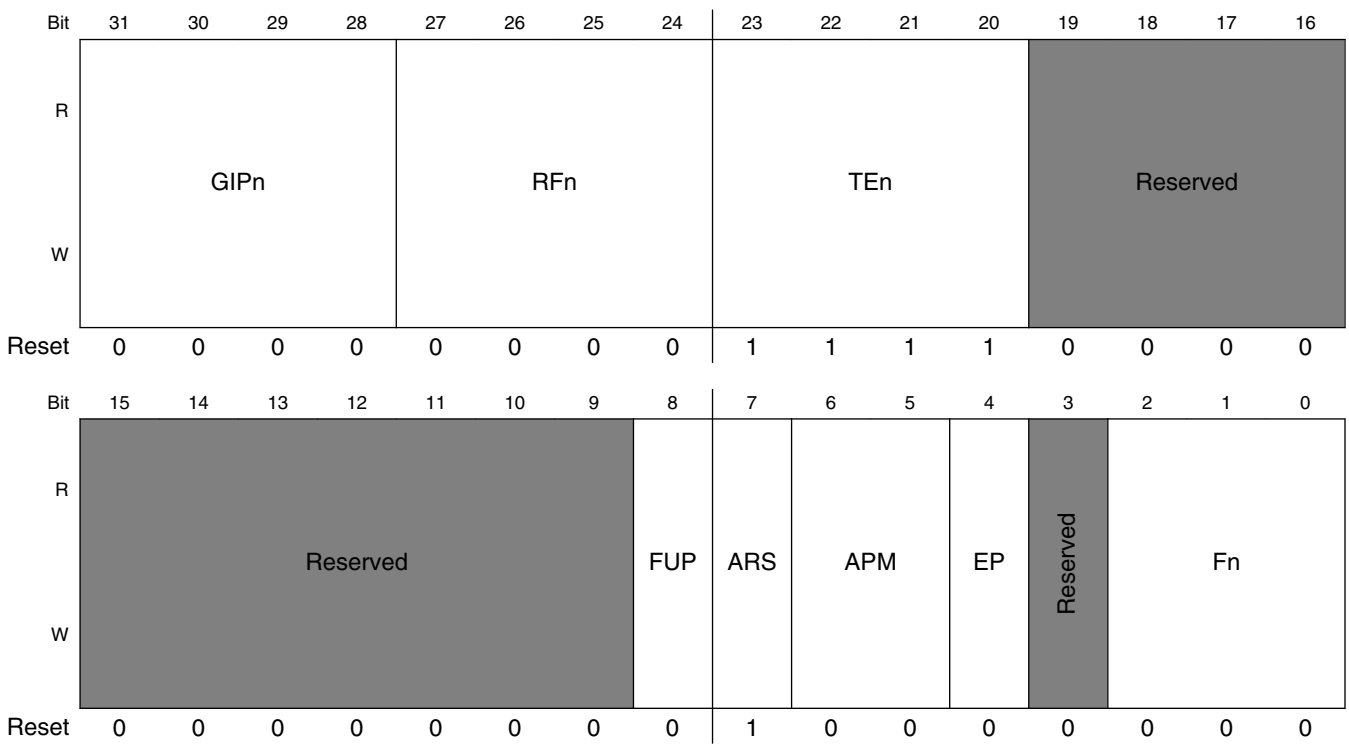
Field	Description
BRR3	Processor B Receive Register 3. (Read-only) • Reflects the data written to Processor A Transmit Register 3 (ATR3). • Reading the BRR3 register clears the “receiver full” bit (RF3) in the Processor B Status Register (BSR) on the receiver side, and sets the “transmitter empty” bit (TE3) in the Processor A Status Register on the transmitter side (optionally triggering a transmit interrupt 3 on the Processor A-side). • Any read of the BRR3 register will update all status information.

4.3.5.9 Processor B Status Register (MUB_BSR)

Use the Processor B Status Register (BSR, 32-bit, read-write) to show interrupt status from the Processor B, general purpose flags, the Processor A power mode, and to set dual function control-status bits.

- Dual-purpose bits are set by the Processor B-side programmer, and cleared by the MU logic.

Address: 30AB_0000h base + 20h offset = 30AB_0020h



MUB_BSR field descriptions

Field	Description
31–28 GIPn	For n = {0, 1, 2, 3} Processor B General Interrupt Request n Pending. (Read-Write) • GIPn bit signals the Processor B that the GIRn bit in the ACR register on the Processor A-side was set from “0” to “1”. If the GIEn bit in the BCR register is set to “1”, a General Interrupt n request is issued. • The GIPn bit is cleared by writing it back as “1”. Writing “0”, or writing “1” when the GIPn bit is cleared is ignored. Use this feature in the interrupt routine, where the GIPn bit is cleared in order to de-assert the interrupt request source at the interrupt controller. • GIPn bit is cleared when the MU is reset. 0 Processor B general purpose interrupt n is not pending. (default) 1 Processor B general purpose interrupt n is pending.

Table continues on the next page...

MUB_BSR field descriptions (continued)

Field	Description
27–24 RFn	For n = {0, 1, 2, 3} Processor B Receive Register n Full. (Read-only) • RFn bit signals to the Processor B-side that new data was written by the Processor A to the ATRn register, and is ready to be read by the Processor B in the BRRn register. • The RFn bit is set to “1” when the ATRn register is written on the Processor A-side. • After the RFn bit is set to “1”, the RFn bit signals the Processor B-side that new data is ready to be read by the Processor B in the BRRn register, and a Receive n interrupt is issued on the Processor A-side (if the RIEn bit in the BCR register has been set to “1”). • RFn bit is cleared when the BRRn register is read, and when the MU is reset. 0 BRRn register is not full (default). 1 BRRn register has received data from ATRn register and is ready to be read by the Processor B.
23–20 TEn	For n = {0, 1, 2, 3} Processor B Transmit Register n Empty. (Read-only) • When TEn = “1”, it signals to the Processor B-side that the BTRn register is ready to be written on the Processor B-side. • The TEn bit is set to “1” after the ARRn register is read on the Processor A-side. • Setting TEn bit will issue a transmit n interrupt on the Processor B-side (if the TIEn bit in the BCR register is set to “1”. • TEn bit is cleared after the BTRn register is written on the Processor B-side. • TEn bit is set to “1” when the MU is reset. 0 BTRn register is not empty. 1 BTRn register is empty (default).
19–9 Reserved	This field is reserved.
8 FUP	Processor B Flags Update Pending. (Read-only) • FUP bit is set to “1” when the Processor B-side sends a Flags Update request to the Processor A-side. • A Flags Update request is generated when the BAF[2:0] bits of the BCR register change. No flag update changes are allowed while the FUP bit is set to “1”. Any write to the BAF[2:0] bits, while the FUP bit is set to “1”, will not generate a Flags Update event, and the BAF[2:0] bits will stay unchanged. • FUP bit is cleared when this Flags Update request is internally acknowledged (that the flag is updated) from the MU Processor A-side, and during MU reset. 0 No flags updated, initiated by the Processor B, in progress (default) 1 Processor B initiated flags update, processing
7 ARS	Processor A Reset State. (Read-only) • ARS bit indicates if the Processor A-side of the MU is in a reset state or not. • If the ARS bit is set to “1”, then the Processor A-side of the MU is still in the reset state. • If the ARS bit is cleared, then both the Processor A and the Processor A-side of the MU are out of reset. • The ARS bit is set to “1” during: a Processor A system reset, or an MU reset (caused by setting the MUR bit at the BCR register).

Table continues on the next page...

MUB_BSR field descriptions (continued)

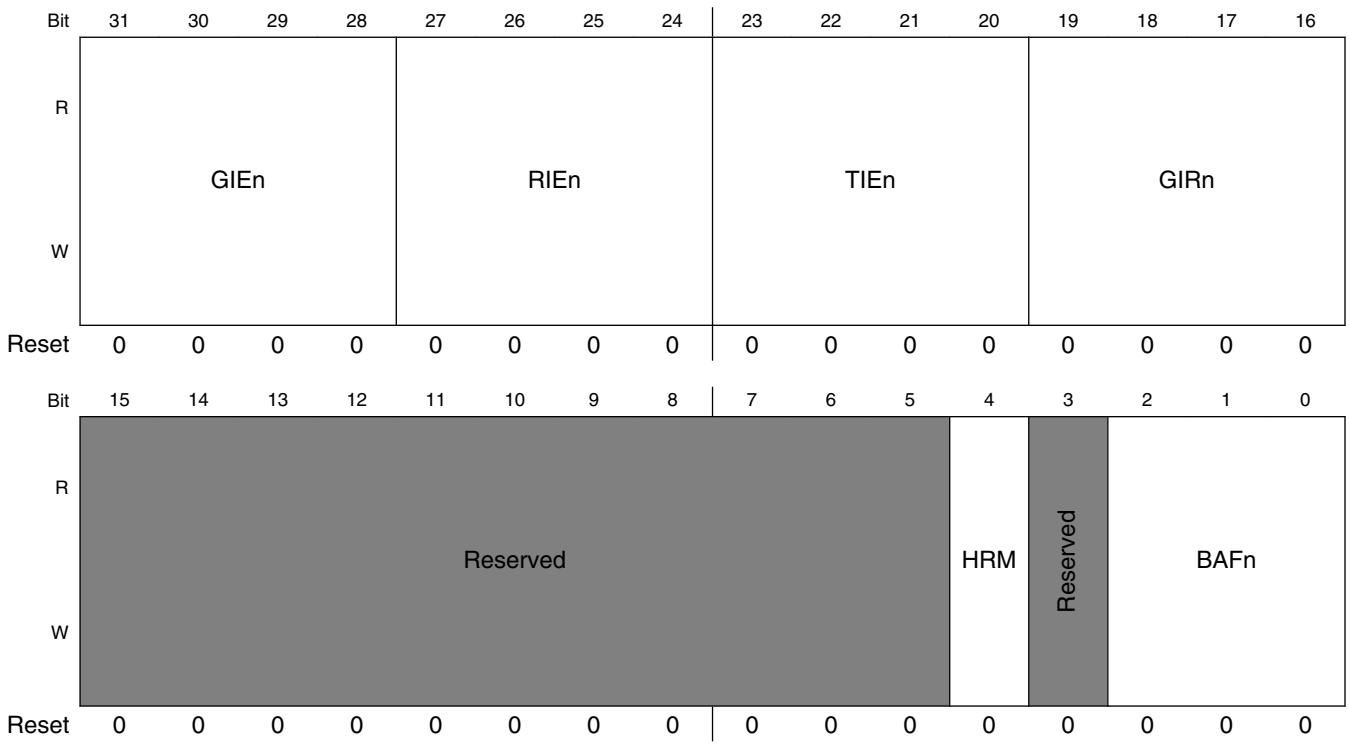
Field	Description
	• The ARS bit is cleared when the reset sequence on the Processor A-side of the MU ends. After issuing any of the three reset events mentioned previously, you should verify that the ARS bit is cleared before starting any accesses. • When Processor B side of MU comes out of reset ARS bit has value “1”(default).Then Processor B sees the status of Processor A side and if Processor A has come out of reset then ARS bit goes low.This takes 5-6 clock cycles.And if you read ARS bit now you will see it as low although its reset value was "1" . 0 The Processor A or the Processor A-side of the MU is not in reset. 1 The Processor A or the Processor A-side of the MU is in reset.
6–5 APM	Processor A Power Mode. (Read-only) • APM[1:0] bits indicate the Processor A power mode. 00 The System is in Run Mode. 01 The System is in WAIT Mode. 10 Reserved. 11 The System is in STOP Mode.
4 EP	Processor B-Side Event Pending. (Read-only) • EP bit is set to “1” when the Processor B-side mechanism sends an event update request to the Processor A-side. • EP bit is cleared when the event update acknowledge is received. An “event” is any hardware message that is reflected in the ASR register on the Processor A-side (for example, “transmit register 0 written”). During normal operations, you do not have to deal with the state of the EP bit because the event update mechanism works automatically. • To ensure events have been posted to Processor A before entering STOP mode, you should verify that the EP bit is cleared. If EP bit is set to “1”, you should wait and continue to poll it (EP bit) before entering STOP mode. • Reading the BSR register (to check the EP bit) should be the last access to the MU that should be performed before entering STOP mode; otherwise, the EP bit may be set by subsequent additional actions. • Due to Processor B pipeline effects, three NOP operations (or their timing equivalent) should be given after an instruction that sets an event before the EP bit can reflect this event. • The EP bit is cleared when the MU resets. 0 The Processor B-side event is not pending (default). 1 The Processor B-side event is pending.
3 Reserved	This field is reserved.
Fn	For n = {0, 1, 2} Processor B-Side Flag n. (Read-only) • Fn bit is the Processor B-side flag that reflects the values written to the ABFn bit in the Processor A control register. • Every time that the ABFn bit is written, the ABFn bit write event updates the Fn bit after the event update latency, which is measured in terms of the number of clocks of the Processor A and the Processor B. 0 ABFn bit in ACR register is written 0 (default). 1 ABFn bit in ACR register is written 1.

4.3.5.10 Processor B Control Register (MUB_BCR)

Use the Processor B Control Register (BCR, 32-bit, read-write) to enable the MU interrupts on the Processor B-side, and trigger events and interrupts on the Processor A-side (wake from STOP, hardware reset, flag update).

For the fields GIEn, RIEn, TIEn and GIRn, n=0 corresponds to the high order bit and n=3 corresponds to the low order bit.

Address: 30AB_0000h base + 24h offset = 30AB_0024h



MUB_BCR field descriptions

Field	Description
31–28 GIEn	For n = {0, 1, 2, 3} Processor B General Purpose Interrupt Enable n. (Read-Write) • GIEn bit enables Processor B General Interrupt n. • If GIEn bit is set to “1” (enabled), then a General Interrupt n request is issued when the GIPn bit in the BSR register is set to “1”. • If GIEn is cleared (disabled), then the value of the GIPn bit is ignored and no General Interrupt n request will be issued. • GIEn bit is cleared when the MU resets. 0 Disables Processor B General Interrupt n. (default) 1 Enables Processor B General Interrupt n.
27–24 RIEn	For n = {0, 1, 2, 3} Processor B Receive Interrupt Enable n. (Read-Write)

Table continues on the next page...

MUB_BCR field descriptions (continued)

Field	Description
	• RIEn bit enables Processor B Receive Interrupt n. • If RIEn bit is set to “1” (enabled), then an Processor B Receive Interrupt n request is issued when the RFn bit in the BSR register is set to “1”. • If RIEn bit is cleared (disabled), then the value of the RFn bit is ignored and no Processor B Receive Interrupt n request will be issued. • RIEn bit is cleared when the MU resets. 0 Disables Processor B Receive Interrupt n. (default) 1 Enables Processor B Receive Interrupt n.
23–20 TIEn	For n = {0, 1, 2, 3} Processor B Transmit Interrupt Enable n. (Read-Write) • TIEn bit enables Processor B Transmit Interrupt n. • If TIEn bit is set to “1” (enabled), then an Processor B Transmit Interrupt n request is issued when the TEn bit in the BSR register is set to “1”. • If TIEn bit is cleared (disabled), then the value of the TEn bit is ignored and no Processor B Transmit Interrupt n request will be issued. • TIEn bit is cleared when the MU resets. 0 Disables Processor B Transmit Interrupt n. (default) 1 Enables Processor B Transmit Interrupt n.
19–16 GIRn	For n = {0, 1, 2, 3} Processor B General Purpose Interrupt Request n. (Read-Write) • Writing “1” to the GIRn bit sets the GIPn bit in the ASR register on the Processor A-side. If the GIEn bit in the ACR register is set to “1” on the Processor A-side, a General Purpose Interrupt n request is triggered. • The GIRn bit is cleared if the GIPn bit (in the ASR register on the Processor A-side) is cleared by writing it (GIPn bit) as “1”, thereby signalling the Processor B that the interrupt was accepted (cleared by the software). The GIPn bit cannot be written as “0” on the Processor B-side. • To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit). • GIRn bit is cleared when the MU resets. 0 Processor B General Interrupt n is not requested to the Processor A (default). 1 Processor B General Interrupt n is requested to the Processor A.
15–5 Reserved	This field is reserved.
4 HRM	Processor B Hardware Reset Mask. (Read-Write) • The Processor A can give a hardware reset to the Processor B by setting the BHR bit in the ACR Register to “1”. • When the HRM bit is set to “1” by the Processor B, the BHR reset issued by the Processor A is masked (disabled by the Processor B). • When the HRM bit is cleared, the BHR reset issued by the Processor A to the Processor B is not masked (enabled by the Processor B). 0 BHR bit in ACR is not masked, enables the hardware reset to the Processor B (default after hardware reset). 1 BHR bit in ACR is masked, disables the hardware reset request to the Processor B.
3 Reserved	This field is reserved.
BAFn	For n = {0, 1, 2} Processor B to Processor A Flag n. (Read-Write)

Table continues on the next page...

MUB_BCR field descriptions (continued)

Field	Description
	• BAFn bit is a read-write flag that is reflected in Fn bit in the ASR register on the Processor A-side. • BAFn bit is cleared when the MU resets. 0 Clears the Fn bit in the ASR register. 1 Sets the Fn bit in the ASR register.

4.4 Semaphore (SEMA4)

4.4.1 Overview

The IPS_Semaphores module provides a platform IPS slave device which implements 16 hardware-enforced gates with the following features:

- Module definition supports 16 hardware-enforced gates in a dual-processor configuration, where cp0 is core processor 0 and cp1 is core processor 1
  - Hardware gates appear as a 16-entry byte-size array with read and write accesses
    - Processors lock gates by writing "processor_number+1" to the appropriate gate and must read back the gate value to verify the lock operation was successful
    - Once locked, the gate is unlocked by a write of zeroes from the locking processor
  - Optional interrupt notification after a failed lock write provides a mechanism to indicate when the gate is unlocked
  - Secure reset mechanisms are supported to clear the contents of individual semaphore gates or notification logic, as well as a clear_all capability
  - Programming model allocates memory space to support up to 8 processors and up to 64 gates

A simplified block diagram of the Semaphores module is shown in Figure 4-10. In the diagram, the register blocks named gate0, gate1,…, gate 15 include the finite state machines implementing the semaphore gates plus the interrupt notification logic.

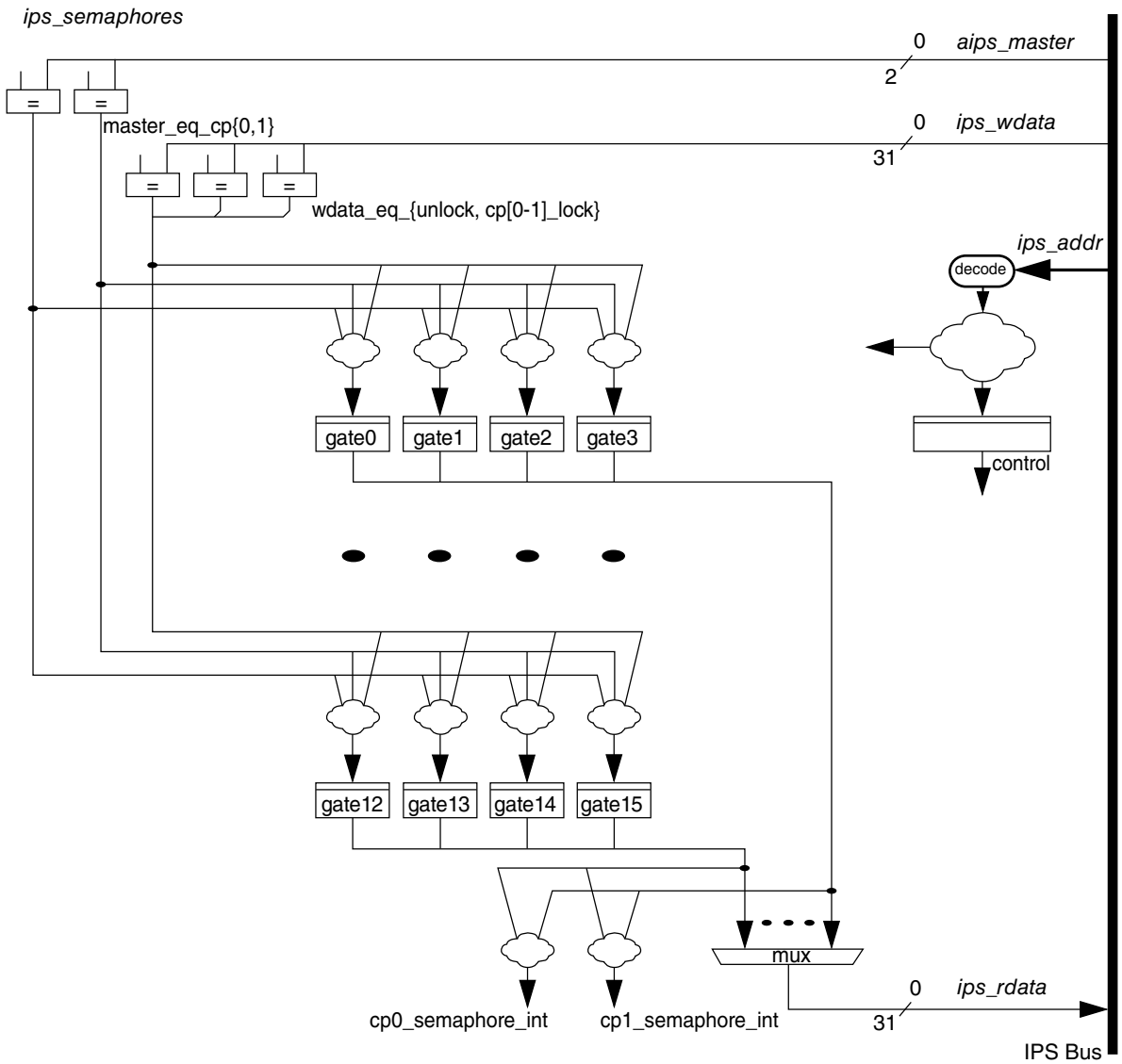


Figure 4-10. IPS_Semaphores Block Diagram

4.4.1.1 Features

The Semaphores module implements hardware-enforced semaphores as an IPS-mapped slave peripheral device. The feature set includes:

- Support for 16 hardware-enforced gates in a dual-processor configuration
  - Each hardware gate appears as a 3-state, 2-bit state machine, with all 16 gates mapped as a byte-size array
    - 3-state implementation

      if gate = 0b00, then state = unlocked
      if gate = 0b01, then state = locked by processor 0

if gate = 0b10, then state = locked by processor 1

- Uses the bus master number as a reference attribute plus the specified data patterns to validate all write operations
- Once locked, the gate can (and must) be unlocked by a write of zeroes from the locking processor

- Optional interrupt notification after a failed lock write provides a mechanism to indicate when the gate is unlocked
- Secure reset mechanisms are supported to clear the contents of individual gates or notification logic, as well as a clear_all capability

- Memory-mapped IPS slave peripheral platform module
  - Interface to the IPS bus for programming-model accesses
  - Two outputs (one per processor) for interrupt notification of failed lock writes

4.4.1.2 Modes of Operation

The Semaphores module does not support any special modes of operation. As a slave peripheral memory-mapped device located on the platform's IPS slave bus, it responds based strictly on the memory addresses of the connected bus. The IPS bus is used to access the Semaphores ' programming model.

4.4.2 External Signal Description

The Semaphores module does not include any external interfaces.

4.4.3 Functional Description

In this section, the functional operation of the Semaphores module, specifically the state machines of the SEMA4_GATEn and SEMA4_CPnNTF registers are detailed.

4.4.3.1 SEMA4_GATEn Operation

Recall each of the SEMA4_GATEn registers implements a 2-bit, 3-state machine. The state transitions for each gate are shown in the following figure.

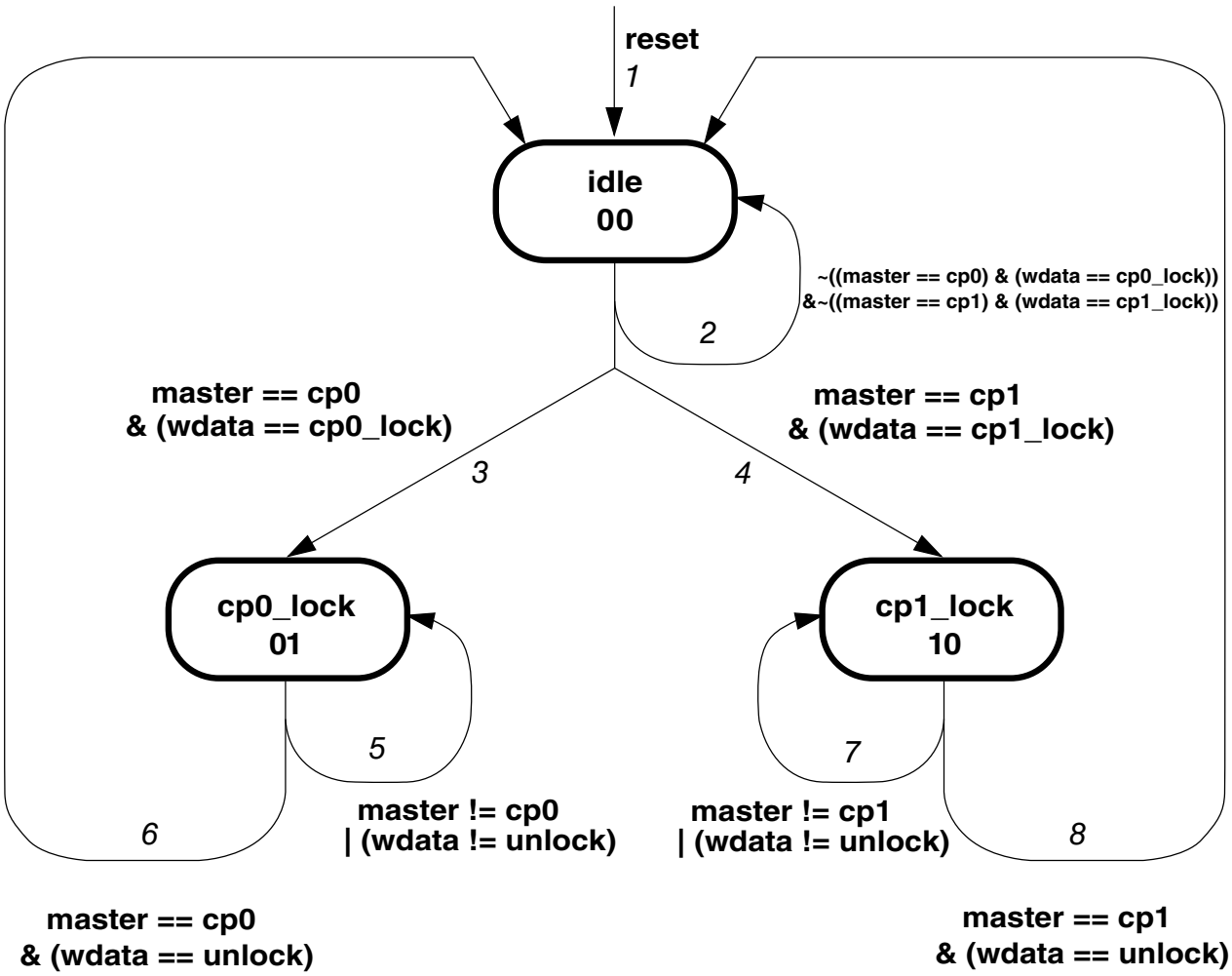


Figure 4-11. SEMA4_GATEn State Machine

The bus master number is used to identify core processor 0 (cp0) or core processor 1 (cp1). The Standard (or Reduced) Product Platform passes the AHB bus master number (`hmaster[2:0]`) through the AIPS (or AIPS-Lite) controller and drives an `aips_master[2:0]` output to the Semaphores module as an IPS sideband signal.

The state transitions for SEMA4_GATEn are defined in the following table.

Table 4-16. SEMA4_GATEn State Transitions

Current State	Next State	Transition	Description
–	idle	1	Any reset, whether a system reset or an individual gate reset, unconditionally forces the gate into the idle state.
idle	idle	2	Unless a write of the appropriate lock value from the corresponding processor occurs, the gate remains in the idle state.
idle	cp0_lock	3	When a write of the "cp0_lock" data value is initiated by processor 0, the gate transitions into the cp0_lock state.

Table continues on the next page...

Table 4-16. SEMA4_GATEn State Transitions (continued)

Current State	Next State	Transition	Description
idle	cp1_lock	4	When a write of the "cp1_lock" value is initiated by processor 1, the gate transitions into the cp1_lock state.
cp0_lock	cp0_lock	5	Once in this state, the gate remains here if any attempted write is not from cp0 with the unlock data value.
cp0_lock	idle	6	The gate returns to the idle (unlocked) state once a write from cp0 with the unlock data value occurs.
cp1_lock	cp1_lock	7	Once in this state, the gate remains here if any attempted write is not from cp1 with the unlock data value.
cp1_lock	idle	8	The gate returns to the idle (unlocked) state once a write from cp1 with the unlock data value occurs.

4.4.3.2 SEMA4_CPnNTF Operation

The failed lock write notification interrupt request is implemented in a 3-bit, 5-state machine which records failed lock attempts and transitions based on gate locking and unlocking. Two specific states are encoded and program-visible as SEMA4_CP0NTF[GNn] and SEMA4_CP1NTF[GNn]. See the following figure.

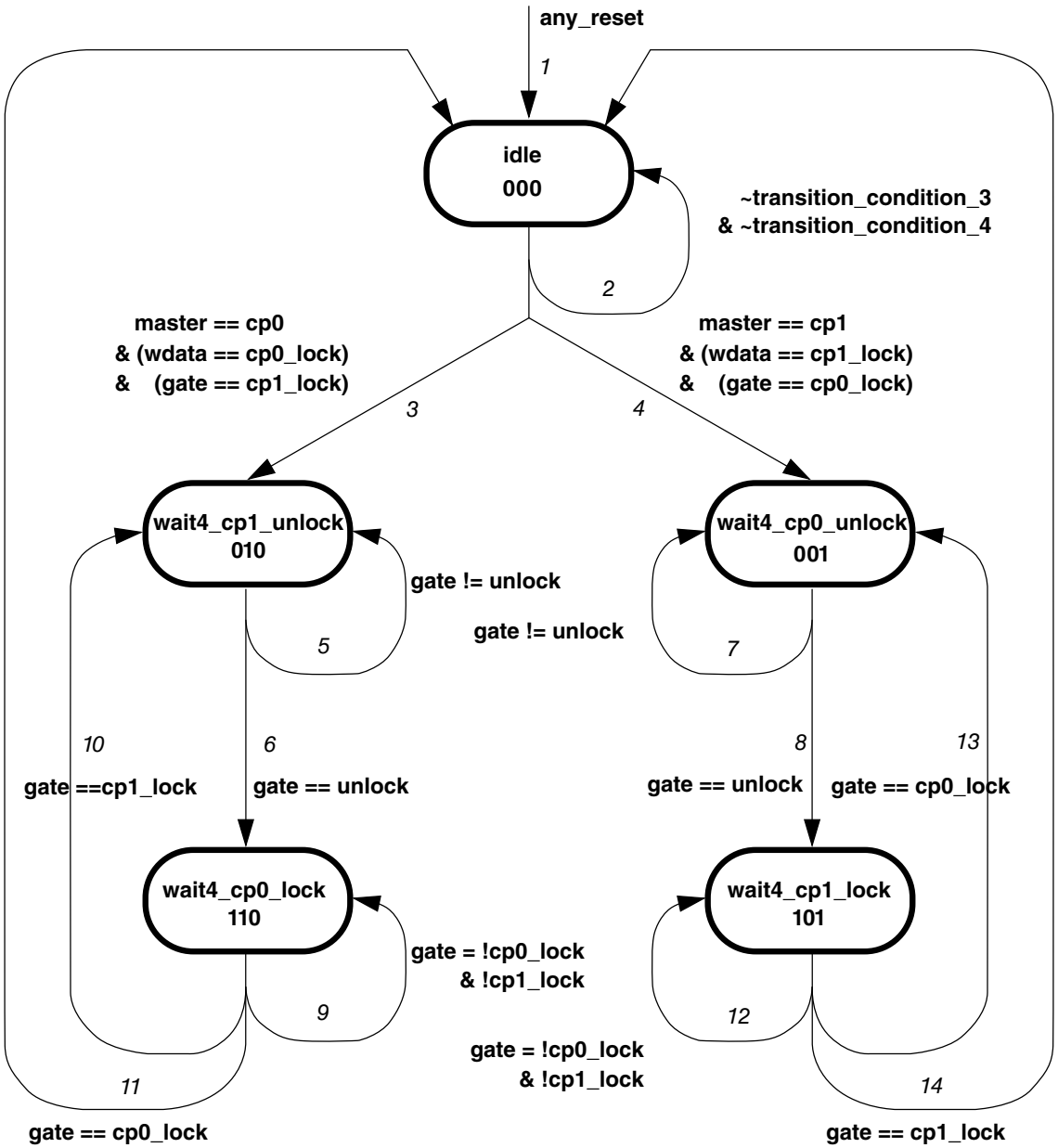


Figure 4-12. IRQ Notification State Machine

The state transitions of the IRQ notification function are defined in the following. Specific states of this machine are program-visible as the SEMA4_CPnNTF registers. In particular, two states are program-visible:

```
if state = wait4_cp0_lock (0b110) // generate cp0_semaphore_int if properly enabled
               then SEMA4_CP0NTF[GNn] = 1; else SEMA4_CP0NTF[GNn] = 0
if state = wait4_cp1_lock (0b101) // generate cp1_semaphore_int if properly enabled
               then SEMA4_CP1NTF[GNn] = 1; else SEMA4_CP1NTF[GNn] = 0
```

Table 4-17. IRQ Notification State Transitions

Current State	Next State	Transition	Description
–	idle	1	Any reset, including a system reset or an individual notification or secure gate reset, unconditionally forces the machine into the idle state.
idle	idle	2	Unless a write of the appropriate lock value from the corresponding processor to an already-locked gate occurs, the machine remains in the idle state.
idle	wait4_cp1_unlock	3	When a write of the "cp0_lock" data value is initiated by processor 0 but the gate is already locked by cp1, the machine transitions into this state, where it waits for cp1 to unlock the gate.
idle	wait4_cp0_unlock	4	When a write of the "cp1_lock" data value is initiated by processor 1but the gate is already locked by cp0, the machine transitions into this state, where it waits for cp0 to unlock the gate
wait4_cp1_unlock	wait4_cp1_unlock	5	Once in this state, the machine remains here until the gate is unlocked.
wait4_cp1_unlock	wait4_cp0_lock	6	From this state, the machine transitions into the next state, waiting for cp0 to lock the gate, once it has been unlocked.
wait4_cp0_unlock	wait4_cp0_unlock	7	Once in this state, the machine remains here until the gate is unlocked.
wait4_cp0_unlock	wait4_cp1_lock	8	From this state, the machine transitions into the next state, waiting for cp1 to lock the gate, once it has been unlocked.
wait4_cp0_lock	wait4_cp0_lock	9	In this state, the machine generates the notification interrupt (if properly-enabled) and remains here until the gate is locked by processor 0 or the gate is again locked by processor 1.
wait4_cp0_lock	wait4_cp1_unlock	10	In this state, the machine generates the notification interrupt (if properly-enabled) and transitions if the gate is again locked by processor 1. With this transition, the notification interrupt request is negated.
wait4_cp0_lock	idle	11	In this state, the machine generates the notification interrupt (if properly-enabled) and transitions if the gate is finally locked by processor 0. With this transition, the notification interrupt request is negated.
wait4_cp1_lock	wait4_cp1_lock	12	In this state, the machine generates the notification interrupt (if properly-enabled) and remains here until the gate is locked by processor 1or the gate is again locked by processor 0.
wait4_cp1_lock	wait4_cp0_unlock	13	In this state, the machine generates the notification interrupt (if properly-enabled) and transitions if the gate is again locked by processor 0. With this transition, the notification interrupt request is negated.
wait4_cp1_lock	idle	14	In this state, the machine generates the notification interrupt (if properly-enabled) and transitions if the gate is finally locked by processor 1. With this transition, the notification interrupt request is negated.

The Semaphores module generates two interrupt request output signals, one per processor, combining the SEMA4_CPnINE and SEMA4_CPnNTF registers, where the boolean equations are:

```
cp0_semaphore_int
    =    sema4_cp0ine[ine0]    &    sema4_cp0ntf[gn0]
    |    sema4_cp0ine[ine1]    &    sema4_cp0ntf[gn1]
    |    sema4_cp0ine[ine2]    &    sema4_cp0ntf[gn2]
    ...
    |    sema4_cp0ine[ine15]    &    sema4_cp0ntf[gn15]
cp1_semaphore_int
    =    sema4_cp1ine[ine0]    &    sema4_cp1ntf[gn0]
    |    sema4_cp1ine[ine1]    &    sema4_cp1ntf[gn1]
    |    sema4_cp1ine[ine2]    &    sema4_cp1ntf[gn2]
    ...
    |    sema4_cp1ine[ine15]    &    sema4_cp1ntf[gn15]
```

4.4.4 Initialization Information

The reset state of the IPS_Semaphores module allows it to begin operation without the need for any further initialization. All the internal state machines are cleared by any reset event, allowing the module to immediately begin operation.

4.4.5 Application Information

In an operational multi-core system, most interactions involving the Semaphores module involves reads and writes to the SEMA4_GATEn registers for implementation of the hardware-enforced software gate functions. Typical code segments for gate functions perform the following operations:

- To lock (close) a gate
    - The processor performs a byte write of "logical_processor_number + 1" to gate[i]
    - The processor reads back gate[i] and checks for a value of "logical_processor_number + 1"

If the compare indicates the expected value, then the gate is locked; proceed with the protected code segment. If the compare does not indicate the expected value, the lock operation failed; repeat the process beginning with byte write to gate[i] in spin-wait loop, or proceed with another execution path and wait for failed lock interrupt notification.

A simple C-language example of a `gateLock` function is shown in the following figure. This function follows the Hennessy/Patterson example described in Multi-Core Programming 101: Software Gates.

```
#define UNLOCK      0
#define CP0_LOCK    1
#define CP1_LOCK    2

void gateLock (n)
int   n;                                        /* gate number to lock */
{
   int i;
   int current_value;
   int locked_value;

   i = processor_number();  /* obtain logical CPU number */

   if (i == 0)
      locked_value = CP0_LOCK;
   else
      locked_value = CP1_LOCK;

   /* read the current value of the gate and wait until the state == UNLOCK */
   do {
        current_value = gate[n];
   } while (current_value != UNLOCK);

   /* the current value of the gate == UNLOCK. attempt to lock the gate for this
        processor. spin-wait in this loop until gate ownership is obtained */
   do {
      gate[n] = locked_value;  /* write gate with processor_number + 1 */
      current_value = gate[n]; /* read gate to verify ownership was obtained */
   } while (current_value != locked_value);
```

Figure 4-13. Sample gateLock Function

- To unlock (open) a gate
    - After completing the protected code segment, the locking processor performs a byte write of zeroes to gate[i], opening (unlocking) the gate

A few comments on the logical CPU number are appropriate. In this example, a reference to `processor_number()` is used to retrieve this hardware configuration value. Typically, the logical processor numbers are defined by a hardwired input vector to the individual cores. The exact method for accessing the logical processor number varies by architecture. For PowerPC cores, there is a processor ID register (PIR) which is SPR 286 and contains this value. A single instruction can be used to move the contents of the PIR into a general-purpose register: mfspr rx,286 where rx is the destination GPRn. Other architectures may support a specific instruction to move the contents of the logical processor number into a general-purpose register, e.g., rdcpn rx for a "read CPU number" instruction.

If the optional failed lock IRQ notification mechanisms are used, then accesses to the related registers (SEMA4_CPnINE, SEMA4_ CPnNTF) are required. Note that there is no required negation of the failed lock write notification interrupt as the request is automatically negated by the Semaphores module once the gate has been successfully locked by the "failing" processor.

Finally, in the event a system state requires a software-controlled reset of a gate or IRQ notification register(s), accesses to the secure reset control registers (SEMA4_RSTGT, SEMA4_RSTNTF) are required. For these situations, it is recommended that the appropriate IRQ notification enable(s) (SEMA4_CPnINE) bits be disabled *before* initiating the secure reset 2-write sequence to avoid any race conditions involving spurious notification interrupt requests.

4.4.6 Memory map and register definition

The Semaphores module provides an IPS programming model mapped to an SPP-standard on-platform 16 KB space. The description here specifies a dual-core configuration with 16 semaphore gates. All the register names are prefixed with "Sema4" as an abbreviation for the full module name.

The programming model is referenced using 8-, 16- and 32-bit accesses. Reads can use any reference size, while writes are generally restricted to the size of the register. Exceptions to the write size restrictions are detailed in the individual register descriptions. Attempted references using inappropriate access sizes, to undefined (reserved) addresses, or with a non-supported access type (for example, a write to a read-only register) generate an IPS error termination.

Finally, the programming model allocates space for a definition with up to 64 gates and up to 8 processor cores, even though this definition is considerably larger than any currently-planned module implementations. The number of gates and supported processor cores are independent; there is no relationship between these two system variables.

The 16 KB Semaphores programming model map is shown in the following table.

SEMA4 memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30AC_0000	Semaphores Gate 0 Register (SEMA4_Gate00)	8	R/W	00h	4.4.6.1/163
30AC_0001	Semaphores Gate 1 Register (SEMA4_Gate01)	8	R/W	00h	4.4.6.2/164
30AC_0002	Semaphores Gate 2 Register (SEMA4_Gate02)	8	R/W	00h	4.4.6.3/165
30AC_0003	Semaphores Gate 3 Register (SEMA4_Gate03)	8	R/W	00h	4.4.6.4/166
30AC_0004	Semaphores Gate 4 Register (SEMA4_Gate04)	8	R/W	00h	4.4.6.5/167
30AC_0005	Semaphores Gate 5 Register (SEMA4_Gate05)	8	R/W	00h	4.4.6.6/168
30AC_0006	Semaphores Gate 6 Register (SEMA4_Gate06)	8	R/W	00h	4.4.6.7/169
30AC_0007	Semaphores Gate 7 Register (SEMA4_Gate07)	8	R/W	00h	4.4.6.8/170

Table continues on the next page...

SEMA4 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30AC_0008	Semaphores Gate 8 Register (SEMA4_Gate08)	8	R/W	00h	4.4.6.9/171
30AC_0009	Semaphores Gate 9 Register (SEMA4_Gate09)	8	R/W	00h	4.4.6.10/ 172
30AC_000A	Semaphores Gate 10 Register (SEMA4_Gate10)	8	R/W	00h	4.4.6.11/ 173
30AC_000B	Semaphores Gate 11 Register (SEMA4_Gate11)	8	R/W	00h	4.4.6.12/ 174
30AC_000C	Semaphores Gate 12 Register (SEMA4_Gate12)	8	R/W	00h	4.4.6.13/ 175
30AC_000D	Semaphores Gate 13 Register (SEMA4_Gate13)	8	R/W	00h	4.4.6.14/ 176
30AC_000E	Semaphores Gate 14 Register (SEMA4_Gate14)	8	R/W	00h	4.4.6.15/ 177
30AC_000F	Semaphores Gate 15 Register (SEMA4_Gate15)	8	R/W	00h	4.4.6.16/ 178
30AC_0040	Semaphores Processor n IRQ Notification Enable (SEMA4_CP0INE)	16	R/W	0000h	4.4.6.17/ 179
30AC_0048	Semaphores Processor n IRQ Notification Enable (SEMA4_CP1INE)	16	R/W	0000h	4.4.6.17/ 179
30AC_0080	Semaphores Processor n IRQ Notification (SEMA4_CP0NTF)	16	R	0000h	4.4.6.18/ 181
30AC_0088	Semaphores Processor n IRQ Notification (SEMA4_CP1NTF)	16	R	0000h	4.4.6.18/ 181
30AC_0100	Semaphores (Secure) Reset Gate n (SEMA4_RSTGT)	16	R/W	0000h	4.4.6.19/ 183
30AC_0104	Semaphores (Secure) Reset IRQ Notification (SEMA4_RSTNTF)	16	R/W	0000h	4.4.6.20/ 184

4.4.6.1 Semaphores Gate 0 Register (SEMA4_Gate00)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate

register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + 0h offset = 30AC_0000h

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate00 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.2 Semaphores Gate 1 Register (SEMA4_Gate01)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + 1h offset = 30AC_0001h

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate01 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.3 Semaphores Gate 2 Register (SEMA4_Gate02)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + 2h offset = 30AC_0002h

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate02 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.4 Semaphores Gate 3 Register (SEMA4_Gate03)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + 3h offset = 30AC_0003h

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate03 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine.

Table continues on the next page...

SEMA4_Gate03 field descriptions (continued)

Field	Description
	Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.5 Semaphores Gate 4 Register (SEMA4_Gate04)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + 4h offset = 30AC_0004h

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate04 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation .

Table continues on the next page...

SEMA4_Gate04 field descriptions (continued)

Field	Description
	NOTE: The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.6 Semaphores Gate 5 Register (SEMA4_Gate05)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + 5h offset = 30AC_0005h

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate05 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free).

Table continues on the next page...

SEMA4_Gate05 field descriptions (continued)

Field	Description
	01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.7 Semaphores Gate 6 Register (SEMA4_Gate06)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + 6h offset = 30AC_0006h

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate06 field descriptions

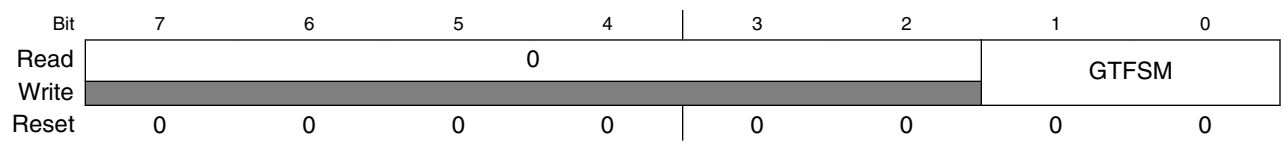
Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.8 Semaphores Gate 7 Register (SEMA4_Gate07)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + 7h offset = 30AC_0007h



SEMA4_Gate07 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.9 Semaphores Gate 8 Register (SEMA4_Gate08)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + 8h offset = 30AC_0008h

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate08 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.10 Semaphores Gate 9 Register (SEMA4_Gate09)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + 9h offset = 30AC_0009h

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate09 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.11 Semaphores Gate 10 Register (SEMA4_Gate10)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + Ah offset = 30AC_000Ah

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate10 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.12 Semaphores Gate 11 Register (SEMA4_Gate11)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + Bh offset = 30AC_000Bh

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate11 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.13 Semaphores Gate 12 Register (SEMA4_Gate12)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + Ch offset = 30AC_000Ch

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate12 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.14 Semaphores Gate 13 Register (SEMA4_Gate13)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + Dh offset = 30AC_000Dh

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate13 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.15 Semaphores Gate 14 Register (SEMA4_Gate14)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + Eh offset = 30AC_000Eh

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate14 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.16 Semaphores Gate 15 Register (SEMA4_Gate15)

Each semaphore gate is implemented in a 2-bit finite state machine, right-justified in a byte data structure. The hardware uses the bus master number in conjunction with the data patterns to validate all attempted write operations. Only processor bus masters can modify the gate registers. Once locked, a gate can (and must) be opened (unlocked) by the locking processor core.

Multiple gate values can be read in a single access, but only a single gate can be updated via a write operation at a time. 16- and 32-bit writes to multiple gates are allowed, but the write data operand must only update the state of a single gate. A byte write data value of 0x03 is defined as "no operation" and does not affect the state of the corresponding gate register. Attempts to write multiple gates in a single aligned access with a size larger than an 8-bit (byte) reference generate an error termination and do not allow any gate state changes

Address: 30AC_0000h base + Fh offset = 30AC_000Fh

Bit	7	6	5	4	3	2	1	0
Read	0						GTFSM	
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_Gate15 field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GTFSM	Gate Finite State Machine. Gate Finite State Machine. The hardware gate is maintained in a 3-state implementation-unlocked, locked by processor 0 or locked by processor 1. For more details, see SEMA4_GATEn Operation . **NOTE:** The state of the gate reflects the last processor that locked it, which can be useful during system debug. 00 The gate is unlocked (free). 01 The gate has been locked by processor 0. 10 The gate has been locked by processor 1. 11 This state encoding is never used and therefore reserved. Attempted writes of 0x03 are treated as "no operation" and do not affect the gate state machine.

4.4.6.17 Semaphores Processor n IRQ Notification Enable (SEMA4_CP*n*INE)

The application of a hardware semaphore module provides an opportunity for implementation of helpful system-level features. An example is an optional mechanism to generate a processor interrupt after a failed lock attempt. Recall traditional software gate functions execute a spin-wait loop in an effort to obtain and lock the referenced gate. With this module, the processor that fails in the lock attempt could continue with other tasks and allow a properly-enabled notification interrupt to return its execution to the original lock function.

The optional notification interrupt function consists of two registers for each processor: an interrupt notification enable register (SEMA4_CPnINE) and the interrupt request register (SEMA4_CPnNTF). To support implementations with more than 16 gates, these registers can be referenced with aligned 16- or 32-bit accesses. For the SEMA4_CPnINE registers, unimplemented bits read as zeroes, and writes are ignored.

Address: 30AC_0000h base + 40h offset + (8d × i), where i=0d to 1d

Bit	15	14	13	12	11	10	9	8
Read Write	INE8	INE9	INE10	INE11	INE12	INE13	INE14	INE15
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read Write	INE0	INE1	INE2	INE3	INE4	INE5	INE6	INE7
Reset	0	0	0	0	0	0	0	0

SEMA4_CP*n*INE field descriptions

Field	Description
15 INE8	Interrupt Request Notification Enable 8. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 8. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
14 INE9	Interrupt Request Notification Enable 9. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 9. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
13 INE10	Interrupt Request Notification Enable 10. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 10. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
12 INE11	Interrupt Request Notification Enable 11. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 11.

Table continues on the next page...

SEMA4_CP*n*INE field descriptions (continued)

Field	Description
	0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
11 INE12	Interrupt Request Notification Enable 12. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 12. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
10 INE13	Interrupt Request Notification Enable 13. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 13. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
9 INE14	Interrupt Request Notification Enable 14. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 14. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
8 INE15	Interrupt Request Notification Enable 15. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 15. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
7 INE0	Interrupt Request Notification Enable 0. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 0. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
6 INE1	Interrupt Request Notification Enable 1. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 1. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
5 INE2	Interrupt Request Notification Enable 2. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 2. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
4 INE3	Interrupt Request Notification Enable 3. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 3. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
3 INE4	Interrupt Request Notification Enable 4. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 4. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
2 INE5	Interrupt Request Notification Enable 5. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 5.

Table continues on the next page...

SEMA4_CP*n*INE field descriptions (continued)

Field	Description
	0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
1 INE6	Interrupt Request Notification Enable 6. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 6. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.
0 INE7	Interrupt Request Notification Enable 7. This field is a bitmap to enable the generation of an interrupt notification from a failed attempt to lock gate 7. 0 The generation of the notification interrupt is disabled. 1 The generation of the notification interrupt is enabled.

4.4.6.18 Semaphores Processor n IRQ Notification (SEMA4_CP*n*NTF)

The Semaphores module optionally allows the processor that fails in the lock attempt to continue with other tasks and allow a properly-enabled notification interrupt to return its execution to the original lock function rather than simply execute in a spin-wait loop.

The optional notification interrupt mechanism consists of two registers for each processor: an interrupt notification enable register (SEMA4_CPnINE) and the read-only notification interrupt request register (SEMA4_CPnNTF). To support implementations with more than 16 gates, these registers can be referenced with aligned 16- or 32-bit accesses. For the SEMA4_CPnNTF registers, unimplemented bits read as zeroes.

The notification interrupt is generated via a unique finite state machine, one per hardware gate. This machine operates in the following manner:

1. When an attempted lock fails, the FSM enters a first state where it waits until the gate is unlocked.
2. Once unlocked, the FSM enters a second state where it generates an interrupt request to the “failed lock” processor.
3. When the “failed lock” processor succeeds in locking the gate, the IRQ is automatically negated and the FSM returns to the idle state. However, if the other processor again locks the gate, the FSM returns to the first state, negates the interrupt request, and then waits for the gate to be unlocked (again).

The notification interrupt request is implemented in a 3-bit, 5-state machine, where tw specific states are encoded and program-visible as SEMA4_CP0NTF[GNn] and SEMA4_CP1NTF[GNn].

Address: 30AC_0000h base + 80h offset + (8d × i), where i=0d to 1d

Bit	15	14	13	12	11	10	9	8
Read	GN8	GN9	GN10	GN11	GN12	GN13	GN14	GN15
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	GN0	GN1	GN2	GN3	GN4	GN5	GN6	GN7
Write								
Reset	0	0	0	0	0	0	0	0

SEMA4_CP*n*NTF field descriptions

Field	Description
15 GN8	Gate 8 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 8. For more details, see SEMA4_CPnNTF Operation .
14 GN9	Gate 9 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 9. For more details, see SEMA4_CPnNTF Operation .
13 GN10	Gate 10 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 10. For more details, see SEMA4_CPnNTF Operation .
12 GN11	Gate 11 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 11. For more details, see SEMA4_CPnNTF Operation .
11 GN12	Gate 12 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 12. For more details, see SEMA4_CPnNTF Operation .
10 GN13	Gate 13 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 13. For more details, see SEMA4_CPnNTF Operation .
9 GN14	Gate 14 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 14. For more details, see SEMA4_CPnNTF Operation .
8 GN15	Gate 15 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 15. For more details, see SEMA4_CPnNTF Operation .
7 GN0	Gate 0 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 0. For more details, see SEMA4_CPnNTF Operation .
6 GN1	Gate 1 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 1. For more details, see SEMA4_CPnNTF Operation .
5 GN2	Gate 2 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 2. For more details, see SEMA4_CPnNTF Operation .
4 GN3	Gate 3 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 3. For more details, see SEMA4_CPnNTF Operation .
3 GN4	Gate 4 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 4. For more details, see SEMA4_CPnNTF Operation .
2 GN5	Gate 5 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 5. For more details, see SEMA4_CPnNTF Operation .
1 GN6	Gate 6 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 6. For more details, see SEMA4_CPnNTF Operation .
0 GN7	Gate 7 Notification. This read-only field is a bitmap of the interrupt request notification from a failed attempt to lock gate 7. For more details, see SEMA4_CPnNTF Operation .

4.4.6.19 Semaphores (Secure) Reset Gate n (SEMA4_RSTGT)

Although the intent of the hardware gate implementation specifies a protocol where the locking processor must unlock the gate, it is recognized that system operation may require a reset function to re-initialize the state of any gate(s) without requiring a system-level reset.

To support this special gate reset requirement, the Semaphores module implements a "secure" reset mechanism which allows a hardware gate (or all the gates) to be initialized by following a specific dual-write access pattern. Using a technique similar to that required for the servicing of a software watchdog timer, the secure gate reset requires two consecutive writes with predefined data patterns from the same processor to force the clearing of the specified gate(s). The required access pattern is:

1. A processor performs a 16-bit write to the SEMA4_RSTGT memory location. The most significant byte (SEMA4_RSTGT[RSTGDP]) must be 0xe2; the least significant byte is a "don't_care" for this reference.
2. The same processor then performs a second 16-bit write to the SEMA4_RSTGT location. For this write, the upper byte (SEMA4_RSTGT[RSTGDP]) is the logical complement of the first data pattern (0x1d) and the lower byte (SEMA4_RSTGT[RSTGTN]) specifies the gate(s) to be reset. This gate field can specify a single gate be cleared, or that all gates are cleared.
3. Reads of the SEMA4_RSTGT location return information on the 2-bit state machine (SEMA4_RSTGT[RSTGSM]) which implements this function, the bus master performing the reset (SEMA4_RSTGT[RSTGMS]) and the gate number(s) last cleared (SEMA4_RSTGT[RSTGTN]). Reads of the SEMA4_RSTGT register do not affect the secure reset finite state machine in any manner.

Address: 30AC_0000h base + 100h offset = 30AC_0100h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	RSTGTN								RSTGSM_RSTGMS_RSTGDP							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SEMA4_RSTGT field descriptions

Field	Description
15–8 RSTGTN	Reset Gate Number. This 8-bit field specifies the specific hardware gate to be reset. This field is updated by the second write. If RSTGTN < 64, then reset the single gate defined by RSTGTN, else reset all the gates. The corresponding secure IRQ notification state machine(s) are also reset.
RSTGSM_ RSTGMS_ RSTGDP	**NOTE:** This field contains subfields that vary depending on whether it is being read or written. Sub-fields indicated as having read access are valid only for read operations. Sub-fields indicated as having

Table continues on the next page...

SEMA4_RSTGT field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>write access are valid only for write operations. Bit numbering in the descriptions begins with the most significant bit numbered 0. See the following table for details.

<table>
<tr><th>Access</th><th>Sub-Field</th><th>Description</th></tr>
<tr><td rowspan="4">Read-Only</td><td>7-6
Reserved</td><td>Reserved. Always reads 0.</td></tr>
<tr><td>5-4
RSTGSM</td><td>Reset Gate Finite State Machine. Reads of the SEMA4_RSTGT register return the encoded state machine value. The reset state machine is maintained in a 2-bit, 3-state implementation, defined as:
00 Idle, waiting for the first data pattern write.
01 Waiting for the second data pattern write.
10 The 2-write sequence has completed. Generate the specified gate reset(s). After the reset is performed, this machine returns to the idle (waiting for first data pattern write) state. Note that the RSTGSM = 0b10 state is valid for only a single machine cycle, so it is impossible for a read to return this value
11 This state encoding is never used and therefore reserved.</td></tr>
<tr><td>3
Reserved</td><td>Reserved. Always reads 0.</td></tr>
<tr><td>2-0
RSTGMS</td><td>Reset Gate Bus Master. This 3-bit read-only field records the logical number of the bus master performing the gate reset function. The reset function requires that the two consecutive writes to this register be initiated by the same bus master to succeed. This field is updated each time a write to this register occurs. The association between system bus master port numbers, the associated bus master device and the logical processor number is SoC-specific. See the chip configuration chapter for this information.</td></tr>
<tr><td>Write-Only</td><td>7-0
RSTGDP</td><td>Reset Gate Data Pattern. This write-only field is accessed with the specified data patterns on the two consecutive writes to enable the gate reset mechanism. For the first write, RSTGDP = 0xe2 while the second write requires RSTGDP = 0x1d.</td></tr>
</table>
</td></tr>
</table>

4.4.6.20 Semaphores (Secure) Reset IRQ Notification (SEMA4_RSTNTF)

As with the case of the secure reset function and the hardware gates, it is recognized that system operation may require a reset function to re-initialize the state of the IRQ notification logic without requiring a system-level reset.

To support this special notification reset requirement, the Semaphores module implements a "secure" reset mechanism which allows an IRQ notification (or all the notifications) to be initialized by following a specific dual-write access pattern. When successful, the specified IRQ notification state machine(s) are reset. Using a technique

similar to that required for the servicing of a software watchdog timer, the secure reset mechanism requires two consecutive writes with predefined data patterns from the same processor to force the clearing of the IRQ notification(s). The required access pattern is:

1. A processor performs a 16-bit write to the SEMA4_RSTNTF memory location. The most significant byte (SEMA4_RSTNTF[RSTNDP]) must be 0x47; the least significant byte is a "don't_care" for this reference.
2. The same processor then performs a second 16-bit write to the SEMA4_RSTNTF location. For this write, the upper byte (SEMA4_RSTNTF[RSTNDP]) is the logical complement of the first data pattern (0xb8) and the lower byte (SEMA4_RSTNTF[RSTNTN]) specifies the notification(s) to be reset. This field can specify a single notification be cleared, or that all notifications are cleared.
3. Reads of the SEMA4_RSTNTF location return information on the 2-bit state machine (SEMA4_RSTNTF[RSTNSM]) which implements this function, the bus master performing the reset (SEMA4_RSTNTF[RSTNMS]) and the notification number(s) last cleared (SEMA4_RSTNTF[RSTNTN]). Reads of the SEMA4_RSTNTF register do not affect the secure reset finite state machine in any manner.

Address: 30AC_0000h base + 104h offset = 30AC_0104h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	RSTNTN								RSTNSM_RSTNMS_RSTNDP							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SEMA4_RSTNTF field descriptions

Field	Description
15–8 RSTNTN	Reset Notification Number. This 8-bit field specifies the specific IRQ notification state machine to be reset. This field is updated by the second write. If RSTNTN < 64, then reset the single IRQ notification machine defined by RSTNTN, else reset all the notifications.
RSTNSM_ RSTNMS_ RSTNDP	**NOTE:** This field contains subfields that vary depending on whether it is being read or written. Sub-fields indicated as having read access are valid only for read operations. Sub-fields indicated as having write access are valid only for write operations. Bit numbering in the descriptions begins with the most significant bit numbered 0. See the following table for details.

Access	Sub-Field	Description
Read-Only	7-6 Reserved	Reserved. Always reads 0.
	5-4 RSTNSM	Reset Notification Finite State Machine. Reads of the SEMA4_RSTNTF register return the encoded state machine value. The reset state machine is maintained in a 2-bit, 3-state implementation, defined as: 00 Idle, waiting for the first data pattern write. 01 Waiting for the second data pattern write.

Table continues on the next page...

SEMA4_RSTNTF field descriptions (continued)

Field	Description		
	Access	**Sub-Field**	**Description**
			10 The 2-write sequence has completed. Generate the specified notification reset(s). After the reset is performed, this machine returns to the idle (waiting for first data pattern write) state. Note the RSTNSM = 10 state is valid for only a single machine cycle, so it is impossible for a read to return this value. 11 This state encoding is never used and therefore reserved..
		3 Reserved	Reserved. Always reads 0.
		2-0 RSTNMS	Reset Notification Bus Master. This 3-bit read-only field records the logical number of the bus master performing the notification reset function. The reset function requires that the two consecutive writes to this register be initiated by the same bus master to succeed. This field is updated each time a write to this register occurs. The association between system bus master port numbers, the associated bus master device and the logical processor number is SoC-specific. See the chip configuration chapter for this information.
	Write-Only	7-0 RSTNDP	Reset Notification Data Pattern. This write-only field is accessed with the specified data patterns on the two consecutive writes to enable the notification reset mechanism. For the first write, RSTNDP = 0x47 while the second write requires RSTNDP = 0xb8.

4.5 On-Chip RAM Memory Controller (OCRAM)

4.5.1 Overview

Various options are provided for adding a pipeline or wait-states in a read/write access, in order to ensure flexible timing control at both high and low frequencies.

The internal block diagram is shown in the figure below.

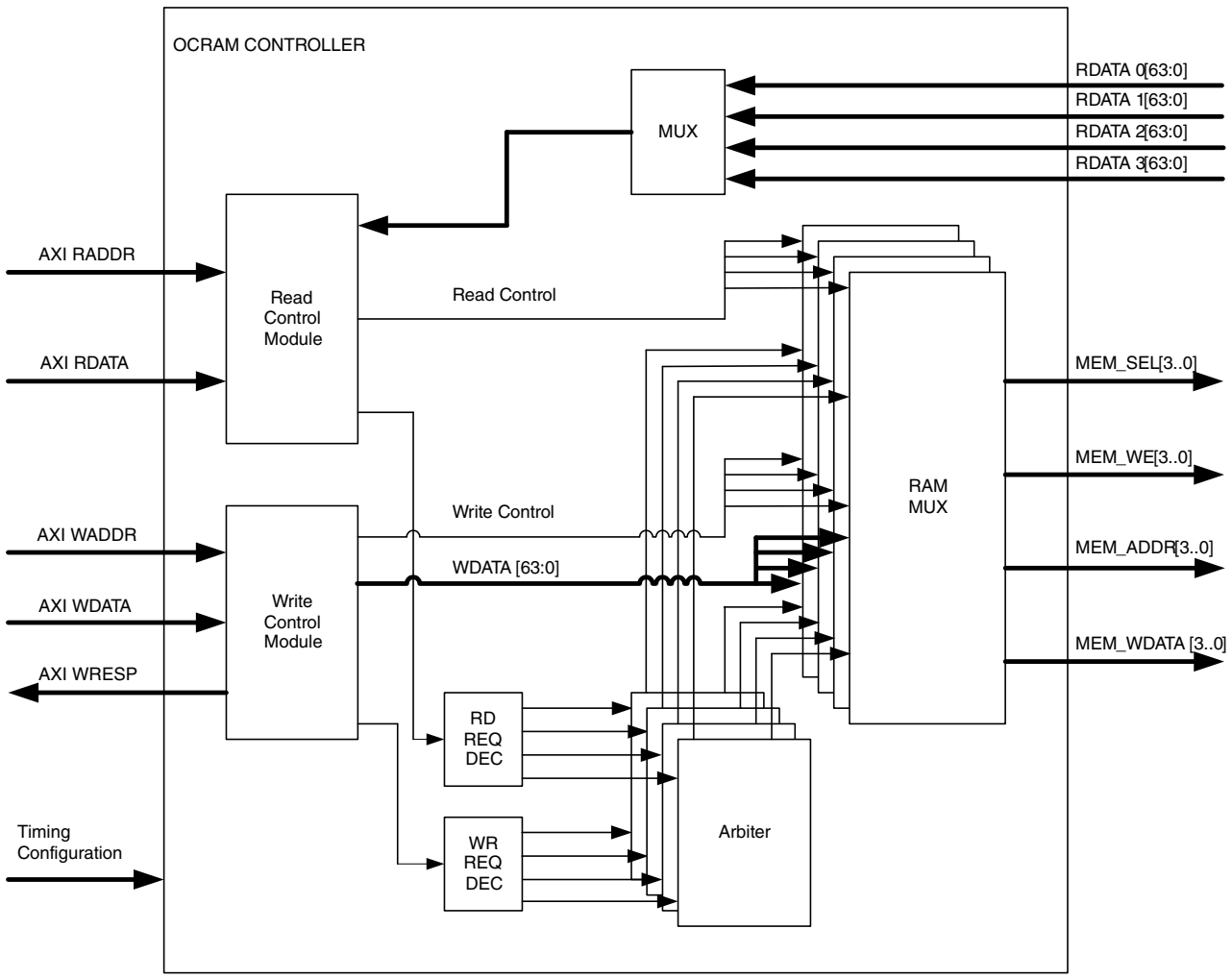


Figure 4-14. On-chip RAM Block Diagram

4.5.2 Basic Functions

4.5.2.1 Read/Write Arbitration

The detailed rules used in arbitration are as follows:

- If there is no granted read or write in the last cycle, and there is only a read request or a write request, the request will be granted.
- If there is no granted read or write in the last cycle, and there are both read or write requests coming in at the same time, the read request will be granted first.

- If a granted read/write transaction has just finished, the write/read request will have the higher priority in the next cycle.
- If the first read/write access request in a transaction is granted, all the data transfer in this burst will be finished before the next arbitration begins, that is, the round-robin arbitration mechanism is based on AXI transaction, not data access.

4.5.3 Advanced Features

This section describes some advanced features designed to avoid timing issues when the on-chip RAM is working at high frequency.

4.5.3.1 Read Data Wait State

When the wait state is enabled, it will take 2 cycles for each read access (each beat of a read burst).

This can avoid the potential timing problem caused by the longer memory access time at higher frequency.

When this feature is disabled, it only takes 1 clock cycle to finish a read transaction. That is, read data is available in the next cycle of read request becomes valid on the bus.

4.5.3.2 Read Address Pipeline

When this feature is enabled, the read address from the AXI master is delayed 1 cycle before it can be accepted by the on-chip RAM.

This can avoid setup time issues for the read access on the memory cell at high frequency. Enabling this feature can cost, at most, 1 more clock cycle for each AXI read transaction, that is, at most 1 more clock cycle for each read burst with multiple beats of data.

When this feature is disabled, the read address from the AXI master can be accepted by the on-chip RAM without delay, and data can become ready for master at next clock cycle (if no other access and no read data wait).

4.5.3.3 Write Data Pipeline

When this feature is enabled, the write data from the AXI master would be delayed 1 cycle before it can be accepted by the on-chip RAM.

This can avoid setup time issue for the write access on the memory cell at high frequency. Enabling this feature would cost at most 1 more clock cycle for each AXI write transaction, that is, at most 1 more clock cycle for each write burst with multiple beats of data.

When this feature is disabled, the write data from the AXI master can be accepted by the on-chip RAM without delay, and data can be written to memory at this cycle (if no other access and write address is also ready at this cycle).

4.5.3.4 Write Address Pipeline

When this feature is enabled, the write address from the AXI master would be delayed 1 cycle before it can be accepted by the on-chip RAM.

This can avoid setup time issue for the write access on the memory cell at high frequency. Enabling this feature would take at most 1 more clock cycle for each AXI write transaction, that is, at most 1 more clock cycle for each write burst with multiple beats of data.

When this feature is disabled, the write address from the AXI master can be accepted by the on-chip RAM without delay, and data can be written to memory at this cycle (if no other access and write data is also ready at this cycle).

4.5.4 Programmable Registers

There are no programmable registers in this block;

4.6 Network Interconnect Bus System (NIC)

4.6.1 Overview

This section provides an overview of the NIC-301 (Network Inter-Connect) AXI arbiter IP.

The NIC-301 (by Arm Ltd.) is a configurable AXI arbiter between several masters and slaves. The NIC-301 IP is designed so that many configuration options are selected at the hardware design stage, determined by SoC characteristics and needs, while several other configuration options are software-controlled.

NOTE

The NIC-301 default settings are configured by NXP's board support package (BSP), and in most cases should not be modified by the customer. The default settings have gone through exhaustive testing during the validation of the part, and have proven to work well for the part's intended target applications. Changes to the default settings may result in a degradation in system performance.

4.6.2 External Signals

There are no external I/O interfaces for NIC-301.

4.7 AHB to IP Bridge (AIPSTZ)

4.7.1 Overview

This section provides an overview of the AHB to IP Bridge (AIPSTZ). This particular peripheral is designed as the bridge between AHB bus and peripherals with the lower bandwidth IP Slave (IPS) buses.

4.7.1.1 Features

The following list summarizes the key features of the bridge:

- The bridge supports the IPS slave bus signals.
- The bridge supports 8-, 16-, and 32-bit IPS peripherals. (Accesses larger than the size of a peripheral are not supported, except to 32-bit memory.)
- The bridge supports a pair of IPS accesses for 64-bit and certain misaligned AHB transfers to 32-bit memory in 64-bit platforms.
- The bridge directly supports up to 32 64-Kbyte external IPS peripherals, and 2 global external IPS peripheral spaces. The bridge occupies 1 MBytes of total address space.
- The bridge provides configurable per-block and per-master access protections. Access permissions are based on bus master (e.g. DMA or core) privilege levels and resource domain. More details on the protection features and configuration can be found in the Security Reference Manual

- Peripheral read transactions require a minimum of 2 hclk clocks, and unbuffered write transactions require a minimum of 3 hclk clocks.
- The bridge uses one single asynchronous reset and one global clock.

4.7.2 Clocks

The following table describes the clock sources for AIPSTZ. Please see Clock Controller Module (CCM) for clock setting, configuration and gating information.

Table 4-18. AIPSTZ Clocks

Clock name	Clock Root	Description
hclk	ahb_clk_root	Module clock

4.7.3 Functional Description

The AIPS bridge serves as a protocol translator between the AHB system bus and the IP bus.

Support is provided for generating a pair of 32-bit IP bus accesses when targeted by a 64-bit system bus access, or a misaligned access which crosses a 32-bit boundary. No other bus-sizing access support is provided.

The AHB to IP bridge is the interface between the AHB and on-chip IPS peripherals, which are sub-blocks containing readable/writable control and status registers.

The AHB master reads and writes these registers through the AIPSTZ. The bridge generates block enables, the block address, transfer attributes, byte enables and write data as inputs to the IPS peripherals. The bridge captures read data from the IPS interface and drives it on the AHB.

Each bridge that connects to the IPS (or peripherals) are referred as AIPS. The chip has three separate AIPS modules, and peripherals are grouped and assigned under each AIPS block. The list of peripherals are indicated as n-1, n-2, and n-3 for AIPS-1, AIPS-2, and AIPS-3 respectively.

AIPS occupies a 1-Mbyte portion of the address space. The register maps of the IPS peripherals are located on 64-Kbyte boundaries. Each IPS peripheral is allocated one 64-Kbyte block of the memory map, and is activated by one of the block enables from the bridge. Up to thirty-two 64-Kbyte external IPS peripherals may be implemented, occupying contiguous blocks of 64-Kbytes. Two global external IPS block enables are

available for the remaining address space to allow for customization and expansion of addressed peripheral devices. In addition, a single "non-global" block enable is also asserted whenever any of the thirty-two non-global block enables is asserted.

The bridge is responsible for indicating to IPS peripherals if an access is in supervisor or user mode. It may block user mode accesses to certain IPS peripherals or it may allow the individual IPS peripherals to determine if user mode accesses are allowed. In addition, peripherals may be designated as write-protected.

The bridge supports the notion of "trusted" masters for security purposes. Masters may be individually designated as trusted for reads, trusted for writes, or trusted for both reads and writes, as well as being forced to look as though all accesses from a master are in user-mode privilege level. Refer to AIPSTZ Memory Map/Register Definition for more information.

The AIPSTZ prevents access to a peripheral if the transaction originated from a source from a resource domain that has been explicitly omitted. Resource domains are assigned in the RDC submodule. Please refer to the RDC chapter for programming details.

All peripheral devices are expected to only require aligned accesses equal to or smaller in size than the peripheral size. An exception to this rule is supported for 32-bit peripherals to allow memory to be placed on the IPS.

4.7.4 Access Protections

The AIPSTZ bridge provides programmable access protections for both masters and peripherals. It allows the privilege level of a master to be overridden, forcing it to user-mode privilege, and allows masters to be designated as trusted or untrusted.

Peripherals may require supervisor privilege level for access, may restrict access to a trusted master only, and may be write-protected. IP bus peripherals are subject to access control policies set in both CSU registers and AIPSTZ registers. An access is blocked if it is denied by either policy.

Masters and peripherals are assigned to one or more resource domains in the RDC submodule (see the RDC chapter for details). Depending on RDC programming, masters transactions through the AIPSTZ may or may not be allowed access to peripherals in different resource domains.

4.7.5 Access Support

Aligned 64-bit accesses, aligned and misaligned word and half word accesses, as well as byte accesses are supported for 32-bit peripherals. Misaligned accesses are supported to allow memory to be placed on the IPS.

Peripheral registers must not be misaligned, although no explicit checking is performed by the AIPS bridge. The bridge will perform two IPS transfers for 64-bit accesses, word accesses with byte offsets of 1, 2, or 3, and for half word accesses with a byte offset of 3. All other accesses will be performed with a single IPS transfer.

Only aligned half word and byte accesses are supported for 16-bit peripherals. All other accesses types are unsupported, and results of such accesses are undefined. They are not terminated with an error response.

Only byte accesses are supported for 8-bit peripherals. All other accesses types are unsupported, and results of such accesses are undefined. They are not terminated with an error response.

4.7.6 Initialization Information

The AIPS bridge should be programmed before use.

The following registers should be initialized: The Master Privilege Registers (AIPSTZ_MPRs), the Peripheral Access Control registers (AIPSTZ_PACRs), and the Off-platform Peripheral Access Control registers (AIPSTZ_OPACRs) described in AIPSTZ Memory Map/Register Definition.

4.7.6.1 Security Block

The AIPSTZ contains a security block that is connected to each off-platform peripheral. This block filters accesses based on write/read, non-secure, and supervisor signals.

Each peripheral can be individually configured to allow or deny each of the following transactions as described in the table below:

Table 4-19. Peripheral Access Configuration options

Config Bit	Write	Non-Secure	Supervisor	Meaning
0	0	0	0	Secure User Read
1	0	0	1	Secure Supervisor Read

Table continues on the next page...

Table 4-19. Peripheral Access Configuration options (continued)

Config Bit	Write	Non-Secure	Supervisor	Meaning
2	0	1	0	Non-Secure User Read
3	0	1	1	Non-Secure Supervisor Read
4	1	0	0	Secure User Write
5	1	0	1	Secure Supervisor Write
6	1	1	0	Non-Secure User Write
7	1	1	1	Non-Secure Supervisor Write

Each peripheral has a security configuration (sec_config_X) input for determining whether to allow or deny a given access type. These are 8-bit vectors, with each bit corresponding to one of the transactions above as listed in the Config Bit column of Table 4-19. If the bit is asserted (1'b1), the transaction is allowed. If the bit is negated (1'b0), the transaction is not allowed.

For example, if peripheral 0 is configured as follows:

sec_config_0 [7:0] = 8'b0011_0011

This peripheral can only be accessed by secure transactions. Bits 0, 1, 4, and 5 are asserted and these bits refer to the four types of secure transactions. If an insecure transaction is attempted to this peripheral, it will result in an error.

Eight bits per peripheral across an entire system can result in a large number of configuration bits that must be assigned and controlled, most likely in a series of registers in another block. To reduce the number of register bits required predefined sets of security profiles can be defined and encapsulated in an external security translation block. The table below describes one set of security profiles that has been proposed for use with the AIPSTZ.

Table 4-20. Security Levels

CSU_SEC_LEVEL	Non-Secure User	Non-Secure Supervisor	Secure User	Secure Supervisor
0	RD+WR	RD+WR	RD+WR	RD+WR
1	NOT ALLOWED	RD+WR	RD+WR	RD+WR
2	Read Only	Read Only	RD+WR	RD+WR
3	NOT ALLOWED	Read Only	RD+WR	RD+WR
4	NOT ALLOWED	NOT ALLOWED	RD+WR	RD+WR
5	NOT ALLOWED	NOT ALLOWED	NOT ALLOWED	RD+WR
6	NOT ALLOWED	NOT ALLOWED	Read Only	Read Only
7	NOT ALLOWED	NOT ALLOWED	NOT ALLOWED	NOT ALLOWED

Information regarding CSU is provided in the Security Reference Manual. Contact your NXP representative for information about obtaining this document.

A 3-bit input, 8-bit output translation block can be used such that only three register bits are required to set the security profile and the translation block will drive the correct 8-bit configuration vector. Each peripheral connected to the AIPSTZ would require this translation block. The top level AIPSTZ has this three bit input line `csu_sec_level[2:0]' corresponding to each peripheral X.

4.7.7 AIPSTZ Memory Map/Register Definition

The memory map for the AIPS SW-visible registers is shown in the table below.

The MPROT and OPACR fields are 4 bits in width. Some bits may be reserved depending on device.

AIPSTZ memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3000_0000	Master Priviledge Registers (AIPSTZ_MPR)	32	R/W	7700_0000h	4.7.7.1/195
3000_0040	Off-Platform Peripheral Access Control Registers (AIPSTZ_OPACR)	32	R/W	4444_4444h	4.7.7.2/198
3000_0044	Off-Platform Peripheral Access Control Registers (AIPSTZ_OPACR1)	32	R/W	4444_4444h	4.7.7.3/201
3000_0048	Off-Platform Peripheral Access Control Registers (AIPSTZ_OPACR2)	32	R/W	4444_4444h	4.7.7.4/204
3000_004C	Off-Platform Peripheral Access Control Registers (AIPSTZ_OPACR3)	32	R/W	4444_4444h	4.7.7.5/207
3000_0050	Off-Platform Peripheral Access Control Registers (AIPSTZ_OPACR4)	32	R/W	4444_4444h	4.7.7.6/210

4.7.7.1 Master Priviledge Registers (AIPSTZ_MPR)

Each AIPSTZ_MPR specifies 16 4-bit fields defining the access privilege level associated with a bus master in the platform, as well as specifying whether write accesses from this master are bufferable shown in Table 4-21

The registers provide one field per bus master, where field 15 corresponds to master 15, field 14 to master 14,... field 0 to master 0 (typically the processor core). The master index allocation is shown in the table below.

Table 4-21. MPROT Field

Bit	Field	Description
3	MBW	**Master Buffer Writes** - This bit determines whether the AIPSTZ is enabled to buffer writes from this master.
2	MTR	**Master Trusted for Reads** - This bit determines whether the master is trusted for read accesses.
1	MTW	**Master Trusted for Writes** - This bit determines whether the master is trusted for write accesses.
0	MPL	**Master Privilege Level** - This bit determines how the privilege level of the master is determined.

NOTE

The reset value is set to 0000_0000_7700_0000, which makes master 0 and master 1 (Arm CORE) the trusted masters. Trusted software can change the settings after reset.

Table 4-22. Master Index Allocation

Master Index	Master Name	Comments
Master 0	All masters excluding Arm core	Share the same number allocation.
Master 1	Arm A53	
Master 4	SDMA	
Master 6	M4	

Address: 3000_0000h base + 0h offset = 3000_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MPROT0				MPROT1				MPROT2				MPROT3				Reserved				MPROT5				Reserved							
Reset	0	1	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

AIPSTZ_MPR field descriptions

Field	Description
31–28 MPROT0	Master 0 Priviledge, Buffer, Read, Write Control xxx0 **MPL0** — Accesses from this master are forced to user-mode (ips_supervisor_access is forced to zero) regardless of the hprot[1] access attribute. xxx1 **MPL1** — Accesses from this master are not forced to user-mode. The hprot[1] access attribute is used directly to determine ips_supervisor_access. xx0x **MTW0** — This master is not trusted for write accesses. xx1x **MTW1** — This master is trusted for write accesses. x0xx **MTR0** — This master is not trusted for read accesses. x1xx **MTR1** — This master is trusted for read accesses.

Table continues on the next page...

AIPSTZ_MPR field descriptions (continued)

Field	Description
	0xxx **MBW0** — Write accesses from this master are not bufferable 1xxx **MBW1** — Write accesses from this master are allowed to be buffered
27–24 MPROT1	Master 1 Priviledge, Buffer, Read, Write Control xxx0 **MPL0** — Accesses from this master are forced to user-mode (ips_supervisor_access is forced to zero) regardless of the hprot[1] access attribute. xxx1 **MPL1** — Accesses from this master are not forced to user-mode. The hprot[1] access attribute is used directly to determine ips_supervisor_access. xx0x **MTW0** — This master is not trusted for write accesses. xx1x **MTW1** — This master is trusted for write accesses. x0xx **MTR0** — This master is not trusted for read accesses. x1xx **MTR1** — This master is trusted for read accesses. 0xxx **MBW0** — Write accesses from this master are not bufferable 1xxx **MBW1** — Write accesses from this master are allowed to be buffered
23–20 MPROT2	Master 2 Priviledge, Buffer, Read, Write Control xxx0 **MPL0** — Accesses from this master are forced to user-mode (ips_supervisor_access is forced to zero) regardless of the hprot[1] access attribute. xxx1 **MPL1** — Accesses from this master are not forced to user-mode. The hprot[1] access attribute is used directly to determine ips_supervisor_access. xx0x **MTW0** — This master is not trusted for write accesses. xx1x **MTW1** — This master is trusted for write accesses. x0xx **MTR0** — This master is not trusted for read accesses. x1xx **MTR1** — This master is trusted for read accesses. 0xxx **MBW0** — Write accesses from this master are not bufferable 1xxx **MBW1** — Write accesses from this master are allowed to be buffered
19–16 MPROT3	Master 3 Priviledge, Buffer, Read, Write Control. xxx0 **MPL0** — Accesses from this master are forced to user-mode (ips_supervisor_access is forced to zero) regardless of the hprot[1] access attribute. xxx1 **MPL1** — Accesses from this master are not forced to user-mode. The hprot[1] access attribute is used directly to determine ips_supervisor_access. xx0x **MTW0** — This master is not trusted for write accesses. xx1x **MTW1** — This master is trusted for write accesses. x0xx **MTR0** — This master is not trusted for read accesses. x1xx **MTR1** — This master is trusted for read accesses. 0xxx **MBW0** — Write accesses from this master are not bufferable 1xxx **MBW1** — Write accesses from this master are allowed to be buffered
15–12 -	This field is reserved. Reserved
11–8 MPROT5	Master 5 Priviledge, Buffer, Read, Write Control. xxx0 **MPL0** — Accesses from this master are forced to user-mode (ips_supervisor_access is forced to zero) regardless of the hprot[1] access attribute. xxx1 **MPL1** — Accesses from this master are not forced to user-mode. The hprot[1] access attribute is used directly to determine ips_supervisor_access. xx0x **MTW0** — This master is not trusted for write accesses. xx1x **MTW1** — This master is trusted for write accesses. x0xx **MTR0** — This master is not trusted for read accesses.

Table continues on the next page...

AIPSTZ_MPR field descriptions (continued)

Field	Description
	x1xx **MTR1** — This master is trusted for read accesses. 0xxx **MBW0** — Write accesses from this master are not bufferable 1xxx **MBW1** — Write accesses from this master are allowed to be buffered
-	This field is reserved. Reserved

4.7.7.2 Off-Platform Peripheral Access Control Registers (AIPSTZ_OPACR)

Each of the off-platform peripherals have an Off-platform Peripheral Access Control Register (AIPSTZ_OPACR) which defines the access levels supported by the given block.

Each AIPSTZ_OPACR has the following format shown in Table 4-23

Table 4-23. OPAC Field

Bit	Field	Description
3	BW	**Buffer Writes** - This bit determines whether write accesses to this peripheral are allowed to be buffered.[1]
2	SP	**Supervisor Protect** - This bit determines whether the peripheral requires supervisor privilege level for access.
1	WP	**Write Protect** - This bit determines whether the peripheral allows write accesses.
0	TP	**Trusted Protect** - This bit determines whether the peripheral allows accesses from an untrusted master.

1. Buffered writes are not available for AIPSTZ. This bit should be set to '0'.

Address: 3000_0000h base + 40h offset = 3000_0040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OPAC0				OPAC1				OPAC2				OPAC3				OPAC4				OPAC5				OPAC6				OPAC7			
Reset	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0

AIPSTZ_OPACR field descriptions

Field	Description
31–28 OPAC0	Off-platform Peripheral Access Control 0 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.

Table continues on the next page...

AIPSTZ_OPACR field descriptions (continued)

Field	Description
	x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
27–24 OPAC1	Off-platform Peripheral Access Control 1 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
23–20 OPAC2	Off-platform Peripheral Access Control 2 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
19–16 OPAC3	Off-platform Peripheral Access Control 3 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses.

Table continues on the next page...

AIPSTZ_OPACR field descriptions (continued)

Field	Description
	x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
15–12 OPAC4	Off-platform Peripheral Access Control 4 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
11–8 OPAC5	Off-platform Peripheral Access Control 5 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
7–4 OPAC6	Off-platform Peripheral Access Control 6 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the

Table continues on the next page...

AIPSTZ_OPACR field descriptions (continued)

Field	Description
	master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
OPAC7	Off-platform Peripheral Access Control 7 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.

4.7.7.3 Off-Platform Peripheral Access Control Registers (AIPSTZ_OPACR1)

Each of the off-platform peripherals have an Off-platform Peripheral Access Control Register (AIPSTZ_OPACR) which defines the access levels supported by the given block.

Each AIPSTZ_OPACR has the following format shown in Table 4-23

Address: 3000_0000h base + 44h offset = 3000_0044h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OPAC8				OPAC9				OPAC10				OPAC11				OPAC12				OPAC13				OPAC14				OPAC15			
Reset	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0

AIPSTZ_OPACR1 field descriptions

Field	Description
31–28 OPAC8	Off-platform Peripheral Access Control 8 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.

Table continues on the next page...

AIPSTZ_OPACR1 field descriptions (continued)

Field	Description
	x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
27–24 OPAC9	Off-platform Peripheral Access Control 9 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
23–20 OPAC10	Off-platform Peripheral Access Control 10 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
19–16 OPAC11	Off-platform Peripheral Access Control 11 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses.

Table continues on the next page...

AIPSTZ_OPACR1 field descriptions (continued)

Field	Description
	x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
15–12 OPAC12	Off-platform Peripheral Access Control 12 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
11–8 OPAC13	Off-platform Peripheral Access Control 13 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
7–4 OPAC14	Off-platform Peripheral Access Control 14 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the

Table continues on the next page...

AIPSTZ_OPACR1 field descriptions (continued)

Field	Description
	master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
OPAC15	Off-platform Peripheral Access Control 15 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.

4.7.7.4 Off-Platform Peripheral Access Control Registers (AIPSTZ_OPACR2)

Each of the off-platform peripherals have an Off-platform Peripheral Access Control Register (AIPSTZ_OPACR) which defines the access levels supported by the given block.

Each AIPSTZ_OPACR has the following format shown in Table 4-23

Address: 3000_0000h base + 48h offset = 3000_0048h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OPAC16				OPAC17				OPAC18				OPAC19				OPAC20				OPAC21				OPAC22				OPAC23			
Reset	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0

AIPSTZ_OPACR2 field descriptions

Field	Description
31–28 OPAC16	Off-platform Peripheral Access Control 16 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses.

Table continues on the next page...

AIPSTZ_OPACR2 field descriptions (continued)

Field	Description
	xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
27–24 OPAC17	Off-platform Peripheral Access Control 17 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
23–20 OPAC18	Off-platform Peripheral Access Control 18 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
19–16 OPAC19	Off-platform Peripheral Access Control 19 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.

Table continues on the next page...

AIPSTZ_OPACR2 field descriptions (continued)

Field	Description
	x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
15–12 OPAC20	Off-platform Peripheral Access Control 20 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
11–8 OPAC21	Off-platform Peripheral Access Control 21 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
7–4 OPAC22	Off-platform Peripheral Access Control 22 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses.

Table continues on the next page...

AIPSTZ_OPACR2 field descriptions (continued)

Field	Description
	x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
OPAC23	Off-platform Peripheral Access Control 23 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.

4.7.7.5 Off-Platform Peripheral Access Control Registers (AIPSTZ_OPACR3)

Each of the off-platform peripherals have an Off-platform Peripheral Access Control Register (AIPSTZ_OPACR) which defines the access levels supported by the given block.

Each AIPSTZ_OPACR has the following format shown in Table 4-23

Address: 3000_0000h base + 4Ch offset = 3000_004Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OPAC24				OPAC25				OPAC26				OPAC27				OPAC28				OPAC29				OPAC30				OPAC31			
Reset	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0

AIPSTZ_OPACR3 field descriptions

Field	Description
31–28 OPAC24	Off-platform Peripheral Access Control 24 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.

Table continues on the next page...

AIPSTZ_OPACR3 field descriptions (continued)

Field	Description
	xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
27–24 OPAC25	Off-platform Peripheral Access Control 25 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
23–20 OPAC26	Off-platform Peripheral Access Control 26 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
19–16 OPAC27	Off-platform Peripheral Access Control 27 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses.

Table continues on the next page...

AIPSTZ_OPACR3 field descriptions (continued)

Field	Description
	xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
15–12 OPAC28	Off-platform Peripheral Access Control 28 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
11–8 OPAC29	Off-platform Peripheral Access Control 29 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
7–4 OPAC30	Off-platform Peripheral Access Control 30 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.

Table continues on the next page...

AIPSTZ_OPACR3 field descriptions (continued)

Field	Description
	x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
OPAC31	Off-platform Peripheral Access Control 31 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.

4.7.7.6 Off-Platform Peripheral Access Control Registers (AIPSTZ_OPACR4)

Each of the off-platform peripherals have an Off-platform Peripheral Access Control Register (AIPSTZ_OPACR) which defines the access levels supported by the given block.

Each AIPSTZ_OPACR has the following format shown in Table 4-23

Address: 3000_0000h base + 50h offset = 3000_0050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OPAC32				OPAC33				Reserved																							
Reset	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0

AIPSTZ_OPACR4 field descriptions

Field	Description
31–28 OPAC32	Off-platform Peripheral Access Control 32 xxx0 **TP0** — Accesses from an untrusted master are allowed.

Table continues on the next page...

AIPSTZ_OPACR4 field descriptions (continued)

Field	Description
	xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
27–24 OPAC33	Off-platform Peripheral Access Control 33 xxx0 **TP0** — Accesses from an untrusted master are allowed. xxx1 **TP1** — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x **WP0** — This peripheral allows write accesses. xx1x **WP1** — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx **SP0** — This peripheral does not require supervisor privilege level for accesses. x1xx **SP1** — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx **BW0** — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx **BW1** — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
-	This field is reserved. Reserved

4.8 Shared Peripheral Bus Arbiter (SPBA)

4.8.1 Overview

The Shared Peripheral Bus Arbiter (SPBA) is a three-to-one IP Bus interface arbiter. Three masters arbitrate for shared peripheral access through the SPBA.

The SPBA has three primary functions:

- The IP Bus Line switches a master to one peripheral

- The Masters arbiter arbitrates between the three masters to solve concurrent access or restricted access to peripherals
- The Control Registers and Ownership Control includes a set of registers which are reachable through software and permit the access scheme to be defined for each peripheral (Resource Ownership and Access Control). It generates signals for the external steering logic of interrupts and DMA signals.

The figure below shows the SPBA block diagram

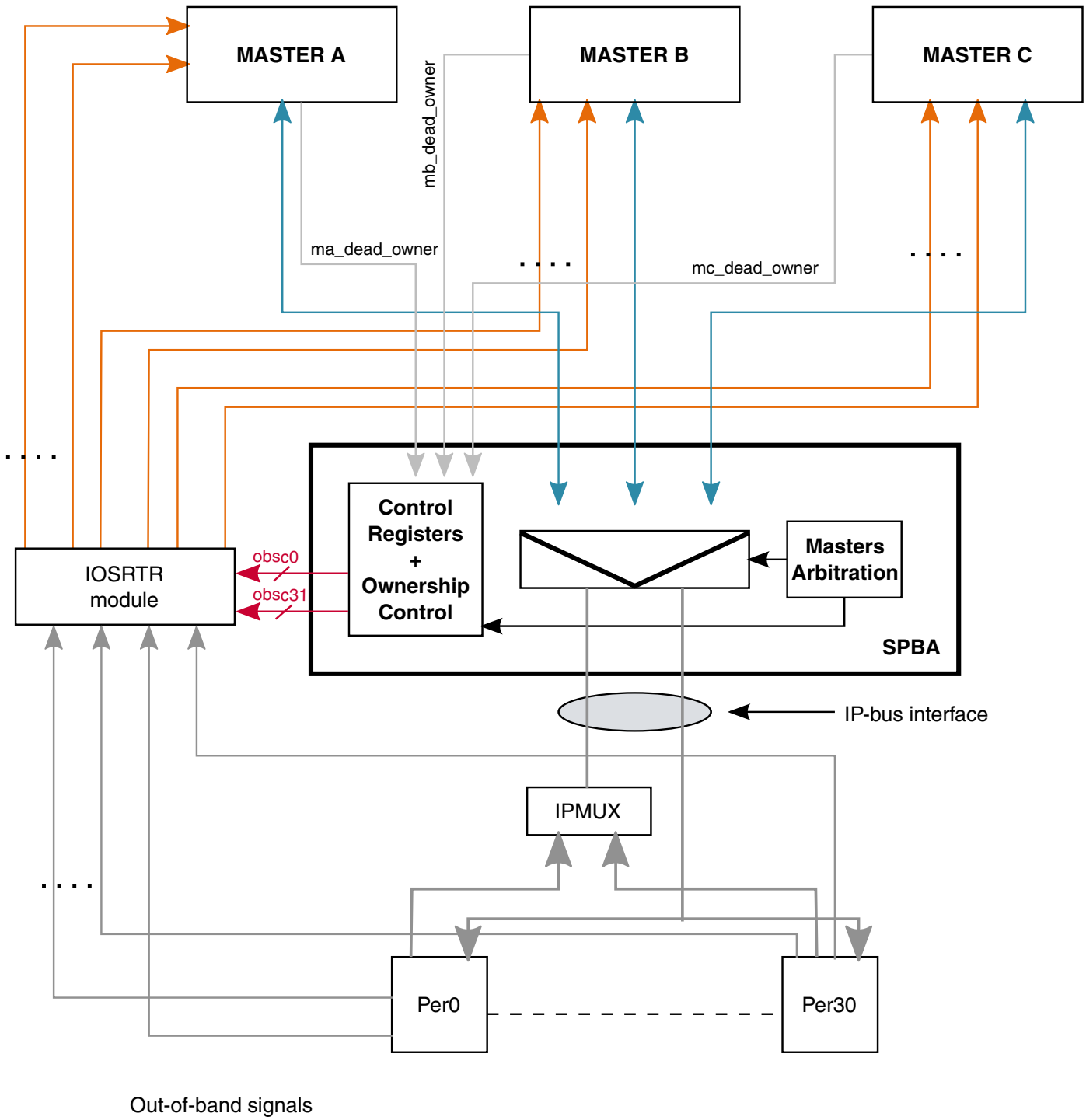


Figure 4-15. SPBA Block Diagram

4.8.1.1 Features

The SPBA includes the following features:

- Three IP Bus masters arbitration: Master A, B and C
- Support for DMA masters
- 32-bit data
- Supports up to 31 shared peripherals, each consuming 64 kilobytes of address space
- SPBA can be considered the 32nd peripheral, used for resource ownership and access control of the 31 peripherals
- Provides 31 sets of out of band steering control (OBSC) signals to the off-block steering logic
- Operating frequency up to 67 MHz
- Clocks: ipg_clk, ipg_clk_s

4.8.1.2 Modes of operation

SPBA behavior is transparent when accessing a peripheral, though it has these distinct modes of operation.

Reset/Abort

The SPBA has a hardware reset which initializes all registers, arbitration and peripherals rights registers (PRRs).

An abort signal input is provided allowing each master to abort its current access and release ownership (in case of master reset sequence).

Functional

Once a master request is granted, its IP Bus signals are steered to the requested peripheral.

Standby

No clock needed. The SPBA needs clocks only during access to the PRRs, arbitration, and abort phases. It generates two clock enable signals indicating when the clocks must be provided.

Configuration

During this phase, a master accesses the SPBA PRRs. The SPBA memory-mapped registers are seen as a shared peripheral.

4.8.2 Clocks

The table found here describes the clock sources for SPBA.

Please see Clock Controller Module (CCM) for clock setting, configuration and gating information.

Table 4-24. SPBA Clocks

Clock name	Clock Root	Description
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_s	ipg_clk_root	Peripheral access clock

4.8.3 Functional description

4.8.3.1 Masters arbitration

The arbitration mechanism determines which port will control the master port, based on a simple round-robin arbitration scheme.

There are several use cases to consider.

- Only one master request per access. The master is switched to the shared peripheral bus, without arbitration. Figure 4-16 shows the MB request on the global module enable signal, served without wait state.
- If two masters simultaneously access SPBA, the last granted master is held off using the <master>_ips_xfr_wait output signal (default value is high). When the master is granted sips_xfr_wait, shared IP Bus peripheral is connected to <master>_ips_xfr_wait outputs.
- If three masters simultaneously access SPBA, then the last two granted masters are held off using <master>_ips_xfr_wait. Figure 4-17 shows a case in which the last two accesses granted are MA and MB. The requests are used even if they are in the same cycle.
- If after reset, at the first multiple access, no master has been granted, the priority is static: Master A (MA), Master B (MB) and last Master C (MC) port.
- No master request. No master switch to shared peripherals.

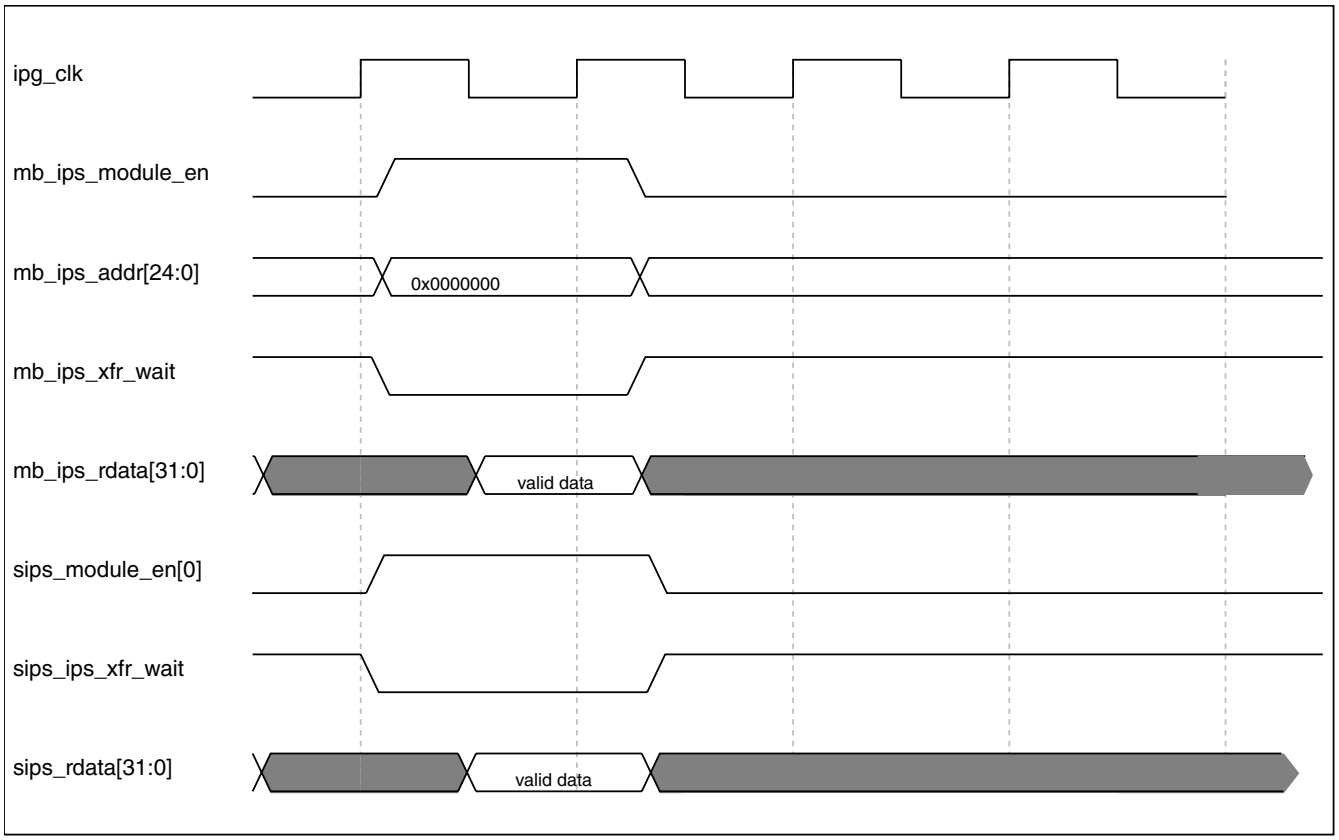


Figure 4-16. Example of one master request, no SPBA arbitration

The following figure assumes MA and MB have been the last two masters granted in the previous transfers (MA then MB).

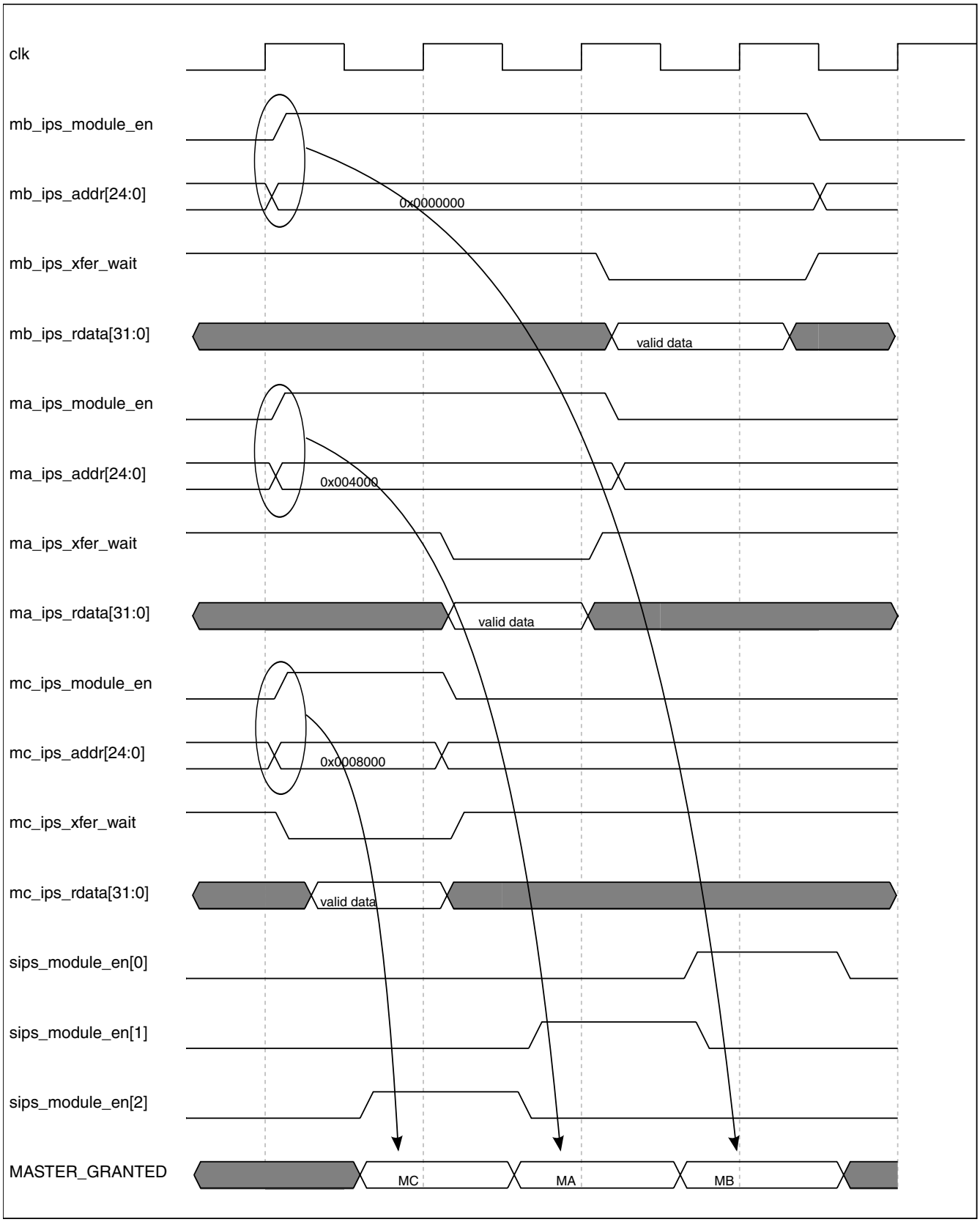


Figure 4-17. Example of three master requests: Masters already granted are "waited";

4.8.4 Resource ownership control

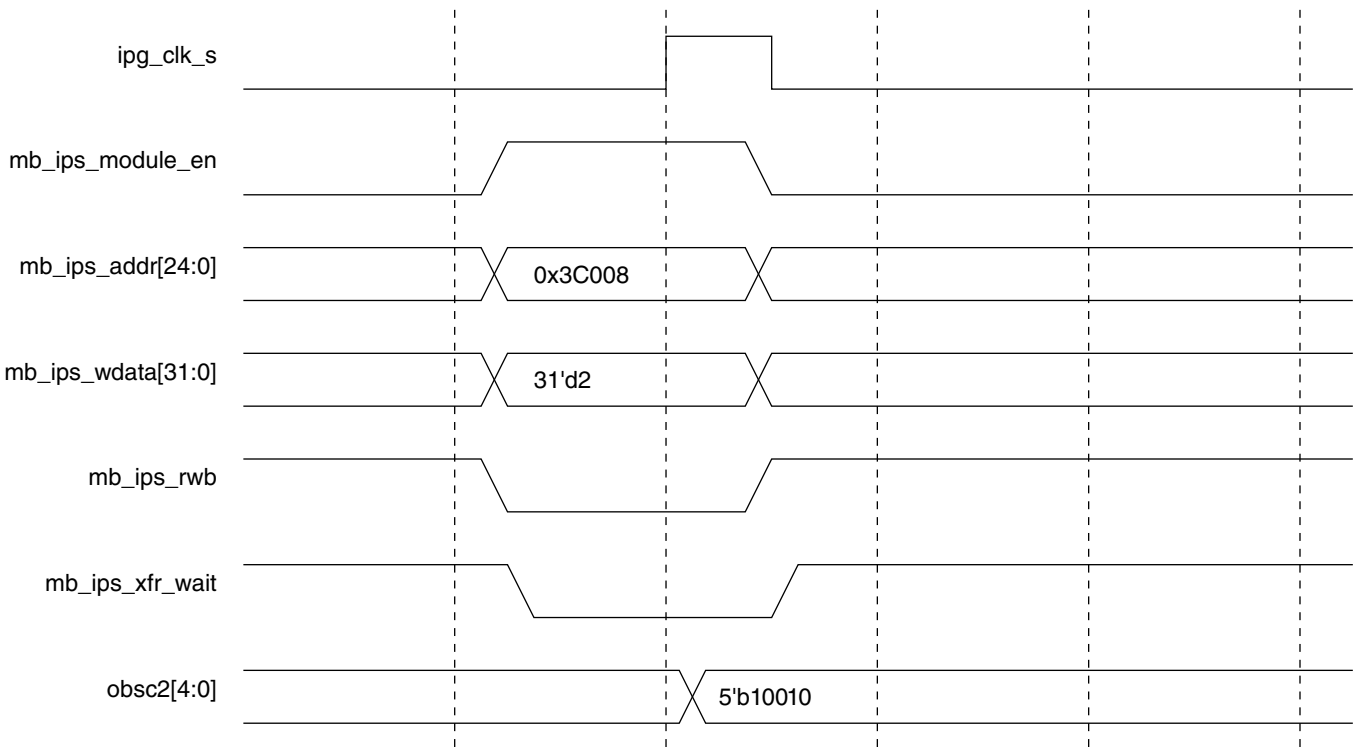
The resource ownership control regulates access to the shared peripherals and determines the steering of out-of-band signals.

4.8.4.1 Access control

4.8.4.1.1 Peripheral access

The peripheral access (resource access) of the requesting master is given by the corresponding RAR bit of the Peripheral Right Register. It determines if the master has access privilege to the resource.

Any attempt at access made by a requesting master whose access privilege bit is not set (in the PRR) is terminated with a bus error (**<master>_ips_xfr_err** is asserted by SPBA logic). The master that owns the resource can lock the peripheral for itself and/or grant other masters access to the peripheral by setting the appropriate bit(s) in the RAR field.



Master B is taking ownership of peripheral 2 by writing 3'b010 in the SPBA peripheral 2 right register (rarfield)
This ownership can be checked on obsc2 output as roi2[1:0] = 2'b10 and rar2[2:0] = 3'b010
(obsc[4:0] = {roi2[1], roi2[0], rar2[2], rar2[1], rar2[0]})

Figure 4-18. Example of one master B gaining ownership of peripheral 2

4.8.4.1.2 Peripheral Right Register access

The ROI bits of the Peripheral Right Register (PRR) determine which master is allowed to make write access to PRR. The identification of the requesting master is compared to the ROI bits of the PRR to determine if the master has ownership of the corresponding register.

Any attempted write access to a PRR already owned by another master will be ignored.

4.8.4.2 Owner election

When the peripheral is not owned by any master (ROI="00", after coming out of reset for instance), the first master to perform successfully a write to the RAR bits of the PRR is granted ownership of the peripheral and its associated PRR.

After writing to the PRR (RAR bit(s)), the master must read it back to make sure that it was granted ownership. If the RMO field is 2'b11, then the ownership claim is successful. If RMO is 2'b10, another master claimed ownership before this master was able to complete its write. This resolves the case in which two or more masters attempt to write the PRR at the same time; only the first master will be granted ownership. However all masters must read the PRR to determine if this case occurred, and if so, whether they were the first master which was granted ownership.

NOTE

A master that has been granted ownership of the PRR does not automatically have the right access to the peripheral; it must still set its own RAR bits in the PRR to access the peripheral.

4.8.4.3 Ending ownership

Ownership may be voluntarily ended by the owning master, or automatically upon assertion of a master-specific dead_owner signal.

The former is appropriate for software-controlled yielding of ownership. The latter is appropriate for automatic yielding of ownership when the owner has gone into reset.

When a master is reset, it clears the ROI bits of the PRRs owned by the corresponding master. When the owner is dead (in reset), all peripherals previously owned by that master must be changed to the un-owned state.

NOTE

It is the programmer's responsibility to make sure the peripherals are placed in an appropriate state before ending ownership.

4.8.4.3.1 Software Controlled Ownership Ending

The ROI bits will be automatically cleared when the master that owns the PRR access right clears (write) the RAR bits (Table 2).

It will then end the ownership of the PRR.

4.8.4.4 The Un-owned State

During the time when the peripheral is un-owned (i.e the ROI field contains all 0's), all masters have full access to it (RAR bits can then be modified by a master if ROI[1:0] = 2'b0).

In such cases it is necessary for software to ensure any necessary coherency in the resource, there is no hardware protection.

4.8.5 SPBA Memory Map/Register Definition

The SPBA control registers (Peripheral Right Registers) are mapped as a virtual shared peripheral.

SPBA can support up to 31 shared peripherals. Each of them has its own Peripheral Right Register (PRR) accessible within the SPBA memory-mapped registers, and consists of the Requesting Master Owner, the Resource Owner ID and the Resource Access Right fields.

SPBA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
300F_0000	Peripheral Rights Register (SPBA2_PRR0)	32	R/W	0000_0007h	4.8.5.1/223
300F_0004	Peripheral Rights Register (SPBA2_PRR1)	32	R/W	0000_0007h	4.8.5.1/223
300F_0008	Peripheral Rights Register (SPBA2_PRR2)	32	R/W	0000_0007h	4.8.5.1/223
300F_000C	Peripheral Rights Register (SPBA2_PRR3)	32	R/W	0000_0007h	4.8.5.1/223
300F_0010	Peripheral Rights Register (SPBA2_PRR4)	32	R/W	0000_0007h	4.8.5.1/223
300F_0014	Peripheral Rights Register (SPBA2_PRR5)	32	R/W	0000_0007h	4.8.5.1/223
300F_0018	Peripheral Rights Register (SPBA2_PRR6)	32	R/W	0000_0007h	4.8.5.1/223
300F_001C	Peripheral Rights Register (SPBA2_PRR7)	32	R/W	0000_0007h	4.8.5.1/223
300F_0020	Peripheral Rights Register (SPBA2_PRR8)	32	R/W	0000_0007h	4.8.5.1/223
300F_0024	Peripheral Rights Register (SPBA2_PRR9)	32	R/W	0000_0007h	4.8.5.1/223
300F_0028	Peripheral Rights Register (SPBA2_PRR10)	32	R/W	0000_0007h	4.8.5.1/223
300F_002C	Peripheral Rights Register (SPBA2_PRR11)	32	R/W	0000_0007h	4.8.5.1/223
300F_0030	Peripheral Rights Register (SPBA2_PRR12)	32	R/W	0000_0007h	4.8.5.1/223
300F_0034	Peripheral Rights Register (SPBA2_PRR13)	32	R/W	0000_0007h	4.8.5.1/223
300F_0038	Peripheral Rights Register (SPBA2_PRR14)	32	R/W	0000_0007h	4.8.5.1/223
300F_003C	Peripheral Rights Register (SPBA2_PRR15)	32	R/W	0000_0007h	4.8.5.1/223

Table continues on the next page...

SPBA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
300F_0040	Peripheral Rights Register (SPBA2_PRR16)	32	R/W	0000_0007h	4.8.5.1/223
300F_0044	Peripheral Rights Register (SPBA2_PRR17)	32	R/W	0000_0007h	4.8.5.1/223
300F_0048	Peripheral Rights Register (SPBA2_PRR18)	32	R/W	0000_0007h	4.8.5.1/223
300F_004C	Peripheral Rights Register (SPBA2_PRR19)	32	R/W	0000_0007h	4.8.5.1/223
300F_0050	Peripheral Rights Register (SPBA2_PRR20)	32	R/W	0000_0007h	4.8.5.1/223
300F_0054	Peripheral Rights Register (SPBA2_PRR21)	32	R/W	0000_0007h	4.8.5.1/223
300F_0058	Peripheral Rights Register (SPBA2_PRR22)	32	R/W	0000_0007h	4.8.5.1/223
300F_005C	Peripheral Rights Register (SPBA2_PRR23)	32	R/W	0000_0007h	4.8.5.1/223
300F_0060	Peripheral Rights Register (SPBA2_PRR24)	32	R/W	0000_0007h	4.8.5.1/223
300F_0064	Peripheral Rights Register (SPBA2_PRR25)	32	R/W	0000_0007h	4.8.5.1/223
300F_0068	Peripheral Rights Register (SPBA2_PRR26)	32	R/W	0000_0007h	4.8.5.1/223
300F_006C	Peripheral Rights Register (SPBA2_PRR27)	32	R/W	0000_0007h	4.8.5.1/223
300F_0070	Peripheral Rights Register (SPBA2_PRR28)	32	R/W	0000_0007h	4.8.5.1/223
300F_0074	Peripheral Rights Register (SPBA2_PRR29)	32	R/W	0000_0007h	4.8.5.1/223
300F_0078	Peripheral Rights Register (SPBA2_PRR30)	32	R/W	0000_0007h	4.8.5.1/223
300F_007C	Peripheral Rights Register (SPBA2_PRR31)	32	R/W	0000_0007h	4.8.5.1/223
308F_0000	Peripheral Rights Register (SPBA1_PRR0)	32	R/W	0000_0007h	4.8.5.1/223
308F_0004	Peripheral Rights Register (SPBA1_PRR1)	32	R/W	0000_0007h	4.8.5.1/223
308F_0008	Peripheral Rights Register (SPBA1_PRR2)	32	R/W	0000_0007h	4.8.5.1/223
308F_000C	Peripheral Rights Register (SPBA1_PRR3)	32	R/W	0000_0007h	4.8.5.1/223
308F_0010	Peripheral Rights Register (SPBA1_PRR4)	32	R/W	0000_0007h	4.8.5.1/223
308F_0014	Peripheral Rights Register (SPBA1_PRR5)	32	R/W	0000_0007h	4.8.5.1/223
308F_0018	Peripheral Rights Register (SPBA1_PRR6)	32	R/W	0000_0007h	4.8.5.1/223
308F_001C	Peripheral Rights Register (SPBA1_PRR7)	32	R/W	0000_0007h	4.8.5.1/223
308F_0020	Peripheral Rights Register (SPBA1_PRR8)	32	R/W	0000_0007h	4.8.5.1/223
308F_0024	Peripheral Rights Register (SPBA1_PRR9)	32	R/W	0000_0007h	4.8.5.1/223
308F_0028	Peripheral Rights Register (SPBA1_PRR10)	32	R/W	0000_0007h	4.8.5.1/223
308F_002C	Peripheral Rights Register (SPBA1_PRR11)	32	R/W	0000_0007h	4.8.5.1/223
308F_0030	Peripheral Rights Register (SPBA1_PRR12)	32	R/W	0000_0007h	4.8.5.1/223
308F_0034	Peripheral Rights Register (SPBA1_PRR13)	32	R/W	0000_0007h	4.8.5.1/223
308F_0038	Peripheral Rights Register (SPBA1_PRR14)	32	R/W	0000_0007h	4.8.5.1/223
308F_003C	Peripheral Rights Register (SPBA1_PRR15)	32	R/W	0000_0007h	4.8.5.1/223
308F_0040	Peripheral Rights Register (SPBA1_PRR16)	32	R/W	0000_0007h	4.8.5.1/223
308F_0044	Peripheral Rights Register (SPBA1_PRR17)	32	R/W	0000_0007h	4.8.5.1/223
308F_0048	Peripheral Rights Register (SPBA1_PRR18)	32	R/W	0000_0007h	4.8.5.1/223
308F_004C	Peripheral Rights Register (SPBA1_PRR19)	32	R/W	0000_0007h	4.8.5.1/223
308F_0050	Peripheral Rights Register (SPBA1_PRR20)	32	R/W	0000_0007h	4.8.5.1/223
308F_0054	Peripheral Rights Register (SPBA1_PRR21)	32	R/W	0000_0007h	4.8.5.1/223

Table continues on the next page...

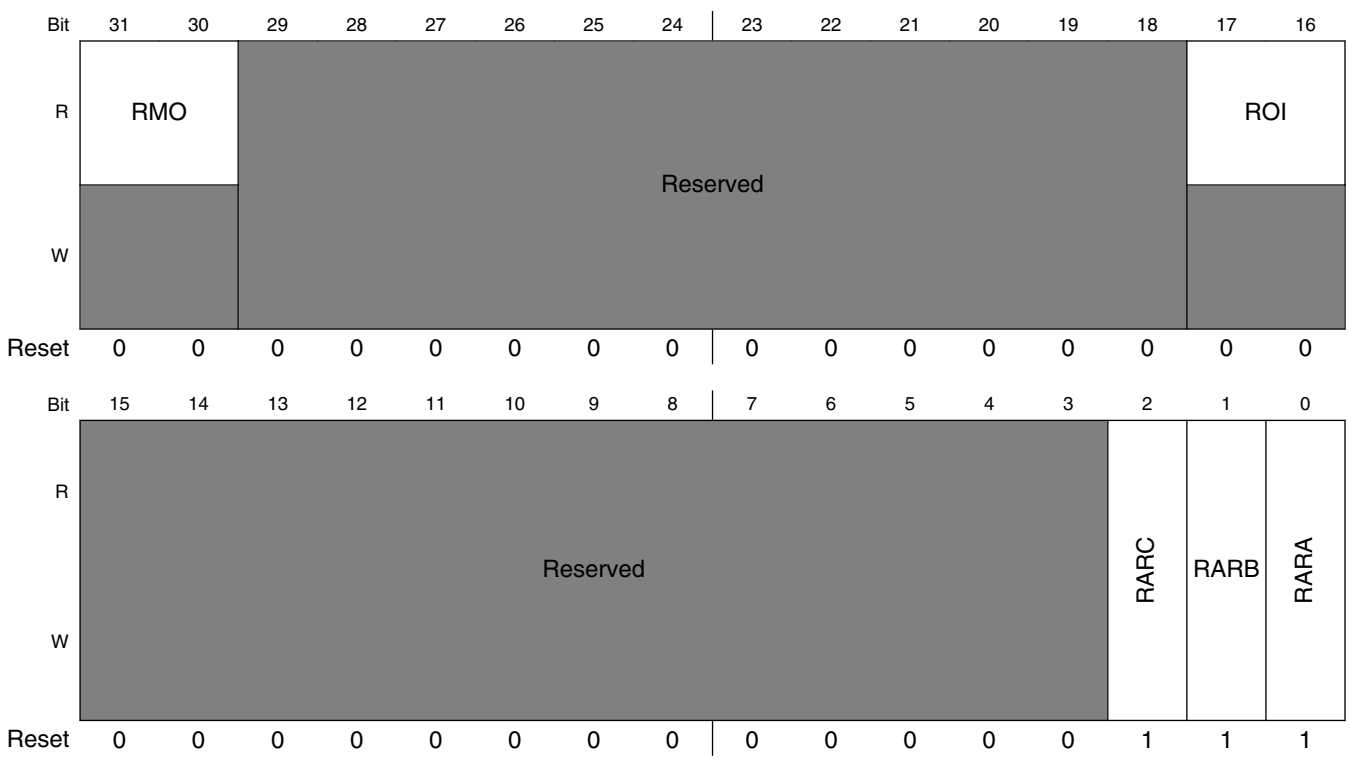
SPBA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
308F_0058	Peripheral Rights Register (SPBA1_PRR22)	32	R/W	0000_0007h	4.8.5.1/223
308F_005C	Peripheral Rights Register (SPBA1_PRR23)	32	R/W	0000_0007h	4.8.5.1/223
308F_0060	Peripheral Rights Register (SPBA1_PRR24)	32	R/W	0000_0007h	4.8.5.1/223
308F_0064	Peripheral Rights Register (SPBA1_PRR25)	32	R/W	0000_0007h	4.8.5.1/223
308F_0068	Peripheral Rights Register (SPBA1_PRR26)	32	R/W	0000_0007h	4.8.5.1/223
308F_006C	Peripheral Rights Register (SPBA1_PRR27)	32	R/W	0000_0007h	4.8.5.1/223
308F_0070	Peripheral Rights Register (SPBA1_PRR28)	32	R/W	0000_0007h	4.8.5.1/223
308F_0074	Peripheral Rights Register (SPBA1_PRR29)	32	R/W	0000_0007h	4.8.5.1/223
308F_0078	Peripheral Rights Register (SPBA1_PRR30)	32	R/W	0000_0007h	4.8.5.1/223
308F_007C	Peripheral Rights Register (SPBA1_PRR31)	32	R/W	0000_0007h	4.8.5.1/223

4.8.5.1 Peripheral Rights Register (SPBAx_PRR*n*)

This register controls master ownership and access for a peripheral.

Address: Base address + 0h offset + (4d × i), where i=0d to 31d



SPBAx_PRR*n* field descriptions

Field	Description
31–30 RMO	Requesting Master Owner. This 2-bit register field indicates if the corresponding resource is owned by the requesting master or not. This register is reset to 2'b0 if ROI = 2'b0. 00 **UNOWNED** — The resource is unowned. 01 Reserved. 10 **ANOTHER_MASTER** — The resource is owned by another master. 11 **REQUESTING_MASTER** — The resource is owned by the requesting master.
29–18 -	This field is reserved. Reserved
17–16 ROI	Resource Owner ID. This field indicates which master (one at a time) can access to the PRR for rights modification. This is a read-only register. After reset, ROI bits are cleared ("00" -> un-owned resource). A master performing a write access to the an un-owned PRR will get its ID automatically written into ROI, while modifying RARx bits. It can then read back the RMO, RAR, ROI bits to make sure RMO returns the right value, ROI bits contain its ID and RARx bits are correctly asserted. Then no other master (whom ID is different from the one stored in ROI) will be able to modify RAR fields. Owner master of a peripheral can assert its dead_owner signal, or write 1'b0 in the RARx to release the ownership (ROI[1:0] reset to 2'b0). 00 **UNOWNED** — Unowned resource. 01 **MASTER_A** — The resource is owned by master A port. 10 **MASTER_B** — The resource is owned by master B port. 11 **MASTER_C** — The resource is owned by master C port.
15–3 -	This field is reserved. Reserved
2 RARC	Resource Access Right. Control and Status bit for master C. This field indicates whether master C can access the peripheral. From 0 up to 3 masters can have permission to access a resource (all the master can be granted on a peripheral, but only one access at a time will be granted by SPBA). 0 **PROHIBITED** — Access to peripheral is not allowed. 1 **ALLOWED** — Access to peripheral is granted.
1 RARB	Resource Access Right. Control and Status bit for master B. This field indicates whether master B can access the peripheral. From 0 up to 3 masters can have permission to access a resource (all the master can be granted on a peripheral, but only one access at a time will be granted by SPBA). 0 **PROHIBITED** — Access to peripheral is not allowed. 1 **ALLOWED** — Access to peripheral is granted.
0 RARA	Resource Access Right. Control and Status bit for master A. This field indicates whether master A can access the peripheral. From 0 up to 3 masters can have permission to access a resource (all the master can be granted on a peripheral, but only one access at a time will be granted by SPBA). 0 **PROHIBITED** — Access to peripheral is not allowed. 1 **ALLOWED** — Access to peripheral is granted.

4.9 TrustZone Address Space Controller (TZASC)

4.9.1 Overview

The TrustZone Address Space Controller (TZASC) protects security-sensitive SW and data in a trusted execution environment against potentially compromised SW running on the platform.

The TZASC block diagram is shown in figure below.

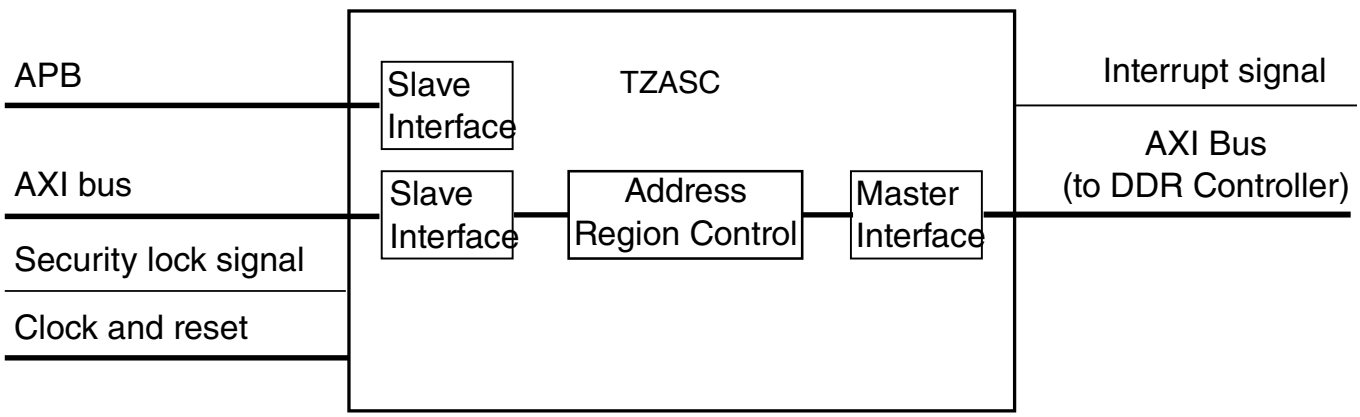


Figure 4-19. TZASC Block Diagram

The TZASC is an IP by Arm ("CoreLink™ TrustZone Address Space Controller TZC-380"), designed to provide configurable protection over program (SW) memory space.

The main features of TZASC are:

- Supports 16 independent address regions
- Access controls are independently programmable for each address region
- Sensitive registers may be locked
- Host interrupt may be programmed to signal attempted access control violations
- AXI master/slave interfaces for transactions
- APB slave interface for configuration and status reporting

NOTE

In this device it is necessary to set TZASC_ID_SWAP_BYPASS in IOMUXC_GPR10[1] to avoid an AXI bus error when using GPU.

4.9.2 Clocks

The table found here describes the clock sources for TZASC.

Table 4-25. TZASC Clocks

Clock name	Clock Root	Description
aclk	ccm_clk_root	Module clock

4.9.3 Address Mapping in various memory mapping modes

The TZASC region base address starts at the beginning of DDR memory space (0x40000000) instead of the beginning of memory map (0x00000000). In this case the addresses configured in TZASC controller will be 1GB (0x40000000) offset and does not match the local addresses.

For example, setting region_setup_low_x=0xBE000000 maps DDR_ADDR=0xFE000000, the same behavior is observed with fail_address_x registers.

Memory "aliasing" implications on TZASC settings - in systems which does not utilize the maximal supported DDR space the controller is designed for, the whole DDR memory map becomes "aliased" (replicated) by the size of the physical memory used. In such cases, the TZASC must be configured to protect all aliased regions as well (i.e. effectively reducing the number of available TZASC regions, since all aliased regions must be handled, for each "real" space needing protection).

For complete details on TZASC functionality and the programming model, see the Arm document, “CoreLink™ TrustZone Address Space Controller TZC-380 Technical Reference Manual, (Rev r0p1 or newer)”, available at **http://infocenter.arm.com**.

4.10 System Debug

4.10.1 Debug

4.10.1.1 Debug Architecture

The chapter describes the debug architecture of the chip.

4.10.1.2 Debug System Features

The chip debug is based on Arm's CoreSight platform, with support for Quad-core A53 platform and Cortex-M4 core. The key features of the debug system include:

- Support 5-pins(JTAG) interface.
- Support both non-intrusive and halt-mode trace/debug options.
- MDM-AP registers for debugger to control mutli-core halt/resume cores.
- Trace Memory Controller (TMC) is used to enable capturing trace.
- 4KB in SOC trace block.
- ETR is used to allow routing trace data to system memory.
- Support ARM real time trace interface (TPIU) 16-bit @133MHz.
- Support cross trigger between Quad Cortex-A53 and Cortex M4.
- 4 JTAG security levels, via SJC security functions together with e-Fuse (challenge response, field return, intrusive detection)

4.10.1.3 System level debug architecture

The debug architecture is shown in the following figure:

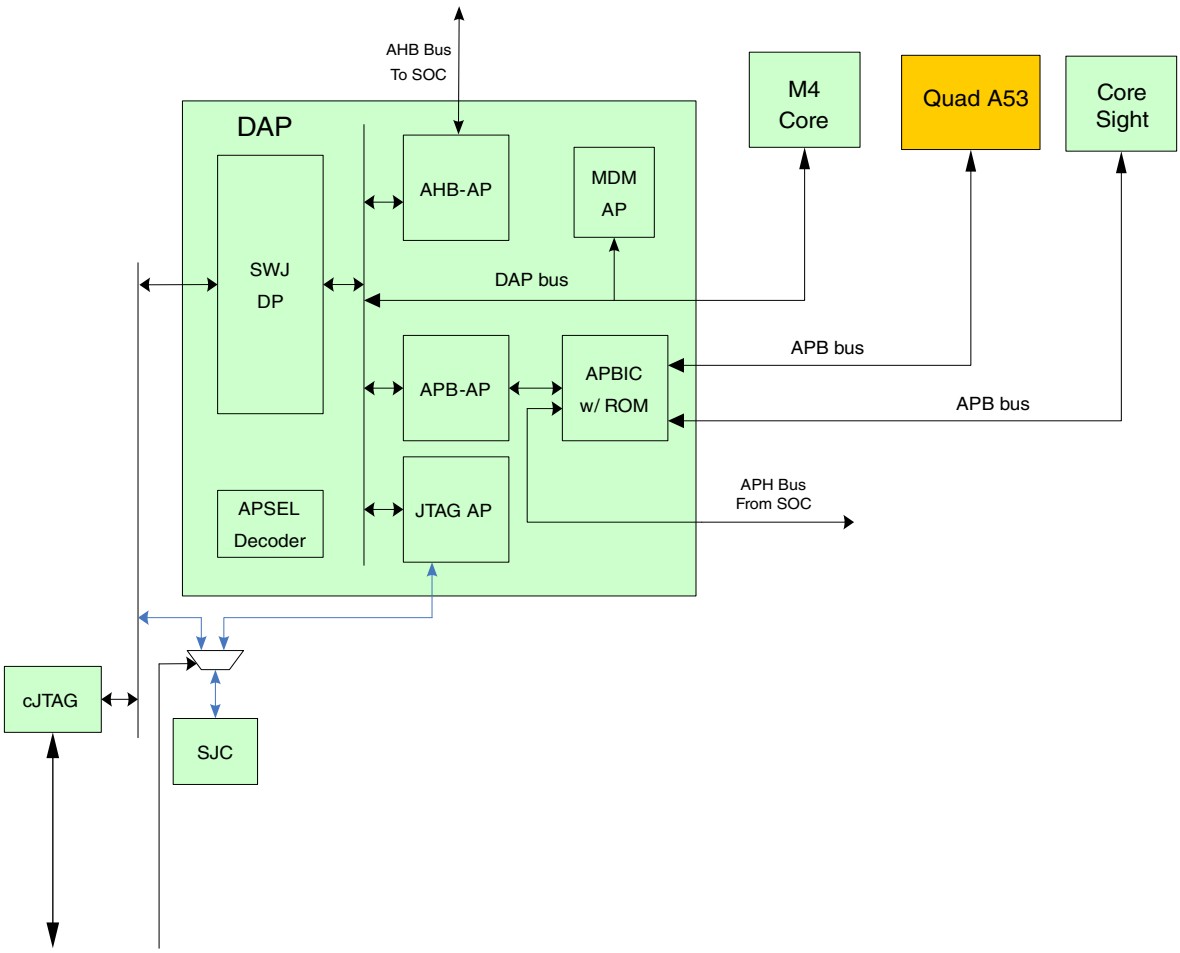


Figure 4-20. Debug Architecture Diagram

4.10.1.4 Functional description

This section gives a brief overview of the modules that are implemented within the Cortex-M/Cortex-A Core Platform. The debug blocks are part of the overall CoreSight platform debug system, which include the ETR, CTM, CTI, ATB replicator, APB address decode, TPIU and DAP. The CoreSight™ compatible Embedded Trace Macrocell (ETM) enables traces of program flow to be collected, compressed, and fed into the trace infrastructure. The Cross Trigger Interface (CTI) is included to provide a common programming model for use by the debug tools, control the trigger sources, and interface to the Cross Trigger Matrix (CTM). The debug is controlled via an ARM Debug Access Port (DAP).

4.10.1.4.1 Embedded Trace Router (ETR)

The Embedded Trace Router (ETR) is a CoreSight trace component which buffers trace data before dumping that data into DRAM through the AXI bus (via AXI master port). The output of the funnel is the ETR, and has the ability to buffer a certain amount before writing the data to DRAM.

4.10.1.4.2 Embedded Trace Macrocell (ETM)

The Embedded Trace Macrocell (ETM) is a CoreSight component. The ETM trace unit is a module that performs real-time instruction flow tracing based on the Embedded Trace Macrocell (ETM) architecture ETMv4.

4.10.1.4.3 Cross Trigger Matrix (CTM)

The CoreSight CTI channel signals from all the cores are combined using a Cross Trigger Matrix (CTM) block so that a single cross trigger channel interface is presented in the Cortex processor. This module can combine four internal channel interfaces corresponding to each core along with one external channel interface.

In the Cortex processor CTM, the external channel output is driven by the OR output of all internal channel outputs. Each internal channel input is driven by the OR output of internal channel outputs of all other CTIs in addition to the external channel input.

4.10.1.4.4 Cross Trigger Interface (CTI)

The Cortex processor has a single external cross trigger channel interface. This external interface is connected to the CoreSight Cross Trigger Interface (CTI) interface corresponding to each core through a Cross Trigger Matrix (CTM). A number of Embedded Cross Trigger (ECT) trigger inputs and trigger outputs are connected between debug components in the Cortex-A processor and CoreSight CTI blocks.

The CTI enables the debug logic, ETM trace unit, and performance monitoring, to interact with each other and with other CoreSight components. This is called cross triggering. For example, you configure the CTI to generate an interrupt when the ETM trace unit trigger event occurs.

4.10.1.4.5 AMBA Trace Bus Interface (ATB)

The AMBA Trace Bus (ATB) CoreSight debug components are part of the Trace stream block. The ATB components consist of the ATB Funnel and ATB Replicator. The ATB Funnel is used to merge several streams together to produce a single output to the ATB Replicator. The ATB Replicator enables two trace sinks to be wired together and receive ATB trace data from the same trace source.

4.10.1.4.6 Advanced Peripheral Bus Interface (APB)

The APB asynchronous interface / bridge connects several debug components. Each component has a single separate APB interface from the APB-IC (w/ ROM), which hooks up to a separate APB asynchronous component. The APB interface connects the core debug functions to the DAP.

4.10.1.4.7 Instrumentation Trace Macrocell (ITM)

The Instrumentation Trace Macrocell (ITM) is a application-driven trace source that supports printf style debugging to trace operating system and application events. The ITM generates diagnostic system information as packets. Multiple sources can generate packets. If multiple sources generate packets at the same time, the ITM arbitrates the order in which packets are output. These sources in decreasing order of priority are:

- **Software trace.** Software can write directly to ITM stimulus registers to generate packets.
- **Hardware trace.** The DWT generates these packets, and the ITM outputs them.
- **Time stamping.** Timestamps are generated relative to packets. The ITM contains a 21-bit counter to generate the timestamp. The Cortex-M4 clock or the bitclock rate of the Serial Wire Viewer (SWV) output clocks the counter.

4.10.1.4.8 Data Watchpoint and Trace (DWT)

The Data Watchpoint and Trace (DWT) debug unit provides watchpoints, data tracing, and system profiling for the processor.

4.10.1.4.9 Debug Access Port (DAP)

The Debug Access Port (DAP) is a physical port that connects to external debug tools. The DAP is part of the standardized ARM Debug Interface, and provides a bridge between a JTAG pin interface and on-chip memory mapped peripherals via the DAP bus. Transactions generated by the DAP are referred to as External Debugger Accesses.

4.10.1.5 JTAG topology

There is only one JTAG on the chip, and two JTAG modes are supported. Select via the JTAG_MOD pin.

- Debug mode: JTAG_MOD == 0, DAP is the only TAP controller in the daisy chain. SJC will be attached to JTAG-AP of DAP.
- Test mode: JTAG_MOD == 1, SJC is the only TAP controller in the daisy chain. 1149.1-compliant, and support 1149.6 AC coupled test.

When the JTAG interface is in Debug Mode, it can be operating in standard 5-pin JTAG interface. cJTAG/SWD interface can be supported by the IP, but since they are not needed by the targeted application, they will not be supported by this chip.

4.11 System JTAG Controller (SJC)

4.11.1 Overview

The System JTAG Controller (SJC) provides debug and test control with the maximum security.

The test access port (TAP) is designed to support features compatible with the IEEE Standard 1149.1 v2001 (JTAG).

The figure below shows an overview of the JTAG architecture.

Figure 4-21. System JTAG Controller (SJC) Block Diagram

4.11.1.1 Features

The System JTAG Controller (SJC) provides the following capabilities:

- JTAG IEEE1149.1 mandatory instructions, see EXTEST Instruction, SAMPLE/ PRELOAD Instruction , and BYPASS Instruction .
- JTAG IEEE1149.1 optional instructions, see ID_CODE Instruction (IDCODE) , and HIGHZ Instruction.
- JTAG IEEE P1149.1 (standard JTAG) interface to off-chip test and development equipment including an SJC-only mode for true IEEE 1149.1 compliance, used primarily for board-level implementation of boundary scan.

- IEEE P1149.6 (JTAG) mandatory instructions, see EXTEST_PULSE instruction and EXTEST_TRAIN instruction. These two instructions enable edge-detecting behavior on the signal path containing AC pins.
- Debug-related control and status, such as putting selected cores into reset and/or debug mode and the ability to monitor individual core status signals via JTAG.
- Provides means for accessing each OnCE/ICE TAP controller independently to control a target system (see Modes of Operation).
- ExtraDebug logic (see ENABLE_ExtraDebug Instruction).
- The maximum clock speed of the SJC is one-eight of the lowest frequency of the accessed OnCE/ICE. For example in normal operation (no core in low-power mode), this frequency is one-eight of the SDMA frequency if this core is present in the TDI-TDO chain (serially connected with other cores or standalone). The user must also consider the 25 MHz frequency limitation on the CE bus.
- Core compliant modes to support standalone core debuggers (see Modes of Operation).
- Multi-cores daisy chained mode (default one) to support multi-core debuggers (see Modes of Operation).

Detailed information about the SJC is provided in the Security Reference Manual. Contact your NXP representative for information about obtaining this document.

4.11.1.2 Modes of Operation

The SJC modes are controlled through both the TAP select register (SJC_TSR) and the MOD input port.

The MOD port (typically connected to pad of the same name) selects between two possible topologies of TAP connections, as seen at SoC level:

- Negating it (this should be the default state) selects all the TAPs (SJC, SDMA, DAP and Arm/ETM) to be connected in the TDI-TDO chain, which is referred to as "daisy chain" mode, throughout this chapter.
- Asserting it only selects the SJC TAP to be connected in the TDI-TDO chain.

IEEE1149.1 standard features are enabled by configuring the SJC input pin: MOD. Refer to the following table for MOD settings details:

Table 4-26. SJC Modes

MOD	Name	Description
0	Daisy chain ALL	For common SW debug (High speed and production)
1	SJC only	IEEE 1149.1 JTAG compliant mode

The following figure shows the SJC mode selection flow. The numbers shown in parenthesis below each block name indicates the TAP's IR length.

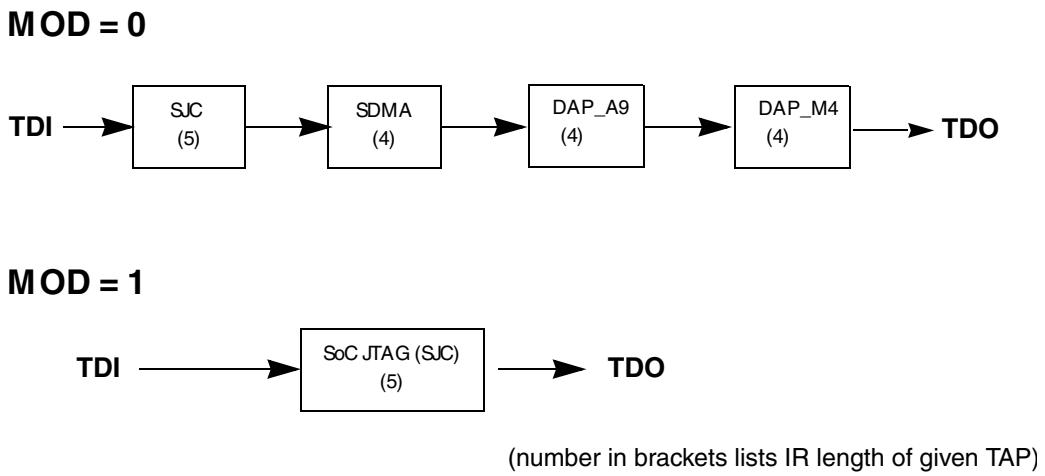


Figure 4-22. SJC Mode Selection Using MOD Pin Sampling

The Connect SDMA bit inside TAP select register controls the SDMA TAP bypass.

- When negated (should be the default state), the SDMA TAP is bypassed with a single D-FF (Flip-flop) during Shift-Dr path
- When asserted SDMA TAP is connected inside the chain
- When taking the SDMA into bypass or out of bypass (by writing to tapsel reg), additional cycle with TMS '0' should be given

The TAP selection block (TSB) provides a simple method of integrating various pieces of IP that have embedded TAPs.

- Provides a way to connect up multiple TAPs within a single SoC
- Identify the SJC TAP as the master TAP which controls the boundary chain (for IEEE 1149.1 standard compliance)
- Follow the state of SJC TAP, and when the Test-Logic-Reset (TLR) state is reached, reset all TAPs

The figure below shows the TAP Selection Block and SOC TAP Chain Scheme.

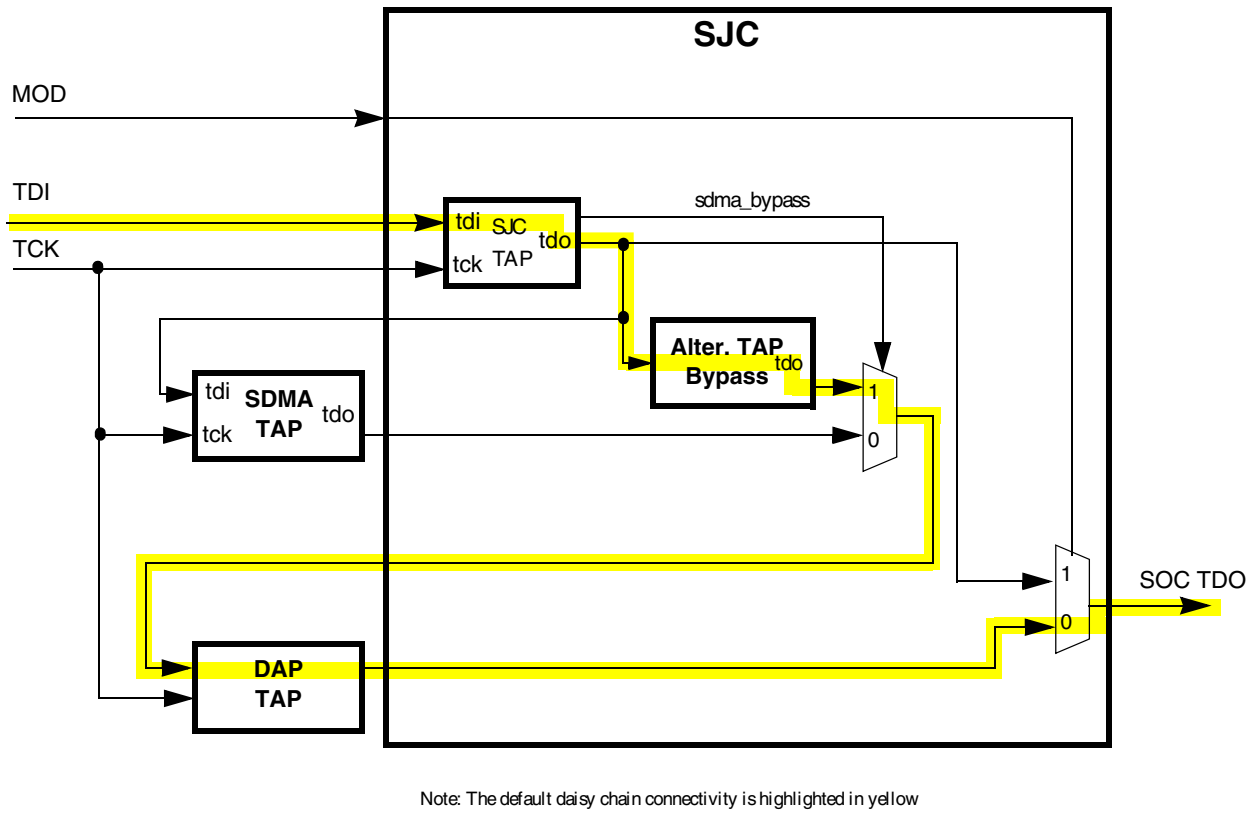


Figure 4-23. TAP Selection Block and SoC TAP Chain Scheme

NOTE

It is the responsibility of the user to ensure that in any configuration of the TAP controllers chosen, all of the TAPs in the chain comply with the demands of TCK clock frequency as well as the required ratio between TCK clock frequency and that of the core's to which the TAP refers.

4.11.2 TAP Selection Block (TSB)

As described in Modes of Operation, the SJC can access cores in different modes selected through a TSB.

4.11.2.1 Select Mode Using Software

Conceptually, the SJC_TSR is a data register which is accessed through Access TSR IR instruction of SJC TAP.

The following figure shows the process of using reserved IR to access the SJC_TSR.

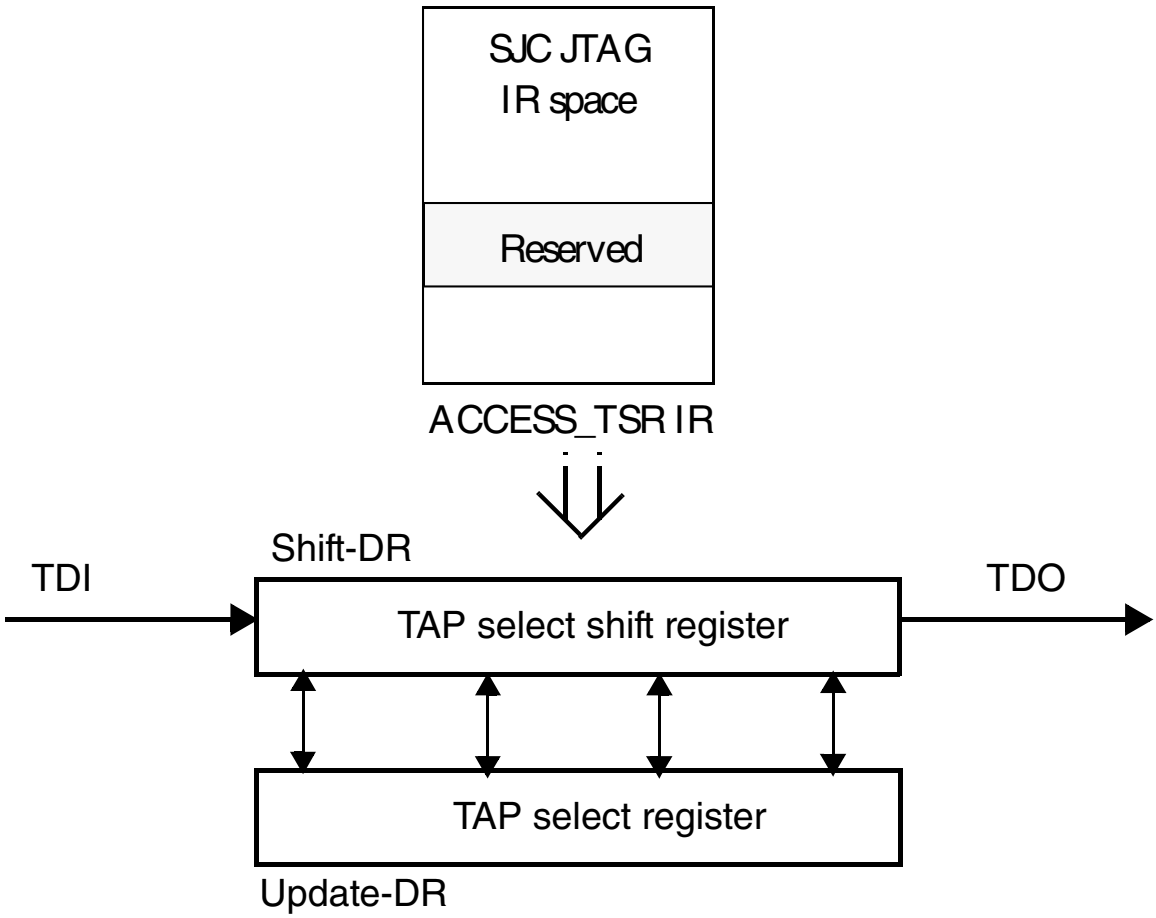


Figure 4-24. Using Reserved IR to Access the TAP Select Register (SJC_TSR)

The SJC_TSR can only be changed during the update-DR state of the TSB JTAG state machine. This is necessary to prevent a TAP that is being selected from losing synchronization with the TSB state machine when the TSB state machine returns to run-test-idle. Therefore, an associated shift register for the SJC_TSR is loaded into the SJC_TSR during the update-DR state (see the figure above). The shift register must also capture the state of the SJC_TSR when in the Capture-DR state for visibility of the contents of the SJC_TSR. See TAP Select Instruction , for more information.

4.11.3 Boundary Scan Register (BSR)

The Boundary Scan Register (BSR) in the JTAG implementation contains bits for all device signal and clock pins and associated control signals.

All SoC bidirectional pins have a single register bit in the boundary scan register for pin data, and are controlled by an associated control bit in the boundary scan register.

4.11.4 SoC JTAG Instruction Register (SJIR)

The SoC JTAG Instruction register is provided in the following table. The SoC JTAG Instruction register is 5 bits wide.

Table 4-27. SoC JTAG Instruction Register (SJIR)

Code					SJC IR
B4	B3	B2	B1	B0	
0	0	0	0	0	IDCODE
0	0	0	0	1	SAMPLE/PRELOAD
0	0	0	1	0	EXTEST
0	0	0	1	1	HI-Z
0	0	1	0	0	ENABLE_ExtraDebug
0	0	1	0	1	ENTER_DEBUG (secured)
0	0	1	1	0	Reserved
0	0	1	1	1	TAP select
0	1	0	0	0	EXTEST_PULSE
0	1	0	0	1	EXTEST_TRAIN
0	1	0	1	0	Reserved
0	1	0	1	1	Reserved
0	1	1	0	0	Security Output challenge
0	1	1	0	1	Security Enter response
-					Reserved
1	1	1	1	1	BYPASS

The instruction register is reset to 0b00000 in the test-logic-reset controller state which is equivalent to the IDCODE instruction.

During the capture-IR controller state, the parallel inputs to the instruction register are loaded with the code 01 in the least significant bits as required by the standard; the most significant bits are loaded with the values 00, leading to a capture value of 0b00001.

4.11.4.1 ID_CODE Instruction (IDCODE)

Selects the ID register, and the system logic controls the I/O pins. This instruction is provided as a public instruction to allow the manufacturer, part number and version of a component to be determined through the TAP.

The table below shows the ID register configuration.

Table 4-28. ID Configuration Register Description (IDCODE)

Field	Description
31-28 Version Information	IC/SoC Version information number. Initial value: '0000' This number is subject to changes, for new IC/SoC (System On A Chip) revision releases.
27-12 Part Number	Customer Part Number The 16-bit Part Number value is unique for every NXP SoC / IC. See System Debug chapter for exact register value for a specific SoC.
11-1 Manufacturer Identity	Manufacturer Identity NXP Manufacturer Identity code. Bits [11:1] - 00000001110
0	Tied to logic 1.

One application of the ID register is to distinguish the manufacturer(s) of components on a board when multiple sourcing is used. As more components emerge which conform to the IEEE 1149.1 standard, it is desirable to allow for a system diagnostic controller unit to blindly interrogate a board design to determine the type of each component in each location. This information is also available for factory process monitoring and for failure mode analysis of assembled boards.

Once the IDCODE instruction is decoded, it selects the ID register which is a 32 Bit data register. Because the bypass register loads a logic 0 at the start of a scan cycle, whereas the ID register loads a logic 1 into its least significant bit, examination of the first bit of data shifted out of a component during a test data scan sequence immediate following exit from Test-Logic-Reset controller state shows whether such a register is included in the design. When the IDCODE instruction is selected, the operation of the test logic has no effect on the operation of the on-chip system logic as required by the IEEE 1149.1 standard.

4.11.4.2 SAMPLE/PRELOAD Instruction

Selects the boundary scan register and the system logic controls the I/O pins.

The SAMPLE/PRELOAD instruction provides two separate functions:

- First, it provides a means to obtain a snapshot of system data and control signals. The snapshot occurs on the rising edge of TCK in the capture-DR controller state. The data can be observed by shifting it transparently through the boundary scan register.
- The second function of SAMPLE/PRELOAD is to initialize the boundary scan register output cells prior to selection of EXTEST. This initialization ensures that known data appears on the outputs when entering the EXTEST instruction.

NOTE

Because there is no internal synchronization between the JTAG clock (TCK) and the system clock (CLK), the user must provide some form of external synchronization to achieve meaningful results.

For more details on the function and use of SAMPLE/PRELOAD, refer to the appropriate IEEE 1149.1 document.

4.11.4.3 EXTEST Instruction

Selects the boundary scan register, and the 1149.1 test logic has control of the I/O pins.

By using the TAP controller, the register is capable of:

- Scanning user-defined values into the output buffers,
- Capturing values presented to input pins
- Controlling the direction of bidirectional pins,
- Controlling the output drive of tri-statable output pins.

For more details on the function and use of EXTEST, refer to the appropriate IEEE 1149.1 document.

The EXTEST instruction also asserts internal reset for the cores (through CCM, refer to Figure 4-27) to force a predictable internal state while performing external boundary scan operations.

4.11.4.4 HIGHZ Instruction

All output drivers, including the two-state drivers, are turned off (that is, high impedance). The instruction selects the bypass register.

In this mode, all internal pullup resistors on all the pins (except for the TMS, TDI, TCK, TRSTB pins) are disabled. This disabling functionality is not built into SJC, but should be implemented by some logic in the SOC/IO Pads.

For more details on the function and use of HIGHZ, refer to the IEEE 1149.1 document.

The HIGHZ instruction also asserts internal reset for the cores (through CCM, refer to Figure 4-27) to force a predictable internal state while performing external boundary scan operations.

4.11.4.5 BYPASS Instruction

Selects the single Bit bypass register and the system logic controls the I/O pins.

This creates a shift-register path from TDI to the bypass register and, finally, to TDO, circumventing the boundary scan register. This instruction is used to enhance test efficiency when a component other than the SoC Core based device becomes the device under test.

When the bypass register is selected by the current instruction, the shift-register stage is set to a logic zero on the rising edge of TCK in the capture-DR controller state. Therefore, the first bit to be shifted out after selecting the bypass register is always a logic zero.

For more details on the function and use of BYPASS, refer to the appropriate IEEE 1149.1 document.

4.11.4.6 ENABLE_ExtraDebug Instruction

The TDI and TDO pins are connected directly to the ExtraDebug registers, the SJC TAP controller remaining connected to TDI and TMS.

The ExtraDebug shift register consists of 38 bits (maximum) comprising a 32-bits data field (maximum length, see Accessing ExtraDebug Registers,), a 5 bits address field and read/$\overline{\text{write}}$ bit. On a register read, the data field does not need to be filled in. The particular ExtraDebug register connected between TDI and TDO at a given time is selected by the ExtraDebug controller depending on the ExtraDebug Address being currently decoded. All communication with the ExtraDebug controller is done through the Select-DR-Scan path of the JTAG TAP Controller.

4.11.4.7 ENTER_DEBUG instruction

The ENTER_DEBUG instruction is used to generate a debug request event to SDMA and the Arm Core Platform simultaneously (practically, inherited minimal skew is expected, due to difference in event signal propagation in the different modules).

The TDI and TDO are connected to the Instruction Register (IR). After the acknowledgment of the Debug Mode is received (can be checked by reading the Core Status Register part of the ExtraDebug logic), the user can perform system debug functions on the cores.

NOTE

The ENTER_DEBUG event issue to the cores, can be masked, by bits in DCR register.

NOTE

It is user's responsibility to shift-in another IR value (like IDCODE) before trying to bring the cores out of debug mode, as the debug request signals to the cores remains asserted as long as ENTER_DEBUG IR is in place.

NOTE

The user need to check that cores are in debug mode (watching debug acknowledge signal) before leaving ENTER_DEBUG instruction, otherwise debug request might not take affect.

4.11.4.8 TAP Select Instruction

By means of TAP select instruction a user can access TAP select register and by controlling its only bit SDMA Bypass, control whether SDMA TAP is bypassed or not.

Table 4-29. TAP Select Register (TSR)

	TAP Select Register
	BIT 0
	Connect SDMA
TYPE	rw
RESET	0
Note:	

Table 4-30. TAP Select Register Description

Field	Description
0 SDMA Bypass	Connect SDMA Control whether SDMA TAP is bypassed or not: • 0 - SDMA TAP is bypassed by the alternate TAP inside SJC (emulating 4-bit IR and 1-bit bypass path). • 1 - SDMA TAP is connected to the TDI-TDO chain. **NOTE:** Additional cycle with TMS '0' should be inserted, after writing to this register, to allow the SDMA tap be sync before SDMA get into / out of bypass.

4.11.4.9 EXTEST_PULSE instruction

The EXTEST_PULSE instruction implements test behaviors for AC pins and simultaneously behaves identically to IEEE Std 1149.1 EXTEST for DC pins.

4.11.4.10 EXTEST_TRAIN instruction

The EXTEST_TRAIN instruction implements test behaviors for AC pins and simultaneously behaves identically to IEEE Std 1149.1 EXTEST for DC pins.

4.11.5 Security

JTAG manipulation is one of the known hackers' ways of executing unauthorized program code, getting control over the OS and run code in privileged modes.

The SJC provides a debug access to several H/W blocks including the Arm processor and the system bus. This allows for program control and manipulation as well as visibility into system peripherals and memory. The ETM and NEXUS interfaces allow bus transactions to be traced. Together these tools provide the hacker all the access needed to completely comprise the system. Means must be provided to block any malicious JTAG access.

The SJC provides a way of regulating the JTAG access.

The following are the different JTAG security modes:

- **Mode #1: No Debug**-Maximum Security. All security sensitive JTAG features are permanently blocked.
- **Mode #2: Secure JTAG**-High security. JTAG use is regulated by secret key based authentication mechanism.
- **Mode #3: JTAG Enabled**-Low security. JTAG always enabled.

The JTAG security modes are configured using eFUSEs which can burned after packaging by applying electrical signals. The fuse burning is irreversible process, once a fuse is burned (e-fuse or laser fuse) it is impossible to change the fuse back to the un-burned state.

4.11.5.1 JTAG Security Modes

JTAG can be in one of JTAG security modes which is selected by setting the SJC eFUSE configuration. The physical location of the fuses is not in the SJC.

4.11.5.1.1 Mode 1: No Debug - Maximum Security

No Debug JTAG security mode provides the highest security level.

In this mode, all JTAG features are disabled except for:

- ScanBoundary Scan
- PLL bypass- Bypass Arm or/and USB PLL.
- Visibility of the following status bits: power mode - normal, standby, stop, shutdown, and so on

These features do not reduce the security level of the product, and they allows to perform important tests and board connectivity checks.

4.11.5.1.2 Mode 2: Secure JTAG - High Security

The Secure JTAG mode limits the JTAG access by using challenge/response based authentication mechanism. Any access to JTAG port is being checked. Only authorized debug devices (that is, devices having the right response) can access the JTAG, unauthorized JTAG access attempts are denied.

The intent of this mode is to allow return field testing. When a secured JTAG device is being returned for debugging, this mode allows authorized re-activation of the JTAG.

4.11.5.1.2.1 Challenge/Response Mechanism in System JTAG Mode

When SJC is in Sysytem JTAG mode the authentication process is as follows:

1. Shift Output Challenge instruction to IR.
2. Passing through Capture-DR state of the SJC and by performing Shift-DR operations Challenge code can be accessed from TDO.
3. Shift Enter Response instruction to IR. By performing Shift-DR, operations enter Response code value through TDI. As Update-DR state is entered, Response code is compared with the correct one.

In Fixed challenge-response pair mode, each part has its individual challenge - response pair which is determined at manufacturing time, and does not change later on. The SJC compares the user's response to the expected response.

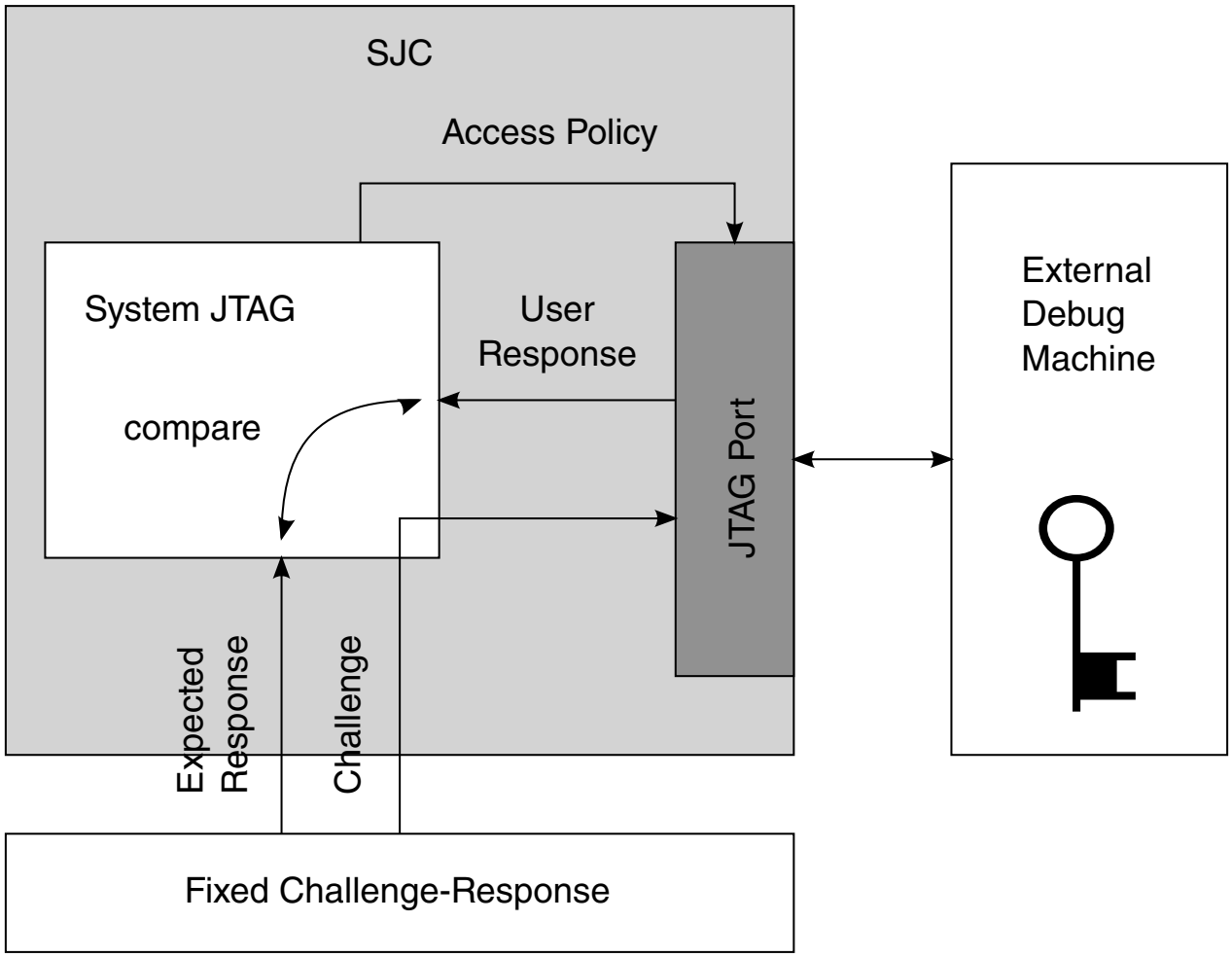


Figure 4-25. Mode #2 - Secure JTAG with Fixed Challenge-response Pair

4.11.5.1.3 Mode 3: JTAG Enabled - Low Security

In the JTAG Enabled JTAG security mode, all JTAG features are enabled.

4.11.5.2 Software Enabled JTAG

To increase the flexibility of the SJC, an option to enable the JTAG via software is added and is available only in Secure JTAG mode. By writing '1' to HAB_JDE (HAB JTAG DEBUG ENABLE) bit in the e-fuse controller module, the JTAG is opened, regardless of its security mode. It is the responsibility of software to assert or negate this bit.

Additionally, a corresponding lock bit is available (in the e-fuse control module) to ensure that only trusted software is able to set the JDE bit. When the LOCK bit is set, no future change of JDE is possible, until the next POR (power-on-reset) cycle.

The platform initialization software should set the LOCK bit for JDE bit before transferring control to the application code.

The S/W JTAG enable allows JTAG enabling without activating the challenge-Response mechanism (which requires JTAG access tool enhancement or special H/W). The JTAG S/W enable does not allow debug in case of boot or memory fault as it requires reset before entering debug.

This feature can be permanently blocked by burning the dedicated e-fuse.

NOTE

The S/W enabled JTAG feature reduces the overall security level of the system as it relies on S/W protections. If this feature is not required, it is strongly recommended to burn the JTAG_HEO e-fuse which disables this feature.

4.11.5.3 Kill Trace

The kill trace signal disables any output of the ETM block. The ETM can be accessed either via JTAG port and/or by direct software code. Blocking the JTAG port also yields assertion of the kill trace signal. This resulted in blocking of trace port. The intention of this action is to block any attempt to break into the system via software manipulation of the debug modules. The kill trace, when active, prevents trace output even in case where it can be activated via chip pin.

The kill trace feature needs to be activated by burning a dedicated e-fuse. If the fuse is left intact, kill trace is never activated as seen in Figure 4-26.

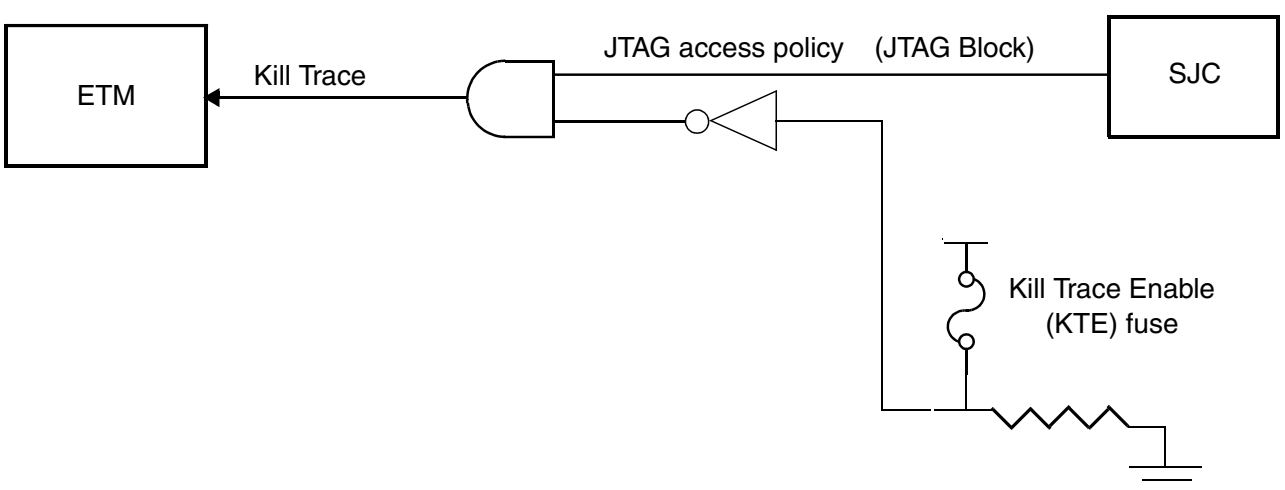


Figure 4-26. Kill Trace eFUSE

The kill trace is asserted when "kill trace enable" fuse is burned and "ipt_secur_block" signal in SJC is asserted, which happens when at least one of the following is true:

- Mode #2 (Secure JTAG) and no code has been entered
- Mode #2 (Secure JTAG) with burned Bypass and Re-enable fuses
- Mode #2 (Secure JTAG) with incorrect response entered
- Mode #1 (No debug)
- TRST_B signal is active
- POR has not ever been asserted

4.11.5.4 SJC Disable Fuse

In addition to the different JTAG security modes that are implemented internally in the System JTAG Controller (SJC), there is an option to disable the SJC functionality by eFUSE configuration. This creates additional JTAG mode that is, JTAG Disabled with highest level of JTAG protection. In this mode all JTAG features are disabled.

Specifically, the following debug features are disabled in addition to the features that were already disabled in No Debug JTAG mode:

- Boundary scan register (SJC_BSR)
- Non-Secure JTAG control registers (PLL configuration, Deterministic Reset, PLL bypass)
- Non-Secure JTAG status registers (Core status)
- Chip Identification Code (IDCODE)

4.11.6 Functional Description

This section provides a complete functional description of the block.

4.11.6.1 Static Core Debug

The SJC JTAG TAP controller is fully compatible with the IEEE 1149.1a-2001 Standard Test Access Port and Boundary Scan Architecture specifications.

The Arm platform has an integrated JTAG interface and a TAP controller to manage its own ICE. Also it can access an embedded trace ETM interface, see Arm core and ETM Technical reference guide for more information.

The SDMA has a TAP controller to manage its own OnCE, see SDMA OnCE specifications for more details.

The OnCE and ICE provide a mean of interacting with the cores and their peripherals non-intrusively so that a user may examine registers, memories to facilitate hardware and software development. Refer to TAP Selection Block (TSB), for more information.

4.11.6.2 Reset Mechanism

The following figure shows the SJC reset logic

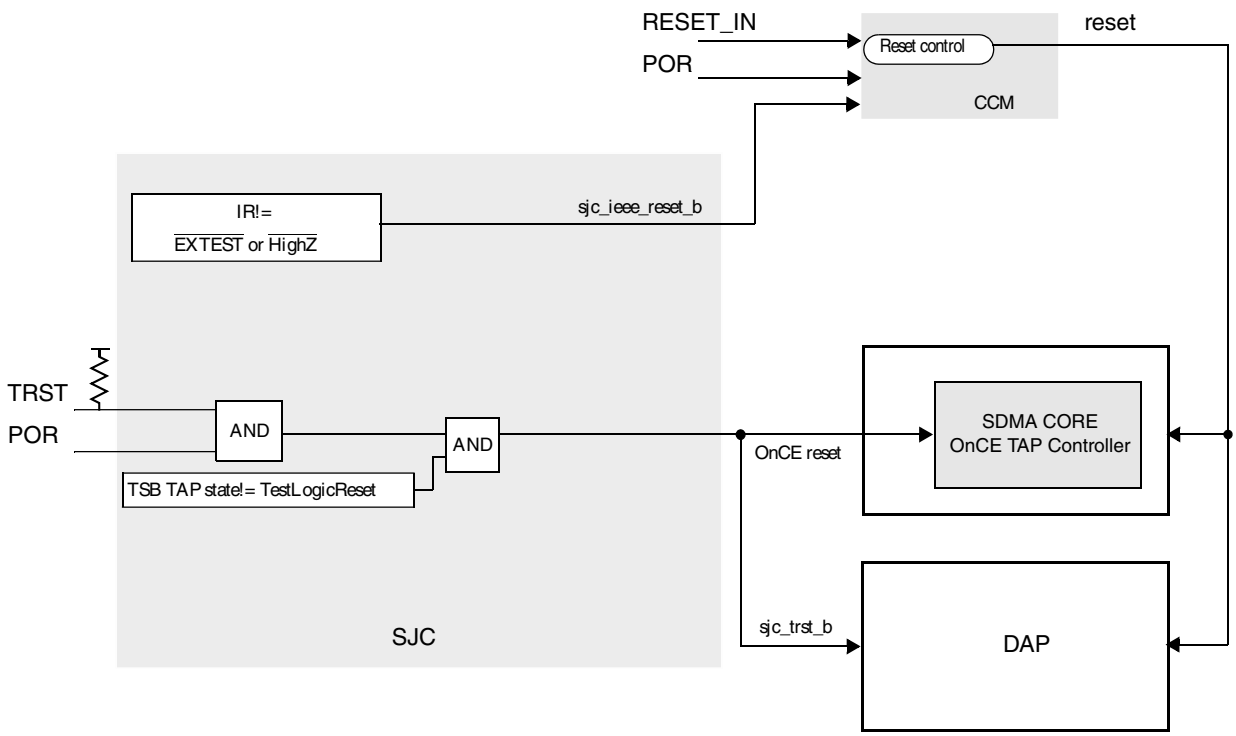


Figure 4-27. SJC Reset Logic

NOTE

- Asserting TRSTB in any scan mode resets the TCR loosing the testmode configuration and selects default TAP.
- SJC generates an IEEE reset signal to the CCM when in one of the IEEE modes HIGHZ or EXTEST. This signal generates a system reset to the cores until exit from one of these modes.
- The TSB generates Once/ICE reset (either TRSTB if implemented or other) when its TAP state reaches Test-Logic-Reset (meaning that TAP accessed is also reaching Test-Logic-Reset).

4.11.7 Initialization/Application Information

The control afforded by the output enable signals using the boundary scan register and the EXTEST instruction requires a compatible circuit-board test environment to avoid device-destructive configurations. The user must avoid situations in which the SJC output drivers are enabled into actively driven networks.

There are two constraints related to the JTAG interface:

- Ensure that the JTAG test logic is kept transparent to the system logic by forcing TAP into the Test-Logic-Reset controller state. During power-up, SJC's internal TRSTB is asserted as IC's POR_B is asserted which forces the TAP controller into this state. After that, if TMS either remains unconnected or is connected to VCC, then the TAP controller cannot leave the Test-Logic-Reset state, regardless of the state of TCK.
- DE_B is an IO pin with pullup and care must be taken of the direction when driving this signal.

4.11.8 SJC Memory Map/Register Definition

In addition to the standard accessible JTAG registers (per IEEE1149.1 standard) listed in SoC JTAG Instruction Register (SJIR) , the chip contains the following registers accessed using the ExtraDebug mechanism, controlled via "ENABLE_ExtraDebug" IR instruction.

NOTE

SJC registers are only accessible by JTAG interface. They are not memory mapped to processor address space, so the absolute addresses provided by default in the SJC memory map are not valid.

This section assumes the JTAG controller is accessed in standalone mode or daisy chained (defined by TAP Selection Block) using the appropriate TSB configuration.

See "System Debug" chapter for more details about the general purpose register descriptions that are unique to this chip.

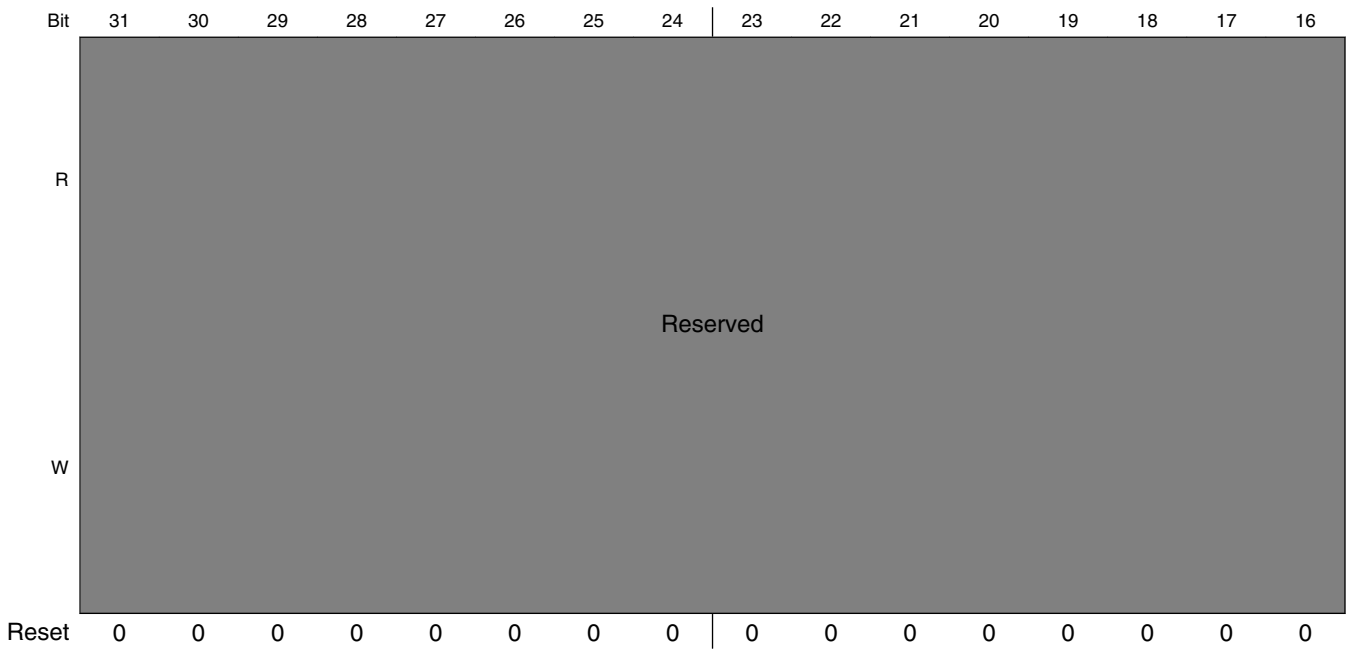
SJC memory map

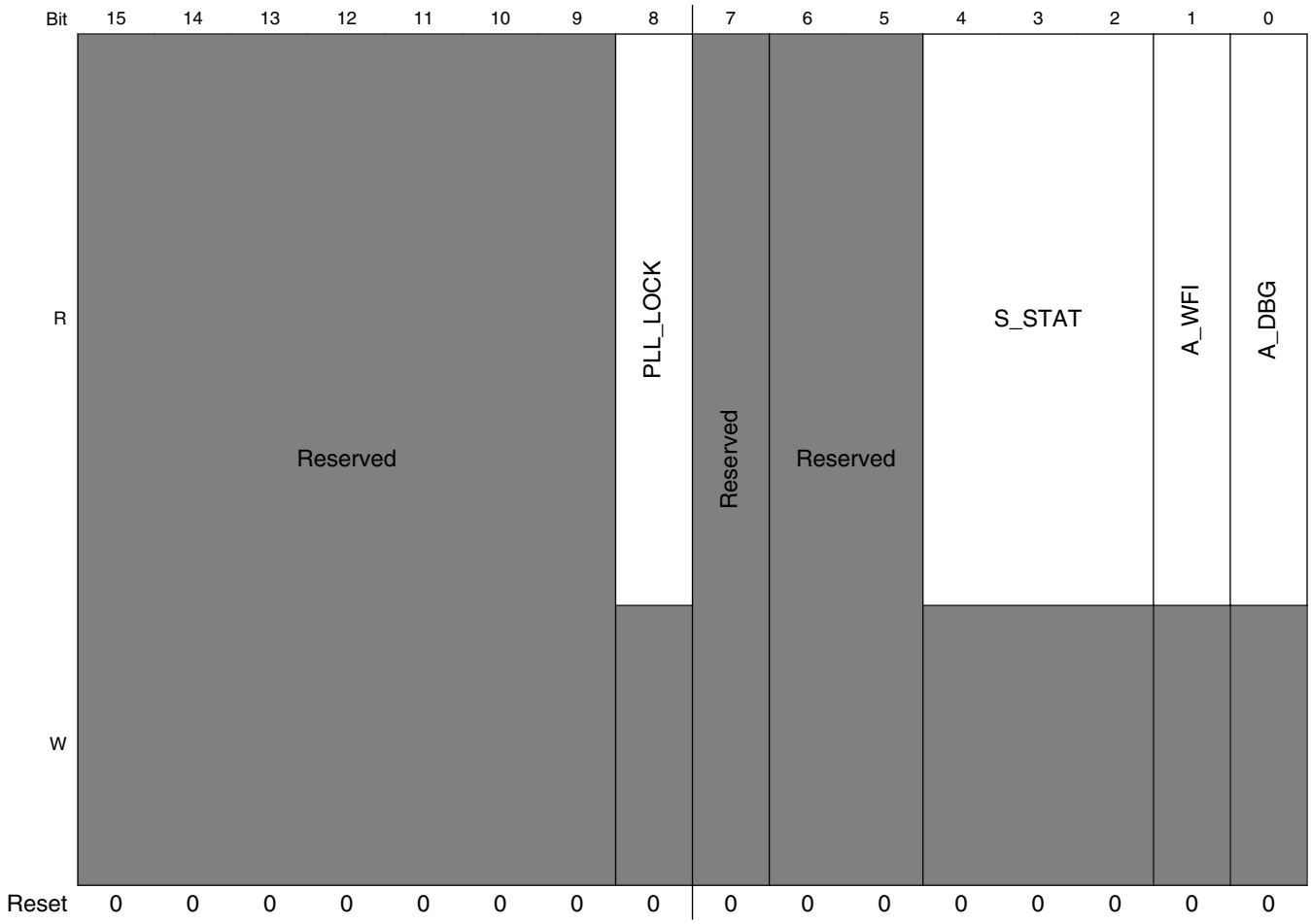
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	General Purpose Unsecured Status Register 1 (SJC_GPUSR1)	32	R	0000_0000h	4.11.8.1/ 249
1	General Purpose Unsecured Status Register 2 (SJC_GPUSR2)	32	R	0000_0000h	4.11.8.2/ 251
2	General Purpose Unsecured Status Register 3 (SJC_GPUSR3)	32	R	0000_0000h	4.11.8.3/ 251
3	General Purpose Secured Status Register (SJC_GPSSR)	32	R	0000_0000h	4.11.8.4/ 252
4	Debug Control Register (SJC_DCR)	32	R/W	0000_0000h	4.11.8.5/ 253
5	Security Status Register (SJC_SSR)	32	R	See section	4.11.8.6/ 255
7	General Purpose Clocks Control Register (SJC_GPCCR)	32	R/W	0000_0000h	4.11.8.7/ 258

4.11.8.1 General Purpose Unsecured Status Register 1 (SJC_GPUSR1)

The General Purpose Unsecured Status Register 1 is a read only register used to check the status of the different Cores and of the PLL. The rest of its bits are for general purpose use.

Address: 0h base + 0h offset = 0h



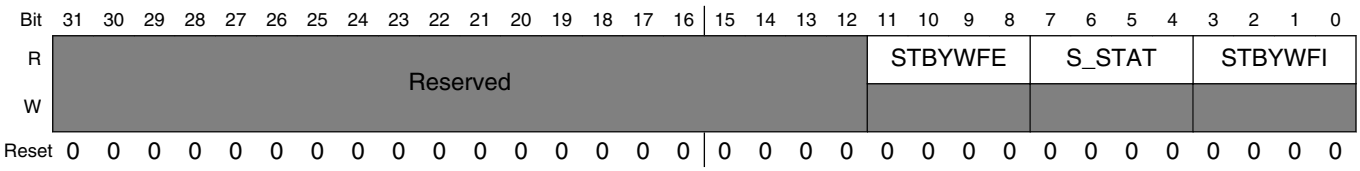


SJC_GPUSR1 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8 PLL_LOCK	PLL_LOCK A Combined PLL-Lock flag indicator, for all the PLL's.
7 -	This field is reserved. Reserved
6–5 -	This field is reserved. Reserved.
4–2 S_STAT	3 LSBits of SDMA core statusH.
1 A_WFI	Arm core wait-for interrupt bit Bit 1 is the Arm core standbywfi (stand by wait-for interrupt). When this bit is HIGH, Arm core is in wait for interrupt mode.
0 A_DBG	Arm core debug status bit Bit 0 is the Arm core DBGACK (debug acknowledge) DBGACK can be overwritten in the Arm core DCR to force a particular DBGACK value. Consequently interpretation of the DBGACK value is highly dependent on the debug sequence. When this bit is HIGH, Arm core is in debug.

4.11.8.2 General Purpose Unsecured Status Register 2 (SJC_GPUSR2)

Address: 0h base + 1h offset = 1h

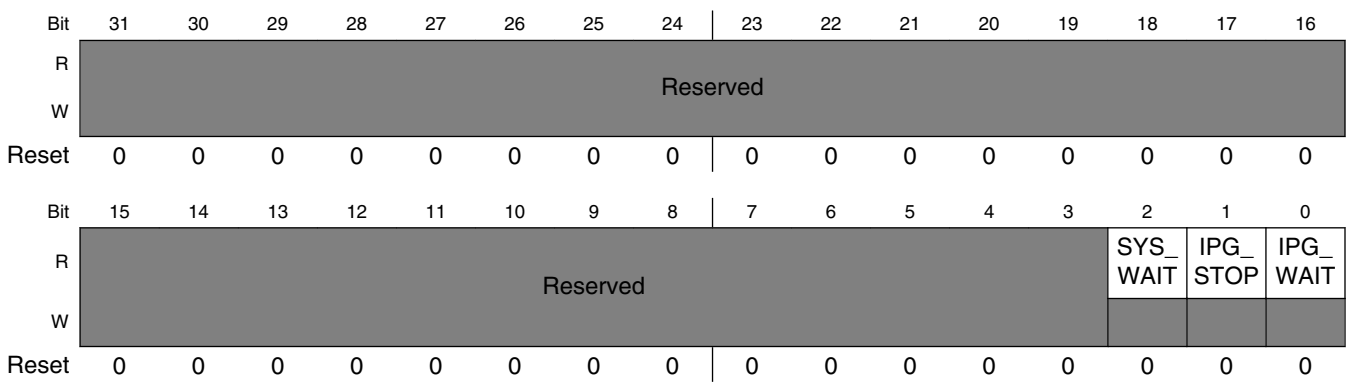


SJC_GPUSR2 field descriptions

Field	Description
31–12 -	This field is reserved. Reserved
11–8 STBYWFE	STBYWFE[3:0] Reflecting the "Standby Wait For Event" signals of all cores.
7–4 S_STAT	S_STAT[3:0] SDMA debug status bits: debug_core_state[3:0]
STBYWFI	STBYWFI[3:0] These bits provide status of "Standby Wait-For-Interrupt" state of all Arm cores.

4.11.8.3 General Purpose Unsecured Status Register 3 (SJC_GPUSR3)

Address: 0h base + 2h offset = 2h



SJC_GPUSR3 field descriptions

Field	Description
31–3 -	This field is reserved. Reserved

Table continues on the next page...

SJC_GPUSR3 field descriptions (continued)

Field	Description
2 SYS_WAIT	System In wait Indication on System in wait mode (from CCM).
1 IPG_STOP	IPG_STOP CCM's "ipg_stop" signal indication
0 IPG_WAIT	IPG_WAIT CCM's "ipg_wait" signal indication

4.11.8.4 General Purpose Secured Status Register (SJC_GPSSR)

The General Purpose Secured Status Register is a read-only register used to check the status of the different critical information in the SoC. This register cannot be accessed in secure modes.

Address: 0h base + 3h offset = 3h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPSSR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

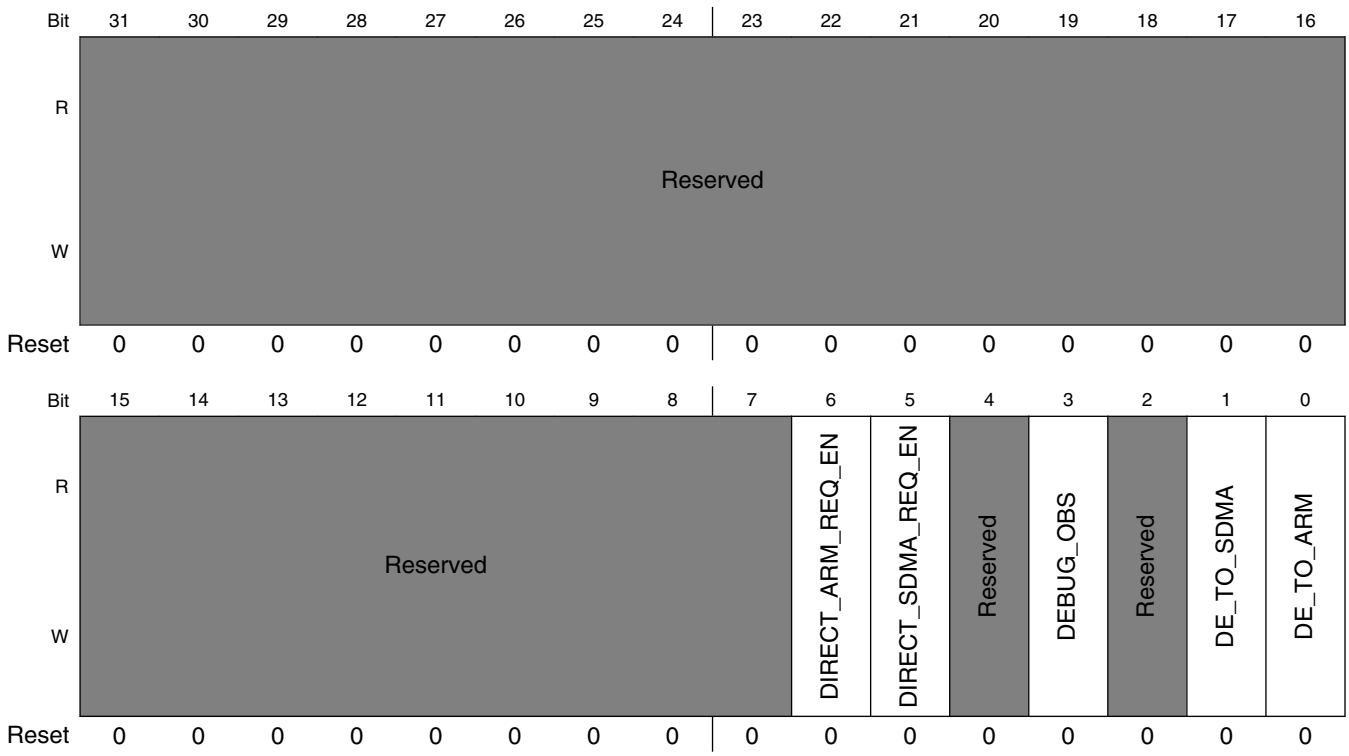
SJC_GPSSR field descriptions

Field	Description
GPSSR	General Purpose Secured Status Register Register is used for testing and debug.

4.11.8.5 Debug Control Register (SJC_DCR)

This register is used to control propagation of debug request from DE_B pad to the cores and debug signals from internal logic to the DE_B pad.

Address: 0h base + 4h offset = 4h



SJC_DCR field descriptions

Field	Description
31–7 -	This field is reserved. Reserved
6 DIRECT_ARM_REQ_EN	Pass Debug Enable event from DE_B pin to Arm platform debug request signal(s). This bit controls the propagation of debug request DE_B to the Arm platform. 0 Disable propagation of system debug to (DE_B pin) to Arm platform. 1 Enable propagation of system debug to (DE_B pin) to Arm platform.
5 DIRECT_SDMA_REQ_EN	Debug enable of the sdma debug request This bit controls the propagation of debug request DE_B to the sdma. 0 Disable propagation of system debug to (DE_B pin) to sdma. 1 Enable propagation of system debug to (DE_B pin) to sdma.
4 -	This field is reserved. Reserved

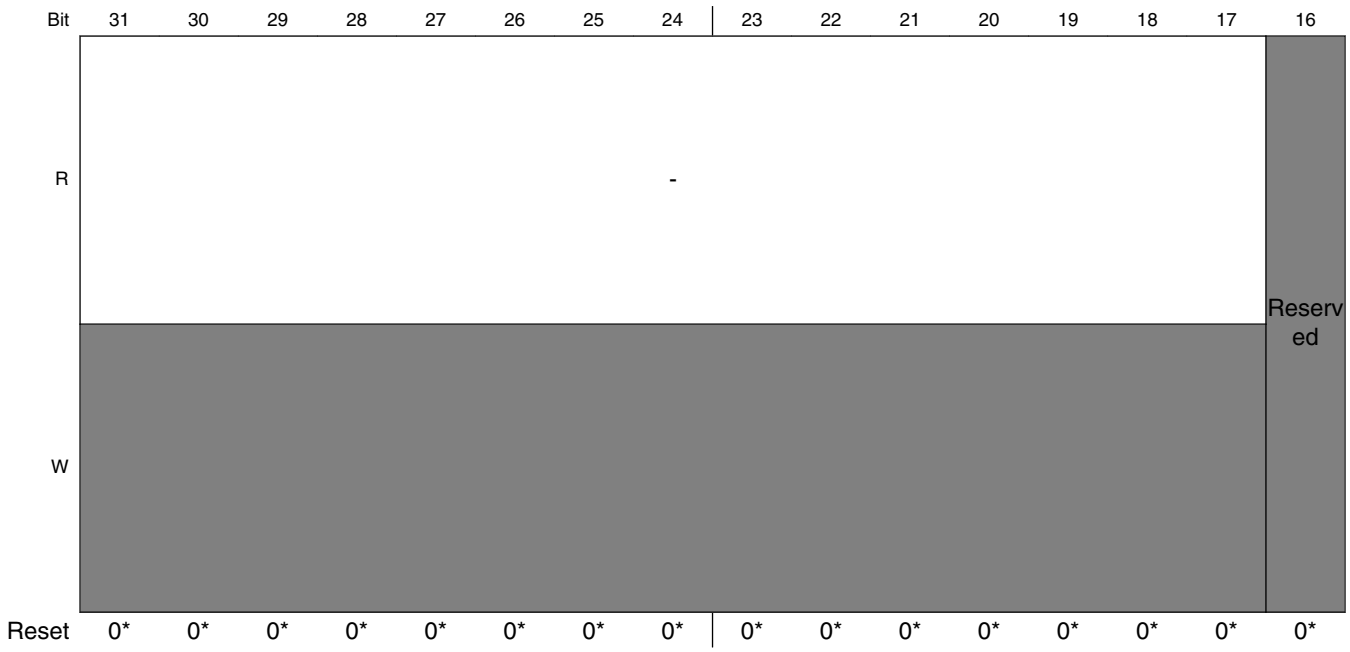
Table continues on the next page...

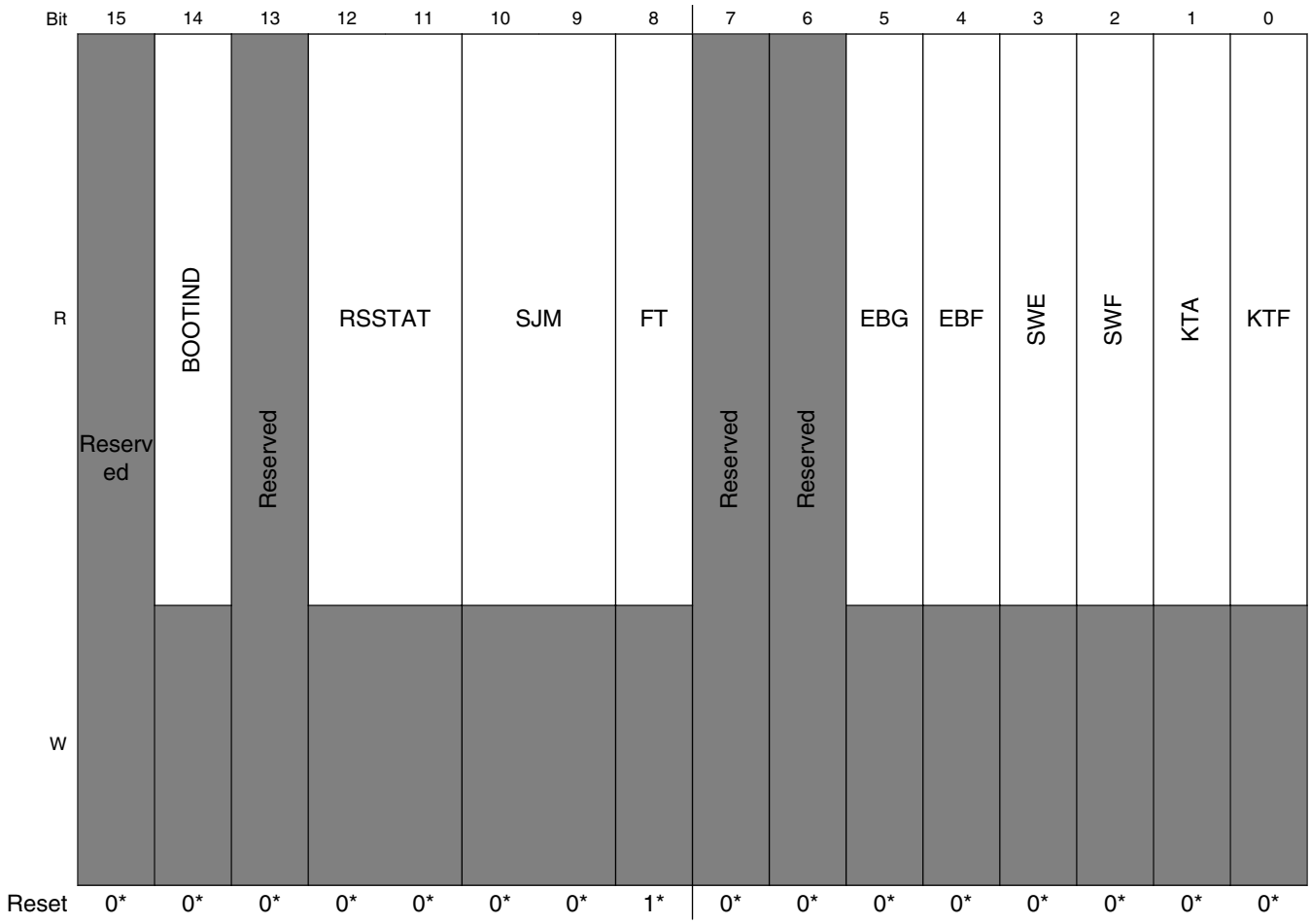
SJC_DCR field descriptions (continued)

Field	Description
3 DEBUG_OBS	Debug observability This bit controls the propagation of the "system debug" input to SJC (driven by the ECT logic), to the DE_B pad. (This logic can be used to pass debug acknowledge event from ECT out to the PAD, for example). The SJC's "system_debug" input is tied to logic HIGH value, therefore, set of "debug_obs" bit, will result in unconditional assertion of DE_B pad. 0 Disable propagation of system debug to DE_B pin 1 Unconditional assertion of pad DE_B
2 -	This field is reserved. Reserved
1 DE_TO_SDMA	SDMA debug request input propagation This bit controls the propagation of debug request to SDMA, when the JTAG state machine is put in "ENTER_DEBUG" IR instruction.. 0 Disable propagation of debug request to SDMA 1 Enable propagation of debug request to SDMA
0 DE_TO_ARM	Arm platform debug request input propagation This bit controls the propagation of debug request to Arm platform ("dbgreq"), when the JTAG state machine is put in "ENTER_DEBUG" IR instruction. 0 Disable propagation of debug request to Arm platform 1 Enable propagation of debug request to Arm platform

4.11.8.6 Security Status Register (SJC_SSR)

Address: 0h base + 5h offset = 5h





* Notes:
- The SJM reset value, reflects the JTAG security state, as defined by status of JTAG_SMODE[1:0] fuses. See the SJM bitfield description for details on valid values.

SJC_SSR field descriptions

Field	Description
31–17 -	Reserved.
16–15 -	This field is reserved. Reserved
14 BOOTIND	Boot Indication Inverted Internal Boot indication, i.e inverse of SRC: "src_int_boot" signal
13 -	This field is reserved. Reserved
12–11 RSSTAT	Response status Response status bits 00 Response wasn't entered 01 Response was entered but not verified

Table continues on the next page...

SJC_SSR field descriptions (continued)

Field	Description
	10 Response was entered and is incorrect 11 Response is correct
10–9 SJM	SJC Secure mode Secure JTAG mode, as set by external fuses. 00 No debug (#1) 01 Secure JTAG (#2) 10 Reserved 11 JTAG enabled (#3)
8 FT	Fuse type Fuse type bit - e-fuse or laser fuse 0 E-fuse technology 1 Laser fuse technology
7 -	This field is reserved. Reserved
6 -	This field is reserved. Reserved
5 EBG	External boot granted External boot enabled, requested and granted 1 granted 0 not granted
4 EBF	External Boot fuse Status of the external boot disable fuse 0 (intact) - external boot is allowed 1 (burned) - external boot is disabled
3 SWE	SW enable SW JTAG enable status 1 enabled 0 disabled
2 SWF	Software JTAG enable fuse Status of the no SW disable JTAG fuse 0 (intact) - SW enable possible 1 (intact) - no SW enable possible
1 KTA	Kill Trace is active 1 active 0 not active
0 KTF	Kill Trace Enable fuse value 0 (intact) - kill trace is never active 1 (burned) - kill trace functionality enabled

4.11.8.7 General Purpose Clocks Control Register (SJC_GPCCR)

This register is used to configure clock related modes in SOC, see System Configuration chapter for more information. Those bits are directly connected to JTAG outputs. Bit 0 of GPCCR controls SDMA clocks invocation. When out of reset, the SDMA is in sleep mode with no SDMA clock running. Unlike events, debug requests does not wake SDMA if it is in sleep mode. The debug request is recognized by the SDMA only when it exits sleep mode upon reception of an event. To be able to enter debug mode even if no event is triggered, the SDMA clock on bit needs to be set prior to sending the debug request (clear at reset).

Address: 0h base + 7h offset = 7h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	-															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	-														ACLKOFFDIS	SCLKR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SJC_GPCCR field descriptions

Field	Description
31–2 -	Reserved
1 ACLKOFFDIS	Disable/prevent Arm platform clock/power shutdown
0 SCLKR	SDMA Clock ON Register - This bit forces the clock on of the SDMA

Chapter 5
Clocks and Power Management

5.1 Clock Control Module (CCM)

5.1.1 Overview

Clock Control Module (CCM) manages the on-chip module clocks. CCM receives clocks from PLLs and oscillators and creates clocks for on-chip peripherals through a set of multiplexers, dividers and gates. When entering or exiting a low power mode, CCM automatically turns on and off PLLs and peripheral clocks.

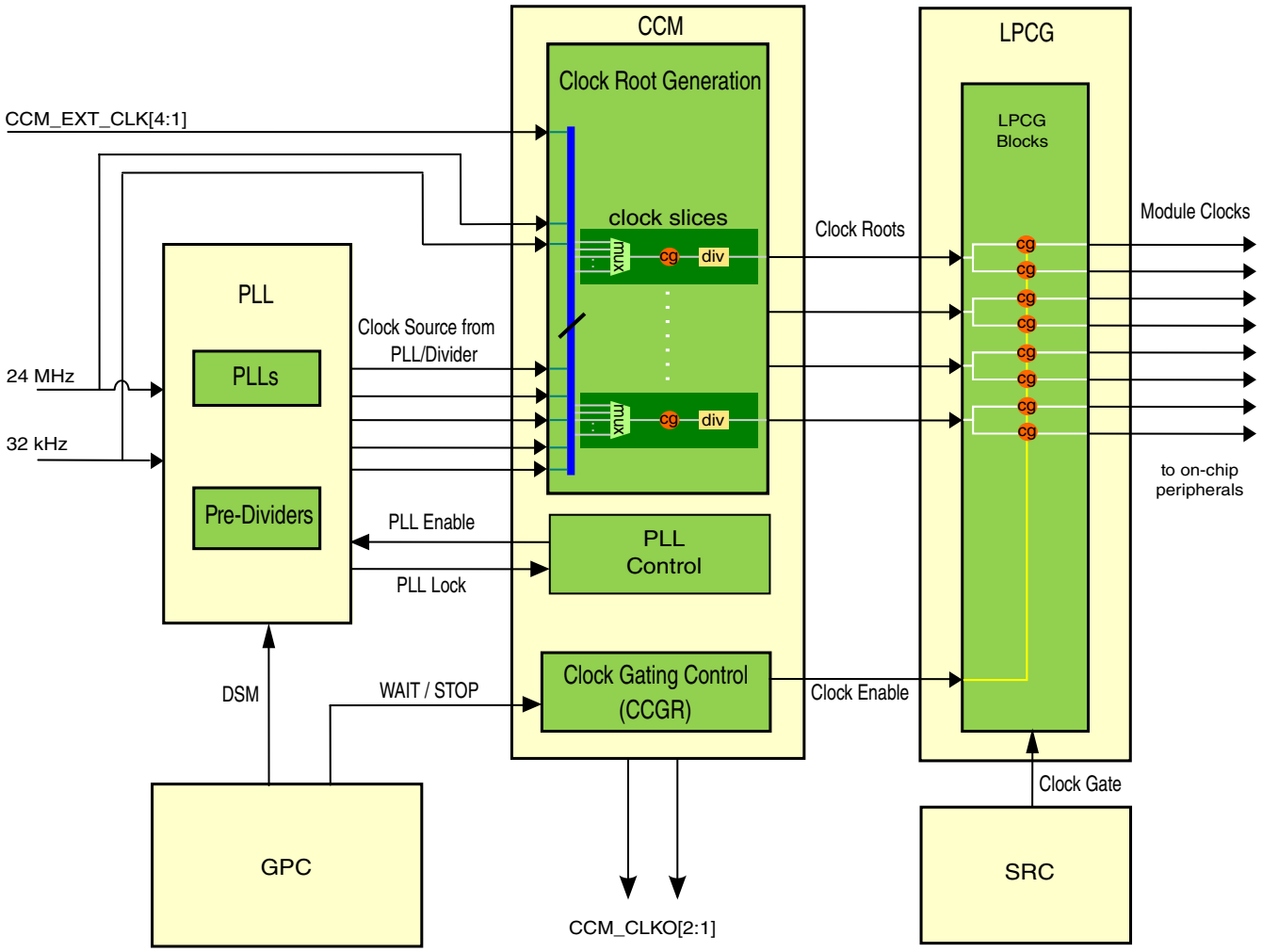


Figure 5-1. CCM Block Diagram

5.1.2 Clock Root Selects

The table below details the clock root slices and clock source inputs.

NOTE

The value of all clock root slice registers are zero after POR with the exception of DRAM clock. Please see System Boot for ROM reset values and default frequency settings for the clock root slices.

Table 5-1. Clock Root Table

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
0	ARM_A53_CLK_ROOT	0x8000	1000	000 - 24M_REF_CLK 001 - ARM_PLL_CLK 100 - SYSTEM_PLL1_CLK 101 - SYSTEM_PLL1_DIV2 011 - SYSTEM_PLL2_CLK 010 - SYSTEM_PLL2_DIV2 111 - SYSTEM_PLL3_CLK 110 - AUDIO_PLL1_CLK
1	ARM_M4_CLK_ROOT	0x8080	400	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL1_DIV3 010 - SYSTEM_PLL2_DIV4 001 - SYSTEM_PLL2_DIV5 111 - SYSTEM_PLL3_CLK 101 - AUDIO_PLL1_CLK 110 - VIDEO_PLL_CLK
2	VPU_A53_CLK_ROOT	0x8100	800	000 - 24M_REF_CLK 001 - ARM_PLL_CLK 111 - VPU_PLL_CLK 100 - SYSTEM_PLL1_CLK 101 - SYSTEM_PLL1_DIV2 011 - SYSTEM_PLL2_CLK 010 - SYSTEM_PLL2_DIV2 110 - AUDIO_PLL1_CLK
3	GPU3D_CLK_ROOT	0x8180	1000	000 - 24M_REF_CLK 001 - GPU_PLL_CLK 010 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL2_CLK 011 - SYSTEM_PLL3_CLK 101 - AUDIO_PLL1_CLK 111 - AUDIO_PLL2_CLK 110 - VIDEO_PLL_CLK
4	GPU2D_CLK_ROOT	0x8200	1000	000 - 24M_REF_CLK 001 - GPU_PLL_CLK 010 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL2_CLK

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				011 - SYSTEM_PLL3_CLK 101 - AUDIO_PLL1_CLK 111 - AUDIO_PLL2_CLK 110 - VIDEO_PLL_CLK
16	MAIN_AXI_CLK_ROOT	0x8800	400	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_CLK 111 - 100 - SYSTEM_PLL2_CLK 001 - SYSTEM_PLL2_DIV3 011 - SYSTEM_PLL2_DIV4 101 - AUDIO_PLL1_CLK 110 - VIDEO_PLL_CLK
17	ENET_AXI_CLK_ROOT	0x8880	266	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_CLK 001 - SYSTEM_PLL1_DIV3 011 - SYSTEM_PLL2_DIV4 100 - SYSTEM_PLL2_DIV5 111 - SYSTEM_PLL3_CLK 101 - AUDIO_PLL1_CLK 110 - VIDEO_PLL_CLK
18	NAND_USDHC_BUS_CLK_ROOT	0x8900	266	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_CLK 001 - SYSTEM_PLL1_DIV3 100 - SYSTEM_PLL1_DIV6 110 - SYSTEM_PLL2_DIV4 011 - SYSTEM_PLL2_DIV5 101 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL1_CLK
19	VPU_BUS_CLK_ROOT	0x8980	800	000 - 24M_REF_CLK 010 - VPU_PLL_CLK 001 - SYSTEM_PLL1_CLK 111 - 101 - SYSTEM_PLL2_CLK 110 - SYSTEM_PLL2_DIV5 100 - SYSTEM_PLL3_CLK 011 - AUDIO_PLL2_CLK
20	DISPLAY_AXI_CLK_ROOT	0x8A00	500	000 - 24M_REF_CLK

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				010 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_CLK 011 - SYSTEM_PLL3_CLK 101 - AUDIO_PLL2_CLK 110 - EXT_CLK_1 111 - EXT_CLK_4
21	DISPLAY_APB_CLK_ROOT	0x8A80	200	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_DIV8 011 - SYSTEM_PLL3_CLK 101 - AUDIO_PLL2_CLK 110 - EXT_CLK_1 111 - EXT_CLK_3
22	DISPLAY_RTRM_CLK_ROOT	0x8B00	500	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL2_CLK 010 - SYSTEM_PLL2_DIV5 100 - AUDIO_PLL1_CLK 101 - VIDEO_PLL_CLK 110 - EXT_CLK_2 111 - EXT_CLK_3
23	USB_BUS_CLK_ROOT	0x8B80	500	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_CLK 001 - SYSTEM_PLL2_DIV2 100 - SYSTEM_PLL2_DIV5 011 - SYSTEM_PLL2_DIV10 111 - AUDIO_PLL2_CLK 101 - EXT_CLK_2 110 - EXT_CLK_4
24	GPU_AXI_CLK_ROOT	0x8C00	800	000 - 24M_REF_CLK 010 - GPU_PLL_CLK 001 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL2_CLK 011 - SYSTEM_PLL3_CLK 101 - AUDIO_PLL1_CLK

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				111 - AUDIO_PLL2_CLK 110 - VIDEO_PLL_CLK
25	GPU_AHB_CLK_ROOT	0x8C80	400	000 - 24M_REF_CLK 010 - GPU_PLL_CLK 001 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL2_CLK 011 - SYSTEM_PLL3_CLK 101 - AUDIO_PLL1_CLK 111 - AUDIO_PLL2_CLK 110 - VIDEO_PLL_CLK
26	NOC_CLK_ROOT	0x8D00	800	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL2_CLK 100 - SYSTEM_PLL2_DIV2 010 - SYSTEM_PLL3_CLK 101 - AUDIO_PLL1_CLK 111 - AUDIO_PLL2_CLK 110 - VIDEO_PLL_CLK
27	NOC_APB_CLK_ROOT	0x8D80	800	000 - 24M_REF_CLK 101 - SYSTEM_PLL1_CLK 001 - SYSTEM_PLL1_DIV2 011 - SYSTEM_PLL2_DIV3 100 - SYSTEM_PLL2_DIV5 010 - SYSTEM_PLL3_CLK 110 - AUDIO_PLL1_CLK 111 - VIDEO_PLL_CLK
32	AHB_CLK_ROOT	0x9000	133	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL1_DIV2 001 - SYSTEM_PLL1_DIV6 100 - SYSTEM_PLL2_DIV8 101 - SYSTEM_PLL3_CLK 110 - AUDIO_PLL1_CLK 111 - VIDEO_PLL_CLK
34	AUDIO_AHB_CLK_ROOT	0x9100	400	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL2_CLK

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				001 - SYSTEM_PLL2_DIV2 100 - SYSTEM_PLL2_DIV6 101 - SYSTEM_PLL3_CLK 110 - AUDIO_PLL1_CLK 111 - VIDEO_PLL_CLK
36	MIPI_DSI_ESC_RX_CLK_ROOT	0x9200	133	000 - 24M_REF_CLK 011 - SYSTEM_PLL1_CLK 010 - SYSTEM_PLL1_DIV10 100 - SYSTEM_PLL2_CLK 001 - SYSTEM_PLL2_DIV10 101 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK 110 - EXT_CLK_3
48	DRAM_SEL_CFG	0x9800	800	000 - DRAM_PLL1_CLK
49	CORE_SEL_CFG	0x9880	800	000 - DRAM_PLL1_CLK
64	DRAM_ALT_CLK_ROOT	0xA000	800	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_CLK 111 - SYSTEM_PLL1_DIV3 010 - SYSTEM_PLL1_DIV8 100 - SYSTEM_PLL2_CLK 011 - SYSTEM_PLL2_DIV2 101 - SYSTEM_PLL3_CLK 110 - AUDIO_PLL1_CLK
65	DRAM_APB_CLK_ROOT	0xA080	200	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL1_DIV5 010 - SYSTEM_PLL1_DIV20 110 - SYSTEM_PLL2_DIV4 001 - SYSTEM_PLL2_DIV5 101 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK
66	VPU_G1_CLK_ROOT	0xA100	800	000 - 24M_REF_CLK 001 - VPU_PLL_CLK 010 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL1_DIV8 011 - SYSTEM_PLL2_CLK 101 - SYSTEM_PLL2_DIV8

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				110 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL1_CLK
67	VPU_G2_CLK_ROOT	0xA180	800	000 - 24M_REF_CLK 001 - VPU_PLL_CLK 010 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL1_DIV8 011 - SYSTEM_PLL2_CLK 101 - SYSTEM_PLL2_DIV8 110 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL1_CLK
68	DISPLAY_DTRC_CLK_ROOT	0xA200	600	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL1_DIV5 011 - SYSTEM_PLL2_CLK 110 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK 101 - VIDEO_PLL_CLK
69	DISPLAY_DC8000_CLK_ROOT	0xA280	600	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL1_DIV5 011 - SYSTEM_PLL2_CLK 110 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK 101 - VIDEO_PLL_CLK
70	PCIE_CTRL_CLK_ROOT	0xA300	333	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL1_DIV3 101 - SYSTEM_PLL2_DIV2 110 - SYSTEM_PLL2_DIV3 001 - SYSTEM_PLL2_DIV4 010 - SYSTEM_PLL2_DIV5 111 - SYSTEM_PLL3_CLK
71	PCIE_PHY_CLK_ROOT	0xA380	100	000 - 24M_REF_CLK 111 - SYSTEM_PLL1_DIV2 010 - SYSTEM_PLL2_DIV2 001 - SYSTEM_PLL2_DIV10 011 - EXT_CLK_1

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				100 - EXT_CLK_2 101 - EXT_CLK_3 110 - EXT_CLK_4
72	PCIE_AUX_CLK_ROOT	0xA400	10	000 - 24M_REF_CLK 111 - SYSTEM_PLL1_DIV4 110 - SYSTEM_PLL1_DIV5 101 - SYSTEM_PLL1_DIV10 001 - SYSTEM_PLL2_DIV5 100 - SYSTEM_PLL2_DIV10 010 - SYSTEM_PLL2_DIV20 011 - SYSTEM_PLL3_CLK
73	DC_PIXEL_CLK_ROOT	0xA480	594	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_CLK 101 - SYSTEM_PLL2_CLK 110 - SYSTEM_PLL3_CLK 011 - AUDIO_PLL1_CLK 010 - AUDIO_PLL2_CLK 001 - VIDEO_PLL_CLK 111 - EXT_CLK_4
74	LCDIF_PIXEL_CLK_ROOT	0xA500	250	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_CLK 101 - SYSTEM_PLL2_CLK 110 - SYSTEM_PLL3_CLK 011 - AUDIO_PLL1_CLK 010 - AUDIO_PLL2_CLK 001 - VIDEO_PLL_CLK 111 - EXT_CLK_4
75	SAI1_CLK_ROOT	0xA580	66	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_DIV6 001 - AUDIO_PLL1_CLK 010 - AUDIO_PLL2_CLK 011 - VIDEO_PLL_CLK 110 - EXT_CLK_1 111 - EXT_CLK_2
76	SAI2_CLK_ROOT	0xA600	66	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_DIV6 001 - AUDIO_PLL1_CLK

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				010 - AUDIO_PLL2_CLK 011 - VIDEO_PLL_CLK 110 - EXT_CLK_2 111 - EXT_CLK_3
77	SAI3_CLK_ROOT	0xA680	66	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_DIV6 001 - AUDIO_PLL1_CLK 010 - AUDIO_PLL2_CLK 011 - VIDEO_PLL_CLK 110 - EXT_CLK_3 111 - EXT_CLK_4
79	SAI5_CLK_ROOT	0xA780	66	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_DIV6 001 - AUDIO_PLL1_CLK 010 - AUDIO_PLL2_CLK 011 - VIDEO_PLL_CLK 110 - EXT_CLK_2 111 - EXT_CLK_3
80	SAI6_CLK_ROOT	0xA800	66	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_DIV6 001 - AUDIO_PLL1_CLK 010 - AUDIO_PLL2_CLK 011 - VIDEO_PLL_CLK 110 - EXT_CLK_3 111 - EXT_CLK_4
81	SPDIF1_CLK_ROOT	0xA880	66	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_DIV6 001 - AUDIO_PLL1_CLK 010 - AUDIO_PLL2_CLK 011 - VIDEO_PLL_CLK 110 - EXT_CLK_2 111 - EXT_CLK_3
82	SPDIF2_CLK_ROOT	0xA900	66	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_DIV6 001 - AUDIO_PLL1_CLK 010 - AUDIO_PLL2_CLK 011 - VIDEO_PLL_CLK

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				110 - EXT_CLK_3 111 - EXT_CLK_4
83	ENET_REF_CLK_ROOT	0xA980	125	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_DIV5 001 - SYSTEM_PLL2_DIV8 011 - SYSTEM_PLL2_DIV10 010 - SYSTEM_PLL2_DIV20 101 - AUDIO_PLL1_CLK 110 - VIDEO_PLL_CLK 111 - EXT_CLK_4
84	ENET_TIMER_CLK_ROOT	0xAA00	125	000 - 24M_REF_CLK 001 - SYSTEM_PLL2_DIV10 010 - AUDIO_PLL1_CLK 111 - VIDEO_PLL_CLK 011 - EXT_CLK_1 100 - EXT_CLK_2 101 - EXT_CLK_3 110 - EXT_CLK_4
85	ENET_PHY_REF_CLK_ROOT	0xAA80	125	000 - 24M_REF_CLK 100 - SYSTEM_PLL2_DIV2 011 - SYSTEM_PLL2_DIV5 010 - SYSTEM_PLL2_DIV8 001 - SYSTEM_PLL2_DIV20 101 - AUDIO_PLL1_CLK 111 - AUDIO_PLL2_CLK 110 - VIDEO_PLL_CLK
86	NAND_CLK_ROOT	0xAB00	500	000 - 24M_REF_CLK 011 - SYSTEM_PLL1_DIV2 001 - SYSTEM_PLL2_DIV2 110 - SYSTEM_PLL2_DIV4 101 - SYSTEM_PLL3_CLK 010 - AUDIO_PLL1_CLK 100 - AUDIO_PLL2_CLK 111 - VIDEO_PLL_CLK
87	QSPI_CLK_ROOT	0xAB80	400	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_DIV2 101 - SYSTEM_PLL1_DIV3

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				111 - SYSTEM_PLL1_DIV8 011 - SYSTEM_PLL2_DIV2 010 - SYSTEM_PLL2_DIV3 110 - SYSTEM_PLL3_CLK 100 - AUDIO_PLL2_CLK
88	USDHC1_CLK_ROOT	0xAC00	400	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_CLK 001 - SYSTEM_PLL1_DIV2 101 - SYSTEM_PLL1_DIV3 111 - SYSTEM_PLL1_DIV8 011 - SYSTEM_PLL2_DIV2 100 - SYSTEM_PLL3_CLK 110 - AUDIO_PLL2_CLK
89	USDHC2_CLK_ROOT	0xAC80	400	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_CLK 001 - SYSTEM_PLL1_DIV2 101 - SYSTEM_PLL1_DIV3 111 - SYSTEM_PLL1_DIV8 011 - SYSTEM_PLL2_DIV2 100 - SYSTEM_PLL3_CLK 110 - AUDIO_PLL2_CLK
90	I2C1_CLK_ROOT	0xAD00	66	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_DIV5 111 - SYSTEM_PLL1_DIV6 010 - SYSTEM_PLL2_DIV20 011 - SYSTEM_PLL3_CLK 100 - AUDIO_PLL1_CLK 110 - AUDIO_PLL2_CLK 101 - VIDEO_PLL_CLK
91	I2C2_CLK_ROOT	0xAD80	66	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_DIV5 111 - SYSTEM_PLL1_DIV6 010 - SYSTEM_PLL2_DIV20 011 - SYSTEM_PLL3_CLK 100 - AUDIO_PLL1_CLK 110 - AUDIO_PLL2_CLK 101 - VIDEO_PLL_CLK

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
92	I2C3_CLK_ROOT	0xAE00	66	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_DIV5 111 - SYSTEM_PLL1_DIV6 010 - SYSTEM_PLL2_DIV20 011 - SYSTEM_PLL3_CLK 100 - AUDIO_PLL1_CLK 110 - AUDIO_PLL2_CLK 101 - VIDEO_PLL_CLK
93	I2C4_CLK_ROOT	0xAE80	66	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_DIV5 111 - SYSTEM_PLL1_DIV6 010 - SYSTEM_PLL2_DIV20 011 - SYSTEM_PLL3_CLK 100 - AUDIO_PLL1_CLK 110 - AUDIO_PLL2_CLK 101 - VIDEO_PLL_CLK
94	UART1_CLK_ROOT	0xAF00	80	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_DIV10 010 - SYSTEM_PLL2_DIV5 011 - SYSTEM_PLL2_DIV10 100 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK 101 - EXT_CLK_2 110 - EXT_CLK_4
95	UART2_CLK_ROOT	0xAF80	80	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_DIV10 010 - SYSTEM_PLL2_DIV5 011 - SYSTEM_PLL2_DIV10 100 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK 101 - EXT_CLK_2 110 - EXT_CLK_3
96	UART3_CLK_ROOT	0xB000	80	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_DIV10 010 - SYSTEM_PLL2_DIV5 011 - SYSTEM_PLL2_DIV10

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				100 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK 101 - EXT_CLK_2 110 - EXT_CLK_4
97	UART4_CLK_ROOT	0xB080	80	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_DIV10 010 - SYSTEM_PLL2_DIV5 011 - SYSTEM_PLL2_DIV10 100 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK 101 - EXT_CLK_2 110 - EXT_CLK_3
98	USB_CORE_REF_CLK_ROOT	0xB100	125	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_DIV8 010 - SYSTEM_PLL1_DIV20 100 - SYSTEM_PLL2_DIV5 011 - SYSTEM_PLL2_DIV10 111 - AUDIO_PLL2_CLK 101 - EXT_CLK_2 110 - EXT_CLK_3
99	USB_PHY_REF_CLK_ROOT	0xB180	100	000 - 24M_REF_CLK 001 - SYSTEM_PLL1_DIV8 010 - SYSTEM_PLL1_DIV20 100 - SYSTEM_PLL2_DIV5 011 - SYSTEM_PLL2_DIV10 111 - AUDIO_PLL2_CLK 101 - EXT_CLK_2 110 - EXT_CLK_3
100	GIC_CLK_ROOT	0xB200	400	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_CLK 010 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_DIV5 011 - SYSTEM_PLL2_DIV10 111 - AUDIO_PLL2_CLK 101 - EXT_CLK_2 110 - EXT_CLK_4
101	ECSPI1_CLK_ROOT	0xB280	80	000 - 24M_REF_CLK

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				100 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL1_DIV5 010 - SYSTEM_PLL1_DIV20 110 - SYSTEM_PLL2_DIV4 001 - SYSTEM_PLL2_DIV5 101 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK
102	ECSPI2_CLK_ROOT	0xB300	80	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL1_DIV5 010 - SYSTEM_PLL1_DIV20 110 - SYSTEM_PLL2_DIV4 001 - SYSTEM_PLL2_DIV5 101 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK
103	PWM1_CLK_ROOT	0xB380	66	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_DIV5 110 - SYSTEM_PLL1_DIV10 011 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_DIV10 100 - SYSTEM_PLL3_CLK 111 - VIDEO_PLL_CLK 101 - EXT_CLK_1
104	PWM2_CLK_ROOT	0xB400	66	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_DIV5 110 - SYSTEM_PLL1_DIV10 011 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_DIV10 100 - SYSTEM_PLL3_CLK 111 - VIDEO_PLL_CLK 101 - EXT_CLK_1
105	PWM3_CLK_ROOT	0xB480	66	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_DIV5 110 - SYSTEM_PLL1_DIV10 011 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_DIV10 100 - SYSTEM_PLL3_CLK

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				111 - VIDEO_PLL_CLK 101 - EXT_CLK_2
106	PWM4_CLK_ROOT	0xB500	66	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_DIV5 110 - SYSTEM_PLL1_DIV10 011 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_DIV10 100 - SYSTEM_PLL3_CLK 111 - VIDEO_PLL_CLK 101 - EXT_CLK_2
107	GPT1_CLK_ROOT	0xB580	100	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_DIV2 101 - SYSTEM_PLL1_DIV10 011 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_DIV10 110 - AUDIO_PLL1_CLK 100 - VIDEO_PLL_CLK 111 - EXT_CLK_1
108	GPT2_CLK_ROOT	0xB600	100	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_DIV2 101 - SYSTEM_PLL1_DIV10 011 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_DIV10 110 - AUDIO_PLL1_CLK 100 - VIDEO_PLL_CLK 111 - EXT_CLK_2
109	GPT3_CLK_ROOT	0xB680	100	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_DIV2 101 - SYSTEM_PLL1_DIV10 011 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_DIV10 110 - AUDIO_PLL1_CLK 100 - VIDEO_PLL_CLK 111 - EXT_CLK_3
110	GPT4_CLK_ROOT	0xB700	100	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_DIV2 101 - SYSTEM_PLL1_DIV10

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				011 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_DIV10 110 - AUDIO_PLL1_CLK 100 - VIDEO_PLL_CLK 111 - EXT_CLK_1
111	GPT5_CLK_ROOT	0xB780	100	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_DIV2 101 - SYSTEM_PLL1_DIV10 011 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_DIV10 110 - AUDIO_PLL1_CLK 100 - VIDEO_PLL_CLK 111 - EXT_CLK_2
112	GPT6_CLK_ROOT	0xB800	100	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_DIV2 101 - SYSTEM_PLL1_DIV10 011 - SYSTEM_PLL1_DIV20 001 - SYSTEM_PLL2_DIV10 110 - AUDIO_PLL1_CLK 100 - VIDEO_PLL_CLK 111 - EXT_CLK_3
113	TRACE_CLK_ROOT	0xB880	133	000 - 24M_REF_CLK 011 - VPU_PLL_CLK 010 - SYSTEM_PLL1_DIV5 001 - SYSTEM_PLL1_DIV6 100 - SYSTEM_PLL2_DIV8 101 - SYSTEM_PLL3_CLK 110 - EXT_CLK_1 111 - EXT_CLK_3
114	WDOG_CLK_ROOT	0xB900	66	000 - 24M_REF_CLK 011 - VPU_PLL_CLK 010 - SYSTEM_PLL1_DIV5 001 - SYSTEM_PLL1_DIV6 110 - SYSTEM_PLL1_DIV10 111 - SYSTEM_PLL2_DIV6 100 - SYSTEM_PLL2_DIV8 101 - SYSTEM_PLL3_CLK

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
115	WRCLK_CLK_ROOT	0xB980	40	000 - 24M_REF_CLK 010 - VPU_PLL_CLK 101 - SYSTEM_PLL1_DIV3 111 - SYSTEM_PLL1_DIV8 001 - SYSTEM_PLL1_DIV20 110 - SYSTEM_PLL2_DIV2 100 - SYSTEM_PLL2_DIV5 011 - SYSTEM_PLL3_CLK
116	IPP_DO_CLKO1	0xBA00	266	000 - 24M_REF_CLK 110 - VPU_PLL_CLK 001 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL1_DIV4 111 - SYSTEM_PLL1_DIV10 101 - SYSTEM_PLL2_DIV2 100 - AUDIO_PLL2_CLK
117	IPP_DO_CLKO2	0xBA80	266	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_DIV2 001 - SYSTEM_PLL2_DIV5 011 - SYSTEM_PLL2_DIV6 100 - SYSTEM_PLL3_CLK 101 - AUDIO_PLL1_CLK 110 - VIDEO_PLL_CLK 111 - 32K_REF_CLK
118	MIPI_DSI_CORE_CLK_ROOT	0xBB00	266	000 - 24M_REF_CLK 011 - SYSTEM_PLL1_CLK 001 - SYSTEM_PLL1_DIV3 100 - SYSTEM_PLL2_CLK 010 - SYSTEM_PLL2_DIV4 101 - SYSTEM_PLL3_CLK 110 - AUDIO_PLL2_CLK 111 - VIDEO_PLL_CLK
119	MIPI_DSI_PHY_REF_CLK_ROOT	0xBB80	125	000 - 24M_REF_CLK 011 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL2_CLK 001 - SYSTEM_PLL2_DIV8 010 - SYSTEM_PLL2_DIV10

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				110 - AUDIO_PLL2_CLK 111 - VIDEO_PLL_CLK 101 - EXT_CLK_2
120	MIPI_DSI_DBI_CLK_ROOT	0xBC00	266	000 - 24M_REF_CLK 011 - SYSTEM_PLL1_CLK 001 - SYSTEM_PLL1_DIV3 100 - SYSTEM_PLL2_CLK 010 - SYSTEM_PLL2_DIV10 101 - SYSTEM_PLL3_CLK 110 - AUDIO_PLL2_CLK 111 - VIDEO_PLL_CLK
121	USDHC3_CLK_ROOT	0xBC80	400	000 - 24M_REF_CLK 010 - SYSTEM_PLL1_CLK 001 - SYSTEM_PLL1_DIV2 101 - SYSTEM_PLL1_DIV3 111 - SYSTEM_PLL1_DIV8 011 - SYSTEM_PLL2_DIV2 100 - SYSTEM_PLL3_CLK 110 - AUDIO_PLL2_CLK
122	MIPI_CSI1_CORE_CLK_ROOT	0xBD00	333	000 - 24M_REF_CLK 011 - SYSTEM_PLL1_CLK 001 - SYSTEM_PLL1_DIV3 100 - SYSTEM_PLL2_CLK 010 - SYSTEM_PLL2_DIV4 101 - SYSTEM_PLL3_CLK 110 - AUDIO_PLL2_CLK 111 - VIDEO_PLL_CLK
123	MIPI_CSI1_PHY_REF_CLK_ROOT	0xBD80	125	000 - 24M_REF_CLK 011 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL2_CLK 001 - SYSTEM_PLL2_DIV3 010 - SYSTEM_PLL2_DIV10 110 - AUDIO_PLL2_CLK 111 - VIDEO_PLL_CLK 101 - EXT_CLK_2
124	MIPI_CSI1_ESC_CLK_ROOT	0xBE00	133	000 - 24M_REF_CLK 011 - SYSTEM_PLL1_CLK

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				010 - SYSTEM_PLL1_DIV10 100 - SYSTEM_PLL2_CLK 001 - SYSTEM_PLL2_DIV10 101 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK 110 - EXT_CLK_3
125	MIPI_CSI2_CORE_CLK_ROOT	0xBE80	266	000 - 24M_REF_CLK 011 - SYSTEM_PLL1_CLK 001 - SYSTEM_PLL1_DIV3 100 - SYSTEM_PLL2_CLK 010 - SYSTEM_PLL2_DIV4 101 - SYSTEM_PLL3_CLK 110 - AUDIO_PLL2_CLK 111 - VIDEO_PLL_CLK
126	MIPI_CSI2_PHY_REF_CLK_ROOT	0xBF00	125	000 - 24M_REF_CLK 011 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL2_CLK 001 - SYSTEM_PLL2_DIV3 010 - SYSTEM_PLL2_DIV10 110 - AUDIO_PLL2_CLK 111 - VIDEO_PLL_CLK 101 - EXT_CLK_2
127	MIPI_CSI2_ESC_CLK_ROOT	0xBF80	133	000 - 24M_REF_CLK 011 - SYSTEM_PLL1_CLK 010 - SYSTEM_PLL1_DIV10 100 - SYSTEM_PLL2_CLK 001 - SYSTEM_PLL2_DIV10 101 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK 110 - EXT_CLK_3
128	PCIE2_CTRL_CLK_ROOT	0xC000	500	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL1_DIV3 101 - SYSTEM_PLL2_DIV2 110 - SYSTEM_PLL2_DIV3 001 - SYSTEM_PLL2_DIV4 010 - SYSTEM_PLL2_DIV5

Table continues on the next page...

Table 5-1. Clock Root Table (continued)

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				111 - SYSTEM_PLL3_CLK
129	PCIE2_PHY_CLK_ROOT	0xC080	100	000 - 24M_REF_CLK 111 - SYSTEM_PLL1_DIV2 010 - SYSTEM_PLL2_DIV2 001 - SYSTEM_PLL2_DIV10 011 - EXT_CLK_1 100 - EXT_CLK_2 101 - EXT_CLK_3 110 - EXT_CLK_4
130	PCIE2_AUX_CLK_ROOT	0xC100	10	000 - 24M_REF_CLK 111 - SYSTEM_PLL1_DIV4 110 - SYSTEM_PLL1_DIV5 101 - SYSTEM_PLL1_DIV10 001 - SYSTEM_PLL2_DIV5 100 - SYSTEM_PLL2_DIV10 010 - SYSTEM_PLL2_DIV20 011 - SYSTEM_PLL3_CLK
131	ECSPI3_CLK_ROOT	0xC180	80	000 - 24M_REF_CLK 100 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL1_DIV5 010 - SYSTEM_PLL1_DIV20 110 - SYSTEM_PLL2_DIV4 001 - SYSTEM_PLL2_DIV5 101 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL2_CLK
132	PDM_CLK_ROOT	0xC200	200	000 - 24M_REF_CLK 011 - SYSTEM_PLL1_CLK 100 - SYSTEM_PLL2_CLK 001 - SYSTEM_PLL2_DIV10 101 - SYSTEM_PLL3_CLK 010 - AUDIO_PLL1_CLK 111 - AUDIO_PLL2_CLK 110 - EXT_CLK_3
133	VPU_H1_CLK_ROOT	0xC280	800	000 - 24M_REF_CLK 001 - VPU_PLL_CLK 010 - SYSTEM_PLL1_CLK 011 - SYSTEM_PLL2_CLK

Table 5-1. Clock Root Table

Slice	Clock Root	Offset	Max Freq (MHz)	Source Select (CCM_TARGET_ROOT[MUX])
				101 - SYSTEM_PLL2_DIV8 110 - SYSTEM_PLL3_CLK 111 - AUDIO_PLL1_CLK 100 - AUDIO_PLL2_CLK

5.1.3 Clock Tree

The figure below illustrates the clock sources from the PLLs.

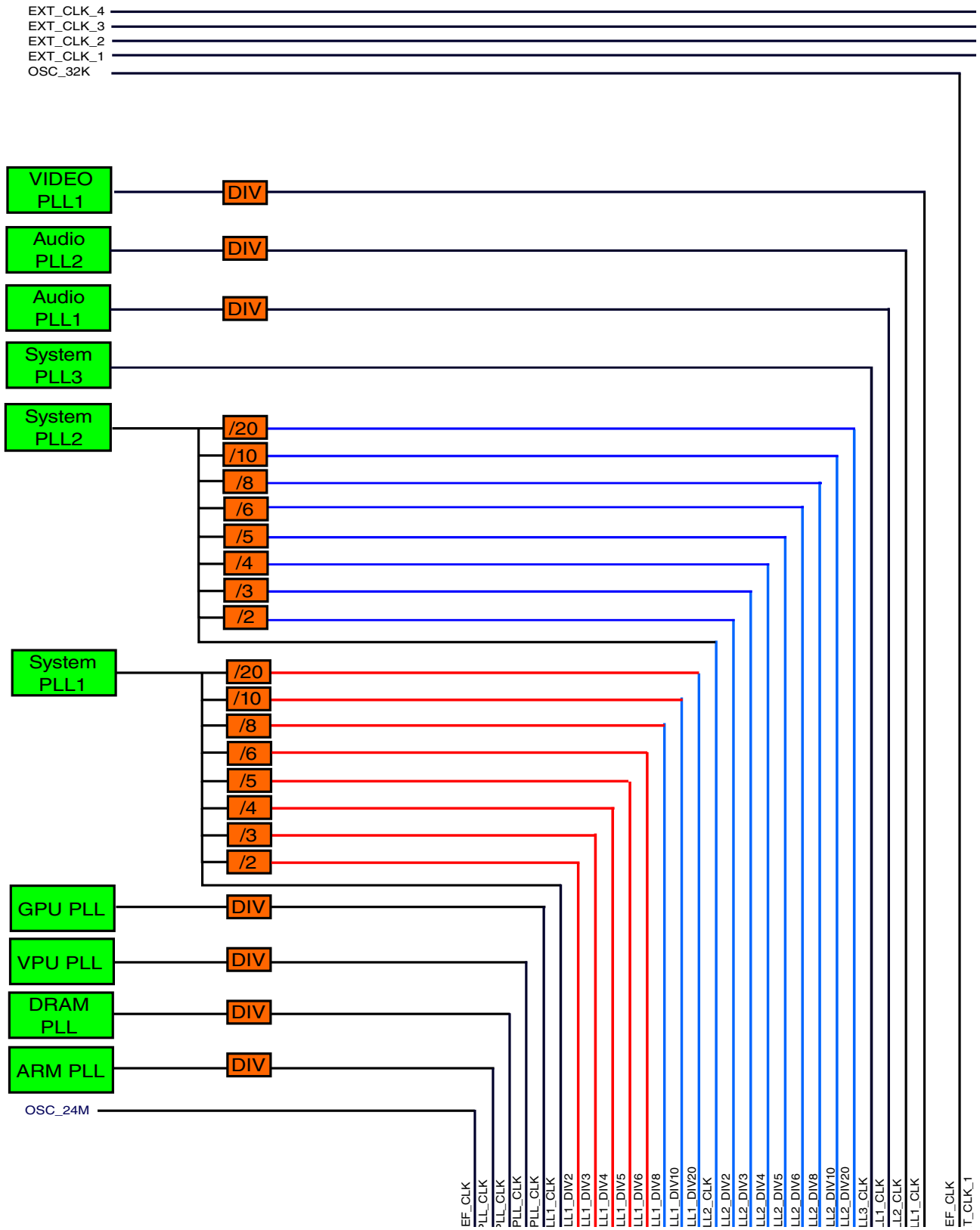
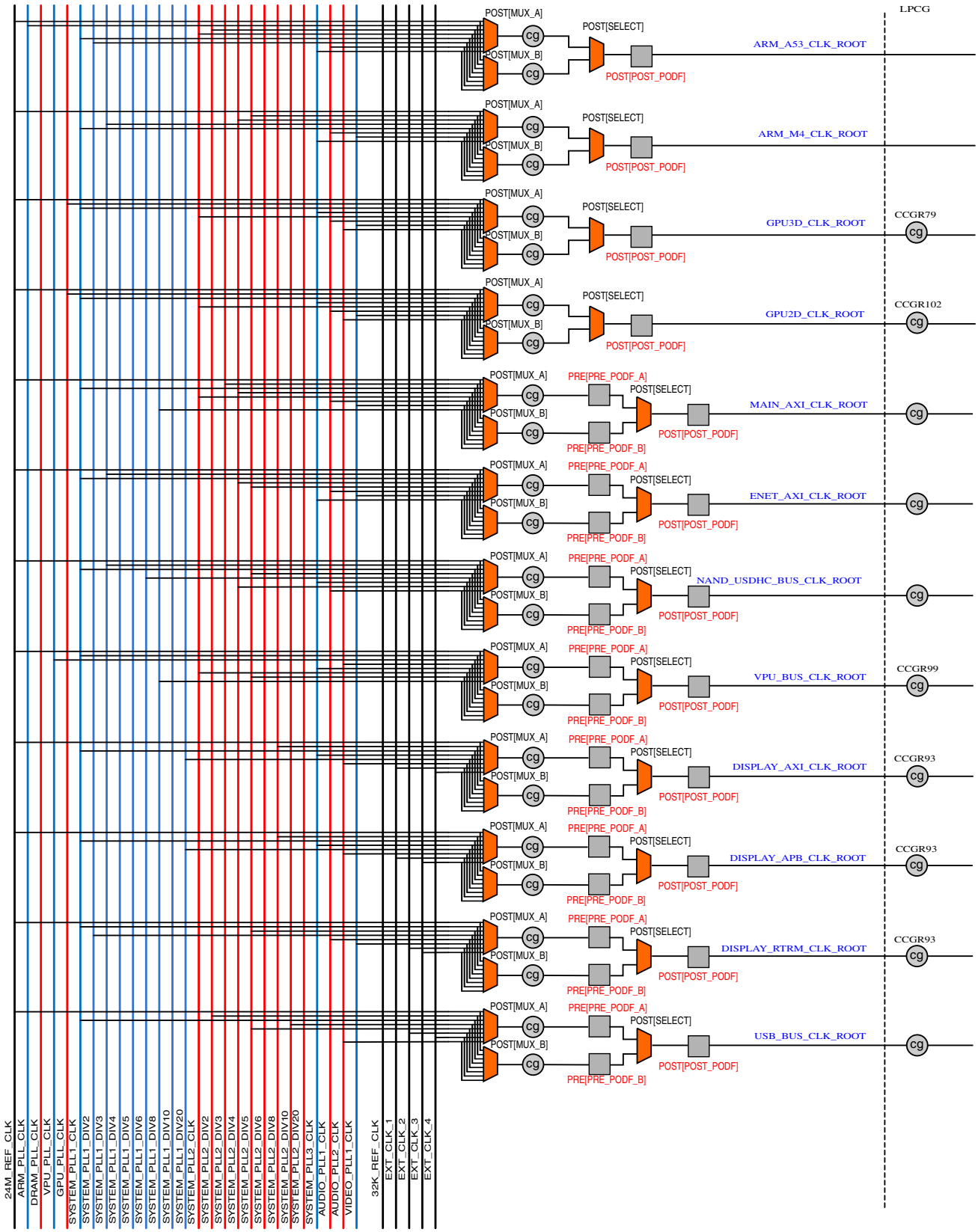
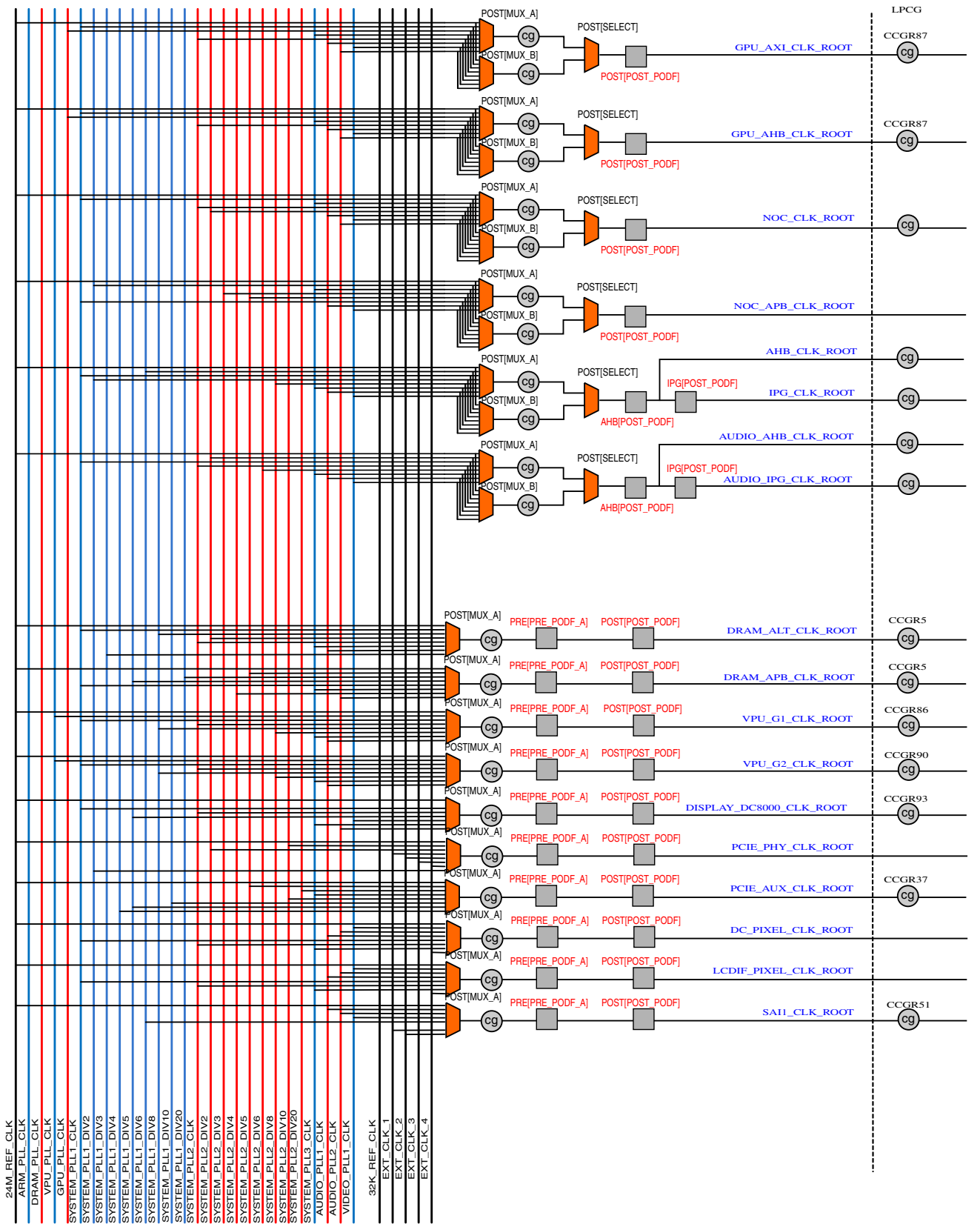
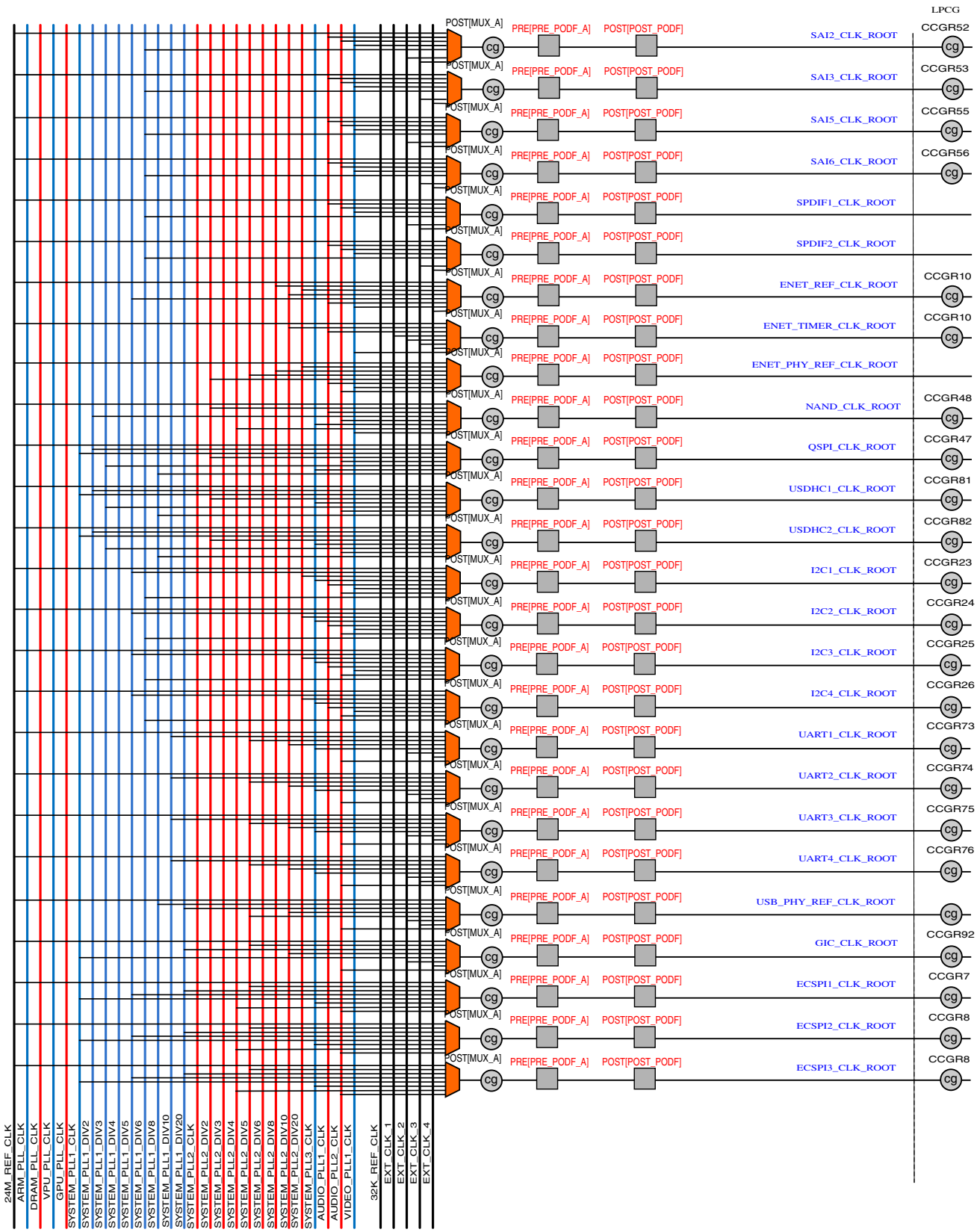


Figure 5-2. CCM input clock sources

The figure below illustrates the clock slices of CCM and clock root generation.


LPCG
ARM_A53_CLK_ROOT
ARM_M4_CLK_ROOT
GPU3D_CLK_ROOT
CCGR79
GPU2D_CLK_ROOT
CCGR102
MAIN_AXI_CLK_ROOT
ENET_AXI_CLK_ROOT
NAND_USDHC_BUS_CLK_ROOT
VPU_BUS_CLK_ROOT
CCGR99
DISPLAY_AXI_CLK_ROOT
CCGR93
DISPLAY_APB_CLK_ROOT
CCGR93
DISPLAY_RTRM_CLK_ROOT
CCGR93
USB_BUS_CLK_ROOT
POST[MUX_A]
POST[MUX_B]
POST[SELECT]
POST[POST_PODF]
PRE[PRE_PODF_A]
PRE[PRE_PODF_B]
cg
24M_REF_CLK
ARM_PLL_CLK
DRAM_PLL_CLK
VPU_PLL_CLK
GPU_PLL_CLK
SYSTEM_PLL1_CLK
SYSTEM_PLL1_DIV2
SYSTEM_PLL1_DIV3
SYSTEM_PLL1_DIV4
SYSTEM_PLL1_DIV5
SYSTEM_PLL1_DIV6
SYSTEM_PLL1_DIV8
SYSTEM_PLL1_DIV10
SYSTEM_PLL1_DIV20
SYSTEM_PLL2_CLK
SYSTEM_PLL2_DIV2
SYSTEM_PLL2_DIV3
SYSTEM_PLL2_DIV4
SYSTEM_PLL2_DIV5
SYSTEM_PLL2_DIV6
SYSTEM_PLL2_DIV8
SYSTEM_PLL2_DIV10
SYSTEM_PLL2_DIV20
SYSTEM_PLL3_CLK
AUDIO_PLL1_CLK
AUDIO_PLL2_CLK
VIDEO_PLL1_CLK
32K_REF_CLK
EXT_CLK_1
EXT_CLK_2
EXT_CLK_3
EXT_CLK_4


LPCG
POST[MUX_A]
POST[MUX_B]
POST[SELECT]
POST[POST_PODF]
GPU_AXI_CLK_ROOT
CCGR87
GPU_AHB_CLK_ROOT
CCGR87
NOC_CLK_ROOT
NOC_APB_CLK_ROOT
AHB_CLK_ROOT
IPG[POST_PODF]
IPG_CLK_ROOT
AHB[POST_PODF]
AUDIO_AHB_CLK_ROOT
AUDIO_IPG_CLK_ROOT
PRE[PRE_PODF_A]
DRAM_ALT_CLK_ROOT
CCGR5
DRAM_APB_CLK_ROOT
CCGR5
VPU_G1_CLK_ROOT
CCGR86
VPU_G2_CLK_ROOT
CCGR90
DISPLAY_DC8000_CLK_ROOT
CCGR93
PCIE_PHY_CLK_ROOT
PCIE_AUX_CLK_ROOT
CCGR37
DC_PIXEL_CLK_ROOT
LCDIF_PIXEL_CLK_ROOT
SAI1_CLK_ROOT
CCGR51
24M_REF_CLK
ARM_PLL_CLK
DRAM_PLL_CLK
VPU_PLL_CLK
GPU_PLL_CLK
SYSTEM_PLL1_CLK
SYSTEM_PLL1_DIV2
SYSTEM_PLL1_DIV3
SYSTEM_PLL1_DIV4
SYSTEM_PLL1_DIV5
SYSTEM_PLL1_DIV6
SYSTEM_PLL1_DIV8
SYSTEM_PLL1_DIV10
SYSTEM_PLL1_DIV20
SYSTEM_PLL2_CLK
SYSTEM_PLL2_DIV2
SYSTEM_PLL2_DIV3
SYSTEM_PLL2_DIV4
SYSTEM_PLL2_DIV5
SYSTEM_PLL2_DIV6
SYSTEM_PLL2_DIV8
SYSTEM_PLL2_DIV10
SYSTEM_PLL2_DIV20
SYSTEM_PLL3_CLK
AUDIO_PLL1_CLK
AUDIO_PLL2_CLK
VIDEO_PLL1_CLK
32K_REF_CLK
EXT_CLK_1
EXT_CLK_2
EXT_CLK_3
EXT_CLK_4


LPCG
POST[MUX_A]
PRE[PRE_PODF_A]
POST[POST_PODF]
cg
SAI2_CLK_ROOT
CCGR52
SAI3_CLK_ROOT
CCGR53
SAI5_CLK_ROOT
CCGR55
SAI6_CLK_ROOT
CCGR56
SPDIF1_CLK_ROOT
SPDIF2_CLK_ROOT
ENET_REF_CLK_ROOT
CCGR10
ENET_TIMER_CLK_ROOT
CCGR10
ENET_PHY_REF_CLK_ROOT
NAND_CLK_ROOT
CCGR48
QSPI_CLK_ROOT
CCGR47
USDHC1_CLK_ROOT
CCGR81
USDHC2_CLK_ROOT
CCGR82
I2C1_CLK_ROOT
CCGR23
I2C2_CLK_ROOT
CCGR24
I2C3_CLK_ROOT
CCGR25
I2C4_CLK_ROOT
CCGR26
UART1_CLK_ROOT
CCGR73
UART2_CLK_ROOT
CCGR74
UART3_CLK_ROOT
CCGR75
UART4_CLK_ROOT
CCGR76
USB_PHY_REF_CLK_ROOT
GIC_CLK_ROOT
CCGR92
ECSPI1_CLK_ROOT
CCGR7
ECSPI2_CLK_ROOT
CCGR8
ECSPI3_CLK_ROOT
CCGR8
24M_REF_CLK
ARM_PLL_CLK
DRAM_PLL_CLK
VPU_PLL_CLK
GPU_PLL_CLK
SYSTEM_PLL1_CLK
SYSTEM_PLL1_DIV2
SYSTEM_PLL1_DIV3
SYSTEM_PLL1_DIV4
SYSTEM_PLL1_DIV5
SYSTEM_PLL1_DIV6
SYSTEM_PLL1_DIV8
SYSTEM_PLL1_DIV10
SYSTEM_PLL1_DIV20
SYSTEM_PLL2_CLK
SYSTEM_PLL2_DIV2
SYSTEM_PLL2_DIV3
SYSTEM_PLL2_DIV4
SYSTEM_PLL2_DIV5
SYSTEM_PLL2_DIV6
SYSTEM_PLL2_DIV8
SYSTEM_PLL2_DIV10
SYSTEM_PLL2_DIV20
SYSTEM_PLL3_CLK
AUDIO_PLL1_CLK
AUDIO_PLL2_CLK
VIDEO_PLL1_CLK
32K_REF_CLK
EXT_CLK_1
EXT_CLK_2
EXT_CLK_3
EXT_CLK_4

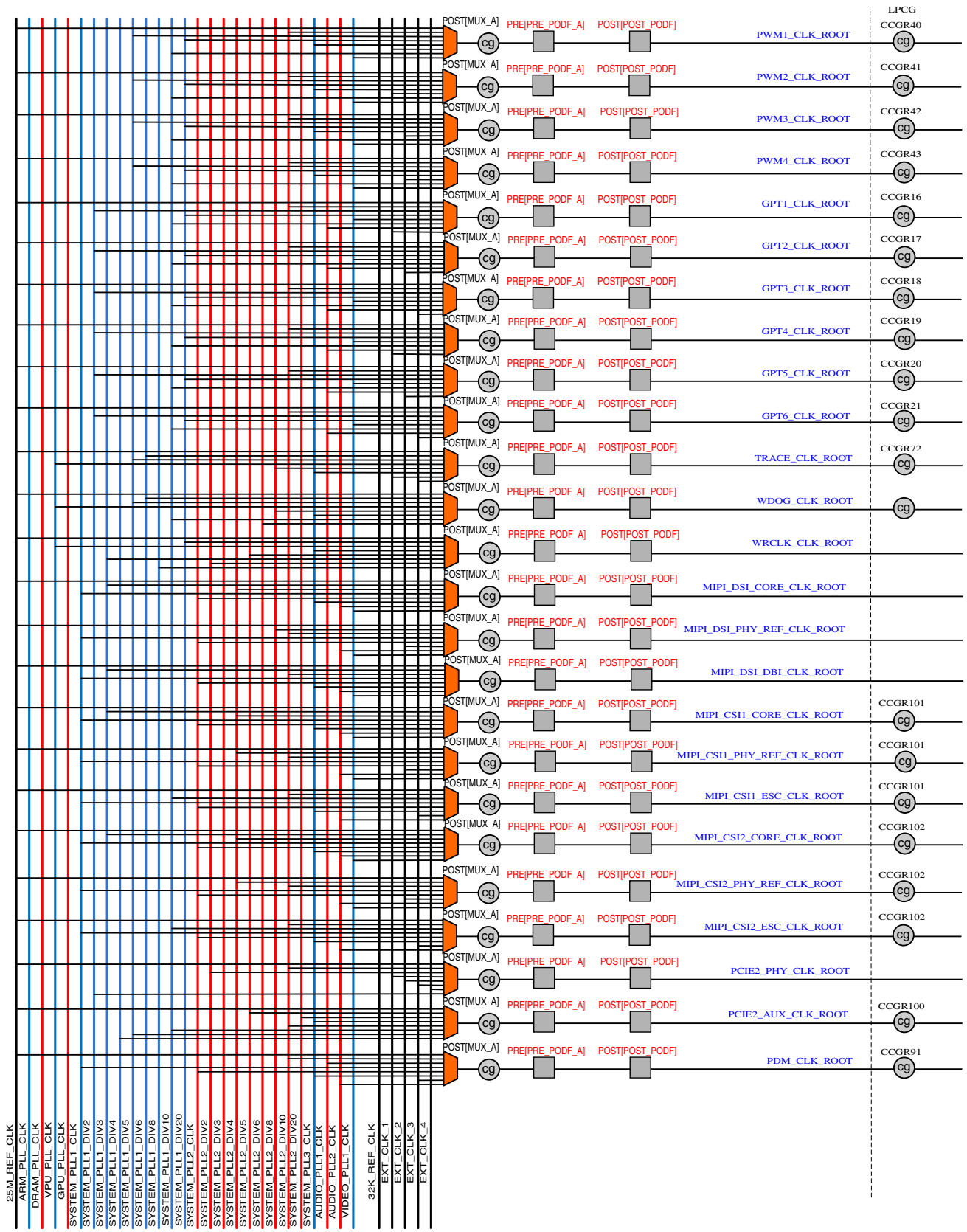


Figure 5-3. CCM Clock Tree Root Slices

5.1.4 System Clocks

The table below shows the CCM output module clock connectivity and gating.

Table 5-2. System Clocks and Gating

Module	Module Clock (instance.clock)	Clock Root	Module Clock Gating Enable (CCGR)
AIPS_TZ_V2	aips_tz1.hclk	AHB_CLK_ROOT	clk_enable_ipmux1 (CCGR28)
	ipmux1. master_clk	AHB_CLK_ROOT	clk_enable_ipmux1 (CCGR28)
	ipmux1. slave_clk	IPG_CLK_ROOT	clk_enable_ipmux1 (CCGR28)
	aips_tz2.hclk	AHB_CLK_ROOT	clk_enable_ipmux2 (CCGR29)
	Ipmux2. master_clk	AHB_CLK_ROOT	clk_enable_ipmux2 (CCGR29)
	Ipmux2. slave_clk	IPG_CLK_ROOT	clk_enable_ipmux2 (CCGR29)
	aips_tz3.hclk	AHB_CLK_ROOT	clk_enable_ipmux3 (CCGR30)
	Ipmux3. master_clk	AHB_CLK_ROOT	clk_enable_ipmux3 (CCGR30)
	Ipmux3. slave_clk	IPG_CLK_ROOT	clk_enable_ipmux3 (CCGR30)
APBHDMA	apbhdma.hclk	NAND_USDHC_BUS_CLK_ROOT	clk_enable_rawnand (CCGR48)
	apbhdma_sec.mst_hclk	NAND_USDHC_BUS_CLK_ROOT	clk_enable_rawnand (CCGR48)
BCH	rawnand.u_bch_input_apb_clk	NAND_USDHC_BUS_CLK_ROOT	clk_enable_rawnand (CCGR48)
	rawnand.u_gpmi_bch_input_bch_clk	NAND_USDHC_BUS_CLK_ROOT	clk_enable_rawnand (CCGR48)
DAP	dap.dapclk_2_2	AHB_CLK_ROOT	clk_enable_debug (CCGR4)
CORESIGHT	coresight.DBGCLK	MAIN_AXI_CLK_ROOT	clk_enable_trace (CCGR72)
	coresight.traceclkin	TRACE_CLK_ROOT	clk_enable_trace (CCGR72)
	coresight_mem.cs_etf_clk	MAIN_AXI_CLK_ROOT	clk_enable_trace (CCGR72)
eCSPI	ecspi1. ipg_clk	IPG_CLK_ROOT	clk_enable_ecspi1 (CCGR7)
	ecspi1. ipg_clk_per	ECSPI1_CLK_ROOT	clk_enable_ecspi1 (CCGR7)
	ecspi1. ipg_clk_s	IPG_CLK_ROOT	clk_enable_ecspi1 (CCGR7)
	ecspi2. ipg_clk	IPG_CLK_ROOT	clk_enable_ecspi2 (CCGR8)
	ecspi2. ipg_clk_per	ECSPI2_CLK_ROOT	clk_enable_ecspi2 (CCGR8)
	ecspi2. ipg_clk_s	IPG_CLK_ROOT	clk_enable_ecspi2 (CCGR8)
	ecspi3. ipg_clk	IPG_CLK_ROOT	clk_enable_ecspi3 (CCGR9)

Table continues on the next page...

Table 5-2. System Clocks and Gating (continued)

Module	Module Clock (instance.clock)	Clock Root	Module Clock Gating Enable (CCGR)
	ecspi3. ipg_clk_per	ECSPI3_CLK_ROOT	clk_enable_ecspi3 (CCGR9)
	ecspi3. ipg_clk_s	IPG_CLK_ROOT	clk_enable_ecspi3 (CCGR9)
ENET	enet1. ipp_ind_mac0_txclk	ENET_REF_CLK_ROOT	clk_enable_enet1 (CCGR10)
	enet1. ipg_clk	ENET_AXI_CLK_ROOT	clk_enable_enet1 (CCGR10)
	enet1. ipg_clk_mac0	ENET_AXI_CLK_ROOT	clk_enable_enet1 (CCGR10)
	enet1. ipg_clk_mac0_s	ENET_AXI_CLK_ROOT	clk_enable_enet1 (CCGR10)
	enet1. ipg_clk_s	ENET_AXI_CLK_ROOT	clk_enable_enet1 (CCGR10)
	enet1. ipg_clk_time	ENET_TIMER_CLK_ROOT	clk_enable_enet1 (CCGR10)
	enet1_mem. mac0_rxmem_clk	ENET_AXI_CLK_ROOT	clk_enable_enet1 (CCGR10)
	enet1_mem. mac0_txmem_clk	ENET_AXI_CLK_ROOT	clk_enable_enet1 (CCGR10)
GPIO	gpio1.ipg_clk_s	IPG_CLK_ROOT	clk_enable_gpio1 (CCGR11)
	gpio2.ipg_clk_s	IPG_CLK_ROOT	clk_enable_gpio2 (CCGR12)
	gpio3.ipg_clk_s	IPG_CLK_ROOT	clk_enable_gpio3 (CCGR13)
	gpio4.ipg_clk_s	IPG_CLK_ROOT	clk_enable_gpio4 (CCGR14)
	gpio5.ipg_clk_s	IPG_CLK_ROOT	clk_enable_gpio5 (CCGR15)
GPMI	rawnand.u_gpmi_bch_input_gpmi_io_clk	NAND_CLK_ROOT	clk_enable_rawnand (CCGR48)
	rawnand.u_gpmi_input_apb_clk	NAND_USDHC_BUS_CLK_ROOT	clk_enable_rawnand (CCGR48)
GPT	gpt1.ipg_clk	GPT1_CLK_ROOT	clk_enable_gpt1 (CCGR16)
	gpt1.ipg_clk_24m	ANAMIX_OSC_24M_CLK	clk_enable_gpt1 (CCGR16)
	gpt1.ipg_clk_highfreq	GPT1_CLK_ROOT	clk_enable_gpt1 (CCGR16)
	gpt1.ipg_clk_s	GPT1_CLK_ROOT	clk_enable_gpt1 (CCGR16)
	gpt2.ipg_clk	GPT2_CLK_ROOT	clk_enable_gpt2 (CCGR17)
	gpt2.ipg_clk_24m	ANAMIX_OSC_24M_CLK	clk_enable_gpt2 (CCGR17)
	gpt2.ipg_clk_highfreq	GPT2_CLK_ROOT	clk_enable_gpt2 (CCGR17)
	gpt2.ipg_clk_s	GPT2_CLK_ROOT	clk_enable_gpt2 (CCGR17)
	gpt3.ipg_clk	GPT3_CLK_ROOT	clk_enable_gpt3 (CCGR18)
	gpt3.ipg_clk_24m	ANAMIX_OSC_24M_CLK	clk_enable_gpt3 (CCGR18)
	gpt3.ipg_clk_highfreq	GPT3_CLK_ROOT	clk_enable_gpt3 (CCGR18)
	gpt3.ipg_clk_s	GPT3_CLK_ROOT	clk_enable_gpt3 (CCGR18)
	gpt4.ipg_clk	GPT4_CLK_ROOT	clk_enable_gpt4 (CCGR19)
	gpt4.ipg_clk_24m	ANAMIX_OSC_24M_CLK	clk_enable_gpt4 (CCGR19)
	gpt4.ipg_clk_highfreq	GPT4_CLK_ROOT	clk_enable_gpt4 (CCGR19)
	gpt4.ipg_clk_s	GPT4_CLK_ROOT	clk_enable_gpt4 (CCGR19)
	gpt5.ipg_clk	GPT5_CLK_ROOT	clk_enable_gpt5 (CCGR20)
	gpt5.ipg_clk_24m	ANAMIX_OSC_24M_CLK	clk_enable_gpt5 (CCGR20)
	gpt5.ipg_clk_highfreq	GPT5_CLK_ROOT	clk_enable_gpt5 (CCGR20)
	gpt5.ipg_clk_s	GPT5_CLK_ROOT	clk_enable_gpt5 (CCGR20)

Table continues on the next page...

Table 5-2. System Clocks and Gating (continued)

Module	Module Clock (instance.clock)	Clock Root	Module Clock Gating Enable (CCGR)
	gpt6.ipg_clk	GPT6_CLK_ROOT	clk_enable_gpt6 (CCGR21)
	gpt6.ipg_clk_24m	ANAMIX_OSC_24M_CLK	clk_enable_gpt6 (CCGR21)
	gpt6.ipg_clk_highfreq	GPT6_CLK_ROOT	clk_enable_gpt6 (CCGR21)
	gpt6.ipg_clk_s	GPT6_CLK_ROOT	clk_enable_gpt6 (CCGR21)
I2C	i2c1. ipg_clk_patref	I2C1_CLK_ROOT	clk_enable_i2c1 (CCGR23)
	i2c1. ipg_clk_s	I2C1_CLK_ROOT	clk_enable_i2c1 (CCGR23)
	i2c2. ipg_clk_patref	I2C2_CLK_ROOT	clk_enable_i2c2 (CCGR24)
	i2c2. ipg_clk_s	I2C2_CLK_ROOT	clk_enable_i2c2 (CCGR24)
	i2c3. ipg_clk_patref	I2C3_CLK_ROOT	clk_enable_i2c3 (CCGR25)
	i2c3. ipg_clk_s	I2C3_CLK_ROOT	clk_enable_i2c3 (CCGR25)
	i2c4. ipg_clk_patref	I2C4_CLK_ROOT	clk_enable_i2c4 (CCGR26)
	i2c4. ipg_clk_s	I2C4_CLK_ROOT	clk_enable_i2c4 (CCGR26)
IOMUXC	iomuxc.ipg_clk_s	IPG_CLK_ROOT	clk_enable_iomux (CCGR27)
	iomuxc_gpr.ipg_clk_s	IPG_CLK_ROOT	clk_enable_iomux (CCGR27)
MU	mu.ipg_clk_dsp	IPG_CLK_ROOT	clk_enable_mu (CCGR33)
	mu.ipg_clk_mcu	IPG_CLK_ROOT	clk_enable_mu (CCGR33)
	mu.ipg_clk_s_dsp	IPG_CLK_ROOT	clk_enable_mu (CCGR33)
	mu.ipg_clk_s_mcu	IPG_CLK_ROOT	clk_enable_mu (CCGR33)
OCOTP	ocotp.ipg_clk	IPG_CLK_ROOT	clk_enable_ocotp (CCGR34)
	ocotp.ipg_clk_s	IPG_CLK_ROOT	clk_enable_ocotp (CCGR34)
OCRAM	ocram_ctrl.clk	MAIN_AXI_CLK_ROOT	clk_enable_ocram (CCGR35)
	ocram_ctrl_s.clk	AHB_CLK_ROOT	clk_enable_ocram_s (CCGR36)
	ocram_exsc.aclk_exsc	MAIN_AXI_CLK_ROOT	clk_enable_ocram (CCGR35)
	ocram_exsc.ipg_clk	IPG_CLK_ROOT	clk_enable_ocram (CCGR35)
	ocram_mem.clk	MAIN_AXI_CLK_ROOT	clk_enable_ocram (CCGR35)
	ocram_s_exsc.aclk_exsc	AHB_CLK_ROOT	clk_enable_ocram_s (CCGR36)
	ocram_s_exsc.ipg_clk	IPG_CLK_ROOT	clk_enable_ocram_s (CCGR36)
	ocram_s_mem.clk	AHB_CLK_ROOT	clk_enable_ocram_s (CCGR36)
PCIe	pcie_ctrl.auxclk	PCIE_AUX_CLK_ROOT	clk_enable_pcie (CCGR37)
	pcie_ctrl.mstr_aclk	PCIE_CTRL_CLK_ROOT	clk_enable_pcie (CCGR37)
	pcie_ctrl.slv_aclk	PCIE_CTRL_CLK_ROOT	clk_enable_pcie (CCGR37)
	pcie_exsc.aclk_exsc	PCIE_CTRL_CLK_ROOT	clk_enable_pcie (CCGR37)
	pcie_exsc.ipg_clk	IPG_CLK_ROOT	clk_enable_pcie (CCGR37)
	pcie_mem.mstr_axi_clk	PCIE_CTRL_CLK_ROOT	clk_enable_pcie (CCGR37)
	pcie_mem.slv_axi_clk	PCIE_CTRL_CLK_ROOT	clk_enable_pcie (CCGR37)
PWM	pwm1.ipg_clk	PWM1_CLK_ROOT	clk_enable_pwm1 (CCGR40)

Table continues on the next page...

Table 5-2. System Clocks and Gating (continued)

Module	Module Clock (instance.clock)	Clock Root	Module Clock Gating Enable (CCGR)
	pwm1.ipg_clk_highfreq	PWM1_CLK_ROOT	clk_enable_pwm1 (CCGR40)
	pwm1.ipg_clk_s	PWM1_CLK_ROOT	clk_enable_pwm1 (CCGR40)
	pwm2.ipg_clk	PWM2_CLK_ROOT	clk_enable_pwm2 (CCGR41)
	pwm2.ipg_clk_highfreq	PWM2_CLK_ROOT	clk_enable_pwm2 (CCGR41)
	pwm2.ipg_clk_s	PWM2_CLK_ROOT	clk_enable_pwm2 (CCGR41)
	pwm3.ipg_clk	PWM3_CLK_ROOT	clk_enable_pwm3 (CCGR42)
	pwm3.ipg_clk_highfreq	PWM3_CLK_ROOT	clk_enable_pwm3 (CCGR42)
	pwm3.ipg_clk_s	PWM3_CLK_ROOT	clk_enable_pwm3 (CCGR42)
	pwm4.ipg_clk	PWM4_CLK_ROOT	clk_enable_pwm4 (CCGR43)
	pwm4.ipg_clk_highfreq	PWM4_CLK_ROOT	clk_enable_pwm4 (CCGR43)
	pwm4.ipg_clk_s	PWM4_CLK_ROOT	clk_enable_pwm4 (CCGR43)
FLEXSPI	flexspi_wrapper.hclk	AHB_CLK_ROOT	clk_enable_qspi (CCGR47)
	flexspi_wrapper.ipg_clk	IPG_CLK_ROOT	clk_enable_qspi (CCGR47)
	flexspi_wrapper.ipg_clk_s	IPG_CLK_ROOT	clk_enable_qspi (CCGR47)
	flexspi_wrapper.ipg_clk_4xsfif	QSPI_CLK_ROOT	clk_enable_qspi (CCGR47)
	flexspi_wrapper.ipg_clk_s	IPG_CLK_ROOT	clk_enable_qspi (CCGR47)
	qspi_sec.ipg_clk	IPG_CLK_ROOT	clk_enable_qspi (CCGR47)
	qspi_sec.ipg_clk_s	IPG_CLK_ROOT	clk_enable_qspi (CCGR47)
	qspi_sec.mst_hclk	AHB_CLK_ROOT	clk_enable_qspi (CCGR47)
RDC	rdc. ipg_clk_s	IPG_CLK_ROOT	clk_enable_rdc (CCGR49)
	rdc. ipg_clk	IPG_CLK_ROOT	clk_enable_rdc (CCGR49)
	rdc_mem. ipg_clk	IPG_CLK_ROOT	clk_enable_rdc (CCGR49)
SAI	sai1.ipg_clk	AUDIO_AHB_CLK_ROOT	clk_enable_sai1 (CCGR51)
	sai1.ipg_clk_s	AUDIO_AHB_CLK_ROOT	clk_enable_sai1 (CCGR51)
	sai1.ipg_clk_sai_mclk	SAI1_CLK_ROOT	clk_enable_sai1 (CCGR51)
	sai1.ipt_clk_sai_bclk	SAI1_CLK_ROOT	clk_enable_sai1 (CCGR51)
	sai1.ipt_clk_sai_bclk_b	SAI1_CLK_ROOT	clk_enable_sai1 (CCGR51)
	sai2.ipg_clk	AUDIO_AHB_CLK_ROOT	clk_enable_sai2 (CCGR52)
	sai2.ipg_clk_s	AUDIO_AHB_CLK_ROOT	clk_enable_sai2 (CCGR52)
	sai2.ipg_clk_sai_mclk	SAI2_CLK_ROOT	clk_enable_sai2 (CCGR52)
	sai2.ipt_clk_sai_bclk	SAI2_CLK_ROOT	clk_enable_sai2 (CCGR52)
	sai2.ipt_clk_sai_bclk_b	SAI2_CLK_ROOT	clk_enable_sai2 (CCGR52)
	sai3.ipg_clk	AUDIO_AHB_CLK_ROOT	clk_enable_sai3 (CCGR53)
	sai3.ipg_clk_s	AUDIO_AHB_CLK_ROOT	clk_enable_sai3 (CCGR53)
	sai3.ipg_clk_sai_mclk	SAI3_CLK_ROOT	clk_enable_sai3 (CCGR53)
	sai3.ipt_clk_sai_bclk	SA3_CLK_ROOT	clk_enable_sai3 (CCGR53)
	sai3.ipt_clk_sai_bclk_b	SA3_CLK_ROOT	clk_enable_sai3 (CCGR53)
	sai5.ipg_clk	AUDIO_AHB_CLK_ROOT	clk_enable_sai5 (CCGR55)
	sai5.ipg_clk_s	AUDIO_AHB_CLK_ROOT	clk_enable_sai5 (CCGR55)

Table continues on the next page...

Table 5-2. System Clocks and Gating (continued)

Module	Module Clock (instance.clock)	Clock Root	Module Clock Gating Enable (CCGR)
	sai5.ipg_clk_sai_mclk	SAI5_CLK_ROOT	clk_enable_sai5 (CCGR55)
	sai5.ipt_clk_sai_bclk	SAI5_CLK_ROOT	clk_enable_sai5 (CCGR55)
	sai5.ipt_clk_sai_bclk_b	SAI5_CLK_ROOT	clk_enable_sai5 (CCGR55)
	sai6.ipg_clk	AUDIO_AHB_CLK_ROOT	clk_enable_sai6 (CCGR56)
	sai6.ipg_clk_s	AUDIO_AHB_CLK_ROOT	clk_enable_sai6 (CCGR56)
	sai6.ipg_clk_sai_mclk	SAI6_CLK_ROOT	clk_enable_sai6 (CCGR56)
	sai6.ipt_clk_sai_bclk	SAI6_CLK_ROOT	clk_enable_sai6 (CCGR56)
	sai6.ipt_clk_sai_bclk_b	SAI6_CLK_ROOT	clk_enable_sai6 (CCGR56)
SDMA	sdma1. ips_hostctrl_clk	IPG_CLK_ROOT	clk_enable_sdma1 (CCGR58)
	sdma1. sdma_ap_ahb_clk	AHB_CLK_ROOT	clk_enable_sdma1 (CCGR58)
	sdma1. sdma_core_clk	IPG_CLK_ROOT	clk_enable_sdma1 (CCGR58)
	sdma2. ips_hostctrl_clk	AUDIO_IPG_CLK_ROOT	clk_enable_sdma2 (CCGR59)
	sdma2. sdma_ap_ahb_clk	AUDIO_AHB_CLK_ROOT	clk_enable_sdma2 (CCGR59)
	sdma2. sdma_core_clk	AUDIO_IPG_CLK_ROOT	clk_enable_sdma2 (CCGR59)
SEMA4	sema1.clk	IPG_CLK_ROOT	clk_enable_sema1 (CCGR61)
	sema2.clk	IPG_CLK_ROOT	clk_enable_sema2 (CCGR62)
SIM	sim_display.cm4clk	ARM_M4_CLK_ROOT	GPC controlled
	sim_display.mainclk	MAIN_AXI_CLK_ROOT	clk_enable_sim_display (CCGR63)
	sim_display.mainclk_r	MAIN_AXI_CLK_ROOT	clk_enable_sim_display (CCGR63)
	sim_enet.mainclk	ENET_AXI_CLK_ROOT	clk_enable_sim_enet (CCGR64)
	sim_enet.mainclk_r	ENET_AXI_CLK_ROOT	clk_enable_sim_enet (CCGR64)
	sim_m.mainclk	AHB_CLK_ROOT	clk_enable_sim_m (CCGR65)
	sim_m.mainclk_r	AHB_CLK_ROOT	clk_enable_sim_m (CCGR65)
	sim_m.usdhcclk	NAND_USDHC_BUS_CLK_ROOT	clk_enable_sim_m (CCGR65)
	sim_m.usdhcclk_r	NAND_USDHC_BUS_CLK_ROOT	clk_enable_sim_m (CCGR65)
	sim_main.cm4clk	ARM_M4_CLK_ROOT	GPC controlled
	sim_main.enetclk	ENET_AXI_CLK_ROOT	clk_enable_sim_enet (CCGR64)
	sim_main.mainclk	MAIN_AXI_CLK_ROOT	clk_enable_sim_main (CCGR66)
	sim_main.mainclk_r	MAIN_AXI_CLK_ROOT	clk_enable_sim_main (CCGR66)
	sim_main.per_mclk	AHB_CLK_ROOT	clk_enable_sim_m (CCGR65)
	sim_main.per_sclk	AHB_CLK_ROOT	clk_enable_sim_s (CCGR67)
	sim_main.usdhcclk	NAND_USDHC_BUS_CLK_ROOT	clk_enable_sim_m (CCGR65)

Table continues on the next page...

Table 5-2. System Clocks and Gating (continued)

Module	Module Clock (instance.clock)	Clock Root	Module Clock Gating Enable (CCGR)
	sim_main.wakeupclk	AHB_CLK_ROOT	clk_enable_sim_wakeup (CCGR68)
	sim_s.apbhdmaclk	NAND_USDHC_BUS_CLK_ROOT	clk_enable_rawnand (CCGR48)
	sim_s.gpv4clk	ENET_AXI_CLK_ROOT	clk_enable_sim_enet (CCGR64)
	sim_s.mainclk	AHB_CLK_ROOT	clk_enable_sim_s (CCGR67)
	sim_s.mainclk_r	AHB_CLK_ROOT	clk_enable_sim_s (CCGR67)
	sim_s.weimclk	AHB_CLK_ROOT	clk_enable_sim_s_weimclk (CCGR4)
	sim_wakeup.mainclk	AHB_CLK_ROOT	clk_enable_sim_wakeup (CCGR68)
	sim_wakeup.mainclk_r	AHB_CLK_ROOT	clk_enable_sim_wakeup (CCGR68)
SNVS	snvs_hs_wrapper.ipg_clk	IPG_CLK_ROOT	clk_enable_snvs (CCGR71)
	snvs_hs_wrapper.ipg_clk_s	IPG_CLK_ROOT	clk_enable_snvs (CCGR71)
SPBA	spba1.ipg_clk	IPG_CLK_ROOT	clk_enable_ipmux3 (CCGR30)
	spba1.ipg_clk_s	IPG_CLK_ROOT	clk_enable_ipmux3 (CCGR30)
UART	uart1.ipg_clk	IPG_CLK_ROOT	clk_enable_uart1 (CCGR73)
	uart1.ipg_clk_s	IPG_CLK_ROOT	clk_enable_uart1 (CCGR73)
	uart1.ipg_perclk	UART1_CLK_ROOT	clk_enable_uart1 (CCGR73)
	uart2.ipg_clk	IPG_CLK_ROOT	clk_enable_uart2 (CCGR74)
	uart2.ipg_clk_s	IPG_CLK_ROOT	clk_enable_uart2 (CCGR74)
	uart2.ipg_perclk	UART2_CLK_ROOT	clk_enable_uart2 (CCGR74)
	uart3.ipg_clk	IPG_CLK_ROOT	clk_enable_uart3 (CCGR75)
	uart3.ipg_clk_s	IPG_CLK_ROOT	clk_enable_uart3 (CCGR75)
	uart3.ipg_perclk	UART3_CLK_ROOT	clk_enable_uart3 (CCGR75)
	uart4.ipg_clk	IPG_CLK_ROOT	clk_enable_uart4 (CCGR76)
	uart4.ipg_clk_s	IPG_CLK_ROOT	clk_enable_uart4 (CCGR76)
	uart4.ipg_perclk	UART4_CLK_ROOT	clk_enable_uart4 (CCGR76)
uSDHC	usdhc1.hclk	NAND_USDHC_BUS_CLK_ROOT	clk_enable_usdhc1 (CCGR81)
	usdhc1.ipg_clk	IPG_CLK_ROOT	clk_enable_usdhc1 (CCGR81)
	usdhc1.ipg_clk_perclk	USDHC1_CLK_ROOT	clk_enable_usdhc1 (CCGR81)
	usdhc1.ipg_clk_s	IPG_CLK_ROOT	clk_enable_usdhc1 (CCGR81)
	usdhc2.hclk	NAND_USDHC_BUS_CLK_ROOT	clk_enable_usdhc2 (CCGR82)

Table continues on the next page...

Table 5-2. System Clocks and Gating (continued)

Module	Module Clock (instance.clock)	Clock Root	Module Clock Gating Enable (CCGR)
	usdhc2.ipg_clk	IPG_CLK_ROOT	clk_enable_usdhc2 (CCGR82)
	usdhc2.ipg_clk_perclk	USDHC2_CLK_ROOT	clk_enable_usdhc2 (CCGR82)
	usdhc2.ipg_clk_s	IPG_CLK_ROOT	clk_enable_usdhc2 (CCGR82)
	usdhc3.hclk	NAND_USDHC_BUS_CLK_ROOT	clk_enable_usdhc3 (CCGR94)
	usdhc3.ipg_clk	IPG_CLK_ROOT	clk_enable_usdhc3 (CCGR94)
	usdhc3.ipg_clk_perclk	USDHC3_CLK_ROOT	clk_enable_usdhc3 (CCGR94)
	usdhc3.ipg_clk_s	IPG_CLK_ROOT	clk_enable_usdhc3 (CCGR94)
WDOG	wdog1.ipg_clk	WDOG_CLK_ROOT	clk_enable_wdog1 (CCGR83)
	wdog1.ipg_clk_s	WDOG_CLK_ROOT	clk_enable_wdog1 (CCGR83)
	wdog2.ipg_clk	WDOG_CLK_ROOT	clk_enable_wdog2 (CCGR84)
	wdog2.ipg_clk_s	WDOG_CLK_ROOT	clk_enable_wdog2 (CCGR84)
	wdog3.ipg_clk	WDOG_CLK_ROOT	clk_enable_wdog3 (CCGR85)
	wdog3.ipg_clk_s	WDOG_CLK_ROOT	clk_enable_wdog3 (CCGR85)

5.1.5 Functional Description

The following sections describe the functional details of CCM.

5.1.5.1 Input Clocks

The table below describes the input clock sources that supply the muxes to the clock slices in the clock root generator. Please see PLL Interface for PLL programming information.

Control Register (CCM_PLL_CTRLn)	Input Clock	Frequency (MHz)	Description
12	ARM_PLL_CLK	1600	Arm PLL
13	GPU_PLL_CLK	1600	GPU PLL clock output
14	VPU_PLL_CLK	800	VPU PLL clock output
15	DRAM_PLL1_CLK	800	DDR PLL
16	SYSTEM_PLL1_CLK	800	System PLL1 output clock

Table continues on the next page...

Control Register (CCM_PLL_CTRLn)	Input Clock	Frequency (MHz)	Description
17	SYSTEM_PLL1_DIV2	400	System PLL1 divided 2 clock output
18	SYSTEM_PLL1_DIV3	266	System PLL1 divided 3 clock output
19	SYSTEM_PLL1_DIV4	200	System PLL1 divided 4 clock output
20	SYSTEM_PLL1_DIV5	160	System PLL1 divided 5 clock output
21	SYSTEM_PLL1_DIV6	133	System PLL1 divided 6 clock output
22	SYSTEM_PLL1_DIV8	100	System PLL1 divided 8 clock output
23	SYSTEM_PLL1_DIV10	80	System PLL1 divided 10 clock output
24	SYSTEM_PLL1_DIV20	40	System PLL divided 20 clock output
25	SYSTEM_PLL2_CLK	1000	System PLL2 output clock
26	SYSTEM_PLL2_DIV2	500	System PLL2 divided 2 clock output
27	SYSTEM_PLL2_DIV3	333	System PLL2 divided 3 clock output
28	SYSTEM_PLL2_DIV4	250	System PLL2 divided 4 clock output
29	SYSTEM_PLL2_DIV5	200	System PLL2 divided 5 clock output
30	SYSTEM_PLL2_DIV6	166	System PLL2 divided 6 clock output
31	SYSTEM_PLL2_DIV8	125	System PLL2 divided 8 clock output
32	SYSTEM_PLL2_DIV10	100	System PLL2 divided 10 clock output
33	SYSTEM_PLL2_DIV20	50	System PLL2 divided 20 clock output
34	SYSTEM_PLL3_CLK	1000	System PLL3 output clock
35	AUDIO_PLL1_CLK	650	Audio PLL1 clock output
36	AUDIO_PLL2_CLK	650	Audio PLL2 clock output
37	VIDEO_PLL1_CLK	650	Video PLL1 clock output
external source	32K_REF_CLK	0.032	32K oscillator clock output
external source	24M_REF_CLK	24	24M oscillator clock output
external source	EXT_CLK_1	133	Clock input from external IO
external source	EXT_CLK_2	133	Clock input from external IO
external source	EXT_CLK_3	133	Clock input from external IO
external source	EXT_CLK_4	133	Clock input from external IO

NOTE

CKIL_SYNC needs to be configured the same as the PLL for ipg_clk

5.1.5.2 CKIL Synchronizer

CCM provides a synchronized version of the 32K clock called CKIL Synchronizer (CKIL_SYNC). The CKIL_SYNC clock is generated and synchronized by the IPG_CLK_ROOT when IPG_CLK_ROOT is active.

When the system enters low-power mode that requires the shutdown of IPG_CLK_ROOT, the CKIL Synchonrizer needs to be bypassed and fed directly from the XTALOSC 32K source.

The control for the CKIL_SYNC bypass comes from source control. The control for CKIL_SYNC bypass needs to be configured exactly the same as the contols for the PLL output that is used to generate IPG_CLK_ROOT.

In cases where a module ipg_clk is not tied to IPG_CLK_ROOT, the bypass signal to the ipg_clk is controlled by LPCG. To avoid issues on their 32K input, the LPCG must also be shutdown at the same time as the PLL clock output for that particular ipg_clk generation.

5.1.5.3 Clock Components

The details of the CCM clock components are detailed below.

5.1.5.3.1 Clock Divider

Synchronized clock dividers can be used to perform integer division on the source clock frequency. The divider guarantees a clean clock signal on its output during the change of the divide factor. Dividers perform a 1/(N+1) divide. A glitch can cause a sync-divider to go into an unrecoverable state, therefore a clean clock signal must be provided to a sync-divider.

When updating the divider value, a read enable signal (read_en) deasserts to stop the related control registers from fetching the divider value. To change a divider factor value, the bus enters a transition state and the divide logic latches the divider factor value. After this value is latched, the read_en is reasserted. The divider will begin to update its internal divide logic. After it finishes updating, the divider sends an acknowledge signal (sw_ack) to indicate a new value is in effect.

5.1.5.3.2 Clock Switching Multiplexer

Clock switching multiplexers can guarantee a clean clock signal when switching between 2 clock sources. Both clock inputs must be active when switching. Glitches in the selected source may be transferred to the output clock while the de-selected source is blocked. Glitches that occur during switching may cause an unpredictable state, and will recover in 2~3 cycles.

Clock switching multiplexers first shutdown the current active clock, then switch. After receiving an acknowledge that the current clock stopped. The multiplexer turns on the new selected clock source. As the acknowledge of turning on the new clock source is received, the source switch finishes.

5.1.5.3.3 Clock Gate

A clock gate cell is used to gate clocks. When gated, the clock will stop at 0. A clock gate cell can accept glitches on its clock inputs. If the gate cell is off, it will block or absorb glitches. If the gate cell is on, it may pass glitches to its output. A gate cell needs 2~3 clock cycles to change states. The state during this period is either on or off and will recover in 2~3 cycles, but cannot be predicted which state it is in during this period.

5.1.5.3.4 8 to 1 Multiplexer

The 8-to-1 multiplexer is a combinational multiplexer that can switch anytime. The multiplexer output does not guarantee a clean clock signal. During switching, the output must be gated or all 8 inputs will require the same value.

5.1.5.4 Clock Slices

There are several types of clock generation slices in CCM. The slices are categorized as Core, Bus, Peripheral (IP), and DRAM.

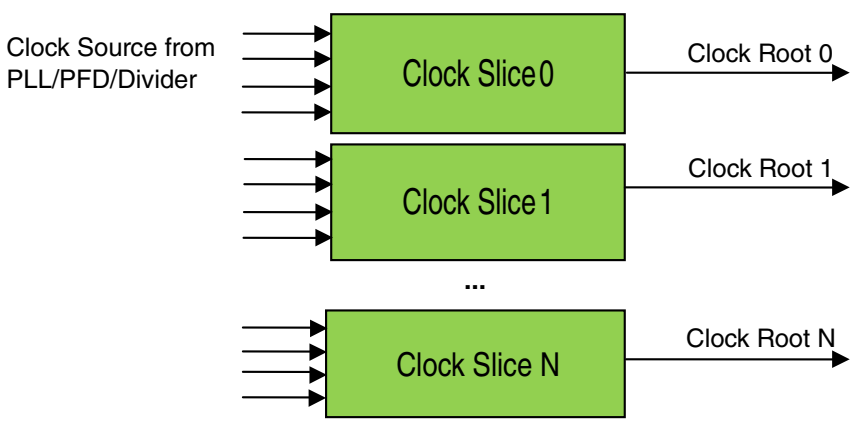


Figure 5-4. Clock Slices

5.1.5.4.1 Core clock slice

Core clock slices are designed for high speed, non-stop clock generation, typically for an Arm core. A core clock slice is comprised of a post divider and clock switching multiplexer. Each has a clock gate and a clock multiplexer inside. To run at high frequency, post divider is 3 bits, which is half the bit width of other slices. The two 8-to-1 multiplexers are not glitch-less, so switching them should only be done when their output is not routed to a clock output.

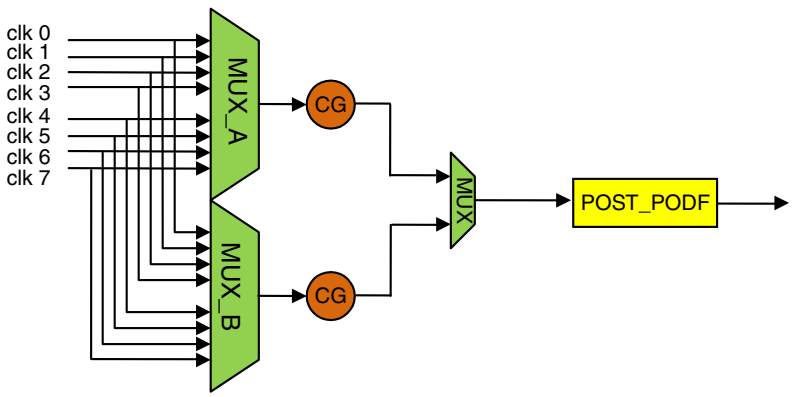


Figure 5-5. Core clock slice

5.1.5.4.2 Bus clock slice

Bus clock slices are comprised of a post divider and a clock switching multiplexer. Each has a pre-divider, clock gate and a clock multiplexer inside. The pre-divider is three bits and provides a maximum division factor of 8. The post-divider is six bits and can divide down the clock input to 1MHz on clock output. The two 8-to-1 multiplexers are not glitch-less, so switching should only be done when their output is not routed to clock output.

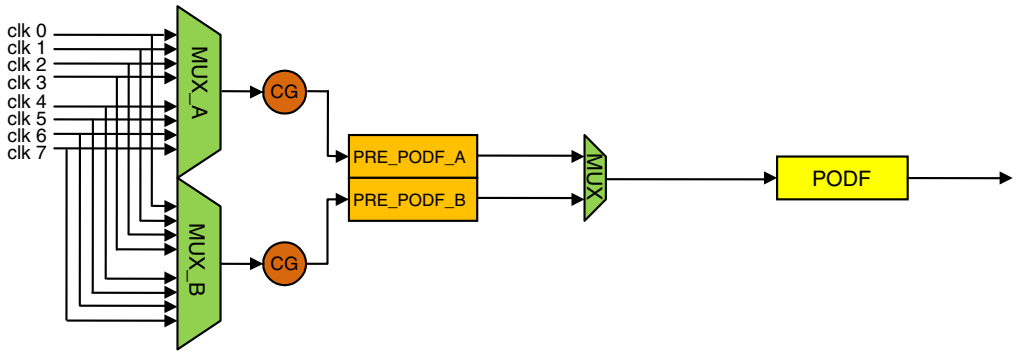


Figure 5-6. Bus clock slice

5.1.5.4.3 Peripheral clock slice

IP peripheral clock slices are comprised of a post-divider, pre-divider, clock gate and a clock multiplexer. The pre-divider is three bit and can divide down by a factor of 8. The post-divider is six bit and can divide down the input clock to 1MHz on clock output. The 8-to-1 multiplexer is not glitch-less, so switching should only be done when their output is not routed to clock output.

IP clock slices must be stopped to change the clock source.

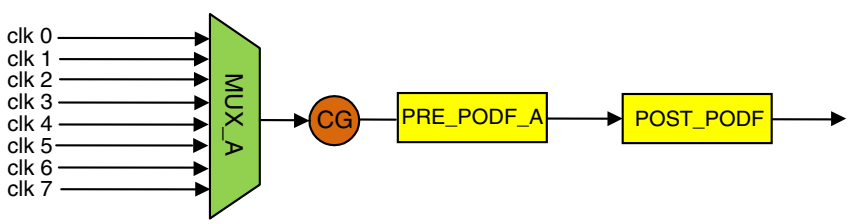


Figure 5-7. Peripheral clock slice

5.1.5.4.4 SSCG and Fractional PLLs

The SSCG and Fractional-N PLLs are detailed below. The SSCG block diagram is shown below. Please see CCM ANALOG for more information.

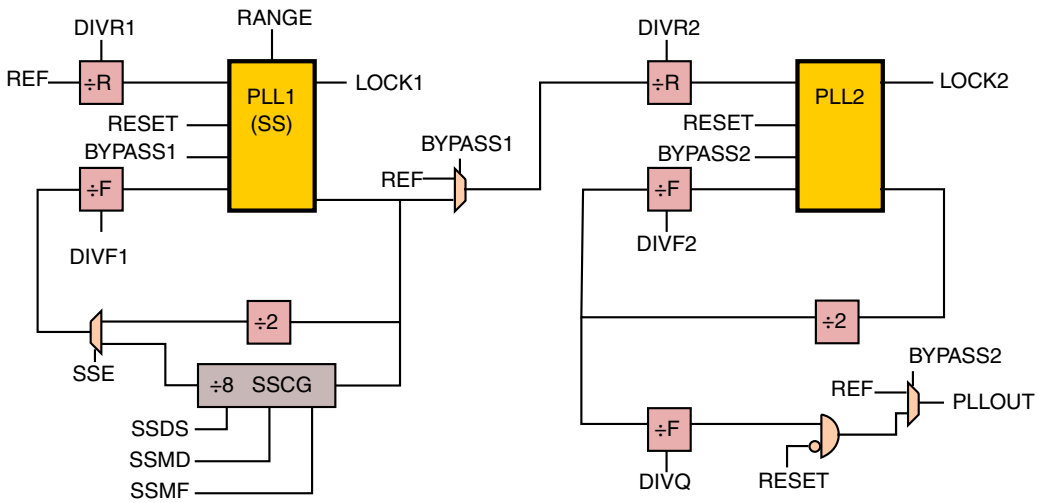


Figure 5-8. SSCG PLL Block Diagram

Formula for SSCG PLLOUT:

- SSE=0: PLLOUT = REF/DIVR1 * 2 * DIVF1/DIVR2 * DIVF2/DIVQ
- SSE=1: PLLOUT = REF/DIVR1 * 8 * DIVF1/DIVR2 * DIVF2/DIVQ

The ARM PLL, GPU PLL, VPU PLL, Audio PLL1/2, and Video PLL1 are fractional PLLs. The frequency on these can be tuned to be very accurate to meet audio and video interface requirements. The figure below shows the Fractional PLL block diagram.

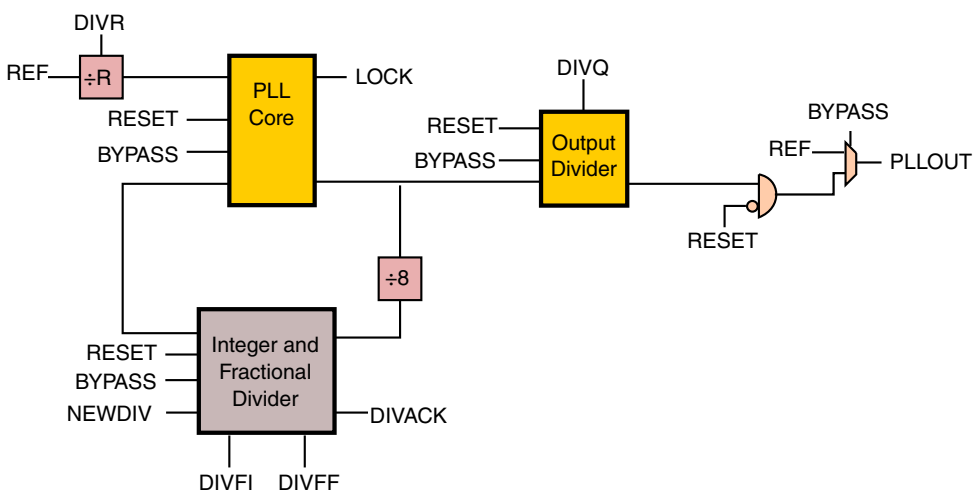


Figure 5-9. Fractional PLL Block Diagram

Formula for Fraction PLLOUT:

- PLLOUT = REF/DIVR_VAL * 8 * DIVF_VAL/DIVQ_VAL
- Where, DIVF_VAL = 1 + DIVFI + (DIVFF/2^24)

5.1.5.5 Clock gate control

CCM can perform automatic clock shutdown of on-chip peripherals according to system low power mode. Each logic domain can respectively declare its dependency level on each clock. If a clock is detected that is not dependant on any domain, it will be shutdown to save power.

Clock generation inside the CCM creates a clock root for on-chip peripherals. Before the clock root goes into peripherals via low power clock gating cells. By controlling these LPCGs, CCM can manage on-chip peripheral clocks.

Clock gate controls use active clock gating, which means the low power clock gating (LPCG) requires an active clock root. The clock generation module only performs multiplexing, gating and dividing on clock sources. Therefore, the clock root from the generation module will stop when the corresponding clock source stops.

The ENABLE bit must be set for the clock root that LPCG is actively gating.

5.1.5.6 Clock source control

CCM can perform automatic PLL shutdown of CCM_ANALOG according to system low power mode. Each logic domain can respectively declare its dependency level on each clock. If a PLL is detected not depended by any domain, it will be shutdown to save power.

PLL in ANATOP can be set be controlled by CCM via setting some CCM override bits. Also there is control on 32K IPG_CLK_ROOT synchronized version inside clock source control module.

For PLLs, we have channel on every PLL, every PFD and every divider. PLL is the source of PFDs and dividers. For any clock, its source must be left on when it is kept on. Behavior is undefined if this rule is violated.

For a shutdown clock source, if it is declared depended by writing clock source control registers, the controlling logic will turn on the source immediately, while the setting goes into a shadow register. After clock source get ready, the setting will be accepted by source control logic, and copied from shadow register to setting register.

5.1.5.7 Access control

CCM can implement its own access control based on domain to provide more precise control on shared resources. Access controls are implemented on clock root generation, clock gate control, and clock source control. Access control logic does not impact read access, but blocks unauthentic write access.

Access controls on clock root generation are independent between every clock root. A sticky authentic fail flag is set when a domain writes to a register and authentication fails. The access control logic contains a whitelist and a semaphore. By default, each clock root's access control logic is disabled after power-on reset. Software can enable access control anytime after reset.

NOTE

Once access logic is enabled, it cannot be disabled until the next power-on reset.

Table 5-3. Whitelist

Enabled	Write access will be authenticated before being performed.
Disabled	every access on protected item will be performed.

NOTE

Only domains that are on the whitelist can perform write access to this clock root when access control is enabled.

Table 5-4. Semaphore

Enabled	A domain must obtain the semaphore’s ownership before its write access can be authenticated. Only a domain on the whitelist can obtain the ownership and the ownership will last until the domain explicitly releases the ownership. Semaphore obtain will fail if it is already fetched by some other domain.
Disabled	Authentic check will check only on whitelist.

NOTE

Semaphore is intended to help software keep the clock root from unexpectedly changing.

Access control of clock gate and clock source control is performed in a simple operation. Every domain can only write on the bits for it's own setting. Any write to irrespective domain will be ignored.

5.1.5.8 System level considerations

Clock shutdown strategy

Any clock shutdown should first shutdown the LPCG, then the PLL. If the LPCG is configured not to shutdown, the clock root for the LPCG should not stop either. Violating this rule leads the system into an unpredictable state.

Core clock root frequency

If the core clock is set lower that one third of the IPG clock, SRC needs to generate a longer reset signal to match the requirement from the Arm core. This typically happens when the Arm core runs at some divided value of the XTAL 24M while IPG clock is supplied by the PLL.

DRAM clock

The DRAM PHYM clock needs a clock frequency faster that 400MHz.

USB OTG CLOCK

The USB clock may not be a reliable clock source in some applications. This clock may stop when USB cable is disconnected.

5.1.6 Programming Guide

5.1.6.1 Set, Clear, and Toggle register features

Every register of the CCM has set, clear, and toggle features.

The set feature for a given register is located at:
Base Address of the register + 0x04

The clear feature for a given register is located at:
Base Address of the regsister + 0x08

The toggle feature is located at:
Base Address of the register + 0x0c

Read from all 4 locations to get the current register value. Writing to the base register bits sets the register to the write value. Writing 1 to the set register bits sets them to 1, while writing 0 has no effect. Writing 1 to the clear register bits clears them to 0, while writing 0 has no effect. Writing 1 to the toggle register bits sets them to invert the value, while writing 0 has no effect.

5.1.6.2 PLL Interface

CCM can control PLLs inside CCM_ANALOG when entering or leaving low-power mode. Software must set the PLL override inside CCM_ANALOG before entering low-power mode after power-on reset (POR).

There are four levels of low-power modes in a logic domain:

- Not needed
- Needed in RUN
- Needed in RUN and WAIT
- Needed in RUN, WAIT, and STOP

CCM only takes action while domain status are switching between STOP (DEEP SLEEP mode is considered the same as STOP). There are 4 domains that can be assigned. Any CPU platform can be assigned to any domain by RDC. If a domain is empty, the domain is considered as STOP.

Each domain can declare its dependency to CCM. The use of any clock, without declaring it in its own domain, is not permitted. A domain declares its dependency on a clock by writing the dependency level. Settings against behavior in low-power mode are as follows:

Table 5-5. Domain Dependency

Domain Level	Run	Wait	Stop / Deep sleep
0			
1	Required		
2	Required	Required	
3	Required	Required	Required

Table 5-6. CCGR Program Interface

CCGR		Domain3				Domain2				Domain1				Domain0			
	31-16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Level1	Level0			Level1	Level0			Level1	Level0			Level1	Level0

Each domain can change only bits assigned to control access. Any irrelevant write to it will be ignored. For example, Domain 0 can only write to bits [1:0]. Any bits written to, other than bits [1:0], will be ignored. Other domains can read all of the other domain settings. The default value for domain 0 is 2, and will enter STOP mode after shutting down. When the default value of the other domain setting is 0, it will not be required.

When setting clock source, the settings will not take effect immediately. The setting will enter the shadow register first. If a PLL shutdown or new setting enters the shadow register to declare dependency on the PLL, the PLL will turn on immediately. When the PLL is ready, the setting in shadow register will be updated to the new setting. During this period, the pending bit will be set and cleared. Then CCM will send the PLL control signal as a shadow register and inform GPC the PLL status according to the setting register. In other cases, the setting will be updated from the shadow register immediately. Clock sources have dependency on each other.

NOTE

Do not shutdown the parent clock when the required child clock is active. Attempting to do so will lead to unpredicable and unrecoverable behavior. It is recommended to shutdown the parent clock and child clock together.

5.1.6.3 CCGR Interface

Before a clock root goes to on–chip peripherals, the clock root is distributed through low power clock gates (LPCG). These LPCG are implemented to automatically perform clock shutdown when a domain enters and leaves a low-power state.

There are four levels of low-power modes in a logic domain:

- Not needed
- Needed in RUN
- Needed in RUN and WAIT
- Needed in RUN, WAIT, and STOP

CCM only takes action while domain status are switching between STOP (DEEP SLEEP mode is considered the same as STOP). There are 4 domains that can be assigned. Any CPU platform can be assigned to any domain by RDC. If a domain is empty, the domain is considered as STOP.

Each domain can declare its dependency to CCM. The use of any clock, without declaring it in its own domain, is not permitted. A domain declares its dependency on a clock by writing the dependency level. Settings against behavior in low-power mode are as follows:

Table 5-7. Domain Dependency

Domain Level	RUN	WAIT	STOP/ DEEP SLEEP
0			
1	Required		
2	Required	Required	

Table continues on the next page...

Table 5-7. Domain Dependency (continued)

3	Required	Required	Required

Table 5-8. CCGR Program Interface

CCGR		Domain3				Domain2				Domain1				Domain0			
	31-16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Level1	Level0			Level1	Level0			Level1	Level0			Level1	Level0

Each domain can change only bits assigned to control access. Any irrelevant write to it will be ignored. For example, Domain 0 can only write to bits [1:0]. Any bits written to, other than bits [1:0], will be ignored. Other domains can read all of the other domain settings. The default value for domain 0 is 2, and will enter STOP mode after shutting down. When the default value of the other domain setting is 0, it will not be required.

The table below lists the CCM Clock Gating Register (CCGR) and associated offset for each LPCG enable.

NOTE

Not all CCGRs are mapped.

NOTE

Sec_debug clock gating (CCGR60) must be active in low power mode. DO NOT gate this clock in low power mode to guarantee the low power mode functions such as stop WDOG counting.

Table 5-9. CCGR Mapping Table

Gating Register	LPCG Enable	Offset
CCM_CCGR0	DVFS (GPC)	0x4000
CCM_CCGR1	Anamix	0x4010
CCM_CCGR2	CPU	0x4020
CCM_CCGR3	CSU	0x4030
CCM_CCGR4	Debug	0x4040
CCM_CCGR5	DDR1	0x4050
CCM_CCGR6	Reserved	0x4060
CCM_CCGR7	ECSPI1	0x4070
CCM_CCGR8	ECSPI2	0x4080
CCM_CCGR9	ECSPI3	0x4090
CCM_CCGR10	ENET1	0x40A0
CCM_CCGR11	GPIO1	0x40B0

Table continues on the next page...

Table 5-9. CCGR Mapping Table (continued)

Gating Register	LPCG Enable	Offset
CCM_CCGR12	GPIO2	0x40C0
CCM_CCGR13	GPIO3	0x40D0
CCM_CCGR14	GPIO4	0x40E0
CCM_CCGR15	GPIO5	0x40F0
CCM_CCGR16	GPT1	0x4100
CCM_CCGR17	GPT2	0x4110
CCM_CCGR18	GPT3	0x4120
CCM_CCGR19	GPT4	0x4130
CCM_CCGR20	GPT5	0x4140
CCM_CCGR21	GPT6	0x4150
CCM_CCGR22	HS	0x4160
CCM_CCGR23	I2C1	0x4170
CCM_CCGR24	I2C2	0x4180
CCM_CCGR25	I2C3	0x4190
CCM_CCGR26	I2C4	0x41A0
CCM_CCGR27	IOMUX	0x41B0
CCM_CCGR28	IOMUX1	0x41C0
CCM_CCGR29	IOMUX2	0x41D0
CCM_CCGR30	IOMUX3	0x41E0
CCM_CCGR31	IOMUX4	0x41F0
CCM_CCGR32	SNVSMIX	0x4200
CCM_CCGR33	MU	0x4210
CCM_CCGR34	OCOTP	0x4220
CCM_CCGR35	OCRAM	0x4230
CCM_CCGR36	OCRAM_s	0x4240
CCM_CCGR37	PCIE	0x4250
CCM_CCGR38	PERFMON1	0x4260
CCM_CCGR39	PERFMON2	0x4270
CCM_CCGR40	PWM1	0x4280
CCM_CCGR41	PWM2	0x4290
CCM_CCGR42	PWM3	0x42A0
CCM_CCGR43	PWM4	0x42B0
CCM_CCGR44	QoS	0x42C0
CCM_CCGR45	QoS_Dispmix	0x42D0
CCM_CCGR46	QoS_ENET	0x42E0
CCM_CCGR47	QSPI	0x42F0
CCM_CCGR48	NAND (APBHDMA, GPMI, BCH)	0x4300
CCM_CCGR49	RDC	0x4310
CCM_CCGR50	ROM	0x4320

Table continues on the next page...

Table 5-9. CCGR Mapping Table (continued)

Gating Register	LPCG Enable	Offset
CCM_CCGR51	SAI1	0x4330
CCM_CCGR52	SAI2	0x4340
CCM_CCGR53	SAI3	0x4350
CCM_CCGR54	SAI4	0x4360
CCM_CCGR55	SAI5	0x4370
CCM_CCGR56	SAI6	0x4380
CCM_CCGR57	SCTR	0x4390
CCM_CCGR58	SDMA1	0x43A0
CCM_CCGR59	SDMA2	0x43B0
CCM_CCGR60	SEC_DEBUG	0x43C0
CCM_CCGR61	SEMA1	0x43D0
CCM_CCGR62	SEMA2	0x43E0
CCM_CCGR63	SIM_display	0x43F0
CCM_CCGR64	SIM_ENET	0x4400
CCM_CCGR65	SIM_m	0x4410
CCM_CCGR66	SIM_main	0x4420
CCM_CCGR67	SIM_s	0x4430
CCM_CCGR68	SIM_wakeup	0x4440
CCM_CCGR69	SIM_HSIO	0x4450
CCM_CCGR70	Reserved	0x4460
CCM_CCGR71	SNVS	0x4470
CCM_CCGR72	Trace	0x4480
CCM_CCGR73	UART1	0x4490
CCM_CCGR74	UART2	0x44A0
CCM_CCGR75	UART3	0x44B0
CCM_CCGR76	UART4	0x44C0
CCM_CCGR77	USB	0x44D0
CCM_CCGR78	Reserved	0x44E0
CCM_CCGR79	GPU3D	0x44F0
CCM_CCGR80	Reserved	0x4500
CCM_CCGR81	USDHC1	0x4510
CCM_CCGR82	USDHC2	0x4520
CCM_CCGR83	WDOG1	0x4530
CCM_CCGR84	WDOG2	0x4540
CCM_CCGR85	WDOG3	0x4550
CCM_CCGR86	VPUG1	0x4560
CCM_CCGR87	GPU	0x4570
CCM_CCGR88	Reserved	0x4580
CCM_CCGR89	VPUH1	0x4590

Table continues on the next page...

Table 5-9. CCGR Mapping Table (continued)

Gating Register	LPCG Enable	Offset
CCM_CCGR90	VPUG2	0x45A0
CCM_CCGR91	PDM	0x45B0
CCM_CCGR92	GIC	0x45C0
CCM_CCGR93	Display	0x45D0
CCM_CCGR94	USDHC3	0x45E0
CCM_CCGR95	SDMA3	0x45F0
CCM_CCGR96	XTALOSC	0x4600
CCM_CCGR97	PLL	0x4610
CCM_CCGR98	TEMPSENSOR	0x4620
CCM_CCGR99	VPUMIX	0x4630
CCM_CCGR100	Reserved	0x4640
CCM_CCGR101	Reserved	0x4650
CCM_CCGR102	GPU2D	0x4660

5.1.6.4 Target Interface

The Target Interface is optimized to simplify software operation. Using this interface, all clock roots are in the same program model with the same register bit field mapping. The software does not handle the details of the clock slice and clock slice types. Software writes the desired settings to the register, and the internal hardware logic generates a required sequence to achieve the desired settings.

The Target Interface requires the software to provide whether a clock is active, the clock source number to be selected, pre-divide value, and post-divide value. If a clock slice does not support a setting, that setting is simply ignored, and will not effect the supported fields.

Freq = (clock source freq)/(pre_div+1)/(post_div+1)

The internal logic sequence of the Target Interface guarantees a clean clock on output without frequency overshoot. A requirement of the Target Interface's software is that the target clock source is active.

The Target Interface sequence begins by opening all clocks, applying highest divider value, switching to the new clock source, then decreasing divider value to the target frequency. If Shutdown is requested, it will be performed last.

The clock output is always active when using the Target Interface. For intermediate frequency requests, the Target Interface choses the lowest frequency source to avoid frequency overshoot on the Peripheral clock slices. For Core and Bus clock slices, the clock switching multiplexer is used to guarantee smooth clock switching.

A write operation on a target interface completes once the output clock is running at the desired setting. Software polling is not necessary to determine clock stability.

STEP	STATE	OPERATION
0	SMART_IDLE	Idle state, no write operation is pending
1	SMART_WAIT_READY	State occurs when a write access is received, wait for every field to be ready
2	SMART_APPLY_GATE1	Open all branches, all gates inside clock slices
3	SMART_WAIT_GATE1	Wait for gate applied
4	SMART_APPLY_PODF1	apply post divider and post divider for if new value generate slower clock
5	SMART_WAIT_PODF1	Wait for divider accept new value
6	SMART_APPLY_GATE2	shutdown spare branch if exist, else shutdown working branch
7	SMART_WAIT_GATE2	Wait for shutdown operation complete
8	SMART_APPLY_MUX	Change multiplexer to new source on spare branch if exist, else switch working one
9	SMART_APPLY_GATE3	open gates on all branches
10	SMART_WAIT_GATE3	Wait for gates opened
11	SMART_APPLY_SWITCH	Switch clock switching multiplexer if there is one
12	SMART_WAIT_SWITCH	Wait for clock switching multiplexer switch
13	SMART_APPLY_PODF2	apply post divider and post divider for if new value generate faster clock
14	SMART_WAIT_PODF2	Wait for divider accept new value
15	SMART_APPLY_GATE4	apply clock gate setting, shutdown spare one if there is
16	SMART_WAIT_GATE4	Wait for gate applied
17	SMART_APPLY_AUTO	apply auto and auto divider
18	SMART_DONE	Finish, wait for bus operation complete

5.1.6.5 Normal Interface

Normal interface provide more controllable thing that target interface. And also provide protections against dangerous operation.

Normal interface provides safe sequences to handle each clock component, divider, gate, multiplexer. But it is software that needs to care the order and relationship between updating components.

Writing to this interface will complete immediately, and internal logic will continue try to apply written values to clock components. A busy flag will be assert during applying.

Field access rule:

1. Only one field can be modified a time
2. No field can be modified when any field pending
3. Not violate change condition in following table

FIELD	CHANGE CONDITION	FINISH CONDITION
Auto		immediate
Auto-divider		immediate
Bypass	Gatea active and gateb active	Bypass switch complete
Post-divider		New divider value applied
Gate B	Bypass disable	New gate value active
Pre-divider B	Bypass disable and gateb not gated	New divider value applied
MUX B	Bypass disable and gateb gating	immediate
Gate A	Bypass	New gate value active
Pre-divider A	Bypass and gatea not gated	New divider value applied
MUX A	Bypass and gatea gating	immediate

- Error will be reported if access rules violated.
- Unsafe or ambiguous access will be ignored.

If a write access as blocked by normal interface, the write operation will be ignored. And a sticky bit “violate” will be set. The bit will last until software clears it explicitly. The violate bit is 4 bits inside CCM, each for a logic domain. Each domain can read and clear the bit for itself, the bits for other domain is neither visible nor clearable.

5.1.7 CCM Memory Map/Register Definition

The Memory Map below represents the full array for CCM.

NOTE

Not all mapped Clock Slices and CCGRs are tied to functional components.

Please see the following for the functional mapping tables and information:

- CCM_PLL_CTRL - Input Clocks
- CCM_TARGET_ROOT - Clock Root Selects
- CCM_CCGR - CCGR Interface

CCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_0000	General Purpose Register (CCM_GPR0)	32	R/W	0000_0000h	5.1.7.1/451
3038_0004	General Purpose Register (CCM_GPR0_SET)	32	R/W	0000_0000h	5.1.7.1/451
3038_0008	General Purpose Register (CCM_GPR0_CLR)	32	R/W	0000_0000h	5.1.7.1/451
3038_000C	General Purpose Register (CCM_GPR0_TOG)	32	R/W	0000_0000h	5.1.7.1/451
3038_0800	CCM PLL Control Register (CCM_PLL_CTRL0)	32	R/W	0000_0002h	5.1.7.2/452
3038_0804	CCM PLL Control Register (CCM_PLL_CTRL0_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0808	CCM PLL Control Register (CCM_PLL_CTRL0_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_080C	CCM PLL Control Register (CCM_PLL_CTRL0_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0810	CCM PLL Control Register (CCM_PLL_CTRL1)	32	R/W	0000_0002h	5.1.7.2/452
3038_0814	CCM PLL Control Register (CCM_PLL_CTRL1_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0818	CCM PLL Control Register (CCM_PLL_CTRL1_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_081C	CCM PLL Control Register (CCM_PLL_CTRL1_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0820	CCM PLL Control Register (CCM_PLL_CTRL2)	32	R/W	0000_0002h	5.1.7.2/452
3038_0824	CCM PLL Control Register (CCM_PLL_CTRL2_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0828	CCM PLL Control Register (CCM_PLL_CTRL2_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_082C	CCM PLL Control Register (CCM_PLL_CTRL2_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0830	CCM PLL Control Register (CCM_PLL_CTRL3)	32	R/W	0000_0002h	5.1.7.2/452
3038_0834	CCM PLL Control Register (CCM_PLL_CTRL3_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0838	CCM PLL Control Register (CCM_PLL_CTRL3_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_083C	CCM PLL Control Register (CCM_PLL_CTRL3_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0840	CCM PLL Control Register (CCM_PLL_CTRL4)	32	R/W	0000_0002h	5.1.7.2/452
3038_0844	CCM PLL Control Register (CCM_PLL_CTRL4_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0848	CCM PLL Control Register (CCM_PLL_CTRL4_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_084C	CCM PLL Control Register (CCM_PLL_CTRL4_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0850	CCM PLL Control Register (CCM_PLL_CTRL5)	32	R/W	0000_0002h	5.1.7.2/452
3038_0854	CCM PLL Control Register (CCM_PLL_CTRL5_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0858	CCM PLL Control Register (CCM_PLL_CTRL5_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_085C	CCM PLL Control Register (CCM_PLL_CTRL5_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0860	CCM PLL Control Register (CCM_PLL_CTRL6)	32	R/W	0000_0002h	5.1.7.2/452
3038_0864	CCM PLL Control Register (CCM_PLL_CTRL6_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0868	CCM PLL Control Register (CCM_PLL_CTRL6_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_086C	CCM PLL Control Register (CCM_PLL_CTRL6_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0870	CCM PLL Control Register (CCM_PLL_CTRL7)	32	R/W	0000_0002h	5.1.7.2/452
3038_0874	CCM PLL Control Register (CCM_PLL_CTRL7_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0878	CCM PLL Control Register (CCM_PLL_CTRL7_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_087C	CCM PLL Control Register (CCM_PLL_CTRL7_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0880	CCM PLL Control Register (CCM_PLL_CTRL8)	32	R/W	0000_0002h	5.1.7.2/452
3038_0884	CCM PLL Control Register (CCM_PLL_CTRL8_SET)	32	R/W	0000_0002h	5.1.7.3/454

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_0888	CCM PLL Control Register (CCM_PLL_CTRL8_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_088C	CCM PLL Control Register (CCM_PLL_CTRL8_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0890	CCM PLL Control Register (CCM_PLL_CTRL9)	32	R/W	0000_0002h	5.1.7.2/452
3038_0894	CCM PLL Control Register (CCM_PLL_CTRL9_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0898	CCM PLL Control Register (CCM_PLL_CTRL9_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_089C	CCM PLL Control Register (CCM_PLL_CTRL9_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_08A0	CCM PLL Control Register (CCM_PLL_CTRL10)	32	R/W	0000_0002h	5.1.7.2/452
3038_08A4	CCM PLL Control Register (CCM_PLL_CTRL10_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_08A8	CCM PLL Control Register (CCM_PLL_CTRL10_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_08AC	CCM PLL Control Register (CCM_PLL_CTRL10_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_08B0	CCM PLL Control Register (CCM_PLL_CTRL11)	32	R/W	0000_0002h	5.1.7.2/452
3038_08B4	CCM PLL Control Register (CCM_PLL_CTRL11_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_08B8	CCM PLL Control Register (CCM_PLL_CTRL11_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_08BC	CCM PLL Control Register (CCM_PLL_CTRL11_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_08C0	CCM PLL Control Register (CCM_PLL_CTRL12)	32	R/W	0000_0002h	5.1.7.2/452
3038_08C4	CCM PLL Control Register (CCM_PLL_CTRL12_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_08C8	CCM PLL Control Register (CCM_PLL_CTRL12_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_08CC	CCM PLL Control Register (CCM_PLL_CTRL12_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_08D0	CCM PLL Control Register (CCM_PLL_CTRL13)	32	R/W	0000_0002h	5.1.7.2/452
3038_08D4	CCM PLL Control Register (CCM_PLL_CTRL13_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_08D8	CCM PLL Control Register (CCM_PLL_CTRL13_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_08DC	CCM PLL Control Register (CCM_PLL_CTRL13_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_08E0	CCM PLL Control Register (CCM_PLL_CTRL14)	32	R/W	0000_0002h	5.1.7.2/452
3038_08E4	CCM PLL Control Register (CCM_PLL_CTRL14_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_08E8	CCM PLL Control Register (CCM_PLL_CTRL14_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_08EC	CCM PLL Control Register (CCM_PLL_CTRL14_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_08F0	CCM PLL Control Register (CCM_PLL_CTRL15)	32	R/W	0000_0002h	5.1.7.2/452
3038_08F4	CCM PLL Control Register (CCM_PLL_CTRL15_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_08F8	CCM PLL Control Register (CCM_PLL_CTRL15_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_08FC	CCM PLL Control Register (CCM_PLL_CTRL15_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0900	CCM PLL Control Register (CCM_PLL_CTRL16)	32	R/W	0000_0002h	5.1.7.2/452
3038_0904	CCM PLL Control Register (CCM_PLL_CTRL16_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0908	CCM PLL Control Register (CCM_PLL_CTRL16_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_090C	CCM PLL Control Register (CCM_PLL_CTRL16_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0910	CCM PLL Control Register (CCM_PLL_CTRL17)	32	R/W	0000_0002h	5.1.7.2/452
3038_0914	CCM PLL Control Register (CCM_PLL_CTRL17_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0918	CCM PLL Control Register (CCM_PLL_CTRL17_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_091C	CCM PLL Control Register (CCM_PLL_CTRL17_TOG)	32	R/W	0000_0002h	5.1.7.5/458

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_0920	CCM PLL Control Register (CCM_PLL_CTRL18)	32	R/W	0000_0002h	5.1.7.2/452
3038_0924	CCM PLL Control Register (CCM_PLL_CTRL18_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0928	CCM PLL Control Register (CCM_PLL_CTRL18_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_092C	CCM PLL Control Register (CCM_PLL_CTRL18_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0930	CCM PLL Control Register (CCM_PLL_CTRL19)	32	R/W	0000_0002h	5.1.7.2/452
3038_0934	CCM PLL Control Register (CCM_PLL_CTRL19_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0938	CCM PLL Control Register (CCM_PLL_CTRL19_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_093C	CCM PLL Control Register (CCM_PLL_CTRL19_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0940	CCM PLL Control Register (CCM_PLL_CTRL20)	32	R/W	0000_0002h	5.1.7.2/452
3038_0944	CCM PLL Control Register (CCM_PLL_CTRL20_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0948	CCM PLL Control Register (CCM_PLL_CTRL20_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_094C	CCM PLL Control Register (CCM_PLL_CTRL20_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0950	CCM PLL Control Register (CCM_PLL_CTRL21)	32	R/W	0000_0002h	5.1.7.2/452
3038_0954	CCM PLL Control Register (CCM_PLL_CTRL21_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0958	CCM PLL Control Register (CCM_PLL_CTRL21_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_095C	CCM PLL Control Register (CCM_PLL_CTRL21_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0960	CCM PLL Control Register (CCM_PLL_CTRL22)	32	R/W	0000_0002h	5.1.7.2/452
3038_0964	CCM PLL Control Register (CCM_PLL_CTRL22_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0968	CCM PLL Control Register (CCM_PLL_CTRL22_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_096C	CCM PLL Control Register (CCM_PLL_CTRL22_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0970	CCM PLL Control Register (CCM_PLL_CTRL23)	32	R/W	0000_0002h	5.1.7.2/452
3038_0974	CCM PLL Control Register (CCM_PLL_CTRL23_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0978	CCM PLL Control Register (CCM_PLL_CTRL23_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_097C	CCM PLL Control Register (CCM_PLL_CTRL23_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0980	CCM PLL Control Register (CCM_PLL_CTRL24)	32	R/W	0000_0002h	5.1.7.2/452
3038_0984	CCM PLL Control Register (CCM_PLL_CTRL24_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0988	CCM PLL Control Register (CCM_PLL_CTRL24_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_098C	CCM PLL Control Register (CCM_PLL_CTRL24_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0990	CCM PLL Control Register (CCM_PLL_CTRL25)	32	R/W	0000_0002h	5.1.7.2/452
3038_0994	CCM PLL Control Register (CCM_PLL_CTRL25_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0998	CCM PLL Control Register (CCM_PLL_CTRL25_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_099C	CCM PLL Control Register (CCM_PLL_CTRL25_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_09A0	CCM PLL Control Register (CCM_PLL_CTRL26)	32	R/W	0000_0002h	5.1.7.2/452
3038_09A4	CCM PLL Control Register (CCM_PLL_CTRL26_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_09A8	CCM PLL Control Register (CCM_PLL_CTRL26_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_09AC	CCM PLL Control Register (CCM_PLL_CTRL26_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_09B0	CCM PLL Control Register (CCM_PLL_CTRL27)	32	R/W	0000_0002h	5.1.7.2/452
3038_09B4	CCM PLL Control Register (CCM_PLL_CTRL27_SET)	32	R/W	0000_0002h	5.1.7.3/454

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_09B8	CCM PLL Control Register (CCM_PLL_CTRL27_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_09BC	CCM PLL Control Register (CCM_PLL_CTRL27_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_09C0	CCM PLL Control Register (CCM_PLL_CTRL28)	32	R/W	0000_0002h	5.1.7.2/452
3038_09C4	CCM PLL Control Register (CCM_PLL_CTRL28_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_09C8	CCM PLL Control Register (CCM_PLL_CTRL28_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_09CC	CCM PLL Control Register (CCM_PLL_CTRL28_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_09D0	CCM PLL Control Register (CCM_PLL_CTRL29)	32	R/W	0000_0002h	5.1.7.2/452
3038_09D4	CCM PLL Control Register (CCM_PLL_CTRL29_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_09D8	CCM PLL Control Register (CCM_PLL_CTRL29_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_09DC	CCM PLL Control Register (CCM_PLL_CTRL29_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_09E0	CCM PLL Control Register (CCM_PLL_CTRL30)	32	R/W	0000_0002h	5.1.7.2/452
3038_09E4	CCM PLL Control Register (CCM_PLL_CTRL30_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_09E8	CCM PLL Control Register (CCM_PLL_CTRL30_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_09EC	CCM PLL Control Register (CCM_PLL_CTRL30_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_09F0	CCM PLL Control Register (CCM_PLL_CTRL31)	32	R/W	0000_0002h	5.1.7.2/452
3038_09F4	CCM PLL Control Register (CCM_PLL_CTRL31_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_09F8	CCM PLL Control Register (CCM_PLL_CTRL31_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_09FC	CCM PLL Control Register (CCM_PLL_CTRL31_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0A00	CCM PLL Control Register (CCM_PLL_CTRL32)	32	R/W	0000_0002h	5.1.7.2/452
3038_0A04	CCM PLL Control Register (CCM_PLL_CTRL32_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0A08	CCM PLL Control Register (CCM_PLL_CTRL32_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_0A0C	CCM PLL Control Register (CCM_PLL_CTRL32_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0A10	CCM PLL Control Register (CCM_PLL_CTRL33)	32	R/W	0000_0002h	5.1.7.2/452
3038_0A14	CCM PLL Control Register (CCM_PLL_CTRL33_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0A18	CCM PLL Control Register (CCM_PLL_CTRL33_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_0A1C	CCM PLL Control Register (CCM_PLL_CTRL33_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0A20	CCM PLL Control Register (CCM_PLL_CTRL34)	32	R/W	0000_0002h	5.1.7.2/452
3038_0A24	CCM PLL Control Register (CCM_PLL_CTRL34_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0A28	CCM PLL Control Register (CCM_PLL_CTRL34_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_0A2C	CCM PLL Control Register (CCM_PLL_CTRL34_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0A30	CCM PLL Control Register (CCM_PLL_CTRL35)	32	R/W	0000_0002h	5.1.7.2/452
3038_0A34	CCM PLL Control Register (CCM_PLL_CTRL35_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0A38	CCM PLL Control Register (CCM_PLL_CTRL35_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_0A3C	CCM PLL Control Register (CCM_PLL_CTRL35_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0A40	CCM PLL Control Register (CCM_PLL_CTRL36)	32	R/W	0000_0002h	5.1.7.2/452
3038_0A44	CCM PLL Control Register (CCM_PLL_CTRL36_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0A48	CCM PLL Control Register (CCM_PLL_CTRL36_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_0A4C	CCM PLL Control Register (CCM_PLL_CTRL36_TOG)	32	R/W	0000_0002h	5.1.7.5/458

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_0A50	CCM PLL Control Register (CCM_PLL_CTRL37)	32	R/W	0000_0002h	5.1.7.2/452
3038_0A54	CCM PLL Control Register (CCM_PLL_CTRL37_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0A58	CCM PLL Control Register (CCM_PLL_CTRL37_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_0A5C	CCM PLL Control Register (CCM_PLL_CTRL37_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_0A60	CCM PLL Control Register (CCM_PLL_CTRL38)	32	R/W	0000_0002h	5.1.7.2/452
3038_0A64	CCM PLL Control Register (CCM_PLL_CTRL38_SET)	32	R/W	0000_0002h	5.1.7.3/454
3038_0A68	CCM PLL Control Register (CCM_PLL_CTRL38_CLR)	32	R/W	0000_0002h	5.1.7.4/456
3038_0A6C	CCM PLL Control Register (CCM_PLL_CTRL38_TOG)	32	R/W	0000_0002h	5.1.7.5/458
3038_4000	CCM Clock Gating Register (CCM_CCGR0)	32	R/W	0000_0002h	5.1.7.6/459
3038_4004	CCM Clock Gating Register (CCM_CCGR0_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4008	CCM Clock Gating Register (CCM_CCGR0_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_400C	CCM Clock Gating Register (CCM_CCGR0_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4010	CCM Clock Gating Register (CCM_CCGR1)	32	R/W	0000_0002h	5.1.7.6/459
3038_4014	CCM Clock Gating Register (CCM_CCGR1_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4018	CCM Clock Gating Register (CCM_CCGR1_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_401C	CCM Clock Gating Register (CCM_CCGR1_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4020	CCM Clock Gating Register (CCM_CCGR2)	32	R/W	0000_0002h	5.1.7.6/459
3038_4024	CCM Clock Gating Register (CCM_CCGR2_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4028	CCM Clock Gating Register (CCM_CCGR2_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_402C	CCM Clock Gating Register (CCM_CCGR2_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4030	CCM Clock Gating Register (CCM_CCGR3)	32	R/W	0000_0002h	5.1.7.6/459
3038_4034	CCM Clock Gating Register (CCM_CCGR3_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4038	CCM Clock Gating Register (CCM_CCGR3_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_403C	CCM Clock Gating Register (CCM_CCGR3_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4040	CCM Clock Gating Register (CCM_CCGR4)	32	R/W	0000_0002h	5.1.7.6/459
3038_4044	CCM Clock Gating Register (CCM_CCGR4_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4048	CCM Clock Gating Register (CCM_CCGR4_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_404C	CCM Clock Gating Register (CCM_CCGR4_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4050	CCM Clock Gating Register (CCM_CCGR5)	32	R/W	0000_0002h	5.1.7.6/459
3038_4054	CCM Clock Gating Register (CCM_CCGR5_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4058	CCM Clock Gating Register (CCM_CCGR5_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_405C	CCM Clock Gating Register (CCM_CCGR5_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4060	CCM Clock Gating Register (CCM_CCGR6)	32	R/W	0000_0002h	5.1.7.6/459
3038_4064	CCM Clock Gating Register (CCM_CCGR6_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4068	CCM Clock Gating Register (CCM_CCGR6_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_406C	CCM Clock Gating Register (CCM_CCGR6_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4070	CCM Clock Gating Register (CCM_CCGR7)	32	R/W	0000_0002h	5.1.7.6/459
3038_4074	CCM Clock Gating Register (CCM_CCGR7_SET)	32	R/W	0000_0002h	5.1.7.7/462

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4078	CCM Clock Gating Register (CCM_CCGR7_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_407C	CCM Clock Gating Register (CCM_CCGR7_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4080	CCM Clock Gating Register (CCM_CCGR8)	32	R/W	0000_0002h	5.1.7.6/459
3038_4084	CCM Clock Gating Register (CCM_CCGR8_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4088	CCM Clock Gating Register (CCM_CCGR8_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_408C	CCM Clock Gating Register (CCM_CCGR8_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4090	CCM Clock Gating Register (CCM_CCGR9)	32	R/W	0000_0002h	5.1.7.6/459
3038_4094	CCM Clock Gating Register (CCM_CCGR9_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4098	CCM Clock Gating Register (CCM_CCGR9_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_409C	CCM Clock Gating Register (CCM_CCGR9_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_40A0	CCM Clock Gating Register (CCM_CCGR10)	32	R/W	0000_0002h	5.1.7.6/459
3038_40A4	CCM Clock Gating Register (CCM_CCGR10_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_40A8	CCM Clock Gating Register (CCM_CCGR10_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_40AC	CCM Clock Gating Register (CCM_CCGR10_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_40B0	CCM Clock Gating Register (CCM_CCGR11)	32	R/W	0000_0002h	5.1.7.6/459
3038_40B4	CCM Clock Gating Register (CCM_CCGR11_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_40B8	CCM Clock Gating Register (CCM_CCGR11_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_40BC	CCM Clock Gating Register (CCM_CCGR11_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_40C0	CCM Clock Gating Register (CCM_CCGR12)	32	R/W	0000_0002h	5.1.7.6/459
3038_40C4	CCM Clock Gating Register (CCM_CCGR12_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_40C8	CCM Clock Gating Register (CCM_CCGR12_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_40CC	CCM Clock Gating Register (CCM_CCGR12_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_40D0	CCM Clock Gating Register (CCM_CCGR13)	32	R/W	0000_0002h	5.1.7.6/459
3038_40D4	CCM Clock Gating Register (CCM_CCGR13_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_40D8	CCM Clock Gating Register (CCM_CCGR13_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_40DC	CCM Clock Gating Register (CCM_CCGR13_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_40E0	CCM Clock Gating Register (CCM_CCGR14)	32	R/W	0000_0002h	5.1.7.6/459
3038_40E4	CCM Clock Gating Register (CCM_CCGR14_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_40E8	CCM Clock Gating Register (CCM_CCGR14_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_40EC	CCM Clock Gating Register (CCM_CCGR14_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_40F0	CCM Clock Gating Register (CCM_CCGR15)	32	R/W	0000_0002h	5.1.7.6/459
3038_40F4	CCM Clock Gating Register (CCM_CCGR15_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_40F8	CCM Clock Gating Register (CCM_CCGR15_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_40FC	CCM Clock Gating Register (CCM_CCGR15_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4100	CCM Clock Gating Register (CCM_CCGR16)	32	R/W	0000_0002h	5.1.7.6/459
3038_4104	CCM Clock Gating Register (CCM_CCGR16_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4108	CCM Clock Gating Register (CCM_CCGR16_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_410C	CCM Clock Gating Register (CCM_CCGR16_TOG)	32	R/W	0000_0002h	5.1.7.9/466

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4110	CCM Clock Gating Register (CCM_CCGR17)	32	R/W	0000_0002h	5.1.7.6/459
3038_4114	CCM Clock Gating Register (CCM_CCGR17_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4118	CCM Clock Gating Register (CCM_CCGR17_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_411C	CCM Clock Gating Register (CCM_CCGR17_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4120	CCM Clock Gating Register (CCM_CCGR18)	32	R/W	0000_0002h	5.1.7.6/459
3038_4124	CCM Clock Gating Register (CCM_CCGR18_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4128	CCM Clock Gating Register (CCM_CCGR18_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_412C	CCM Clock Gating Register (CCM_CCGR18_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4130	CCM Clock Gating Register (CCM_CCGR19)	32	R/W	0000_0002h	5.1.7.6/459
3038_4134	CCM Clock Gating Register (CCM_CCGR19_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4138	CCM Clock Gating Register (CCM_CCGR19_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_413C	CCM Clock Gating Register (CCM_CCGR19_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4140	CCM Clock Gating Register (CCM_CCGR20)	32	R/W	0000_0002h	5.1.7.6/459
3038_4144	CCM Clock Gating Register (CCM_CCGR20_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4148	CCM Clock Gating Register (CCM_CCGR20_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_414C	CCM Clock Gating Register (CCM_CCGR20_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4150	CCM Clock Gating Register (CCM_CCGR21)	32	R/W	0000_0002h	5.1.7.6/459
3038_4154	CCM Clock Gating Register (CCM_CCGR21_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4158	CCM Clock Gating Register (CCM_CCGR21_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_415C	CCM Clock Gating Register (CCM_CCGR21_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4160	CCM Clock Gating Register (CCM_CCGR22)	32	R/W	0000_0002h	5.1.7.6/459
3038_4164	CCM Clock Gating Register (CCM_CCGR22_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4168	CCM Clock Gating Register (CCM_CCGR22_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_416C	CCM Clock Gating Register (CCM_CCGR22_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4170	CCM Clock Gating Register (CCM_CCGR23)	32	R/W	0000_0002h	5.1.7.6/459
3038_4174	CCM Clock Gating Register (CCM_CCGR23_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4178	CCM Clock Gating Register (CCM_CCGR23_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_417C	CCM Clock Gating Register (CCM_CCGR23_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4180	CCM Clock Gating Register (CCM_CCGR24)	32	R/W	0000_0002h	5.1.7.6/459
3038_4184	CCM Clock Gating Register (CCM_CCGR24_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4188	CCM Clock Gating Register (CCM_CCGR24_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_418C	CCM Clock Gating Register (CCM_CCGR24_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4190	CCM Clock Gating Register (CCM_CCGR25)	32	R/W	0000_0002h	5.1.7.6/459
3038_4194	CCM Clock Gating Register (CCM_CCGR25_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4198	CCM Clock Gating Register (CCM_CCGR25_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_419C	CCM Clock Gating Register (CCM_CCGR25_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_41A0	CCM Clock Gating Register (CCM_CCGR26)	32	R/W	0000_0002h	5.1.7.6/459
3038_41A4	CCM Clock Gating Register (CCM_CCGR26_SET)	32	R/W	0000_0002h	5.1.7.7/462

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_41A8	CCM Clock Gating Register (CCM_CCGR26_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_41AC	CCM Clock Gating Register (CCM_CCGR26_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_41B0	CCM Clock Gating Register (CCM_CCGR27)	32	R/W	0000_0002h	5.1.7.6/459
3038_41B4	CCM Clock Gating Register (CCM_CCGR27_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_41B8	CCM Clock Gating Register (CCM_CCGR27_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_41BC	CCM Clock Gating Register (CCM_CCGR27_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_41C0	CCM Clock Gating Register (CCM_CCGR28)	32	R/W	0000_0002h	5.1.7.6/459
3038_41C4	CCM Clock Gating Register (CCM_CCGR28_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_41C8	CCM Clock Gating Register (CCM_CCGR28_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_41CC	CCM Clock Gating Register (CCM_CCGR28_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_41D0	CCM Clock Gating Register (CCM_CCGR29)	32	R/W	0000_0002h	5.1.7.6/459
3038_41D4	CCM Clock Gating Register (CCM_CCGR29_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_41D8	CCM Clock Gating Register (CCM_CCGR29_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_41DC	CCM Clock Gating Register (CCM_CCGR29_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_41E0	CCM Clock Gating Register (CCM_CCGR30)	32	R/W	0000_0002h	5.1.7.6/459
3038_41E4	CCM Clock Gating Register (CCM_CCGR30_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_41E8	CCM Clock Gating Register (CCM_CCGR30_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_41EC	CCM Clock Gating Register (CCM_CCGR30_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_41F0	CCM Clock Gating Register (CCM_CCGR31)	32	R/W	0000_0002h	5.1.7.6/459
3038_41F4	CCM Clock Gating Register (CCM_CCGR31_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_41F8	CCM Clock Gating Register (CCM_CCGR31_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_41FC	CCM Clock Gating Register (CCM_CCGR31_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4200	CCM Clock Gating Register (CCM_CCGR32)	32	R/W	0000_0002h	5.1.7.6/459
3038_4204	CCM Clock Gating Register (CCM_CCGR32_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4208	CCM Clock Gating Register (CCM_CCGR32_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_420C	CCM Clock Gating Register (CCM_CCGR32_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4210	CCM Clock Gating Register (CCM_CCGR33)	32	R/W	0000_0002h	5.1.7.6/459
3038_4214	CCM Clock Gating Register (CCM_CCGR33_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4218	CCM Clock Gating Register (CCM_CCGR33_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_421C	CCM Clock Gating Register (CCM_CCGR33_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4220	CCM Clock Gating Register (CCM_CCGR34)	32	R/W	0000_0002h	5.1.7.6/459
3038_4224	CCM Clock Gating Register (CCM_CCGR34_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4228	CCM Clock Gating Register (CCM_CCGR34_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_422C	CCM Clock Gating Register (CCM_CCGR34_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4230	CCM Clock Gating Register (CCM_CCGR35)	32	R/W	0000_0002h	5.1.7.6/459
3038_4234	CCM Clock Gating Register (CCM_CCGR35_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4238	CCM Clock Gating Register (CCM_CCGR35_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_423C	CCM Clock Gating Register (CCM_CCGR35_TOG)	32	R/W	0000_0002h	5.1.7.9/466

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4240	CCM Clock Gating Register (CCM_CCGR36)	32	R/W	0000_0002h	5.1.7.6/459
3038_4244	CCM Clock Gating Register (CCM_CCGR36_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4248	CCM Clock Gating Register (CCM_CCGR36_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_424C	CCM Clock Gating Register (CCM_CCGR36_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4250	CCM Clock Gating Register (CCM_CCGR37)	32	R/W	0000_0002h	5.1.7.6/459
3038_4254	CCM Clock Gating Register (CCM_CCGR37_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4258	CCM Clock Gating Register (CCM_CCGR37_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_425C	CCM Clock Gating Register (CCM_CCGR37_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4260	CCM Clock Gating Register (CCM_CCGR38)	32	R/W	0000_0002h	5.1.7.6/459
3038_4264	CCM Clock Gating Register (CCM_CCGR38_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4268	CCM Clock Gating Register (CCM_CCGR38_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_426C	CCM Clock Gating Register (CCM_CCGR38_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4270	CCM Clock Gating Register (CCM_CCGR39)	32	R/W	0000_0002h	5.1.7.6/459
3038_4274	CCM Clock Gating Register (CCM_CCGR39_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4278	CCM Clock Gating Register (CCM_CCGR39_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_427C	CCM Clock Gating Register (CCM_CCGR39_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4280	CCM Clock Gating Register (CCM_CCGR40)	32	R/W	0000_0002h	5.1.7.6/459
3038_4284	CCM Clock Gating Register (CCM_CCGR40_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4288	CCM Clock Gating Register (CCM_CCGR40_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_428C	CCM Clock Gating Register (CCM_CCGR40_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4290	CCM Clock Gating Register (CCM_CCGR41)	32	R/W	0000_0002h	5.1.7.6/459
3038_4294	CCM Clock Gating Register (CCM_CCGR41_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4298	CCM Clock Gating Register (CCM_CCGR41_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_429C	CCM Clock Gating Register (CCM_CCGR41_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_42A0	CCM Clock Gating Register (CCM_CCGR42)	32	R/W	0000_0002h	5.1.7.6/459
3038_42A4	CCM Clock Gating Register (CCM_CCGR42_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_42A8	CCM Clock Gating Register (CCM_CCGR42_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_42AC	CCM Clock Gating Register (CCM_CCGR42_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_42B0	CCM Clock Gating Register (CCM_CCGR43)	32	R/W	0000_0002h	5.1.7.6/459
3038_42B4	CCM Clock Gating Register (CCM_CCGR43_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_42B8	CCM Clock Gating Register (CCM_CCGR43_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_42BC	CCM Clock Gating Register (CCM_CCGR43_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_42C0	CCM Clock Gating Register (CCM_CCGR44)	32	R/W	0000_0002h	5.1.7.6/459
3038_42C4	CCM Clock Gating Register (CCM_CCGR44_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_42C8	CCM Clock Gating Register (CCM_CCGR44_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_42CC	CCM Clock Gating Register (CCM_CCGR44_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_42D0	CCM Clock Gating Register (CCM_CCGR45)	32	R/W	0000_0002h	5.1.7.6/459
3038_42D4	CCM Clock Gating Register (CCM_CCGR45_SET)	32	R/W	0000_0002h	5.1.7.7/462

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_42D8	CCM Clock Gating Register (CCM_CCGR45_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_42DC	CCM Clock Gating Register (CCM_CCGR45_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_42E0	CCM Clock Gating Register (CCM_CCGR46)	32	R/W	0000_0002h	5.1.7.6/459
3038_42E4	CCM Clock Gating Register (CCM_CCGR46_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_42E8	CCM Clock Gating Register (CCM_CCGR46_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_42EC	CCM Clock Gating Register (CCM_CCGR46_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_42F0	CCM Clock Gating Register (CCM_CCGR47)	32	R/W	0000_0002h	5.1.7.6/459
3038_42F4	CCM Clock Gating Register (CCM_CCGR47_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_42F8	CCM Clock Gating Register (CCM_CCGR47_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_42FC	CCM Clock Gating Register (CCM_CCGR47_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4300	CCM Clock Gating Register (CCM_CCGR48)	32	R/W	0000_0002h	5.1.7.6/459
3038_4304	CCM Clock Gating Register (CCM_CCGR48_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4308	CCM Clock Gating Register (CCM_CCGR48_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_430C	CCM Clock Gating Register (CCM_CCGR48_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4310	CCM Clock Gating Register (CCM_CCGR49)	32	R/W	0000_0002h	5.1.7.6/459
3038_4314	CCM Clock Gating Register (CCM_CCGR49_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4318	CCM Clock Gating Register (CCM_CCGR49_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_431C	CCM Clock Gating Register (CCM_CCGR49_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4320	CCM Clock Gating Register (CCM_CCGR50)	32	R/W	0000_0002h	5.1.7.6/459
3038_4324	CCM Clock Gating Register (CCM_CCGR50_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4328	CCM Clock Gating Register (CCM_CCGR50_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_432C	CCM Clock Gating Register (CCM_CCGR50_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4330	CCM Clock Gating Register (CCM_CCGR51)	32	R/W	0000_0002h	5.1.7.6/459
3038_4334	CCM Clock Gating Register (CCM_CCGR51_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4338	CCM Clock Gating Register (CCM_CCGR51_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_433C	CCM Clock Gating Register (CCM_CCGR51_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4340	CCM Clock Gating Register (CCM_CCGR52)	32	R/W	0000_0002h	5.1.7.6/459
3038_4344	CCM Clock Gating Register (CCM_CCGR52_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4348	CCM Clock Gating Register (CCM_CCGR52_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_434C	CCM Clock Gating Register (CCM_CCGR52_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4350	CCM Clock Gating Register (CCM_CCGR53)	32	R/W	0000_0002h	5.1.7.6/459
3038_4354	CCM Clock Gating Register (CCM_CCGR53_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4358	CCM Clock Gating Register (CCM_CCGR53_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_435C	CCM Clock Gating Register (CCM_CCGR53_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4360	CCM Clock Gating Register (CCM_CCGR54)	32	R/W	0000_0002h	5.1.7.6/459
3038_4364	CCM Clock Gating Register (CCM_CCGR54_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4368	CCM Clock Gating Register (CCM_CCGR54_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_436C	CCM Clock Gating Register (CCM_CCGR54_TOG)	32	R/W	0000_0002h	5.1.7.9/466

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4370	CCM Clock Gating Register (CCM_CCGR55)	32	R/W	0000_0002h	5.1.7.6/459
3038_4374	CCM Clock Gating Register (CCM_CCGR55_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4378	CCM Clock Gating Register (CCM_CCGR55_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_437C	CCM Clock Gating Register (CCM_CCGR55_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4380	CCM Clock Gating Register (CCM_CCGR56)	32	R/W	0000_0002h	5.1.7.6/459
3038_4384	CCM Clock Gating Register (CCM_CCGR56_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4388	CCM Clock Gating Register (CCM_CCGR56_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_438C	CCM Clock Gating Register (CCM_CCGR56_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4390	CCM Clock Gating Register (CCM_CCGR57)	32	R/W	0000_0002h	5.1.7.6/459
3038_4394	CCM Clock Gating Register (CCM_CCGR57_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4398	CCM Clock Gating Register (CCM_CCGR57_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_439C	CCM Clock Gating Register (CCM_CCGR57_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_43A0	CCM Clock Gating Register (CCM_CCGR58)	32	R/W	0000_0002h	5.1.7.6/459
3038_43A4	CCM Clock Gating Register (CCM_CCGR58_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_43A8	CCM Clock Gating Register (CCM_CCGR58_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_43AC	CCM Clock Gating Register (CCM_CCGR58_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_43B0	CCM Clock Gating Register (CCM_CCGR59)	32	R/W	0000_0002h	5.1.7.6/459
3038_43B4	CCM Clock Gating Register (CCM_CCGR59_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_43B8	CCM Clock Gating Register (CCM_CCGR59_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_43BC	CCM Clock Gating Register (CCM_CCGR59_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_43C0	CCM Clock Gating Register (CCM_CCGR60)	32	R/W	0000_0002h	5.1.7.6/459
3038_43C4	CCM Clock Gating Register (CCM_CCGR60_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_43C8	CCM Clock Gating Register (CCM_CCGR60_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_43CC	CCM Clock Gating Register (CCM_CCGR60_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_43D0	CCM Clock Gating Register (CCM_CCGR61)	32	R/W	0000_0002h	5.1.7.6/459
3038_43D4	CCM Clock Gating Register (CCM_CCGR61_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_43D8	CCM Clock Gating Register (CCM_CCGR61_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_43DC	CCM Clock Gating Register (CCM_CCGR61_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_43E0	CCM Clock Gating Register (CCM_CCGR62)	32	R/W	0000_0002h	5.1.7.6/459
3038_43E4	CCM Clock Gating Register (CCM_CCGR62_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_43E8	CCM Clock Gating Register (CCM_CCGR62_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_43EC	CCM Clock Gating Register (CCM_CCGR62_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_43F0	CCM Clock Gating Register (CCM_CCGR63)	32	R/W	0000_0002h	5.1.7.6/459
3038_43F4	CCM Clock Gating Register (CCM_CCGR63_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_43F8	CCM Clock Gating Register (CCM_CCGR63_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_43FC	CCM Clock Gating Register (CCM_CCGR63_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4400	CCM Clock Gating Register (CCM_CCGR64)	32	R/W	0000_0002h	5.1.7.6/459
3038_4404	CCM Clock Gating Register (CCM_CCGR64_SET)	32	R/W	0000_0002h	5.1.7.7/462

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4408	CCM Clock Gating Register (CCM_CCGR64_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_440C	CCM Clock Gating Register (CCM_CCGR64_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4410	CCM Clock Gating Register (CCM_CCGR65)	32	R/W	0000_0002h	5.1.7.6/459
3038_4414	CCM Clock Gating Register (CCM_CCGR65_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4418	CCM Clock Gating Register (CCM_CCGR65_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_441C	CCM Clock Gating Register (CCM_CCGR65_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4420	CCM Clock Gating Register (CCM_CCGR66)	32	R/W	0000_0002h	5.1.7.6/459
3038_4424	CCM Clock Gating Register (CCM_CCGR66_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4428	CCM Clock Gating Register (CCM_CCGR66_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_442C	CCM Clock Gating Register (CCM_CCGR66_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4430	CCM Clock Gating Register (CCM_CCGR67)	32	R/W	0000_0002h	5.1.7.6/459
3038_4434	CCM Clock Gating Register (CCM_CCGR67_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4438	CCM Clock Gating Register (CCM_CCGR67_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_443C	CCM Clock Gating Register (CCM_CCGR67_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4440	CCM Clock Gating Register (CCM_CCGR68)	32	R/W	0000_0002h	5.1.7.6/459
3038_4444	CCM Clock Gating Register (CCM_CCGR68_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4448	CCM Clock Gating Register (CCM_CCGR68_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_444C	CCM Clock Gating Register (CCM_CCGR68_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4450	CCM Clock Gating Register (CCM_CCGR69)	32	R/W	0000_0002h	5.1.7.6/459
3038_4454	CCM Clock Gating Register (CCM_CCGR69_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4458	CCM Clock Gating Register (CCM_CCGR69_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_445C	CCM Clock Gating Register (CCM_CCGR69_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4460	CCM Clock Gating Register (CCM_CCGR70)	32	R/W	0000_0002h	5.1.7.6/459
3038_4464	CCM Clock Gating Register (CCM_CCGR70_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4468	CCM Clock Gating Register (CCM_CCGR70_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_446C	CCM Clock Gating Register (CCM_CCGR70_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4470	CCM Clock Gating Register (CCM_CCGR71)	32	R/W	0000_0002h	5.1.7.6/459
3038_4474	CCM Clock Gating Register (CCM_CCGR71_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4478	CCM Clock Gating Register (CCM_CCGR71_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_447C	CCM Clock Gating Register (CCM_CCGR71_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4480	CCM Clock Gating Register (CCM_CCGR72)	32	R/W	0000_0002h	5.1.7.6/459
3038_4484	CCM Clock Gating Register (CCM_CCGR72_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4488	CCM Clock Gating Register (CCM_CCGR72_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_448C	CCM Clock Gating Register (CCM_CCGR72_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4490	CCM Clock Gating Register (CCM_CCGR73)	32	R/W	0000_0002h	5.1.7.6/459
3038_4494	CCM Clock Gating Register (CCM_CCGR73_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4498	CCM Clock Gating Register (CCM_CCGR73_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_449C	CCM Clock Gating Register (CCM_CCGR73_TOG)	32	R/W	0000_0002h	5.1.7.9/466

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_44A0	CCM Clock Gating Register (CCM_CCGR74)	32	R/W	0000_0002h	5.1.7.6/459
3038_44A4	CCM Clock Gating Register (CCM_CCGR74_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_44A8	CCM Clock Gating Register (CCM_CCGR74_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_44AC	CCM Clock Gating Register (CCM_CCGR74_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_44B0	CCM Clock Gating Register (CCM_CCGR75)	32	R/W	0000_0002h	5.1.7.6/459
3038_44B4	CCM Clock Gating Register (CCM_CCGR75_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_44B8	CCM Clock Gating Register (CCM_CCGR75_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_44BC	CCM Clock Gating Register (CCM_CCGR75_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_44C0	CCM Clock Gating Register (CCM_CCGR76)	32	R/W	0000_0002h	5.1.7.6/459
3038_44C4	CCM Clock Gating Register (CCM_CCGR76_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_44C8	CCM Clock Gating Register (CCM_CCGR76_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_44CC	CCM Clock Gating Register (CCM_CCGR76_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_44D0	CCM Clock Gating Register (CCM_CCGR77)	32	R/W	0000_0002h	5.1.7.6/459
3038_44D4	CCM Clock Gating Register (CCM_CCGR77_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_44D8	CCM Clock Gating Register (CCM_CCGR77_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_44DC	CCM Clock Gating Register (CCM_CCGR77_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_44E0	CCM Clock Gating Register (CCM_CCGR78)	32	R/W	0000_0002h	5.1.7.6/459
3038_44E4	CCM Clock Gating Register (CCM_CCGR78_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_44E8	CCM Clock Gating Register (CCM_CCGR78_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_44EC	CCM Clock Gating Register (CCM_CCGR78_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_44F0	CCM Clock Gating Register (CCM_CCGR79)	32	R/W	0000_0002h	5.1.7.6/459
3038_44F4	CCM Clock Gating Register (CCM_CCGR79_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_44F8	CCM Clock Gating Register (CCM_CCGR79_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_44FC	CCM Clock Gating Register (CCM_CCGR79_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4500	CCM Clock Gating Register (CCM_CCGR80)	32	R/W	0000_0002h	5.1.7.6/459
3038_4504	CCM Clock Gating Register (CCM_CCGR80_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4508	CCM Clock Gating Register (CCM_CCGR80_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_450C	CCM Clock Gating Register (CCM_CCGR80_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4510	CCM Clock Gating Register (CCM_CCGR81)	32	R/W	0000_0002h	5.1.7.6/459
3038_4514	CCM Clock Gating Register (CCM_CCGR81_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4518	CCM Clock Gating Register (CCM_CCGR81_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_451C	CCM Clock Gating Register (CCM_CCGR81_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4520	CCM Clock Gating Register (CCM_CCGR82)	32	R/W	0000_0002h	5.1.7.6/459
3038_4524	CCM Clock Gating Register (CCM_CCGR82_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4528	CCM Clock Gating Register (CCM_CCGR82_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_452C	CCM Clock Gating Register (CCM_CCGR82_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4530	CCM Clock Gating Register (CCM_CCGR83)	32	R/W	0000_0002h	5.1.7.6/459
3038_4534	CCM Clock Gating Register (CCM_CCGR83_SET)	32	R/W	0000_0002h	5.1.7.7/462

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4538	CCM Clock Gating Register (CCM_CCGR83_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_453C	CCM Clock Gating Register (CCM_CCGR83_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4540	CCM Clock Gating Register (CCM_CCGR84)	32	R/W	0000_0002h	5.1.7.6/459
3038_4544	CCM Clock Gating Register (CCM_CCGR84_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4548	CCM Clock Gating Register (CCM_CCGR84_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_454C	CCM Clock Gating Register (CCM_CCGR84_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4550	CCM Clock Gating Register (CCM_CCGR85)	32	R/W	0000_0002h	5.1.7.6/459
3038_4554	CCM Clock Gating Register (CCM_CCGR85_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4558	CCM Clock Gating Register (CCM_CCGR85_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_455C	CCM Clock Gating Register (CCM_CCGR85_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4560	CCM Clock Gating Register (CCM_CCGR86)	32	R/W	0000_0002h	5.1.7.6/459
3038_4564	CCM Clock Gating Register (CCM_CCGR86_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4568	CCM Clock Gating Register (CCM_CCGR86_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_456C	CCM Clock Gating Register (CCM_CCGR86_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4570	CCM Clock Gating Register (CCM_CCGR87)	32	R/W	0000_0002h	5.1.7.6/459
3038_4574	CCM Clock Gating Register (CCM_CCGR87_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4578	CCM Clock Gating Register (CCM_CCGR87_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_457C	CCM Clock Gating Register (CCM_CCGR87_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4580	CCM Clock Gating Register (CCM_CCGR88)	32	R/W	0000_0002h	5.1.7.6/459
3038_4584	CCM Clock Gating Register (CCM_CCGR88_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4588	CCM Clock Gating Register (CCM_CCGR88_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_458C	CCM Clock Gating Register (CCM_CCGR88_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4590	CCM Clock Gating Register (CCM_CCGR89)	32	R/W	0000_0002h	5.1.7.6/459
3038_4594	CCM Clock Gating Register (CCM_CCGR89_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4598	CCM Clock Gating Register (CCM_CCGR89_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_459C	CCM Clock Gating Register (CCM_CCGR89_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_45A0	CCM Clock Gating Register (CCM_CCGR90)	32	R/W	0000_0002h	5.1.7.6/459
3038_45A4	CCM Clock Gating Register (CCM_CCGR90_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_45A8	CCM Clock Gating Register (CCM_CCGR90_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_45AC	CCM Clock Gating Register (CCM_CCGR90_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_45B0	CCM Clock Gating Register (CCM_CCGR91)	32	R/W	0000_0002h	5.1.7.6/459
3038_45B4	CCM Clock Gating Register (CCM_CCGR91_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_45B8	CCM Clock Gating Register (CCM_CCGR91_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_45BC	CCM Clock Gating Register (CCM_CCGR91_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_45C0	CCM Clock Gating Register (CCM_CCGR92)	32	R/W	0000_0002h	5.1.7.6/459
3038_45C4	CCM Clock Gating Register (CCM_CCGR92_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_45C8	CCM Clock Gating Register (CCM_CCGR92_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_45CC	CCM Clock Gating Register (CCM_CCGR92_TOG)	32	R/W	0000_0002h	5.1.7.9/466

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_45D0	CCM Clock Gating Register (CCM_CCGR93)	32	R/W	0000_0002h	5.1.7.6/459
3038_45D4	CCM Clock Gating Register (CCM_CCGR93_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_45D8	CCM Clock Gating Register (CCM_CCGR93_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_45DC	CCM Clock Gating Register (CCM_CCGR93_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_45E0	CCM Clock Gating Register (CCM_CCGR94)	32	R/W	0000_0002h	5.1.7.6/459
3038_45E4	CCM Clock Gating Register (CCM_CCGR94_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_45E8	CCM Clock Gating Register (CCM_CCGR94_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_45EC	CCM Clock Gating Register (CCM_CCGR94_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_45F0	CCM Clock Gating Register (CCM_CCGR95)	32	R/W	0000_0002h	5.1.7.6/459
3038_45F4	CCM Clock Gating Register (CCM_CCGR95_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_45F8	CCM Clock Gating Register (CCM_CCGR95_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_45FC	CCM Clock Gating Register (CCM_CCGR95_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4600	CCM Clock Gating Register (CCM_CCGR96)	32	R/W	0000_0002h	5.1.7.6/459
3038_4604	CCM Clock Gating Register (CCM_CCGR96_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4608	CCM Clock Gating Register (CCM_CCGR96_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_460C	CCM Clock Gating Register (CCM_CCGR96_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4610	CCM Clock Gating Register (CCM_CCGR97)	32	R/W	0000_0002h	5.1.7.6/459
3038_4614	CCM Clock Gating Register (CCM_CCGR97_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4618	CCM Clock Gating Register (CCM_CCGR97_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_461C	CCM Clock Gating Register (CCM_CCGR97_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4620	CCM Clock Gating Register (CCM_CCGR98)	32	R/W	0000_0002h	5.1.7.6/459
3038_4624	CCM Clock Gating Register (CCM_CCGR98_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4628	CCM Clock Gating Register (CCM_CCGR98_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_462C	CCM Clock Gating Register (CCM_CCGR98_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4630	CCM Clock Gating Register (CCM_CCGR99)	32	R/W	0000_0002h	5.1.7.6/459
3038_4634	CCM Clock Gating Register (CCM_CCGR99_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4638	CCM Clock Gating Register (CCM_CCGR99_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_463C	CCM Clock Gating Register (CCM_CCGR99_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4640	CCM Clock Gating Register (CCM_CCGR100)	32	R/W	0000_0002h	5.1.7.6/459
3038_4644	CCM Clock Gating Register (CCM_CCGR100_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4648	CCM Clock Gating Register (CCM_CCGR100_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_464C	CCM Clock Gating Register (CCM_CCGR100_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4650	CCM Clock Gating Register (CCM_CCGR101)	32	R/W	0000_0002h	5.1.7.6/459
3038_4654	CCM Clock Gating Register (CCM_CCGR101_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4658	CCM Clock Gating Register (CCM_CCGR101_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_465C	CCM Clock Gating Register (CCM_CCGR101_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4660	CCM Clock Gating Register (CCM_CCGR102)	32	R/W	0000_0002h	5.1.7.6/459
3038_4664	CCM Clock Gating Register (CCM_CCGR102_SET)	32	R/W	0000_0002h	5.1.7.7/462

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4668	CCM Clock Gating Register (CCM_CCGR102_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_466C	CCM Clock Gating Register (CCM_CCGR102_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4670	CCM Clock Gating Register (CCM_CCGR103)	32	R/W	0000_0002h	5.1.7.6/459
3038_4674	CCM Clock Gating Register (CCM_CCGR103_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4678	CCM Clock Gating Register (CCM_CCGR103_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_467C	CCM Clock Gating Register (CCM_CCGR103_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4680	CCM Clock Gating Register (CCM_CCGR104)	32	R/W	0000_0002h	5.1.7.6/459
3038_4684	CCM Clock Gating Register (CCM_CCGR104_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4688	CCM Clock Gating Register (CCM_CCGR104_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_468C	CCM Clock Gating Register (CCM_CCGR104_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4690	CCM Clock Gating Register (CCM_CCGR105)	32	R/W	0000_0002h	5.1.7.6/459
3038_4694	CCM Clock Gating Register (CCM_CCGR105_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4698	CCM Clock Gating Register (CCM_CCGR105_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_469C	CCM Clock Gating Register (CCM_CCGR105_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_46A0	CCM Clock Gating Register (CCM_CCGR106)	32	R/W	0000_0002h	5.1.7.6/459
3038_46A4	CCM Clock Gating Register (CCM_CCGR106_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_46A8	CCM Clock Gating Register (CCM_CCGR106_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_46AC	CCM Clock Gating Register (CCM_CCGR106_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_46B0	CCM Clock Gating Register (CCM_CCGR107)	32	R/W	0000_0002h	5.1.7.6/459
3038_46B4	CCM Clock Gating Register (CCM_CCGR107_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_46B8	CCM Clock Gating Register (CCM_CCGR107_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_46BC	CCM Clock Gating Register (CCM_CCGR107_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_46C0	CCM Clock Gating Register (CCM_CCGR108)	32	R/W	0000_0002h	5.1.7.6/459
3038_46C4	CCM Clock Gating Register (CCM_CCGR108_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_46C8	CCM Clock Gating Register (CCM_CCGR108_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_46CC	CCM Clock Gating Register (CCM_CCGR108_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_46D0	CCM Clock Gating Register (CCM_CCGR109)	32	R/W	0000_0002h	5.1.7.6/459
3038_46D4	CCM Clock Gating Register (CCM_CCGR109_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_46D8	CCM Clock Gating Register (CCM_CCGR109_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_46DC	CCM Clock Gating Register (CCM_CCGR109_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_46E0	CCM Clock Gating Register (CCM_CCGR110)	32	R/W	0000_0002h	5.1.7.6/459
3038_46E4	CCM Clock Gating Register (CCM_CCGR110_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_46E8	CCM Clock Gating Register (CCM_CCGR110_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_46EC	CCM Clock Gating Register (CCM_CCGR110_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_46F0	CCM Clock Gating Register (CCM_CCGR111)	32	R/W	0000_0002h	5.1.7.6/459
3038_46F4	CCM Clock Gating Register (CCM_CCGR111_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_46F8	CCM Clock Gating Register (CCM_CCGR111_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_46FC	CCM Clock Gating Register (CCM_CCGR111_TOG)	32	R/W	0000_0002h	5.1.7.9/466

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4700	CCM Clock Gating Register (CCM_CCGR112)	32	R/W	0000_0002h	5.1.7.6/459
3038_4704	CCM Clock Gating Register (CCM_CCGR112_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4708	CCM Clock Gating Register (CCM_CCGR112_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_470C	CCM Clock Gating Register (CCM_CCGR112_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4710	CCM Clock Gating Register (CCM_CCGR113)	32	R/W	0000_0002h	5.1.7.6/459
3038_4714	CCM Clock Gating Register (CCM_CCGR113_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4718	CCM Clock Gating Register (CCM_CCGR113_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_471C	CCM Clock Gating Register (CCM_CCGR113_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4720	CCM Clock Gating Register (CCM_CCGR114)	32	R/W	0000_0002h	5.1.7.6/459
3038_4724	CCM Clock Gating Register (CCM_CCGR114_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4728	CCM Clock Gating Register (CCM_CCGR114_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_472C	CCM Clock Gating Register (CCM_CCGR114_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4730	CCM Clock Gating Register (CCM_CCGR115)	32	R/W	0000_0002h	5.1.7.6/459
3038_4734	CCM Clock Gating Register (CCM_CCGR115_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4738	CCM Clock Gating Register (CCM_CCGR115_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_473C	CCM Clock Gating Register (CCM_CCGR115_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4740	CCM Clock Gating Register (CCM_CCGR116)	32	R/W	0000_0002h	5.1.7.6/459
3038_4744	CCM Clock Gating Register (CCM_CCGR116_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4748	CCM Clock Gating Register (CCM_CCGR116_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_474C	CCM Clock Gating Register (CCM_CCGR116_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4750	CCM Clock Gating Register (CCM_CCGR117)	32	R/W	0000_0002h	5.1.7.6/459
3038_4754	CCM Clock Gating Register (CCM_CCGR117_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4758	CCM Clock Gating Register (CCM_CCGR117_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_475C	CCM Clock Gating Register (CCM_CCGR117_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4760	CCM Clock Gating Register (CCM_CCGR118)	32	R/W	0000_0002h	5.1.7.6/459
3038_4764	CCM Clock Gating Register (CCM_CCGR118_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4768	CCM Clock Gating Register (CCM_CCGR118_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_476C	CCM Clock Gating Register (CCM_CCGR118_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4770	CCM Clock Gating Register (CCM_CCGR119)	32	R/W	0000_0002h	5.1.7.6/459
3038_4774	CCM Clock Gating Register (CCM_CCGR119_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4778	CCM Clock Gating Register (CCM_CCGR119_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_477C	CCM Clock Gating Register (CCM_CCGR119_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4780	CCM Clock Gating Register (CCM_CCGR120)	32	R/W	0000_0002h	5.1.7.6/459
3038_4784	CCM Clock Gating Register (CCM_CCGR120_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4788	CCM Clock Gating Register (CCM_CCGR120_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_478C	CCM Clock Gating Register (CCM_CCGR120_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4790	CCM Clock Gating Register (CCM_CCGR121)	32	R/W	0000_0002h	5.1.7.6/459
3038_4794	CCM Clock Gating Register (CCM_CCGR121_SET)	32	R/W	0000_0002h	5.1.7.7/462

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4798	CCM Clock Gating Register (CCM_CCGR121_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_479C	CCM Clock Gating Register (CCM_CCGR121_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_47A0	CCM Clock Gating Register (CCM_CCGR122)	32	R/W	0000_0002h	5.1.7.6/459
3038_47A4	CCM Clock Gating Register (CCM_CCGR122_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_47A8	CCM Clock Gating Register (CCM_CCGR122_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_47AC	CCM Clock Gating Register (CCM_CCGR122_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_47B0	CCM Clock Gating Register (CCM_CCGR123)	32	R/W	0000_0002h	5.1.7.6/459
3038_47B4	CCM Clock Gating Register (CCM_CCGR123_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_47B8	CCM Clock Gating Register (CCM_CCGR123_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_47BC	CCM Clock Gating Register (CCM_CCGR123_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_47C0	CCM Clock Gating Register (CCM_CCGR124)	32	R/W	0000_0002h	5.1.7.6/459
3038_47C4	CCM Clock Gating Register (CCM_CCGR124_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_47C8	CCM Clock Gating Register (CCM_CCGR124_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_47CC	CCM Clock Gating Register (CCM_CCGR124_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_47D0	CCM Clock Gating Register (CCM_CCGR125)	32	R/W	0000_0002h	5.1.7.6/459
3038_47D4	CCM Clock Gating Register (CCM_CCGR125_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_47D8	CCM Clock Gating Register (CCM_CCGR125_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_47DC	CCM Clock Gating Register (CCM_CCGR125_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_47E0	CCM Clock Gating Register (CCM_CCGR126)	32	R/W	0000_0002h	5.1.7.6/459
3038_47E4	CCM Clock Gating Register (CCM_CCGR126_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_47E8	CCM Clock Gating Register (CCM_CCGR126_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_47EC	CCM Clock Gating Register (CCM_CCGR126_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_47F0	CCM Clock Gating Register (CCM_CCGR127)	32	R/W	0000_0002h	5.1.7.6/459
3038_47F4	CCM Clock Gating Register (CCM_CCGR127_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_47F8	CCM Clock Gating Register (CCM_CCGR127_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_47FC	CCM Clock Gating Register (CCM_CCGR127_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4800	CCM Clock Gating Register (CCM_CCGR128)	32	R/W	0000_0002h	5.1.7.6/459
3038_4804	CCM Clock Gating Register (CCM_CCGR128_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4808	CCM Clock Gating Register (CCM_CCGR128_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_480C	CCM Clock Gating Register (CCM_CCGR128_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4810	CCM Clock Gating Register (CCM_CCGR129)	32	R/W	0000_0002h	5.1.7.6/459
3038_4814	CCM Clock Gating Register (CCM_CCGR129_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4818	CCM Clock Gating Register (CCM_CCGR129_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_481C	CCM Clock Gating Register (CCM_CCGR129_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4820	CCM Clock Gating Register (CCM_CCGR130)	32	R/W	0000_0002h	5.1.7.6/459
3038_4824	CCM Clock Gating Register (CCM_CCGR130_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4828	CCM Clock Gating Register (CCM_CCGR130_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_482C	CCM Clock Gating Register (CCM_CCGR130_TOG)	32	R/W	0000_0002h	5.1.7.9/466

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4830	CCM Clock Gating Register (CCM_CCGR131)	32	R/W	0000_0002h	5.1.7.6/459
3038_4834	CCM Clock Gating Register (CCM_CCGR131_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4838	CCM Clock Gating Register (CCM_CCGR131_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_483C	CCM Clock Gating Register (CCM_CCGR131_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4840	CCM Clock Gating Register (CCM_CCGR132)	32	R/W	0000_0002h	5.1.7.6/459
3038_4844	CCM Clock Gating Register (CCM_CCGR132_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4848	CCM Clock Gating Register (CCM_CCGR132_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_484C	CCM Clock Gating Register (CCM_CCGR132_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4850	CCM Clock Gating Register (CCM_CCGR133)	32	R/W	0000_0002h	5.1.7.6/459
3038_4854	CCM Clock Gating Register (CCM_CCGR133_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4858	CCM Clock Gating Register (CCM_CCGR133_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_485C	CCM Clock Gating Register (CCM_CCGR133_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4860	CCM Clock Gating Register (CCM_CCGR134)	32	R/W	0000_0002h	5.1.7.6/459
3038_4864	CCM Clock Gating Register (CCM_CCGR134_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4868	CCM Clock Gating Register (CCM_CCGR134_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_486C	CCM Clock Gating Register (CCM_CCGR134_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4870	CCM Clock Gating Register (CCM_CCGR135)	32	R/W	0000_0002h	5.1.7.6/459
3038_4874	CCM Clock Gating Register (CCM_CCGR135_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4878	CCM Clock Gating Register (CCM_CCGR135_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_487C	CCM Clock Gating Register (CCM_CCGR135_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4880	CCM Clock Gating Register (CCM_CCGR136)	32	R/W	0000_0002h	5.1.7.6/459
3038_4884	CCM Clock Gating Register (CCM_CCGR136_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4888	CCM Clock Gating Register (CCM_CCGR136_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_488C	CCM Clock Gating Register (CCM_CCGR136_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4890	CCM Clock Gating Register (CCM_CCGR137)	32	R/W	0000_0002h	5.1.7.6/459
3038_4894	CCM Clock Gating Register (CCM_CCGR137_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4898	CCM Clock Gating Register (CCM_CCGR137_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_489C	CCM Clock Gating Register (CCM_CCGR137_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_48A0	CCM Clock Gating Register (CCM_CCGR138)	32	R/W	0000_0002h	5.1.7.6/459
3038_48A4	CCM Clock Gating Register (CCM_CCGR138_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_48A8	CCM Clock Gating Register (CCM_CCGR138_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_48AC	CCM Clock Gating Register (CCM_CCGR138_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_48B0	CCM Clock Gating Register (CCM_CCGR139)	32	R/W	0000_0002h	5.1.7.6/459
3038_48B4	CCM Clock Gating Register (CCM_CCGR139_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_48B8	CCM Clock Gating Register (CCM_CCGR139_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_48BC	CCM Clock Gating Register (CCM_CCGR139_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_48C0	CCM Clock Gating Register (CCM_CCGR140)	32	R/W	0000_0002h	5.1.7.6/459
3038_48C4	CCM Clock Gating Register (CCM_CCGR140_SET)	32	R/W	0000_0002h	5.1.7.7/462

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_48C8	CCM Clock Gating Register (CCM_CCGR140_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_48CC	CCM Clock Gating Register (CCM_CCGR140_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_48D0	CCM Clock Gating Register (CCM_CCGR141)	32	R/W	0000_0002h	5.1.7.6/459
3038_48D4	CCM Clock Gating Register (CCM_CCGR141_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_48D8	CCM Clock Gating Register (CCM_CCGR141_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_48DC	CCM Clock Gating Register (CCM_CCGR141_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_48E0	CCM Clock Gating Register (CCM_CCGR142)	32	R/W	0000_0002h	5.1.7.6/459
3038_48E4	CCM Clock Gating Register (CCM_CCGR142_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_48E8	CCM Clock Gating Register (CCM_CCGR142_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_48EC	CCM Clock Gating Register (CCM_CCGR142_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_48F0	CCM Clock Gating Register (CCM_CCGR143)	32	R/W	0000_0002h	5.1.7.6/459
3038_48F4	CCM Clock Gating Register (CCM_CCGR143_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_48F8	CCM Clock Gating Register (CCM_CCGR143_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_48FC	CCM Clock Gating Register (CCM_CCGR143_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4900	CCM Clock Gating Register (CCM_CCGR144)	32	R/W	0000_0002h	5.1.7.6/459
3038_4904	CCM Clock Gating Register (CCM_CCGR144_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4908	CCM Clock Gating Register (CCM_CCGR144_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_490C	CCM Clock Gating Register (CCM_CCGR144_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4910	CCM Clock Gating Register (CCM_CCGR145)	32	R/W	0000_0002h	5.1.7.6/459
3038_4914	CCM Clock Gating Register (CCM_CCGR145_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4918	CCM Clock Gating Register (CCM_CCGR145_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_491C	CCM Clock Gating Register (CCM_CCGR145_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4920	CCM Clock Gating Register (CCM_CCGR146)	32	R/W	0000_0002h	5.1.7.6/459
3038_4924	CCM Clock Gating Register (CCM_CCGR146_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4928	CCM Clock Gating Register (CCM_CCGR146_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_492C	CCM Clock Gating Register (CCM_CCGR146_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4930	CCM Clock Gating Register (CCM_CCGR147)	32	R/W	0000_0002h	5.1.7.6/459
3038_4934	CCM Clock Gating Register (CCM_CCGR147_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4938	CCM Clock Gating Register (CCM_CCGR147_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_493C	CCM Clock Gating Register (CCM_CCGR147_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4940	CCM Clock Gating Register (CCM_CCGR148)	32	R/W	0000_0002h	5.1.7.6/459
3038_4944	CCM Clock Gating Register (CCM_CCGR148_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4948	CCM Clock Gating Register (CCM_CCGR148_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_494C	CCM Clock Gating Register (CCM_CCGR148_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4950	CCM Clock Gating Register (CCM_CCGR149)	32	R/W	0000_0002h	5.1.7.6/459
3038_4954	CCM Clock Gating Register (CCM_CCGR149_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4958	CCM Clock Gating Register (CCM_CCGR149_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_495C	CCM Clock Gating Register (CCM_CCGR149_TOG)	32	R/W	0000_0002h	5.1.7.9/466

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4960	CCM Clock Gating Register (CCM_CCGR150)	32	R/W	0000_0002h	5.1.7.6/459
3038_4964	CCM Clock Gating Register (CCM_CCGR150_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4968	CCM Clock Gating Register (CCM_CCGR150_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_496C	CCM Clock Gating Register (CCM_CCGR150_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4970	CCM Clock Gating Register (CCM_CCGR151)	32	R/W	0000_0002h	5.1.7.6/459
3038_4974	CCM Clock Gating Register (CCM_CCGR151_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4978	CCM Clock Gating Register (CCM_CCGR151_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_497C	CCM Clock Gating Register (CCM_CCGR151_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4980	CCM Clock Gating Register (CCM_CCGR152)	32	R/W	0000_0002h	5.1.7.6/459
3038_4984	CCM Clock Gating Register (CCM_CCGR152_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4988	CCM Clock Gating Register (CCM_CCGR152_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_498C	CCM Clock Gating Register (CCM_CCGR152_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4990	CCM Clock Gating Register (CCM_CCGR153)	32	R/W	0000_0002h	5.1.7.6/459
3038_4994	CCM Clock Gating Register (CCM_CCGR153_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4998	CCM Clock Gating Register (CCM_CCGR153_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_499C	CCM Clock Gating Register (CCM_CCGR153_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_49A0	CCM Clock Gating Register (CCM_CCGR154)	32	R/W	0000_0002h	5.1.7.6/459
3038_49A4	CCM Clock Gating Register (CCM_CCGR154_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_49A8	CCM Clock Gating Register (CCM_CCGR154_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_49AC	CCM Clock Gating Register (CCM_CCGR154_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_49B0	CCM Clock Gating Register (CCM_CCGR155)	32	R/W	0000_0002h	5.1.7.6/459
3038_49B4	CCM Clock Gating Register (CCM_CCGR155_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_49B8	CCM Clock Gating Register (CCM_CCGR155_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_49BC	CCM Clock Gating Register (CCM_CCGR155_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_49C0	CCM Clock Gating Register (CCM_CCGR156)	32	R/W	0000_0002h	5.1.7.6/459
3038_49C4	CCM Clock Gating Register (CCM_CCGR156_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_49C8	CCM Clock Gating Register (CCM_CCGR156_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_49CC	CCM Clock Gating Register (CCM_CCGR156_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_49D0	CCM Clock Gating Register (CCM_CCGR157)	32	R/W	0000_0002h	5.1.7.6/459
3038_49D4	CCM Clock Gating Register (CCM_CCGR157_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_49D8	CCM Clock Gating Register (CCM_CCGR157_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_49DC	CCM Clock Gating Register (CCM_CCGR157_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_49E0	CCM Clock Gating Register (CCM_CCGR158)	32	R/W	0000_0002h	5.1.7.6/459
3038_49E4	CCM Clock Gating Register (CCM_CCGR158_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_49E8	CCM Clock Gating Register (CCM_CCGR158_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_49EC	CCM Clock Gating Register (CCM_CCGR158_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_49F0	CCM Clock Gating Register (CCM_CCGR159)	32	R/W	0000_0002h	5.1.7.6/459
3038_49F4	CCM Clock Gating Register (CCM_CCGR159_SET)	32	R/W	0000_0002h	5.1.7.7/462

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_49F8	CCM Clock Gating Register (CCM_CCGR159_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_49FC	CCM Clock Gating Register (CCM_CCGR159_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4A00	CCM Clock Gating Register (CCM_CCGR160)	32	R/W	0000_0002h	5.1.7.6/459
3038_4A04	CCM Clock Gating Register (CCM_CCGR160_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4A08	CCM Clock Gating Register (CCM_CCGR160_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4A0C	CCM Clock Gating Register (CCM_CCGR160_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4A10	CCM Clock Gating Register (CCM_CCGR161)	32	R/W	0000_0002h	5.1.7.6/459
3038_4A14	CCM Clock Gating Register (CCM_CCGR161_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4A18	CCM Clock Gating Register (CCM_CCGR161_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4A1C	CCM Clock Gating Register (CCM_CCGR161_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4A20	CCM Clock Gating Register (CCM_CCGR162)	32	R/W	0000_0002h	5.1.7.6/459
3038_4A24	CCM Clock Gating Register (CCM_CCGR162_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4A28	CCM Clock Gating Register (CCM_CCGR162_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4A2C	CCM Clock Gating Register (CCM_CCGR162_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4A30	CCM Clock Gating Register (CCM_CCGR163)	32	R/W	0000_0002h	5.1.7.6/459
3038_4A34	CCM Clock Gating Register (CCM_CCGR163_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4A38	CCM Clock Gating Register (CCM_CCGR163_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4A3C	CCM Clock Gating Register (CCM_CCGR163_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4A40	CCM Clock Gating Register (CCM_CCGR164)	32	R/W	0000_0002h	5.1.7.6/459
3038_4A44	CCM Clock Gating Register (CCM_CCGR164_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4A48	CCM Clock Gating Register (CCM_CCGR164_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4A4C	CCM Clock Gating Register (CCM_CCGR164_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4A50	CCM Clock Gating Register (CCM_CCGR165)	32	R/W	0000_0002h	5.1.7.6/459
3038_4A54	CCM Clock Gating Register (CCM_CCGR165_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4A58	CCM Clock Gating Register (CCM_CCGR165_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4A5C	CCM Clock Gating Register (CCM_CCGR165_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4A60	CCM Clock Gating Register (CCM_CCGR166)	32	R/W	0000_0002h	5.1.7.6/459
3038_4A64	CCM Clock Gating Register (CCM_CCGR166_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4A68	CCM Clock Gating Register (CCM_CCGR166_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4A6C	CCM Clock Gating Register (CCM_CCGR166_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4A70	CCM Clock Gating Register (CCM_CCGR167)	32	R/W	0000_0002h	5.1.7.6/459
3038_4A74	CCM Clock Gating Register (CCM_CCGR167_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4A78	CCM Clock Gating Register (CCM_CCGR167_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4A7C	CCM Clock Gating Register (CCM_CCGR167_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4A80	CCM Clock Gating Register (CCM_CCGR168)	32	R/W	0000_0002h	5.1.7.6/459
3038_4A84	CCM Clock Gating Register (CCM_CCGR168_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4A88	CCM Clock Gating Register (CCM_CCGR168_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4A8C	CCM Clock Gating Register (CCM_CCGR168_TOG)	32	R/W	0000_0002h	5.1.7.9/466

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4A90	CCM Clock Gating Register (CCM_CCGR169)	32	R/W	0000_0002h	5.1.7.6/459
3038_4A94	CCM Clock Gating Register (CCM_CCGR169_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4A98	CCM Clock Gating Register (CCM_CCGR169_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4A9C	CCM Clock Gating Register (CCM_CCGR169_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4AA0	CCM Clock Gating Register (CCM_CCGR170)	32	R/W	0000_0002h	5.1.7.6/459
3038_4AA4	CCM Clock Gating Register (CCM_CCGR170_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4AA8	CCM Clock Gating Register (CCM_CCGR170_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4AAC	CCM Clock Gating Register (CCM_CCGR170_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4AB0	CCM Clock Gating Register (CCM_CCGR171)	32	R/W	0000_0002h	5.1.7.6/459
3038_4AB4	CCM Clock Gating Register (CCM_CCGR171_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4AB8	CCM Clock Gating Register (CCM_CCGR171_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4ABC	CCM Clock Gating Register (CCM_CCGR171_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4AC0	CCM Clock Gating Register (CCM_CCGR172)	32	R/W	0000_0002h	5.1.7.6/459
3038_4AC4	CCM Clock Gating Register (CCM_CCGR172_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4AC8	CCM Clock Gating Register (CCM_CCGR172_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4ACC	CCM Clock Gating Register (CCM_CCGR172_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4AD0	CCM Clock Gating Register (CCM_CCGR173)	32	R/W	0000_0002h	5.1.7.6/459
3038_4AD4	CCM Clock Gating Register (CCM_CCGR173_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4AD8	CCM Clock Gating Register (CCM_CCGR173_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4ADC	CCM Clock Gating Register (CCM_CCGR173_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4AE0	CCM Clock Gating Register (CCM_CCGR174)	32	R/W	0000_0002h	5.1.7.6/459
3038_4AE4	CCM Clock Gating Register (CCM_CCGR174_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4AE8	CCM Clock Gating Register (CCM_CCGR174_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4AEC	CCM Clock Gating Register (CCM_CCGR174_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4AF0	CCM Clock Gating Register (CCM_CCGR175)	32	R/W	0000_0002h	5.1.7.6/459
3038_4AF4	CCM Clock Gating Register (CCM_CCGR175_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4AF8	CCM Clock Gating Register (CCM_CCGR175_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4AFC	CCM Clock Gating Register (CCM_CCGR175_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4B00	CCM Clock Gating Register (CCM_CCGR176)	32	R/W	0000_0002h	5.1.7.6/459
3038_4B04	CCM Clock Gating Register (CCM_CCGR176_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4B08	CCM Clock Gating Register (CCM_CCGR176_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4B0C	CCM Clock Gating Register (CCM_CCGR176_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4B10	CCM Clock Gating Register (CCM_CCGR177)	32	R/W	0000_0002h	5.1.7.6/459
3038_4B14	CCM Clock Gating Register (CCM_CCGR177_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4B18	CCM Clock Gating Register (CCM_CCGR177_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4B1C	CCM Clock Gating Register (CCM_CCGR177_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4B20	CCM Clock Gating Register (CCM_CCGR178)	32	R/W	0000_0002h	5.1.7.6/459
3038_4B24	CCM Clock Gating Register (CCM_CCGR178_SET)	32	R/W	0000_0002h	5.1.7.7/462

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4B28	CCM Clock Gating Register (CCM_CCGR178_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4B2C	CCM Clock Gating Register (CCM_CCGR178_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4B30	CCM Clock Gating Register (CCM_CCGR179)	32	R/W	0000_0002h	5.1.7.6/459
3038_4B34	CCM Clock Gating Register (CCM_CCGR179_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4B38	CCM Clock Gating Register (CCM_CCGR179_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4B3C	CCM Clock Gating Register (CCM_CCGR179_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4B40	CCM Clock Gating Register (CCM_CCGR180)	32	R/W	0000_0002h	5.1.7.6/459
3038_4B44	CCM Clock Gating Register (CCM_CCGR180_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4B48	CCM Clock Gating Register (CCM_CCGR180_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4B4C	CCM Clock Gating Register (CCM_CCGR180_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4B50	CCM Clock Gating Register (CCM_CCGR181)	32	R/W	0000_0002h	5.1.7.6/459
3038_4B54	CCM Clock Gating Register (CCM_CCGR181_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4B58	CCM Clock Gating Register (CCM_CCGR181_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4B5C	CCM Clock Gating Register (CCM_CCGR181_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4B60	CCM Clock Gating Register (CCM_CCGR182)	32	R/W	0000_0002h	5.1.7.6/459
3038_4B64	CCM Clock Gating Register (CCM_CCGR182_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4B68	CCM Clock Gating Register (CCM_CCGR182_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4B6C	CCM Clock Gating Register (CCM_CCGR182_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4B70	CCM Clock Gating Register (CCM_CCGR183)	32	R/W	0000_0002h	5.1.7.6/459
3038_4B74	CCM Clock Gating Register (CCM_CCGR183_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4B78	CCM Clock Gating Register (CCM_CCGR183_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4B7C	CCM Clock Gating Register (CCM_CCGR183_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4B80	CCM Clock Gating Register (CCM_CCGR184)	32	R/W	0000_0002h	5.1.7.6/459
3038_4B84	CCM Clock Gating Register (CCM_CCGR184_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4B88	CCM Clock Gating Register (CCM_CCGR184_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4B8C	CCM Clock Gating Register (CCM_CCGR184_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4B90	CCM Clock Gating Register (CCM_CCGR185)	32	R/W	0000_0002h	5.1.7.6/459
3038_4B94	CCM Clock Gating Register (CCM_CCGR185_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4B98	CCM Clock Gating Register (CCM_CCGR185_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4B9C	CCM Clock Gating Register (CCM_CCGR185_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4BA0	CCM Clock Gating Register (CCM_CCGR186)	32	R/W	0000_0002h	5.1.7.6/459
3038_4BA4	CCM Clock Gating Register (CCM_CCGR186_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4BA8	CCM Clock Gating Register (CCM_CCGR186_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4BAC	CCM Clock Gating Register (CCM_CCGR186_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4BB0	CCM Clock Gating Register (CCM_CCGR187)	32	R/W	0000_0002h	5.1.7.6/459
3038_4BB4	CCM Clock Gating Register (CCM_CCGR187_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4BB8	CCM Clock Gating Register (CCM_CCGR187_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4BBC	CCM Clock Gating Register (CCM_CCGR187_TOG)	32	R/W	0000_0002h	5.1.7.9/466

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_4BC0	CCM Clock Gating Register (CCM_CCGR188)	32	R/W	0000_0002h	5.1.7.6/459
3038_4BC4	CCM Clock Gating Register (CCM_CCGR188_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4BC8	CCM Clock Gating Register (CCM_CCGR188_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4BCC	CCM Clock Gating Register (CCM_CCGR188_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4BD0	CCM Clock Gating Register (CCM_CCGR189)	32	R/W	0000_0002h	5.1.7.6/459
3038_4BD4	CCM Clock Gating Register (CCM_CCGR189_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4BD8	CCM Clock Gating Register (CCM_CCGR189_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4BDC	CCM Clock Gating Register (CCM_CCGR189_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_4BE0	CCM Clock Gating Register (CCM_CCGR190)	32	R/W	0000_0002h	5.1.7.6/459
3038_4BE4	CCM Clock Gating Register (CCM_CCGR190_SET)	32	R/W	0000_0002h	5.1.7.7/462
3038_4BE8	CCM Clock Gating Register (CCM_CCGR190_CLR)	32	R/W	0000_0002h	5.1.7.8/464
3038_4BEC	CCM Clock Gating Register (CCM_CCGR190_TOG)	32	R/W	0000_0002h	5.1.7.9/466
3038_8000	Target Register (CCM_TARGET_ROOT0)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8004	Target Register (CCM_TARGET_ROOT0_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8008	Target Register (CCM_TARGET_ROOT0_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_800C	Target Register (CCM_TARGET_ROOT0_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8010	Miscellaneous Register (CCM_MISC0)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8014	Miscellaneous Register (CCM_MISC_ROOT0_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8018	Miscellaneous Register (CCM_MISC_ROOT0_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_801C	Miscellaneous Register (CCM_MISC_ROOT0_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8020	Post Divider Register (CCM_POST0)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8024	Post Divider Register (CCM_POST_ROOT0_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8028	Post Divider Register (CCM_POST_ROOT0_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_802C	Post Divider Register (CCM_POST_ROOT0_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8030	Pre Divider Register (CCM_PRE0)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8034	Pre Divider Register (CCM_PRE_ROOT0_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8038	Pre Divider Register (CCM_PRE_ROOT0_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_803C	Pre Divider Register (CCM_PRE_ROOT0_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8070	Access Control Register (CCM_ACCESS_CTRL0)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8074	Access Control Register (CCM_ACCESS_CTRL_ROOT0_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8078	Access Control Register (CCM_ACCESS_CTRL_ROOT0_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_807C	Access Control Register (CCM_ACCESS_CTRL_ROOT0_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8080	Target Register (CCM_TARGET_ROOT1)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8084	Target Register (CCM_TARGET_ROOT1_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8088	Target Register (CCM_TARGET_ROOT1_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_808C	Target Register (CCM_TARGET_ROOT1_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8090	Miscellaneous Register (CCM_MISC1)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8094	Miscellaneous Register (CCM_MISC_ROOT1_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8098	Miscellaneous Register (CCM_MISC_ROOT1_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_809C	Miscellaneous Register (CCM_MISC_ROOT1_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_80A0	Post Divider Register (CCM_POST1)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_80A4	Post Divider Register (CCM_POST_ROOT1_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_80A8	Post Divider Register (CCM_POST_ROOT1_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_80AC	Post Divider Register (CCM_POST_ROOT1_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_80B0	Pre Divider Register (CCM_PRE1)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_80B4	Pre Divider Register (CCM_PRE_ROOT1_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_80B8	Pre Divider Register (CCM_PRE_ROOT1_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_80BC	Pre Divider Register (CCM_PRE_ROOT1_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_80F0	Access Control Register (CCM_ACCESS_CTRL1)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_80F4	Access Control Register (CCM_ACCESS_CTRL_ROOT1_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_80F8	Access Control Register (CCM_ACCESS_CTRL_ROOT1_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_80FC	Access Control Register (CCM_ACCESS_CTRL_ROOT1_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8100	Target Register (CCM_TARGET_ROOT2)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8104	Target Register (CCM_TARGET_ROOT2_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8108	Target Register (CCM_TARGET_ROOT2_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_810C	Target Register (CCM_TARGET_ROOT2_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8110	Miscellaneous Register (CCM_MISC2)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8114	Miscellaneous Register (CCM_MISC_ROOT2_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8118	Miscellaneous Register (CCM_MISC_ROOT2_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_811C	Miscellaneous Register (CCM_MISC_ROOT2_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8120	Post Divider Register (CCM_POST2)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8124	Post Divider Register (CCM_POST_ROOT2_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8128	Post Divider Register (CCM_POST_ROOT2_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_812C	Post Divider Register (CCM_POST_ROOT2_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8130	Pre Divider Register (CCM_PRE2)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8134	Pre Divider Register (CCM_PRE_ROOT2_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8138	Pre Divider Register (CCM_PRE_ROOT2_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_813C	Pre Divider Register (CCM_PRE_ROOT2_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8170	Access Control Register (CCM_ACCESS_CTRL2)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8174	Access Control Register (CCM_ACCESS_CTRL_ROOT2_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8178	Access Control Register (CCM_ACCESS_CTRL_ROOT2_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_817C	Access Control Register (CCM_ACCESS_CTRL_ROOT2_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8180	Target Register (CCM_TARGET_ROOT3)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8184	Target Register (CCM_TARGET_ROOT3_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8188	Target Register (CCM_TARGET_ROOT3_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_818C	Target Register (CCM_TARGET_ROOT3_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8190	Miscellaneous Register (CCM_MISC3)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8194	Miscellaneous Register (CCM_MISC_ROOT3_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8198	Miscellaneous Register (CCM_MISC_ROOT3_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_819C	Miscellaneous Register (CCM_MISC_ROOT3_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_81A0	Post Divider Register (CCM_POST3)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_81A4	Post Divider Register (CCM_POST_ROOT3_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_81A8	Post Divider Register (CCM_POST_ROOT3_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_81AC	Post Divider Register (CCM_POST_ROOT3_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_81B0	Pre Divider Register (CCM_PRE3)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_81B4	Pre Divider Register (CCM_PRE_ROOT3_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_81B8	Pre Divider Register (CCM_PRE_ROOT3_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_81BC	Pre Divider Register (CCM_PRE_ROOT3_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_81F0	Access Control Register (CCM_ACCESS_CTRL3)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_81F4	Access Control Register (CCM_ACCESS_CTRL_ROOT3_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_81F8	Access Control Register (CCM_ACCESS_CTRL_ROOT3_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_81FC	Access Control Register (CCM_ACCESS_CTRL_ROOT3_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8200	Target Register (CCM_TARGET_ROOT4)	32	R/W	1000_0000h	5.1.7.10/ 468

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8204	Target Register (CCM_TARGET_ROOT4_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8208	Target Register (CCM_TARGET_ROOT4_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_820C	Target Register (CCM_TARGET_ROOT4_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8210	Miscellaneous Register (CCM_MISC4)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8214	Miscellaneous Register (CCM_MISC_ROOT4_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8218	Miscellaneous Register (CCM_MISC_ROOT4_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_821C	Miscellaneous Register (CCM_MISC_ROOT4_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8220	Post Divider Register (CCM_POST4)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8224	Post Divider Register (CCM_POST_ROOT4_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8228	Post Divider Register (CCM_POST_ROOT4_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_822C	Post Divider Register (CCM_POST_ROOT4_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8230	Pre Divider Register (CCM_PRE4)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8234	Pre Divider Register (CCM_PRE_ROOT4_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8238	Pre Divider Register (CCM_PRE_ROOT4_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_823C	Pre Divider Register (CCM_PRE_ROOT4_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8270	Access Control Register (CCM_ACCESS_CTRL4)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8274	Access Control Register (CCM_ACCESS_CTRL_ROOT4_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8278	Access Control Register (CCM_ACCESS_CTRL_ROOT4_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_827C	Access Control Register (CCM_ACCESS_CTRL_ROOT4_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8280	Target Register (CCM_TARGET_ROOT5)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8284	Target Register (CCM_TARGET_ROOT5_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8288	Target Register (CCM_TARGET_ROOT5_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_828C	Target Register (CCM_TARGET_ROOT5_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8290	Miscellaneous Register (CCM_MISC5)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8294	Miscellaneous Register (CCM_MISC_ROOT5_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8298	Miscellaneous Register (CCM_MISC_ROOT5_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_829C	Miscellaneous Register (CCM_MISC_ROOT5_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_82A0	Post Divider Register (CCM_POST5)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_82A4	Post Divider Register (CCM_POST_ROOT5_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_82A8	Post Divider Register (CCM_POST_ROOT5_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_82AC	Post Divider Register (CCM_POST_ROOT5_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_82B0	Pre Divider Register (CCM_PRE5)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_82B4	Pre Divider Register (CCM_PRE_ROOT5_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_82B8	Pre Divider Register (CCM_PRE_ROOT5_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_82BC	Pre Divider Register (CCM_PRE_ROOT5_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_82F0	Access Control Register (CCM_ACCESS_CTRL5)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_82F4	Access Control Register (CCM_ACCESS_CTRL_ROOT5_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_82F8	Access Control Register (CCM_ACCESS_CTRL_ROOT5_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_82FC	Access Control Register (CCM_ACCESS_CTRL_ROOT5_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8300	Target Register (CCM_TARGET_ROOT6)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8304	Target Register (CCM_TARGET_ROOT6_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8308	Target Register (CCM_TARGET_ROOT6_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_830C	Target Register (CCM_TARGET_ROOT6_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8310	Miscellaneous Register (CCM_MISC6)	32	R/W	0000_0000h	5.1.7.14/ 476

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8314	Miscellaneous Register (CCM_MISC_ROOT6_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8318	Miscellaneous Register (CCM_MISC_ROOT6_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_831C	Miscellaneous Register (CCM_MISC_ROOT6_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8320	Post Divider Register (CCM_POST6)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8324	Post Divider Register (CCM_POST_ROOT6_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8328	Post Divider Register (CCM_POST_ROOT6_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_832C	Post Divider Register (CCM_POST_ROOT6_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8330	Pre Divider Register (CCM_PRE6)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8334	Pre Divider Register (CCM_PRE_ROOT6_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8338	Pre Divider Register (CCM_PRE_ROOT6_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_833C	Pre Divider Register (CCM_PRE_ROOT6_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8370	Access Control Register (CCM_ACCESS_CTRL6)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8374	Access Control Register (CCM_ACCESS_CTRL_ROOT6_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8378	Access Control Register (CCM_ACCESS_CTRL_ROOT6_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_837C	Access Control Register (CCM_ACCESS_CTRL_ROOT6_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8380	Target Register (CCM_TARGET_ROOT7)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8384	Target Register (CCM_TARGET_ROOT7_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8388	Target Register (CCM_TARGET_ROOT7_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_838C	Target Register (CCM_TARGET_ROOT7_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8390	Miscellaneous Register (CCM_MISC7)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8394	Miscellaneous Register (CCM_MISC_ROOT7_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8398	Miscellaneous Register (CCM_MISC_ROOT7_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_839C	Miscellaneous Register (CCM_MISC_ROOT7_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_83A0	Post Divider Register (CCM_POST7)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_83A4	Post Divider Register (CCM_POST_ROOT7_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_83A8	Post Divider Register (CCM_POST_ROOT7_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_83AC	Post Divider Register (CCM_POST_ROOT7_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_83B0	Pre Divider Register (CCM_PRE7)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_83B4	Pre Divider Register (CCM_PRE_ROOT7_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_83B8	Pre Divider Register (CCM_PRE_ROOT7_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_83BC	Pre Divider Register (CCM_PRE_ROOT7_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_83F0	Access Control Register (CCM_ACCESS_CTRL7)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_83F4	Access Control Register (CCM_ACCESS_CTRL_ROOT7_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_83F8	Access Control Register (CCM_ACCESS_CTRL_ROOT7_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_83FC	Access Control Register (CCM_ACCESS_CTRL_ROOT7_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8400	Target Register (CCM_TARGET_ROOT8)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8404	Target Register (CCM_TARGET_ROOT8_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8408	Target Register (CCM_TARGET_ROOT8_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_840C	Target Register (CCM_TARGET_ROOT8_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8410	Miscellaneous Register (CCM_MISC8)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8414	Miscellaneous Register (CCM_MISC_ROOT8_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8418	Miscellaneous Register (CCM_MISC_ROOT8_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_841C	Miscellaneous Register (CCM_MISC_ROOT8_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8420	Post Divider Register (CCM_POST8)	32	R/W	0000_0000h	5.1.7.18/ 480

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8424	Post Divider Register (CCM_POST_ROOT8_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8428	Post Divider Register (CCM_POST_ROOT8_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_842C	Post Divider Register (CCM_POST_ROOT8_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8430	Pre Divider Register (CCM_PRE8)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8434	Pre Divider Register (CCM_PRE_ROOT8_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8438	Pre Divider Register (CCM_PRE_ROOT8_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_843C	Pre Divider Register (CCM_PRE_ROOT8_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8470	Access Control Register (CCM_ACCESS_CTRL8)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8474	Access Control Register (CCM_ACCESS_CTRL_ROOT8_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8478	Access Control Register (CCM_ACCESS_CTRL_ROOT8_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_847C	Access Control Register (CCM_ACCESS_CTRL_ROOT8_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8480	Target Register (CCM_TARGET_ROOT9)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8484	Target Register (CCM_TARGET_ROOT9_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8488	Target Register (CCM_TARGET_ROOT9_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_848C	Target Register (CCM_TARGET_ROOT9_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8490	Miscellaneous Register (CCM_MISC9)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8494	Miscellaneous Register (CCM_MISC_ROOT9_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8498	Miscellaneous Register (CCM_MISC_ROOT9_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_849C	Miscellaneous Register (CCM_MISC_ROOT9_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_84A0	Post Divider Register (CCM_POST9)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_84A4	Post Divider Register (CCM_POST_ROOT9_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_84A8	Post Divider Register (CCM_POST_ROOT9_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_84AC	Post Divider Register (CCM_POST_ROOT9_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_84B0	Pre Divider Register (CCM_PRE9)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_84B4	Pre Divider Register (CCM_PRE_ROOT9_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_84B8	Pre Divider Register (CCM_PRE_ROOT9_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_84BC	Pre Divider Register (CCM_PRE_ROOT9_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_84F0	Access Control Register (CCM_ACCESS_CTRL9)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_84F4	Access Control Register (CCM_ACCESS_CTRL_ROOT9_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_84F8	Access Control Register (CCM_ACCESS_CTRL_ROOT9_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_84FC	Access Control Register (CCM_ACCESS_CTRL_ROOT9_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8500	Target Register (CCM_TARGET_ROOT10)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8504	Target Register (CCM_TARGET_ROOT10_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8508	Target Register (CCM_TARGET_ROOT10_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_850C	Target Register (CCM_TARGET_ROOT10_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8510	Miscellaneous Register (CCM_MISC10)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8514	Miscellaneous Register (CCM_MISC_ROOT10_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8518	Miscellaneous Register (CCM_MISC_ROOT10_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_851C	Miscellaneous Register (CCM_MISC_ROOT10_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8520	Post Divider Register (CCM_POST10)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8524	Post Divider Register (CCM_POST_ROOT10_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8528	Post Divider Register (CCM_POST_ROOT10_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_852C	Post Divider Register (CCM_POST_ROOT10_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8530	Pre Divider Register (CCM_PRE10)	32	R/W	1000_0000h	5.1.7.22/ 492

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8534	Pre Divider Register (CCM_PRE_ROOT10_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8538	Pre Divider Register (CCM_PRE_ROOT10_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_853C	Pre Divider Register (CCM_PRE_ROOT10_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8570	Access Control Register (CCM_ACCESS_CTRL10)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8574	Access Control Register (CCM_ACCESS_CTRL_ROOT10_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8578	Access Control Register (CCM_ACCESS_CTRL_ROOT10_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_857C	Access Control Register (CCM_ACCESS_CTRL_ROOT10_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8580	Target Register (CCM_TARGET_ROOT11)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8584	Target Register (CCM_TARGET_ROOT11_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8588	Target Register (CCM_TARGET_ROOT11_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_858C	Target Register (CCM_TARGET_ROOT11_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8590	Miscellaneous Register (CCM_MISC11)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8594	Miscellaneous Register (CCM_MISC_ROOT11_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8598	Miscellaneous Register (CCM_MISC_ROOT11_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_859C	Miscellaneous Register (CCM_MISC_ROOT11_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_85A0	Post Divider Register (CCM_POST11)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_85A4	Post Divider Register (CCM_POST_ROOT11_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_85A8	Post Divider Register (CCM_POST_ROOT11_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_85AC	Post Divider Register (CCM_POST_ROOT11_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_85B0	Pre Divider Register (CCM_PRE11)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_85B4	Pre Divider Register (CCM_PRE_ROOT11_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_85B8	Pre Divider Register (CCM_PRE_ROOT11_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_85BC	Pre Divider Register (CCM_PRE_ROOT11_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_85F0	Access Control Register (CCM_ACCESS_CTRL11)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_85F4	Access Control Register (CCM_ACCESS_CTRL_ROOT11_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_85F8	Access Control Register (CCM_ACCESS_CTRL_ROOT11_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_85FC	Access Control Register (CCM_ACCESS_CTRL_ROOT11_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8600	Target Register (CCM_TARGET_ROOT12)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8604	Target Register (CCM_TARGET_ROOT12_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8608	Target Register (CCM_TARGET_ROOT12_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_860C	Target Register (CCM_TARGET_ROOT12_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8610	Miscellaneous Register (CCM_MISC12)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8614	Miscellaneous Register (CCM_MISC_ROOT12_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8618	Miscellaneous Register (CCM_MISC_ROOT12_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_861C	Miscellaneous Register (CCM_MISC_ROOT12_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8620	Post Divider Register (CCM_POST12)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8624	Post Divider Register (CCM_POST_ROOT12_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8628	Post Divider Register (CCM_POST_ROOT12_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_862C	Post Divider Register (CCM_POST_ROOT12_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8630	Pre Divider Register (CCM_PRE12)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8634	Pre Divider Register (CCM_PRE_ROOT12_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8638	Pre Divider Register (CCM_PRE_ROOT12_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_863C	Pre Divider Register (CCM_PRE_ROOT12_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8670	Access Control Register (CCM_ACCESS_CTRL12)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8674	Access Control Register (CCM_ACCESS_CTRL_ROOT12_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8678	Access Control Register (CCM_ACCESS_CTRL_ROOT12_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_867C	Access Control Register (CCM_ACCESS_CTRL_ROOT12_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8680	Target Register (CCM_TARGET_ROOT13)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8684	Target Register (CCM_TARGET_ROOT13_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8688	Target Register (CCM_TARGET_ROOT13_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_868C	Target Register (CCM_TARGET_ROOT13_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8690	Miscellaneous Register (CCM_MISC13)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8694	Miscellaneous Register (CCM_MISC_ROOT13_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8698	Miscellaneous Register (CCM_MISC_ROOT13_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_869C	Miscellaneous Register (CCM_MISC_ROOT13_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_86A0	Post Divider Register (CCM_POST13)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_86A4	Post Divider Register (CCM_POST_ROOT13_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_86A8	Post Divider Register (CCM_POST_ROOT13_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_86AC	Post Divider Register (CCM_POST_ROOT13_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_86B0	Pre Divider Register (CCM_PRE13)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_86B4	Pre Divider Register (CCM_PRE_ROOT13_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_86B8	Pre Divider Register (CCM_PRE_ROOT13_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_86BC	Pre Divider Register (CCM_PRE_ROOT13_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_86F0	Access Control Register (CCM_ACCESS_CTRL13)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_86F4	Access Control Register (CCM_ACCESS_CTRL_ROOT13_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_86F8	Access Control Register (CCM_ACCESS_CTRL_ROOT13_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_86FC	Access Control Register (CCM_ACCESS_CTRL_ROOT13_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8700	Target Register (CCM_TARGET_ROOT14)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8704	Target Register (CCM_TARGET_ROOT14_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8708	Target Register (CCM_TARGET_ROOT14_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_870C	Target Register (CCM_TARGET_ROOT14_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8710	Miscellaneous Register (CCM_MISC14)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8714	Miscellaneous Register (CCM_MISC_ROOT14_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8718	Miscellaneous Register (CCM_MISC_ROOT14_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_871C	Miscellaneous Register (CCM_MISC_ROOT14_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8720	Post Divider Register (CCM_POST14)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8724	Post Divider Register (CCM_POST_ROOT14_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8728	Post Divider Register (CCM_POST_ROOT14_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_872C	Post Divider Register (CCM_POST_ROOT14_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8730	Pre Divider Register (CCM_PRE14)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8734	Pre Divider Register (CCM_PRE_ROOT14_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8738	Pre Divider Register (CCM_PRE_ROOT14_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_873C	Pre Divider Register (CCM_PRE_ROOT14_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8770	Access Control Register (CCM_ACCESS_CTRL14)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8774	Access Control Register (CCM_ACCESS_CTRL_ROOT14_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8778	Access Control Register (CCM_ACCESS_CTRL_ROOT14_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_877C	Access Control Register (CCM_ACCESS_CTRL_ROOT14_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8780	Target Register (CCM_TARGET_ROOT15)	32	R/W	1000_0000h	5.1.7.10/ 468

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8784	Target Register (CCM_TARGET_ROOT15_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8788	Target Register (CCM_TARGET_ROOT15_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_878C	Target Register (CCM_TARGET_ROOT15_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8790	Miscellaneous Register (CCM_MISC15)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8794	Miscellaneous Register (CCM_MISC_ROOT15_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8798	Miscellaneous Register (CCM_MISC_ROOT15_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_879C	Miscellaneous Register (CCM_MISC_ROOT15_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_87A0	Post Divider Register (CCM_POST15)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_87A4	Post Divider Register (CCM_POST_ROOT15_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_87A8	Post Divider Register (CCM_POST_ROOT15_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_87AC	Post Divider Register (CCM_POST_ROOT15_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_87B0	Pre Divider Register (CCM_PRE15)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_87B4	Pre Divider Register (CCM_PRE_ROOT15_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_87B8	Pre Divider Register (CCM_PRE_ROOT15_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_87BC	Pre Divider Register (CCM_PRE_ROOT15_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_87F0	Access Control Register (CCM_ACCESS_CTRL15)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_87F4	Access Control Register (CCM_ACCESS_CTRL_ROOT15_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_87F8	Access Control Register (CCM_ACCESS_CTRL_ROOT15_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_87FC	Access Control Register (CCM_ACCESS_CTRL_ROOT15_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8800	Target Register (CCM_TARGET_ROOT16)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8804	Target Register (CCM_TARGET_ROOT16_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8808	Target Register (CCM_TARGET_ROOT16_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_880C	Target Register (CCM_TARGET_ROOT16_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8810	Miscellaneous Register (CCM_MISC16)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8814	Miscellaneous Register (CCM_MISC_ROOT16_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8818	Miscellaneous Register (CCM_MISC_ROOT16_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_881C	Miscellaneous Register (CCM_MISC_ROOT16_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8820	Post Divider Register (CCM_POST16)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8824	Post Divider Register (CCM_POST_ROOT16_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8828	Post Divider Register (CCM_POST_ROOT16_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_882C	Post Divider Register (CCM_POST_ROOT16_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8830	Pre Divider Register (CCM_PRE16)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8834	Pre Divider Register (CCM_PRE_ROOT16_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8838	Pre Divider Register (CCM_PRE_ROOT16_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_883C	Pre Divider Register (CCM_PRE_ROOT16_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8870	Access Control Register (CCM_ACCESS_CTRL16)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8874	Access Control Register (CCM_ACCESS_CTRL_ROOT16_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8878	Access Control Register (CCM_ACCESS_CTRL_ROOT16_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_887C	Access Control Register (CCM_ACCESS_CTRL_ROOT16_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8880	Target Register (CCM_TARGET_ROOT17)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8884	Target Register (CCM_TARGET_ROOT17_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8888	Target Register (CCM_TARGET_ROOT17_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_888C	Target Register (CCM_TARGET_ROOT17_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8890	Miscellaneous Register (CCM_MISC17)	32	R/W	0000_0000h	5.1.7.14/ 476

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8894	Miscellaneous Register (CCM_MISC_ROOT17_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8898	Miscellaneous Register (CCM_MISC_ROOT17_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_889C	Miscellaneous Register (CCM_MISC_ROOT17_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_88A0	Post Divider Register (CCM_POST17)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_88A4	Post Divider Register (CCM_POST_ROOT17_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_88A8	Post Divider Register (CCM_POST_ROOT17_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_88AC	Post Divider Register (CCM_POST_ROOT17_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_88B0	Pre Divider Register (CCM_PRE17)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_88B4	Pre Divider Register (CCM_PRE_ROOT17_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_88B8	Pre Divider Register (CCM_PRE_ROOT17_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_88BC	Pre Divider Register (CCM_PRE_ROOT17_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_88F0	Access Control Register (CCM_ACCESS_CTRL17)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_88F4	Access Control Register (CCM_ACCESS_CTRL_ROOT17_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_88F8	Access Control Register (CCM_ACCESS_CTRL_ROOT17_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_88FC	Access Control Register (CCM_ACCESS_CTRL_ROOT17_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8900	Target Register (CCM_TARGET_ROOT18)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8904	Target Register (CCM_TARGET_ROOT18_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8908	Target Register (CCM_TARGET_ROOT18_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_890C	Target Register (CCM_TARGET_ROOT18_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8910	Miscellaneous Register (CCM_MISC18)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8914	Miscellaneous Register (CCM_MISC_ROOT18_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8918	Miscellaneous Register (CCM_MISC_ROOT18_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_891C	Miscellaneous Register (CCM_MISC_ROOT18_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8920	Post Divider Register (CCM_POST18)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8924	Post Divider Register (CCM_POST_ROOT18_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8928	Post Divider Register (CCM_POST_ROOT18_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_892C	Post Divider Register (CCM_POST_ROOT18_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8930	Pre Divider Register (CCM_PRE18)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8934	Pre Divider Register (CCM_PRE_ROOT18_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8938	Pre Divider Register (CCM_PRE_ROOT18_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_893C	Pre Divider Register (CCM_PRE_ROOT18_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8970	Access Control Register (CCM_ACCESS_CTRL18)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8974	Access Control Register (CCM_ACCESS_CTRL_ROOT18_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8978	Access Control Register (CCM_ACCESS_CTRL_ROOT18_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_897C	Access Control Register (CCM_ACCESS_CTRL_ROOT18_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8980	Target Register (CCM_TARGET_ROOT19)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8984	Target Register (CCM_TARGET_ROOT19_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8988	Target Register (CCM_TARGET_ROOT19_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_898C	Target Register (CCM_TARGET_ROOT19_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8990	Miscellaneous Register (CCM_MISC19)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8994	Miscellaneous Register (CCM_MISC_ROOT19_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8998	Miscellaneous Register (CCM_MISC_ROOT19_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_899C	Miscellaneous Register (CCM_MISC_ROOT19_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_89A0	Post Divider Register (CCM_POST19)	32	R/W	0000_0000h	5.1.7.18/ 480

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_89A4	Post Divider Register (CCM_POST_ROOT19_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_89A8	Post Divider Register (CCM_POST_ROOT19_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_89AC	Post Divider Register (CCM_POST_ROOT19_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_89B0	Pre Divider Register (CCM_PRE19)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_89B4	Pre Divider Register (CCM_PRE_ROOT19_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_89B8	Pre Divider Register (CCM_PRE_ROOT19_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_89BC	Pre Divider Register (CCM_PRE_ROOT19_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_89F0	Access Control Register (CCM_ACCESS_CTRL19)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_89F4	Access Control Register (CCM_ACCESS_CTRL_ROOT19_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_89F8	Access Control Register (CCM_ACCESS_CTRL_ROOT19_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_89FC	Access Control Register (CCM_ACCESS_CTRL_ROOT19_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8A00	Target Register (CCM_TARGET_ROOT20)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8A04	Target Register (CCM_TARGET_ROOT20_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8A08	Target Register (CCM_TARGET_ROOT20_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_8A0C	Target Register (CCM_TARGET_ROOT20_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8A10	Miscellaneous Register (CCM_MISC20)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8A14	Miscellaneous Register (CCM_MISC_ROOT20_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8A18	Miscellaneous Register (CCM_MISC_ROOT20_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_8A1C	Miscellaneous Register (CCM_MISC_ROOT20_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8A20	Post Divider Register (CCM_POST20)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8A24	Post Divider Register (CCM_POST_ROOT20_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8A28	Post Divider Register (CCM_POST_ROOT20_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8A2C	Post Divider Register (CCM_POST_ROOT20_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8A30	Pre Divider Register (CCM_PRE20)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8A34	Pre Divider Register (CCM_PRE_ROOT20_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8A38	Pre Divider Register (CCM_PRE_ROOT20_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_8A3C	Pre Divider Register (CCM_PRE_ROOT20_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8A70	Access Control Register (CCM_ACCESS_CTRL20)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8A74	Access Control Register (CCM_ACCESS_CTRL_ROOT20_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8A78	Access Control Register (CCM_ACCESS_CTRL_ROOT20_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_8A7C	Access Control Register (CCM_ACCESS_CTRL_ROOT20_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8A80	Target Register (CCM_TARGET_ROOT21)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8A84	Target Register (CCM_TARGET_ROOT21_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8A88	Target Register (CCM_TARGET_ROOT21_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_8A8C	Target Register (CCM_TARGET_ROOT21_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8A90	Miscellaneous Register (CCM_MISC21)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8A94	Miscellaneous Register (CCM_MISC_ROOT21_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8A98	Miscellaneous Register (CCM_MISC_ROOT21_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_8A9C	Miscellaneous Register (CCM_MISC_ROOT21_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8AA0	Post Divider Register (CCM_POST21)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8AA4	Post Divider Register (CCM_POST_ROOT21_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8AA8	Post Divider Register (CCM_POST_ROOT21_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_8AAC	Post Divider Register (CCM_POST_ROOT21_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8AB0	Pre Divider Register (CCM_PRE21)	32	R/W	1000_0000h	5.1.7.22/ 492

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8AB4	Pre Divider Register (CCM_PRE_ROOT21_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8AB8	Pre Divider Register (CCM_PRE_ROOT21_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_8ABC	Pre Divider Register (CCM_PRE_ROOT21_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8AF0	Access Control Register (CCM_ACCESS_CTRL21)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8AF4	Access Control Register (CCM_ACCESS_CTRL_ROOT21_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8AF8	Access Control Register (CCM_ACCESS_CTRL_ROOT21_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_8AFC	Access Control Register (CCM_ACCESS_CTRL_ROOT21_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8B00	Target Register (CCM_TARGET_ROOT22)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8B04	Target Register (CCM_TARGET_ROOT22_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8B08	Target Register (CCM_TARGET_ROOT22_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_8B0C	Target Register (CCM_TARGET_ROOT22_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8B10	Miscellaneous Register (CCM_MISC22)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8B14	Miscellaneous Register (CCM_MISC_ROOT22_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8B18	Miscellaneous Register (CCM_MISC_ROOT22_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_8B1C	Miscellaneous Register (CCM_MISC_ROOT22_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8B20	Post Divider Register (CCM_POST22)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8B24	Post Divider Register (CCM_POST_ROOT22_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8B28	Post Divider Register (CCM_POST_ROOT22_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_8B2C	Post Divider Register (CCM_POST_ROOT22_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8B30	Pre Divider Register (CCM_PRE22)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8B34	Pre Divider Register (CCM_PRE_ROOT22_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8B38	Pre Divider Register (CCM_PRE_ROOT22_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8B3C	Pre Divider Register (CCM_PRE_ROOT22_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8B70	Access Control Register (CCM_ACCESS_CTRL22)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8B74	Access Control Register (CCM_ACCESS_CTRL_ROOT22_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8B78	Access Control Register (CCM_ACCESS_CTRL_ROOT22_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_8B7C	Access Control Register (CCM_ACCESS_CTRL_ROOT22_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8B80	Target Register (CCM_TARGET_ROOT23)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8B84	Target Register (CCM_TARGET_ROOT23_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8B88	Target Register (CCM_TARGET_ROOT23_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_8B8C	Target Register (CCM_TARGET_ROOT23_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8B90	Miscellaneous Register (CCM_MISC23)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8B94	Miscellaneous Register (CCM_MISC_ROOT23_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8B98	Miscellaneous Register (CCM_MISC_ROOT23_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_8B9C	Miscellaneous Register (CCM_MISC_ROOT23_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8BA0	Post Divider Register (CCM_POST23)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8BA4	Post Divider Register (CCM_POST_ROOT23_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8BA8	Post Divider Register (CCM_POST_ROOT23_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_8BAC	Post Divider Register (CCM_POST_ROOT23_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8BB0	Pre Divider Register (CCM_PRE23)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8BB4	Pre Divider Register (CCM_PRE_ROOT23_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8BB8	Pre Divider Register (CCM_PRE_ROOT23_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_8BBC	Pre Divider Register (CCM_PRE_ROOT23_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8BF0	Access Control Register (CCM_ACCESS_CTRL23)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8BF4	Access Control Register (CCM_ACCESS_CTRL_ROOT23_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8BF8	Access Control Register (CCM_ACCESS_CTRL_ROOT23_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_8BFC	Access Control Register (CCM_ACCESS_CTRL_ROOT23_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8C00	Target Register (CCM_TARGET_ROOT24)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8C04	Target Register (CCM_TARGET_ROOT24_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8C08	Target Register (CCM_TARGET_ROOT24_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_8C0C	Target Register (CCM_TARGET_ROOT24_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8C10	Miscellaneous Register (CCM_MISC24)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8C14	Miscellaneous Register (CCM_MISC_ROOT24_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8C18	Miscellaneous Register (CCM_MISC_ROOT24_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_8C1C	Miscellaneous Register (CCM_MISC_ROOT24_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8C20	Post Divider Register (CCM_POST24)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8C24	Post Divider Register (CCM_POST_ROOT24_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8C28	Post Divider Register (CCM_POST_ROOT24_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_8C2C	Post Divider Register (CCM_POST_ROOT24_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8C30	Pre Divider Register (CCM_PRE24)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8C34	Pre Divider Register (CCM_PRE_ROOT24_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8C38	Pre Divider Register (CCM_PRE_ROOT24_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_8C3C	Pre Divider Register (CCM_PRE_ROOT24_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8C70	Access Control Register (CCM_ACCESS_CTRL24)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8C74	Access Control Register (CCM_ACCESS_CTRL_ROOT24_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8C78	Access Control Register (CCM_ACCESS_CTRL_ROOT24_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8C7C	Access Control Register (CCM_ACCESS_CTRL_ROOT24_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8C80	Target Register (CCM_TARGET_ROOT25)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8C84	Target Register (CCM_TARGET_ROOT25_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8C88	Target Register (CCM_TARGET_ROOT25_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_8C8C	Target Register (CCM_TARGET_ROOT25_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8C90	Miscellaneous Register (CCM_MISC25)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8C94	Miscellaneous Register (CCM_MISC_ROOT25_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8C98	Miscellaneous Register (CCM_MISC_ROOT25_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_8C9C	Miscellaneous Register (CCM_MISC_ROOT25_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8CA0	Post Divider Register (CCM_POST25)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8CA4	Post Divider Register (CCM_POST_ROOT25_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8CA8	Post Divider Register (CCM_POST_ROOT25_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_8CAC	Post Divider Register (CCM_POST_ROOT25_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8CB0	Pre Divider Register (CCM_PRE25)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8CB4	Pre Divider Register (CCM_PRE_ROOT25_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8CB8	Pre Divider Register (CCM_PRE_ROOT25_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_8CBC	Pre Divider Register (CCM_PRE_ROOT25_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8CF0	Access Control Register (CCM_ACCESS_CTRL25)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8CF4	Access Control Register (CCM_ACCESS_CTRL_ROOT25_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8CF8	Access Control Register (CCM_ACCESS_CTRL_ROOT25_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_8CFC	Access Control Register (CCM_ACCESS_CTRL_ROOT25_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8D00	Target Register (CCM_TARGET_ROOT26)	32	R/W	1000_0000h	5.1.7.10/ 468

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8D04	Target Register (CCM_TARGET_ROOT26_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8D08	Target Register (CCM_TARGET_ROOT26_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_8D0C	Target Register (CCM_TARGET_ROOT26_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8D10	Miscellaneous Register (CCM_MISC26)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8D14	Miscellaneous Register (CCM_MISC_ROOT26_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8D18	Miscellaneous Register (CCM_MISC_ROOT26_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_8D1C	Miscellaneous Register (CCM_MISC_ROOT26_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8D20	Post Divider Register (CCM_POST26)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8D24	Post Divider Register (CCM_POST_ROOT26_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8D28	Post Divider Register (CCM_POST_ROOT26_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_8D2C	Post Divider Register (CCM_POST_ROOT26_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8D30	Pre Divider Register (CCM_PRE26)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8D34	Pre Divider Register (CCM_PRE_ROOT26_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8D38	Pre Divider Register (CCM_PRE_ROOT26_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_8D3C	Pre Divider Register (CCM_PRE_ROOT26_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8D70	Access Control Register (CCM_ACCESS_CTRL26)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8D74	Access Control Register (CCM_ACCESS_CTRL_ROOT26_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8D78	Access Control Register (CCM_ACCESS_CTRL_ROOT26_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_8D7C	Access Control Register (CCM_ACCESS_CTRL_ROOT26_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8D80	Target Register (CCM_TARGET_ROOT27)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8D84	Target Register (CCM_TARGET_ROOT27_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8D88	Target Register (CCM_TARGET_ROOT27_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8D8C	Target Register (CCM_TARGET_ROOT27_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8D90	Miscellaneous Register (CCM_MISC27)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8D94	Miscellaneous Register (CCM_MISC_ROOT27_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8D98	Miscellaneous Register (CCM_MISC_ROOT27_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_8D9C	Miscellaneous Register (CCM_MISC_ROOT27_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8DA0	Post Divider Register (CCM_POST27)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8DA4	Post Divider Register (CCM_POST_ROOT27_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8DA8	Post Divider Register (CCM_POST_ROOT27_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_8DAC	Post Divider Register (CCM_POST_ROOT27_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8DB0	Pre Divider Register (CCM_PRE27)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8DB4	Pre Divider Register (CCM_PRE_ROOT27_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8DB8	Pre Divider Register (CCM_PRE_ROOT27_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_8DBC	Pre Divider Register (CCM_PRE_ROOT27_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8DF0	Access Control Register (CCM_ACCESS_CTRL27)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8DF4	Access Control Register (CCM_ACCESS_CTRL_ROOT27_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8DF8	Access Control Register (CCM_ACCESS_CTRL_ROOT27_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_8DFC	Access Control Register (CCM_ACCESS_CTRL_ROOT27_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8E00	Target Register (CCM_TARGET_ROOT28)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8E04	Target Register (CCM_TARGET_ROOT28_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8E08	Target Register (CCM_TARGET_ROOT28_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_8E0C	Target Register (CCM_TARGET_ROOT28_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8E10	Miscellaneous Register (CCM_MISC28)	32	R/W	0000_0000h	5.1.7.14/ 476

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8E14	Miscellaneous Register (CCM_MISC_ROOT28_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8E18	Miscellaneous Register (CCM_MISC_ROOT28_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_8E1C	Miscellaneous Register (CCM_MISC_ROOT28_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8E20	Post Divider Register (CCM_POST28)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8E24	Post Divider Register (CCM_POST_ROOT28_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8E28	Post Divider Register (CCM_POST_ROOT28_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_8E2C	Post Divider Register (CCM_POST_ROOT28_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8E30	Pre Divider Register (CCM_PRE28)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8E34	Pre Divider Register (CCM_PRE_ROOT28_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8E38	Pre Divider Register (CCM_PRE_ROOT28_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_8E3C	Pre Divider Register (CCM_PRE_ROOT28_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8E70	Access Control Register (CCM_ACCESS_CTRL28)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8E74	Access Control Register (CCM_ACCESS_CTRL_ROOT28_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8E78	Access Control Register (CCM_ACCESS_CTRL_ROOT28_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_8E7C	Access Control Register (CCM_ACCESS_CTRL_ROOT28_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8E80	Target Register (CCM_TARGET_ROOT29)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8E84	Target Register (CCM_TARGET_ROOT29_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8E88	Target Register (CCM_TARGET_ROOT29_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_8E8C	Target Register (CCM_TARGET_ROOT29_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8E90	Miscellaneous Register (CCM_MISC29)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8E94	Miscellaneous Register (CCM_MISC_ROOT29_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8E98	Miscellaneous Register (CCM_MISC_ROOT29_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8E9C	Miscellaneous Register (CCM_MISC_ROOT29_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8EA0	Post Divider Register (CCM_POST29)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8EA4	Post Divider Register (CCM_POST_ROOT29_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8EA8	Post Divider Register (CCM_POST_ROOT29_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_8EAC	Post Divider Register (CCM_POST_ROOT29_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8EB0	Pre Divider Register (CCM_PRE29)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8EB4	Pre Divider Register (CCM_PRE_ROOT29_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8EB8	Pre Divider Register (CCM_PRE_ROOT29_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_8EBC	Pre Divider Register (CCM_PRE_ROOT29_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8EF0	Access Control Register (CCM_ACCESS_CTRL29)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8EF4	Access Control Register (CCM_ACCESS_CTRL_ROOT29_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8EF8	Access Control Register (CCM_ACCESS_CTRL_ROOT29_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_8EFC	Access Control Register (CCM_ACCESS_CTRL_ROOT29_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8F00	Target Register (CCM_TARGET_ROOT30)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8F04	Target Register (CCM_TARGET_ROOT30_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8F08	Target Register (CCM_TARGET_ROOT30_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_8F0C	Target Register (CCM_TARGET_ROOT30_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8F10	Miscellaneous Register (CCM_MISC30)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8F14	Miscellaneous Register (CCM_MISC_ROOT30_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8F18	Miscellaneous Register (CCM_MISC_ROOT30_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_8F1C	Miscellaneous Register (CCM_MISC_ROOT30_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8F20	Post Divider Register (CCM_POST30)	32	R/W	0000_0000h	5.1.7.18/ 480

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8F24	Post Divider Register (CCM_POST_ROOT30_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8F28	Post Divider Register (CCM_POST_ROOT30_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_8F2C	Post Divider Register (CCM_POST_ROOT30_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8F30	Pre Divider Register (CCM_PRE30)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8F34	Pre Divider Register (CCM_PRE_ROOT30_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8F38	Pre Divider Register (CCM_PRE_ROOT30_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_8F3C	Pre Divider Register (CCM_PRE_ROOT30_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8F70	Access Control Register (CCM_ACCESS_CTRL30)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8F74	Access Control Register (CCM_ACCESS_CTRL_ROOT30_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8F78	Access Control Register (CCM_ACCESS_CTRL_ROOT30_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_8F7C	Access Control Register (CCM_ACCESS_CTRL_ROOT30_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_8F80	Target Register (CCM_TARGET_ROOT31)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_8F84	Target Register (CCM_TARGET_ROOT31_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_8F88	Target Register (CCM_TARGET_ROOT31_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_8F8C	Target Register (CCM_TARGET_ROOT31_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_8F90	Miscellaneous Register (CCM_MISC31)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_8F94	Miscellaneous Register (CCM_MISC_ROOT31_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_8F98	Miscellaneous Register (CCM_MISC_ROOT31_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_8F9C	Miscellaneous Register (CCM_MISC_ROOT31_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_8FA0	Post Divider Register (CCM_POST31)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_8FA4	Post Divider Register (CCM_POST_ROOT31_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_8FA8	Post Divider Register (CCM_POST_ROOT31_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_8FAC	Post Divider Register (CCM_POST_ROOT31_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_8FB0	Pre Divider Register (CCM_PRE31)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_8FB4	Pre Divider Register (CCM_PRE_ROOT31_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_8FB8	Pre Divider Register (CCM_PRE_ROOT31_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_8FBC	Pre Divider Register (CCM_PRE_ROOT31_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_8FF0	Access Control Register (CCM_ACCESS_CTRL31)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_8FF4	Access Control Register (CCM_ACCESS_CTRL_ROOT31_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_8FF8	Access Control Register (CCM_ACCESS_CTRL_ROOT31_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_8FFC	Access Control Register (CCM_ACCESS_CTRL_ROOT31_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9000	Target Register (CCM_TARGET_ROOT32)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9004	Target Register (CCM_TARGET_ROOT32_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9008	Target Register (CCM_TARGET_ROOT32_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_900C	Target Register (CCM_TARGET_ROOT32_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9010	Miscellaneous Register (CCM_MISC32)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9014	Miscellaneous Register (CCM_MISC_ROOT32_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9018	Miscellaneous Register (CCM_MISC_ROOT32_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_901C	Miscellaneous Register (CCM_MISC_ROOT32_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9020	Post Divider Register (CCM_POST32)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9024	Post Divider Register (CCM_POST_ROOT32_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9028	Post Divider Register (CCM_POST_ROOT32_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_902C	Post Divider Register (CCM_POST_ROOT32_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9030	Pre Divider Register (CCM_PRE32)	32	R/W	1000_0000h	5.1.7.22/ 492

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9034	Pre Divider Register (CCM_PRE_ROOT32_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9038	Pre Divider Register (CCM_PRE_ROOT32_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_903C	Pre Divider Register (CCM_PRE_ROOT32_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9070	Access Control Register (CCM_ACCESS_CTRL32)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9074	Access Control Register (CCM_ACCESS_CTRL_ROOT32_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9078	Access Control Register (CCM_ACCESS_CTRL_ROOT32_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_907C	Access Control Register (CCM_ACCESS_CTRL_ROOT32_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9080	Target Register (CCM_TARGET_ROOT33)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9084	Target Register (CCM_TARGET_ROOT33_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9088	Target Register (CCM_TARGET_ROOT33_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_908C	Target Register (CCM_TARGET_ROOT33_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9090	Miscellaneous Register (CCM_MISC33)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9094	Miscellaneous Register (CCM_MISC_ROOT33_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9098	Miscellaneous Register (CCM_MISC_ROOT33_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_909C	Miscellaneous Register (CCM_MISC_ROOT33_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_90A0	Post Divider Register (CCM_POST33)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_90A4	Post Divider Register (CCM_POST_ROOT33_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_90A8	Post Divider Register (CCM_POST_ROOT33_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_90AC	Post Divider Register (CCM_POST_ROOT33_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_90B0	Pre Divider Register (CCM_PRE33)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_90B4	Pre Divider Register (CCM_PRE_ROOT33_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_90B8	Pre Divider Register (CCM_PRE_ROOT33_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_90BC	Pre Divider Register (CCM_PRE_ROOT33_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_90F0	Access Control Register (CCM_ACCESS_CTRL33)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_90F4	Access Control Register (CCM_ACCESS_CTRL_ROOT33_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_90F8	Access Control Register (CCM_ACCESS_CTRL_ROOT33_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_90FC	Access Control Register (CCM_ACCESS_CTRL_ROOT33_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9100	Target Register (CCM_TARGET_ROOT34)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9104	Target Register (CCM_TARGET_ROOT34_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9108	Target Register (CCM_TARGET_ROOT34_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_910C	Target Register (CCM_TARGET_ROOT34_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9110	Miscellaneous Register (CCM_MISC34)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9114	Miscellaneous Register (CCM_MISC_ROOT34_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9118	Miscellaneous Register (CCM_MISC_ROOT34_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_911C	Miscellaneous Register (CCM_MISC_ROOT34_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9120	Post Divider Register (CCM_POST34)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9124	Post Divider Register (CCM_POST_ROOT34_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9128	Post Divider Register (CCM_POST_ROOT34_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_912C	Post Divider Register (CCM_POST_ROOT34_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9130	Pre Divider Register (CCM_PRE34)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9134	Pre Divider Register (CCM_PRE_ROOT34_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9138	Pre Divider Register (CCM_PRE_ROOT34_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_913C	Pre Divider Register (CCM_PRE_ROOT34_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9170	Access Control Register (CCM_ACCESS_CTRL34)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9174	Access Control Register (CCM_ACCESS_CTRL_ROOT34_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9178	Access Control Register (CCM_ACCESS_CTRL_ROOT34_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_917C	Access Control Register (CCM_ACCESS_CTRL_ROOT34_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9180	Target Register (CCM_TARGET_ROOT35)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9184	Target Register (CCM_TARGET_ROOT35_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9188	Target Register (CCM_TARGET_ROOT35_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_918C	Target Register (CCM_TARGET_ROOT35_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9190	Miscellaneous Register (CCM_MISC35)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9194	Miscellaneous Register (CCM_MISC_ROOT35_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9198	Miscellaneous Register (CCM_MISC_ROOT35_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_919C	Miscellaneous Register (CCM_MISC_ROOT35_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_91A0	Post Divider Register (CCM_POST35)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_91A4	Post Divider Register (CCM_POST_ROOT35_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_91A8	Post Divider Register (CCM_POST_ROOT35_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_91AC	Post Divider Register (CCM_POST_ROOT35_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_91B0	Pre Divider Register (CCM_PRE35)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_91B4	Pre Divider Register (CCM_PRE_ROOT35_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_91B8	Pre Divider Register (CCM_PRE_ROOT35_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_91BC	Pre Divider Register (CCM_PRE_ROOT35_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_91F0	Access Control Register (CCM_ACCESS_CTRL35)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_91F4	Access Control Register (CCM_ACCESS_CTRL_ROOT35_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_91F8	Access Control Register (CCM_ACCESS_CTRL_ROOT35_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_91FC	Access Control Register (CCM_ACCESS_CTRL_ROOT35_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9200	Target Register (CCM_TARGET_ROOT36)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9204	Target Register (CCM_TARGET_ROOT36_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9208	Target Register (CCM_TARGET_ROOT36_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_920C	Target Register (CCM_TARGET_ROOT36_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9210	Miscellaneous Register (CCM_MISC36)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9214	Miscellaneous Register (CCM_MISC_ROOT36_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9218	Miscellaneous Register (CCM_MISC_ROOT36_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_921C	Miscellaneous Register (CCM_MISC_ROOT36_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9220	Post Divider Register (CCM_POST36)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9224	Post Divider Register (CCM_POST_ROOT36_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9228	Post Divider Register (CCM_POST_ROOT36_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_922C	Post Divider Register (CCM_POST_ROOT36_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9230	Pre Divider Register (CCM_PRE36)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9234	Pre Divider Register (CCM_PRE_ROOT36_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9238	Pre Divider Register (CCM_PRE_ROOT36_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_923C	Pre Divider Register (CCM_PRE_ROOT36_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9270	Access Control Register (CCM_ACCESS_CTRL36)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9274	Access Control Register (CCM_ACCESS_CTRL_ROOT36_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9278	Access Control Register (CCM_ACCESS_CTRL_ROOT36_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_927C	Access Control Register (CCM_ACCESS_CTRL_ROOT36_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9280	Target Register (CCM_TARGET_ROOT37)	32	R/W	1000_0000h	5.1.7.10/ 468

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9284	Target Register (CCM_TARGET_ROOT37_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9288	Target Register (CCM_TARGET_ROOT37_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_928C	Target Register (CCM_TARGET_ROOT37_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9290	Miscellaneous Register (CCM_MISC37)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9294	Miscellaneous Register (CCM_MISC_ROOT37_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9298	Miscellaneous Register (CCM_MISC_ROOT37_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_929C	Miscellaneous Register (CCM_MISC_ROOT37_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_92A0	Post Divider Register (CCM_POST37)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_92A4	Post Divider Register (CCM_POST_ROOT37_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_92A8	Post Divider Register (CCM_POST_ROOT37_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_92AC	Post Divider Register (CCM_POST_ROOT37_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_92B0	Pre Divider Register (CCM_PRE37)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_92B4	Pre Divider Register (CCM_PRE_ROOT37_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_92B8	Pre Divider Register (CCM_PRE_ROOT37_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_92BC	Pre Divider Register (CCM_PRE_ROOT37_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_92F0	Access Control Register (CCM_ACCESS_CTRL37)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_92F4	Access Control Register (CCM_ACCESS_CTRL_ROOT37_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_92F8	Access Control Register (CCM_ACCESS_CTRL_ROOT37_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_92FC	Access Control Register (CCM_ACCESS_CTRL_ROOT37_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9300	Target Register (CCM_TARGET_ROOT38)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9304	Target Register (CCM_TARGET_ROOT38_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9308	Target Register (CCM_TARGET_ROOT38_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_930C	Target Register (CCM_TARGET_ROOT38_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9310	Miscellaneous Register (CCM_MISC38)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9314	Miscellaneous Register (CCM_MISC_ROOT38_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9318	Miscellaneous Register (CCM_MISC_ROOT38_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_931C	Miscellaneous Register (CCM_MISC_ROOT38_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9320	Post Divider Register (CCM_POST38)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9324	Post Divider Register (CCM_POST_ROOT38_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9328	Post Divider Register (CCM_POST_ROOT38_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_932C	Post Divider Register (CCM_POST_ROOT38_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9330	Pre Divider Register (CCM_PRE38)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9334	Pre Divider Register (CCM_PRE_ROOT38_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9338	Pre Divider Register (CCM_PRE_ROOT38_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_933C	Pre Divider Register (CCM_PRE_ROOT38_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9370	Access Control Register (CCM_ACCESS_CTRL38)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9374	Access Control Register (CCM_ACCESS_CTRL_ROOT38_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9378	Access Control Register (CCM_ACCESS_CTRL_ROOT38_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_937C	Access Control Register (CCM_ACCESS_CTRL_ROOT38_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9380	Target Register (CCM_TARGET_ROOT39)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9384	Target Register (CCM_TARGET_ROOT39_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9388	Target Register (CCM_TARGET_ROOT39_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_938C	Target Register (CCM_TARGET_ROOT39_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9390	Miscellaneous Register (CCM_MISC39)	32	R/W	0000_0000h	5.1.7.14/ 476

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9394	Miscellaneous Register (CCM_MISC_ROOT39_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9398	Miscellaneous Register (CCM_MISC_ROOT39_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_939C	Miscellaneous Register (CCM_MISC_ROOT39_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_93A0	Post Divider Register (CCM_POST39)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_93A4	Post Divider Register (CCM_POST_ROOT39_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_93A8	Post Divider Register (CCM_POST_ROOT39_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_93AC	Post Divider Register (CCM_POST_ROOT39_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_93B0	Pre Divider Register (CCM_PRE39)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_93B4	Pre Divider Register (CCM_PRE_ROOT39_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_93B8	Pre Divider Register (CCM_PRE_ROOT39_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_93BC	Pre Divider Register (CCM_PRE_ROOT39_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_93F0	Access Control Register (CCM_ACCESS_CTRL39)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_93F4	Access Control Register (CCM_ACCESS_CTRL_ROOT39_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_93F8	Access Control Register (CCM_ACCESS_CTRL_ROOT39_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_93FC	Access Control Register (CCM_ACCESS_CTRL_ROOT39_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9400	Target Register (CCM_TARGET_ROOT40)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9404	Target Register (CCM_TARGET_ROOT40_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9408	Target Register (CCM_TARGET_ROOT40_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_940C	Target Register (CCM_TARGET_ROOT40_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9410	Miscellaneous Register (CCM_MISC40)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9414	Miscellaneous Register (CCM_MISC_ROOT40_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9418	Miscellaneous Register (CCM_MISC_ROOT40_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_941C	Miscellaneous Register (CCM_MISC_ROOT40_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9420	Post Divider Register (CCM_POST40)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9424	Post Divider Register (CCM_POST_ROOT40_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9428	Post Divider Register (CCM_POST_ROOT40_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_942C	Post Divider Register (CCM_POST_ROOT40_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9430	Pre Divider Register (CCM_PRE40)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9434	Pre Divider Register (CCM_PRE_ROOT40_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9438	Pre Divider Register (CCM_PRE_ROOT40_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_943C	Pre Divider Register (CCM_PRE_ROOT40_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9470	Access Control Register (CCM_ACCESS_CTRL40)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9474	Access Control Register (CCM_ACCESS_CTRL_ROOT40_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9478	Access Control Register (CCM_ACCESS_CTRL_ROOT40_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_947C	Access Control Register (CCM_ACCESS_CTRL_ROOT40_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9480	Target Register (CCM_TARGET_ROOT41)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9484	Target Register (CCM_TARGET_ROOT41_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9488	Target Register (CCM_TARGET_ROOT41_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_948C	Target Register (CCM_TARGET_ROOT41_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9490	Miscellaneous Register (CCM_MISC41)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9494	Miscellaneous Register (CCM_MISC_ROOT41_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9498	Miscellaneous Register (CCM_MISC_ROOT41_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_949C	Miscellaneous Register (CCM_MISC_ROOT41_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_94A0	Post Divider Register (CCM_POST41)	32	R/W	0000_0000h	5.1.7.18/ 480

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_94A4	Post Divider Register (CCM_POST_ROOT41_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_94A8	Post Divider Register (CCM_POST_ROOT41_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_94AC	Post Divider Register (CCM_POST_ROOT41_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_94B0	Pre Divider Register (CCM_PRE41)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_94B4	Pre Divider Register (CCM_PRE_ROOT41_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_94B8	Pre Divider Register (CCM_PRE_ROOT41_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_94BC	Pre Divider Register (CCM_PRE_ROOT41_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_94F0	Access Control Register (CCM_ACCESS_CTRL41)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_94F4	Access Control Register (CCM_ACCESS_CTRL_ROOT41_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_94F8	Access Control Register (CCM_ACCESS_CTRL_ROOT41_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_94FC	Access Control Register (CCM_ACCESS_CTRL_ROOT41_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9500	Target Register (CCM_TARGET_ROOT42)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9504	Target Register (CCM_TARGET_ROOT42_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9508	Target Register (CCM_TARGET_ROOT42_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_950C	Target Register (CCM_TARGET_ROOT42_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9510	Miscellaneous Register (CCM_MISC42)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9514	Miscellaneous Register (CCM_MISC_ROOT42_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9518	Miscellaneous Register (CCM_MISC_ROOT42_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_951C	Miscellaneous Register (CCM_MISC_ROOT42_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9520	Post Divider Register (CCM_POST42)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9524	Post Divider Register (CCM_POST_ROOT42_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9528	Post Divider Register (CCM_POST_ROOT42_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_952C	Post Divider Register (CCM_POST_ROOT42_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9530	Pre Divider Register (CCM_PRE42)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9534	Pre Divider Register (CCM_PRE_ROOT42_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9538	Pre Divider Register (CCM_PRE_ROOT42_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_953C	Pre Divider Register (CCM_PRE_ROOT42_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9570	Access Control Register (CCM_ACCESS_CTRL42)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9574	Access Control Register (CCM_ACCESS_CTRL_ROOT42_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9578	Access Control Register (CCM_ACCESS_CTRL_ROOT42_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_957C	Access Control Register (CCM_ACCESS_CTRL_ROOT42_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9580	Target Register (CCM_TARGET_ROOT43)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9584	Target Register (CCM_TARGET_ROOT43_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9588	Target Register (CCM_TARGET_ROOT43_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_958C	Target Register (CCM_TARGET_ROOT43_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9590	Miscellaneous Register (CCM_MISC43)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9594	Miscellaneous Register (CCM_MISC_ROOT43_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9598	Miscellaneous Register (CCM_MISC_ROOT43_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_959C	Miscellaneous Register (CCM_MISC_ROOT43_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_95A0	Post Divider Register (CCM_POST43)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_95A4	Post Divider Register (CCM_POST_ROOT43_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_95A8	Post Divider Register (CCM_POST_ROOT43_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_95AC	Post Divider Register (CCM_POST_ROOT43_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_95B0	Pre Divider Register (CCM_PRE43)	32	R/W	1000_0000h	5.1.7.22/ 492

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_95B4	Pre Divider Register (CCM_PRE_ROOT43_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_95B8	Pre Divider Register (CCM_PRE_ROOT43_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_95BC	Pre Divider Register (CCM_PRE_ROOT43_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_95F0	Access Control Register (CCM_ACCESS_CTRL43)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_95F4	Access Control Register (CCM_ACCESS_CTRL_ROOT43_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_95F8	Access Control Register (CCM_ACCESS_CTRL_ROOT43_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_95FC	Access Control Register (CCM_ACCESS_CTRL_ROOT43_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9600	Target Register (CCM_TARGET_ROOT44)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9604	Target Register (CCM_TARGET_ROOT44_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9608	Target Register (CCM_TARGET_ROOT44_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_960C	Target Register (CCM_TARGET_ROOT44_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9610	Miscellaneous Register (CCM_MISC44)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9614	Miscellaneous Register (CCM_MISC_ROOT44_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9618	Miscellaneous Register (CCM_MISC_ROOT44_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_961C	Miscellaneous Register (CCM_MISC_ROOT44_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9620	Post Divider Register (CCM_POST44)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9624	Post Divider Register (CCM_POST_ROOT44_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9628	Post Divider Register (CCM_POST_ROOT44_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_962C	Post Divider Register (CCM_POST_ROOT44_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9630	Pre Divider Register (CCM_PRE44)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9634	Pre Divider Register (CCM_PRE_ROOT44_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9638	Pre Divider Register (CCM_PRE_ROOT44_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_963C	Pre Divider Register (CCM_PRE_ROOT44_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9670	Access Control Register (CCM_ACCESS_CTRL44)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9674	Access Control Register (CCM_ACCESS_CTRL_ROOT44_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9678	Access Control Register (CCM_ACCESS_CTRL_ROOT44_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_967C	Access Control Register (CCM_ACCESS_CTRL_ROOT44_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9680	Target Register (CCM_TARGET_ROOT45)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9684	Target Register (CCM_TARGET_ROOT45_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9688	Target Register (CCM_TARGET_ROOT45_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_968C	Target Register (CCM_TARGET_ROOT45_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9690	Miscellaneous Register (CCM_MISC45)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9694	Miscellaneous Register (CCM_MISC_ROOT45_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9698	Miscellaneous Register (CCM_MISC_ROOT45_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_969C	Miscellaneous Register (CCM_MISC_ROOT45_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_96A0	Post Divider Register (CCM_POST45)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_96A4	Post Divider Register (CCM_POST_ROOT45_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_96A8	Post Divider Register (CCM_POST_ROOT45_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_96AC	Post Divider Register (CCM_POST_ROOT45_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_96B0	Pre Divider Register (CCM_PRE45)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_96B4	Pre Divider Register (CCM_PRE_ROOT45_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_96B8	Pre Divider Register (CCM_PRE_ROOT45_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_96BC	Pre Divider Register (CCM_PRE_ROOT45_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_96F0	Access Control Register (CCM_ACCESS_CTRL45)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_96F4	Access Control Register (CCM_ACCESS_CTRL_ROOT45_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_96F8	Access Control Register (CCM_ACCESS_CTRL_ROOT45_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_96FC	Access Control Register (CCM_ACCESS_CTRL_ROOT45_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9700	Target Register (CCM_TARGET_ROOT46)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9704	Target Register (CCM_TARGET_ROOT46_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9708	Target Register (CCM_TARGET_ROOT46_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_970C	Target Register (CCM_TARGET_ROOT46_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9710	Miscellaneous Register (CCM_MISC46)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9714	Miscellaneous Register (CCM_MISC_ROOT46_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9718	Miscellaneous Register (CCM_MISC_ROOT46_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_971C	Miscellaneous Register (CCM_MISC_ROOT46_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9720	Post Divider Register (CCM_POST46)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9724	Post Divider Register (CCM_POST_ROOT46_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9728	Post Divider Register (CCM_POST_ROOT46_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_972C	Post Divider Register (CCM_POST_ROOT46_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9730	Pre Divider Register (CCM_PRE46)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9734	Pre Divider Register (CCM_PRE_ROOT46_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9738	Pre Divider Register (CCM_PRE_ROOT46_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_973C	Pre Divider Register (CCM_PRE_ROOT46_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9770	Access Control Register (CCM_ACCESS_CTRL46)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9774	Access Control Register (CCM_ACCESS_CTRL_ROOT46_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9778	Access Control Register (CCM_ACCESS_CTRL_ROOT46_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_977C	Access Control Register (CCM_ACCESS_CTRL_ROOT46_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9780	Target Register (CCM_TARGET_ROOT47)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9784	Target Register (CCM_TARGET_ROOT47_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9788	Target Register (CCM_TARGET_ROOT47_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_978C	Target Register (CCM_TARGET_ROOT47_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9790	Miscellaneous Register (CCM_MISC47)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9794	Miscellaneous Register (CCM_MISC_ROOT47_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9798	Miscellaneous Register (CCM_MISC_ROOT47_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_979C	Miscellaneous Register (CCM_MISC_ROOT47_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_97A0	Post Divider Register (CCM_POST47)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_97A4	Post Divider Register (CCM_POST_ROOT47_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_97A8	Post Divider Register (CCM_POST_ROOT47_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_97AC	Post Divider Register (CCM_POST_ROOT47_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_97B0	Pre Divider Register (CCM_PRE47)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_97B4	Pre Divider Register (CCM_PRE_ROOT47_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_97B8	Pre Divider Register (CCM_PRE_ROOT47_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_97BC	Pre Divider Register (CCM_PRE_ROOT47_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_97F0	Access Control Register (CCM_ACCESS_CTRL47)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_97F4	Access Control Register (CCM_ACCESS_CTRL_ROOT47_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_97F8	Access Control Register (CCM_ACCESS_CTRL_ROOT47_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_97FC	Access Control Register (CCM_ACCESS_CTRL_ROOT47_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9800	Target Register (CCM_TARGET_ROOT48)	32	R/W	1000_0000h	5.1.7.10/ 468

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9804	Target Register (CCM_TARGET_ROOT48_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9808	Target Register (CCM_TARGET_ROOT48_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_980C	Target Register (CCM_TARGET_ROOT48_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9810	Miscellaneous Register (CCM_MISC48)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9814	Miscellaneous Register (CCM_MISC_ROOT48_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9818	Miscellaneous Register (CCM_MISC_ROOT48_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_981C	Miscellaneous Register (CCM_MISC_ROOT48_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9820	Post Divider Register (CCM_POST48)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9824	Post Divider Register (CCM_POST_ROOT48_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9828	Post Divider Register (CCM_POST_ROOT48_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_982C	Post Divider Register (CCM_POST_ROOT48_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9830	Pre Divider Register (CCM_PRE48)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9834	Pre Divider Register (CCM_PRE_ROOT48_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9838	Pre Divider Register (CCM_PRE_ROOT48_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_983C	Pre Divider Register (CCM_PRE_ROOT48_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9870	Access Control Register (CCM_ACCESS_CTRL48)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9874	Access Control Register (CCM_ACCESS_CTRL_ROOT48_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9878	Access Control Register (CCM_ACCESS_CTRL_ROOT48_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_987C	Access Control Register (CCM_ACCESS_CTRL_ROOT48_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9880	Target Register (CCM_TARGET_ROOT49)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9884	Target Register (CCM_TARGET_ROOT49_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9888	Target Register (CCM_TARGET_ROOT49_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_988C	Target Register (CCM_TARGET_ROOT49_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9890	Miscellaneous Register (CCM_MISC49)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9894	Miscellaneous Register (CCM_MISC_ROOT49_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9898	Miscellaneous Register (CCM_MISC_ROOT49_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_989C	Miscellaneous Register (CCM_MISC_ROOT49_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_98A0	Post Divider Register (CCM_POST49)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_98A4	Post Divider Register (CCM_POST_ROOT49_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_98A8	Post Divider Register (CCM_POST_ROOT49_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_98AC	Post Divider Register (CCM_POST_ROOT49_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_98B0	Pre Divider Register (CCM_PRE49)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_98B4	Pre Divider Register (CCM_PRE_ROOT49_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_98B8	Pre Divider Register (CCM_PRE_ROOT49_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_98BC	Pre Divider Register (CCM_PRE_ROOT49_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_98F0	Access Control Register (CCM_ACCESS_CTRL49)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_98F4	Access Control Register (CCM_ACCESS_CTRL_ROOT49_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_98F8	Access Control Register (CCM_ACCESS_CTRL_ROOT49_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_98FC	Access Control Register (CCM_ACCESS_CTRL_ROOT49_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9900	Target Register (CCM_TARGET_ROOT50)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9904	Target Register (CCM_TARGET_ROOT50_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9908	Target Register (CCM_TARGET_ROOT50_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_990C	Target Register (CCM_TARGET_ROOT50_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9910	Miscellaneous Register (CCM_MISC50)	32	R/W	0000_0000h	5.1.7.14/ 476

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9914	Miscellaneous Register (CCM_MISC_ROOT50_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9918	Miscellaneous Register (CCM_MISC_ROOT50_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_991C	Miscellaneous Register (CCM_MISC_ROOT50_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9920	Post Divider Register (CCM_POST50)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9924	Post Divider Register (CCM_POST_ROOT50_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9928	Post Divider Register (CCM_POST_ROOT50_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_992C	Post Divider Register (CCM_POST_ROOT50_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9930	Pre Divider Register (CCM_PRE50)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9934	Pre Divider Register (CCM_PRE_ROOT50_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9938	Pre Divider Register (CCM_PRE_ROOT50_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_993C	Pre Divider Register (CCM_PRE_ROOT50_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9970	Access Control Register (CCM_ACCESS_CTRL50)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9974	Access Control Register (CCM_ACCESS_CTRL_ROOT50_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9978	Access Control Register (CCM_ACCESS_CTRL_ROOT50_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_997C	Access Control Register (CCM_ACCESS_CTRL_ROOT50_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9980	Target Register (CCM_TARGET_ROOT51)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9984	Target Register (CCM_TARGET_ROOT51_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9988	Target Register (CCM_TARGET_ROOT51_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_998C	Target Register (CCM_TARGET_ROOT51_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9990	Miscellaneous Register (CCM_MISC51)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9994	Miscellaneous Register (CCM_MISC_ROOT51_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9998	Miscellaneous Register (CCM_MISC_ROOT51_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_999C	Miscellaneous Register (CCM_MISC_ROOT51_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_99A0	Post Divider Register (CCM_POST51)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_99A4	Post Divider Register (CCM_POST_ROOT51_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_99A8	Post Divider Register (CCM_POST_ROOT51_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_99AC	Post Divider Register (CCM_POST_ROOT51_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_99B0	Pre Divider Register (CCM_PRE51)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_99B4	Pre Divider Register (CCM_PRE_ROOT51_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_99B8	Pre Divider Register (CCM_PRE_ROOT51_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_99BC	Pre Divider Register (CCM_PRE_ROOT51_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_99F0	Access Control Register (CCM_ACCESS_CTRL51)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_99F4	Access Control Register (CCM_ACCESS_CTRL_ROOT51_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_99F8	Access Control Register (CCM_ACCESS_CTRL_ROOT51_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_99FC	Access Control Register (CCM_ACCESS_CTRL_ROOT51_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9A00	Target Register (CCM_TARGET_ROOT52)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9A04	Target Register (CCM_TARGET_ROOT52_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9A08	Target Register (CCM_TARGET_ROOT52_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_9A0C	Target Register (CCM_TARGET_ROOT52_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9A10	Miscellaneous Register (CCM_MISC52)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9A14	Miscellaneous Register (CCM_MISC_ROOT52_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9A18	Miscellaneous Register (CCM_MISC_ROOT52_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_9A1C	Miscellaneous Register (CCM_MISC_ROOT52_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9A20	Post Divider Register (CCM_POST52)	32	R/W	0000_0000h	5.1.7.18/ 480

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9A24	Post Divider Register (CCM_POST_ROOT52_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9A28	Post Divider Register (CCM_POST_ROOT52_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_9A2C	Post Divider Register (CCM_POST_ROOT52_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9A30	Pre Divider Register (CCM_PRE52)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9A34	Pre Divider Register (CCM_PRE_ROOT52_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9A38	Pre Divider Register (CCM_PRE_ROOT52_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_9A3C	Pre Divider Register (CCM_PRE_ROOT52_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9A70	Access Control Register (CCM_ACCESS_CTRL52)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9A74	Access Control Register (CCM_ACCESS_CTRL_ROOT52_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9A78	Access Control Register (CCM_ACCESS_CTRL_ROOT52_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_9A7C	Access Control Register (CCM_ACCESS_CTRL_ROOT52_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9A80	Target Register (CCM_TARGET_ROOT53)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9A84	Target Register (CCM_TARGET_ROOT53_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9A88	Target Register (CCM_TARGET_ROOT53_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_9A8C	Target Register (CCM_TARGET_ROOT53_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9A90	Miscellaneous Register (CCM_MISC53)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9A94	Miscellaneous Register (CCM_MISC_ROOT53_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9A98	Miscellaneous Register (CCM_MISC_ROOT53_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_9A9C	Miscellaneous Register (CCM_MISC_ROOT53_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9AA0	Post Divider Register (CCM_POST53)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9AA4	Post Divider Register (CCM_POST_ROOT53_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9AA8	Post Divider Register (CCM_POST_ROOT53_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9AAC	Post Divider Register (CCM_POST_ROOT53_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9AB0	Pre Divider Register (CCM_PRE53)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9AB4	Pre Divider Register (CCM_PRE_ROOT53_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9AB8	Pre Divider Register (CCM_PRE_ROOT53_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_9ABC	Pre Divider Register (CCM_PRE_ROOT53_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9AF0	Access Control Register (CCM_ACCESS_CTRL53)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9AF4	Access Control Register (CCM_ACCESS_CTRL_ROOT53_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9AF8	Access Control Register (CCM_ACCESS_CTRL_ROOT53_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_9AFC	Access Control Register (CCM_ACCESS_CTRL_ROOT53_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9B00	Target Register (CCM_TARGET_ROOT54)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9B04	Target Register (CCM_TARGET_ROOT54_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9B08	Target Register (CCM_TARGET_ROOT54_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_9B0C	Target Register (CCM_TARGET_ROOT54_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9B10	Miscellaneous Register (CCM_MISC54)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9B14	Miscellaneous Register (CCM_MISC_ROOT54_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9B18	Miscellaneous Register (CCM_MISC_ROOT54_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_9B1C	Miscellaneous Register (CCM_MISC_ROOT54_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9B20	Post Divider Register (CCM_POST54)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9B24	Post Divider Register (CCM_POST_ROOT54_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9B28	Post Divider Register (CCM_POST_ROOT54_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_9B2C	Post Divider Register (CCM_POST_ROOT54_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9B30	Pre Divider Register (CCM_PRE54)	32	R/W	1000_0000h	5.1.7.22/ 492

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9B34	Pre Divider Register (CCM_PRE_ROOT54_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9B38	Pre Divider Register (CCM_PRE_ROOT54_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_9B3C	Pre Divider Register (CCM_PRE_ROOT54_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9B70	Access Control Register (CCM_ACCESS_CTRL54)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9B74	Access Control Register (CCM_ACCESS_CTRL_ROOT54_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9B78	Access Control Register (CCM_ACCESS_CTRL_ROOT54_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_9B7C	Access Control Register (CCM_ACCESS_CTRL_ROOT54_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9B80	Target Register (CCM_TARGET_ROOT55)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9B84	Target Register (CCM_TARGET_ROOT55_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9B88	Target Register (CCM_TARGET_ROOT55_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_9B8C	Target Register (CCM_TARGET_ROOT55_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9B90	Miscellaneous Register (CCM_MISC55)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9B94	Miscellaneous Register (CCM_MISC_ROOT55_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9B98	Miscellaneous Register (CCM_MISC_ROOT55_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_9B9C	Miscellaneous Register (CCM_MISC_ROOT55_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9BA0	Post Divider Register (CCM_POST55)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9BA4	Post Divider Register (CCM_POST_ROOT55_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9BA8	Post Divider Register (CCM_POST_ROOT55_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_9BAC	Post Divider Register (CCM_POST_ROOT55_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9BB0	Pre Divider Register (CCM_PRE55)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9BB4	Pre Divider Register (CCM_PRE_ROOT55_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9BB8	Pre Divider Register (CCM_PRE_ROOT55_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9BBC	Pre Divider Register (CCM_PRE_ROOT55_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9BF0	Access Control Register (CCM_ACCESS_CTRL55)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9BF4	Access Control Register (CCM_ACCESS_CTRL_ROOT55_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9BF8	Access Control Register (CCM_ACCESS_CTRL_ROOT55_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_9BFC	Access Control Register (CCM_ACCESS_CTRL_ROOT55_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9C00	Target Register (CCM_TARGET_ROOT56)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9C04	Target Register (CCM_TARGET_ROOT56_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9C08	Target Register (CCM_TARGET_ROOT56_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_9C0C	Target Register (CCM_TARGET_ROOT56_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9C10	Miscellaneous Register (CCM_MISC56)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9C14	Miscellaneous Register (CCM_MISC_ROOT56_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9C18	Miscellaneous Register (CCM_MISC_ROOT56_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_9C1C	Miscellaneous Register (CCM_MISC_ROOT56_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9C20	Post Divider Register (CCM_POST56)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9C24	Post Divider Register (CCM_POST_ROOT56_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9C28	Post Divider Register (CCM_POST_ROOT56_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_9C2C	Post Divider Register (CCM_POST_ROOT56_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9C30	Pre Divider Register (CCM_PRE56)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9C34	Pre Divider Register (CCM_PRE_ROOT56_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9C38	Pre Divider Register (CCM_PRE_ROOT56_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_9C3C	Pre Divider Register (CCM_PRE_ROOT56_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9C70	Access Control Register (CCM_ACCESS_CTRL56)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9C74	Access Control Register (CCM_ACCESS_CTRL_ROOT56_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9C78	Access Control Register (CCM_ACCESS_CTRL_ROOT56_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_9C7C	Access Control Register (CCM_ACCESS_CTRL_ROOT56_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9C80	Target Register (CCM_TARGET_ROOT57)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9C84	Target Register (CCM_TARGET_ROOT57_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9C88	Target Register (CCM_TARGET_ROOT57_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_9C8C	Target Register (CCM_TARGET_ROOT57_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9C90	Miscellaneous Register (CCM_MISC57)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9C94	Miscellaneous Register (CCM_MISC_ROOT57_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9C98	Miscellaneous Register (CCM_MISC_ROOT57_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_9C9C	Miscellaneous Register (CCM_MISC_ROOT57_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9CA0	Post Divider Register (CCM_POST57)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9CA4	Post Divider Register (CCM_POST_ROOT57_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9CA8	Post Divider Register (CCM_POST_ROOT57_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_9CAC	Post Divider Register (CCM_POST_ROOT57_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9CB0	Pre Divider Register (CCM_PRE57)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9CB4	Pre Divider Register (CCM_PRE_ROOT57_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9CB8	Pre Divider Register (CCM_PRE_ROOT57_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_9CBC	Pre Divider Register (CCM_PRE_ROOT57_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9CF0	Access Control Register (CCM_ACCESS_CTRL57)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9CF4	Access Control Register (CCM_ACCESS_CTRL_ROOT57_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9CF8	Access Control Register (CCM_ACCESS_CTRL_ROOT57_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9CFC	Access Control Register (CCM_ACCESS_CTRL_ROOT57_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9D00	Target Register (CCM_TARGET_ROOT58)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9D04	Target Register (CCM_TARGET_ROOT58_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9D08	Target Register (CCM_TARGET_ROOT58_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_9D0C	Target Register (CCM_TARGET_ROOT58_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9D10	Miscellaneous Register (CCM_MISC58)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9D14	Miscellaneous Register (CCM_MISC_ROOT58_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9D18	Miscellaneous Register (CCM_MISC_ROOT58_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_9D1C	Miscellaneous Register (CCM_MISC_ROOT58_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9D20	Post Divider Register (CCM_POST58)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9D24	Post Divider Register (CCM_POST_ROOT58_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9D28	Post Divider Register (CCM_POST_ROOT58_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_9D2C	Post Divider Register (CCM_POST_ROOT58_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9D30	Pre Divider Register (CCM_PRE58)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9D34	Pre Divider Register (CCM_PRE_ROOT58_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9D38	Pre Divider Register (CCM_PRE_ROOT58_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_9D3C	Pre Divider Register (CCM_PRE_ROOT58_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9D70	Access Control Register (CCM_ACCESS_CTRL58)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9D74	Access Control Register (CCM_ACCESS_CTRL_ROOT58_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9D78	Access Control Register (CCM_ACCESS_CTRL_ROOT58_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_9D7C	Access Control Register (CCM_ACCESS_CTRL_ROOT58_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9D80	Target Register (CCM_TARGET_ROOT59)	32	R/W	1000_0000h	5.1.7.10/ 468

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9D84	Target Register (CCM_TARGET_ROOT59_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9D88	Target Register (CCM_TARGET_ROOT59_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_9D8C	Target Register (CCM_TARGET_ROOT59_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9D90	Miscellaneous Register (CCM_MISC59)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9D94	Miscellaneous Register (CCM_MISC_ROOT59_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9D98	Miscellaneous Register (CCM_MISC_ROOT59_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_9D9C	Miscellaneous Register (CCM_MISC_ROOT59_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9DA0	Post Divider Register (CCM_POST59)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9DA4	Post Divider Register (CCM_POST_ROOT59_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9DA8	Post Divider Register (CCM_POST_ROOT59_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_9DAC	Post Divider Register (CCM_POST_ROOT59_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9DB0	Pre Divider Register (CCM_PRE59)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9DB4	Pre Divider Register (CCM_PRE_ROOT59_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9DB8	Pre Divider Register (CCM_PRE_ROOT59_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_9DBC	Pre Divider Register (CCM_PRE_ROOT59_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9DF0	Access Control Register (CCM_ACCESS_CTRL59)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9DF4	Access Control Register (CCM_ACCESS_CTRL_ROOT59_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9DF8	Access Control Register (CCM_ACCESS_CTRL_ROOT59_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_9DFC	Access Control Register (CCM_ACCESS_CTRL_ROOT59_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9E00	Target Register (CCM_TARGET_ROOT60)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9E04	Target Register (CCM_TARGET_ROOT60_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9E08	Target Register (CCM_TARGET_ROOT60_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9E0C	Target Register (CCM_TARGET_ROOT60_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9E10	Miscellaneous Register (CCM_MISC60)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9E14	Miscellaneous Register (CCM_MISC_ROOT60_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9E18	Miscellaneous Register (CCM_MISC_ROOT60_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_9E1C	Miscellaneous Register (CCM_MISC_ROOT60_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9E20	Post Divider Register (CCM_POST60)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9E24	Post Divider Register (CCM_POST_ROOT60_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9E28	Post Divider Register (CCM_POST_ROOT60_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_9E2C	Post Divider Register (CCM_POST_ROOT60_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9E30	Pre Divider Register (CCM_PRE60)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9E34	Pre Divider Register (CCM_PRE_ROOT60_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9E38	Pre Divider Register (CCM_PRE_ROOT60_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_9E3C	Pre Divider Register (CCM_PRE_ROOT60_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9E70	Access Control Register (CCM_ACCESS_CTRL60)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9E74	Access Control Register (CCM_ACCESS_CTRL_ROOT60_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9E78	Access Control Register (CCM_ACCESS_CTRL_ROOT60_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_9E7C	Access Control Register (CCM_ACCESS_CTRL_ROOT60_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9E80	Target Register (CCM_TARGET_ROOT61)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9E84	Target Register (CCM_TARGET_ROOT61_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9E88	Target Register (CCM_TARGET_ROOT61_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_9E8C	Target Register (CCM_TARGET_ROOT61_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9E90	Miscellaneous Register (CCM_MISC61)	32	R/W	0000_0000h	5.1.7.14/ 476

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9E94	Miscellaneous Register (CCM_MISC_ROOT61_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9E98	Miscellaneous Register (CCM_MISC_ROOT61_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_9E9C	Miscellaneous Register (CCM_MISC_ROOT61_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9EA0	Post Divider Register (CCM_POST61)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9EA4	Post Divider Register (CCM_POST_ROOT61_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9EA8	Post Divider Register (CCM_POST_ROOT61_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_9EAC	Post Divider Register (CCM_POST_ROOT61_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9EB0	Pre Divider Register (CCM_PRE61)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9EB4	Pre Divider Register (CCM_PRE_ROOT61_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9EB8	Pre Divider Register (CCM_PRE_ROOT61_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_9EBC	Pre Divider Register (CCM_PRE_ROOT61_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9EF0	Access Control Register (CCM_ACCESS_CTRL61)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9EF4	Access Control Register (CCM_ACCESS_CTRL_ROOT61_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9EF8	Access Control Register (CCM_ACCESS_CTRL_ROOT61_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_9EFC	Access Control Register (CCM_ACCESS_CTRL_ROOT61_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9F00	Target Register (CCM_TARGET_ROOT62)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9F04	Target Register (CCM_TARGET_ROOT62_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9F08	Target Register (CCM_TARGET_ROOT62_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_9F0C	Target Register (CCM_TARGET_ROOT62_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9F10	Miscellaneous Register (CCM_MISC62)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9F14	Miscellaneous Register (CCM_MISC_ROOT62_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9F18	Miscellaneous Register (CCM_MISC_ROOT62_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9F1C	Miscellaneous Register (CCM_MISC_ROOT62_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9F20	Post Divider Register (CCM_POST62)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_9F24	Post Divider Register (CCM_POST_ROOT62_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9F28	Post Divider Register (CCM_POST_ROOT62_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_9F2C	Post Divider Register (CCM_POST_ROOT62_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9F30	Pre Divider Register (CCM_PRE62)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9F34	Pre Divider Register (CCM_PRE_ROOT62_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9F38	Pre Divider Register (CCM_PRE_ROOT62_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_9F3C	Pre Divider Register (CCM_PRE_ROOT62_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9F70	Access Control Register (CCM_ACCESS_CTRL62)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9F74	Access Control Register (CCM_ACCESS_CTRL_ROOT62_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9F78	Access Control Register (CCM_ACCESS_CTRL_ROOT62_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_9F7C	Access Control Register (CCM_ACCESS_CTRL_ROOT62_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_9F80	Target Register (CCM_TARGET_ROOT63)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_9F84	Target Register (CCM_TARGET_ROOT63_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_9F88	Target Register (CCM_TARGET_ROOT63_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_9F8C	Target Register (CCM_TARGET_ROOT63_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_9F90	Miscellaneous Register (CCM_MISC63)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_9F94	Miscellaneous Register (CCM_MISC_ROOT63_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_9F98	Miscellaneous Register (CCM_MISC_ROOT63_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_9F9C	Miscellaneous Register (CCM_MISC_ROOT63_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_9FA0	Post Divider Register (CCM_POST63)	32	R/W	0000_0000h	5.1.7.18/ 480

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_9FA4	Post Divider Register (CCM_POST_ROOT63_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_9FA8	Post Divider Register (CCM_POST_ROOT63_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_9FAC	Post Divider Register (CCM_POST_ROOT63_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_9FB0	Pre Divider Register (CCM_PRE63)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_9FB4	Pre Divider Register (CCM_PRE_ROOT63_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_9FB8	Pre Divider Register (CCM_PRE_ROOT63_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_9FBC	Pre Divider Register (CCM_PRE_ROOT63_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_9FF0	Access Control Register (CCM_ACCESS_CTRL63)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_9FF4	Access Control Register (CCM_ACCESS_CTRL_ROOT63_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_9FF8	Access Control Register (CCM_ACCESS_CTRL_ROOT63_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_9FFC	Access Control Register (CCM_ACCESS_CTRL_ROOT63_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A000	Target Register (CCM_TARGET_ROOT64)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A004	Target Register (CCM_TARGET_ROOT64_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A008	Target Register (CCM_TARGET_ROOT64_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A00C	Target Register (CCM_TARGET_ROOT64_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A010	Miscellaneous Register (CCM_MISC64)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A014	Miscellaneous Register (CCM_MISC_ROOT64_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A018	Miscellaneous Register (CCM_MISC_ROOT64_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A01C	Miscellaneous Register (CCM_MISC_ROOT64_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A020	Post Divider Register (CCM_POST64)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A024	Post Divider Register (CCM_POST_ROOT64_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A028	Post Divider Register (CCM_POST_ROOT64_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A02C	Post Divider Register (CCM_POST_ROOT64_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A030	Pre Divider Register (CCM_PRE64)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A034	Pre Divider Register (CCM_PRE_ROOT64_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A038	Pre Divider Register (CCM_PRE_ROOT64_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A03C	Pre Divider Register (CCM_PRE_ROOT64_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A070	Access Control Register (CCM_ACCESS_CTRL64)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A074	Access Control Register (CCM_ACCESS_CTRL_ROOT64_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A078	Access Control Register (CCM_ACCESS_CTRL_ROOT64_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A07C	Access Control Register (CCM_ACCESS_CTRL_ROOT64_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A080	Target Register (CCM_TARGET_ROOT65)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A084	Target Register (CCM_TARGET_ROOT65_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A088	Target Register (CCM_TARGET_ROOT65_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A08C	Target Register (CCM_TARGET_ROOT65_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A090	Miscellaneous Register (CCM_MISC65)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A094	Miscellaneous Register (CCM_MISC_ROOT65_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A098	Miscellaneous Register (CCM_MISC_ROOT65_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A09C	Miscellaneous Register (CCM_MISC_ROOT65_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A0A0	Post Divider Register (CCM_POST65)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A0A4	Post Divider Register (CCM_POST_ROOT65_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A0A8	Post Divider Register (CCM_POST_ROOT65_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A0AC	Post Divider Register (CCM_POST_ROOT65_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A0B0	Pre Divider Register (CCM_PRE65)	32	R/W	1000_0000h	5.1.7.22/ 492

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A0B4	Pre Divider Register (CCM_PRE_ROOT65_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A0B8	Pre Divider Register (CCM_PRE_ROOT65_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A0BC	Pre Divider Register (CCM_PRE_ROOT65_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A0F0	Access Control Register (CCM_ACCESS_CTRL65)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A0F4	Access Control Register (CCM_ACCESS_CTRL_ROOT65_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A0F8	Access Control Register (CCM_ACCESS_CTRL_ROOT65_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A0FC	Access Control Register (CCM_ACCESS_CTRL_ROOT65_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A100	Target Register (CCM_TARGET_ROOT66)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A104	Target Register (CCM_TARGET_ROOT66_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A108	Target Register (CCM_TARGET_ROOT66_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A10C	Target Register (CCM_TARGET_ROOT66_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A110	Miscellaneous Register (CCM_MISC66)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A114	Miscellaneous Register (CCM_MISC_ROOT66_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A118	Miscellaneous Register (CCM_MISC_ROOT66_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A11C	Miscellaneous Register (CCM_MISC_ROOT66_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A120	Post Divider Register (CCM_POST66)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A124	Post Divider Register (CCM_POST_ROOT66_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A128	Post Divider Register (CCM_POST_ROOT66_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A12C	Post Divider Register (CCM_POST_ROOT66_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A130	Pre Divider Register (CCM_PRE66)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A134	Pre Divider Register (CCM_PRE_ROOT66_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A138	Pre Divider Register (CCM_PRE_ROOT66_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A13C	Pre Divider Register (CCM_PRE_ROOT66_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A170	Access Control Register (CCM_ACCESS_CTRL66)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A174	Access Control Register (CCM_ACCESS_CTRL_ROOT66_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A178	Access Control Register (CCM_ACCESS_CTRL_ROOT66_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A17C	Access Control Register (CCM_ACCESS_CTRL_ROOT66_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A180	Target Register (CCM_TARGET_ROOT67)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A184	Target Register (CCM_TARGET_ROOT67_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A188	Target Register (CCM_TARGET_ROOT67_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A18C	Target Register (CCM_TARGET_ROOT67_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A190	Miscellaneous Register (CCM_MISC67)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A194	Miscellaneous Register (CCM_MISC_ROOT67_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A198	Miscellaneous Register (CCM_MISC_ROOT67_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A19C	Miscellaneous Register (CCM_MISC_ROOT67_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A1A0	Post Divider Register (CCM_POST67)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A1A4	Post Divider Register (CCM_POST_ROOT67_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A1A8	Post Divider Register (CCM_POST_ROOT67_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A1AC	Post Divider Register (CCM_POST_ROOT67_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A1B0	Pre Divider Register (CCM_PRE67)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A1B4	Pre Divider Register (CCM_PRE_ROOT67_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A1B8	Pre Divider Register (CCM_PRE_ROOT67_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A1BC	Pre Divider Register (CCM_PRE_ROOT67_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A1F0	Access Control Register (CCM_ACCESS_CTRL67)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A1F4	Access Control Register (CCM_ACCESS_CTRL_ROOT67_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A1F8	Access Control Register (CCM_ACCESS_CTRL_ROOT67_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A1FC	Access Control Register (CCM_ACCESS_CTRL_ROOT67_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A200	Target Register (CCM_TARGET_ROOT68)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A204	Target Register (CCM_TARGET_ROOT68_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A208	Target Register (CCM_TARGET_ROOT68_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A20C	Target Register (CCM_TARGET_ROOT68_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A210	Miscellaneous Register (CCM_MISC68)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A214	Miscellaneous Register (CCM_MISC_ROOT68_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A218	Miscellaneous Register (CCM_MISC_ROOT68_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A21C	Miscellaneous Register (CCM_MISC_ROOT68_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A220	Post Divider Register (CCM_POST68)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A224	Post Divider Register (CCM_POST_ROOT68_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A228	Post Divider Register (CCM_POST_ROOT68_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A22C	Post Divider Register (CCM_POST_ROOT68_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A230	Pre Divider Register (CCM_PRE68)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A234	Pre Divider Register (CCM_PRE_ROOT68_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A238	Pre Divider Register (CCM_PRE_ROOT68_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A23C	Pre Divider Register (CCM_PRE_ROOT68_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A270	Access Control Register (CCM_ACCESS_CTRL68)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A274	Access Control Register (CCM_ACCESS_CTRL_ROOT68_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A278	Access Control Register (CCM_ACCESS_CTRL_ROOT68_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A27C	Access Control Register (CCM_ACCESS_CTRL_ROOT68_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A280	Target Register (CCM_TARGET_ROOT69)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A284	Target Register (CCM_TARGET_ROOT69_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A288	Target Register (CCM_TARGET_ROOT69_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A28C	Target Register (CCM_TARGET_ROOT69_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A290	Miscellaneous Register (CCM_MISC69)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A294	Miscellaneous Register (CCM_MISC_ROOT69_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A298	Miscellaneous Register (CCM_MISC_ROOT69_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A29C	Miscellaneous Register (CCM_MISC_ROOT69_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A2A0	Post Divider Register (CCM_POST69)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A2A4	Post Divider Register (CCM_POST_ROOT69_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A2A8	Post Divider Register (CCM_POST_ROOT69_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A2AC	Post Divider Register (CCM_POST_ROOT69_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A2B0	Pre Divider Register (CCM_PRE69)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A2B4	Pre Divider Register (CCM_PRE_ROOT69_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A2B8	Pre Divider Register (CCM_PRE_ROOT69_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A2BC	Pre Divider Register (CCM_PRE_ROOT69_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A2F0	Access Control Register (CCM_ACCESS_CTRL69)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A2F4	Access Control Register (CCM_ACCESS_CTRL_ROOT69_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A2F8	Access Control Register (CCM_ACCESS_CTRL_ROOT69_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A2FC	Access Control Register (CCM_ACCESS_CTRL_ROOT69_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A300	Target Register (CCM_TARGET_ROOT70)	32	R/W	1000_0000h	5.1.7.10/ 468

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A304	Target Register (CCM_TARGET_ROOT70_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A308	Target Register (CCM_TARGET_ROOT70_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A30C	Target Register (CCM_TARGET_ROOT70_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A310	Miscellaneous Register (CCM_MISC70)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A314	Miscellaneous Register (CCM_MISC_ROOT70_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A318	Miscellaneous Register (CCM_MISC_ROOT70_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A31C	Miscellaneous Register (CCM_MISC_ROOT70_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A320	Post Divider Register (CCM_POST70)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A324	Post Divider Register (CCM_POST_ROOT70_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A328	Post Divider Register (CCM_POST_ROOT70_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A32C	Post Divider Register (CCM_POST_ROOT70_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A330	Pre Divider Register (CCM_PRE70)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A334	Pre Divider Register (CCM_PRE_ROOT70_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A338	Pre Divider Register (CCM_PRE_ROOT70_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A33C	Pre Divider Register (CCM_PRE_ROOT70_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A370	Access Control Register (CCM_ACCESS_CTRL70)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A374	Access Control Register (CCM_ACCESS_CTRL_ROOT70_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A378	Access Control Register (CCM_ACCESS_CTRL_ROOT70_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A37C	Access Control Register (CCM_ACCESS_CTRL_ROOT70_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A380	Target Register (CCM_TARGET_ROOT71)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A384	Target Register (CCM_TARGET_ROOT71_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A388	Target Register (CCM_TARGET_ROOT71_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A38C	Target Register (CCM_TARGET_ROOT71_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A390	Miscellaneous Register (CCM_MISC71)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A394	Miscellaneous Register (CCM_MISC_ROOT71_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A398	Miscellaneous Register (CCM_MISC_ROOT71_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A39C	Miscellaneous Register (CCM_MISC_ROOT71_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A3A0	Post Divider Register (CCM_POST71)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A3A4	Post Divider Register (CCM_POST_ROOT71_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A3A8	Post Divider Register (CCM_POST_ROOT71_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A3AC	Post Divider Register (CCM_POST_ROOT71_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A3B0	Pre Divider Register (CCM_PRE71)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A3B4	Pre Divider Register (CCM_PRE_ROOT71_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A3B8	Pre Divider Register (CCM_PRE_ROOT71_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A3BC	Pre Divider Register (CCM_PRE_ROOT71_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A3F0	Access Control Register (CCM_ACCESS_CTRL71)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A3F4	Access Control Register (CCM_ACCESS_CTRL_ROOT71_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A3F8	Access Control Register (CCM_ACCESS_CTRL_ROOT71_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A3FC	Access Control Register (CCM_ACCESS_CTRL_ROOT71_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A400	Target Register (CCM_TARGET_ROOT72)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A404	Target Register (CCM_TARGET_ROOT72_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A408	Target Register (CCM_TARGET_ROOT72_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A40C	Target Register (CCM_TARGET_ROOT72_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A410	Miscellaneous Register (CCM_MISC72)	32	R/W	0000_0000h	5.1.7.14/ 476

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A414	Miscellaneous Register (CCM_MISC_ROOT72_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A418	Miscellaneous Register (CCM_MISC_ROOT72_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A41C	Miscellaneous Register (CCM_MISC_ROOT72_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A420	Post Divider Register (CCM_POST72)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A424	Post Divider Register (CCM_POST_ROOT72_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A428	Post Divider Register (CCM_POST_ROOT72_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A42C	Post Divider Register (CCM_POST_ROOT72_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A430	Pre Divider Register (CCM_PRE72)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A434	Pre Divider Register (CCM_PRE_ROOT72_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A438	Pre Divider Register (CCM_PRE_ROOT72_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A43C	Pre Divider Register (CCM_PRE_ROOT72_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A470	Access Control Register (CCM_ACCESS_CTRL72)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A474	Access Control Register (CCM_ACCESS_CTRL_ROOT72_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A478	Access Control Register (CCM_ACCESS_CTRL_ROOT72_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A47C	Access Control Register (CCM_ACCESS_CTRL_ROOT72_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A480	Target Register (CCM_TARGET_ROOT73)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A484	Target Register (CCM_TARGET_ROOT73_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A488	Target Register (CCM_TARGET_ROOT73_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A48C	Target Register (CCM_TARGET_ROOT73_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A490	Miscellaneous Register (CCM_MISC73)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A494	Miscellaneous Register (CCM_MISC_ROOT73_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A498	Miscellaneous Register (CCM_MISC_ROOT73_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A49C	Miscellaneous Register (CCM_MISC_ROOT73_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A4A0	Post Divider Register (CCM_POST73)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A4A4	Post Divider Register (CCM_POST_ROOT73_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A4A8	Post Divider Register (CCM_POST_ROOT73_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A4AC	Post Divider Register (CCM_POST_ROOT73_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A4B0	Pre Divider Register (CCM_PRE73)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A4B4	Pre Divider Register (CCM_PRE_ROOT73_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A4B8	Pre Divider Register (CCM_PRE_ROOT73_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A4BC	Pre Divider Register (CCM_PRE_ROOT73_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A4F0	Access Control Register (CCM_ACCESS_CTRL73)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A4F4	Access Control Register (CCM_ACCESS_CTRL_ROOT73_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A4F8	Access Control Register (CCM_ACCESS_CTRL_ROOT73_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A4FC	Access Control Register (CCM_ACCESS_CTRL_ROOT73_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A500	Target Register (CCM_TARGET_ROOT74)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A504	Target Register (CCM_TARGET_ROOT74_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A508	Target Register (CCM_TARGET_ROOT74_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A50C	Target Register (CCM_TARGET_ROOT74_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A510	Miscellaneous Register (CCM_MISC74)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A514	Miscellaneous Register (CCM_MISC_ROOT74_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A518	Miscellaneous Register (CCM_MISC_ROOT74_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A51C	Miscellaneous Register (CCM_MISC_ROOT74_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A520	Post Divider Register (CCM_POST74)	32	R/W	0000_0000h	5.1.7.18/ 480

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A524	Post Divider Register (CCM_POST_ROOT74_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A528	Post Divider Register (CCM_POST_ROOT74_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A52C	Post Divider Register (CCM_POST_ROOT74_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A530	Pre Divider Register (CCM_PRE74)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A534	Pre Divider Register (CCM_PRE_ROOT74_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A538	Pre Divider Register (CCM_PRE_ROOT74_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A53C	Pre Divider Register (CCM_PRE_ROOT74_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A570	Access Control Register (CCM_ACCESS_CTRL74)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A574	Access Control Register (CCM_ACCESS_CTRL_ROOT74_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A578	Access Control Register (CCM_ACCESS_CTRL_ROOT74_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A57C	Access Control Register (CCM_ACCESS_CTRL_ROOT74_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A580	Target Register (CCM_TARGET_ROOT75)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A584	Target Register (CCM_TARGET_ROOT75_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A588	Target Register (CCM_TARGET_ROOT75_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A58C	Target Register (CCM_TARGET_ROOT75_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A590	Miscellaneous Register (CCM_MISC75)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A594	Miscellaneous Register (CCM_MISC_ROOT75_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A598	Miscellaneous Register (CCM_MISC_ROOT75_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A59C	Miscellaneous Register (CCM_MISC_ROOT75_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A5A0	Post Divider Register (CCM_POST75)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A5A4	Post Divider Register (CCM_POST_ROOT75_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A5A8	Post Divider Register (CCM_POST_ROOT75_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A5AC	Post Divider Register (CCM_POST_ROOT75_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A5B0	Pre Divider Register (CCM_PRE75)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A5B4	Pre Divider Register (CCM_PRE_ROOT75_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A5B8	Pre Divider Register (CCM_PRE_ROOT75_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A5BC	Pre Divider Register (CCM_PRE_ROOT75_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A5F0	Access Control Register (CCM_ACCESS_CTRL75)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A5F4	Access Control Register (CCM_ACCESS_CTRL_ROOT75_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A5F8	Access Control Register (CCM_ACCESS_CTRL_ROOT75_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A5FC	Access Control Register (CCM_ACCESS_CTRL_ROOT75_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A600	Target Register (CCM_TARGET_ROOT76)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A604	Target Register (CCM_TARGET_ROOT76_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A608	Target Register (CCM_TARGET_ROOT76_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A60C	Target Register (CCM_TARGET_ROOT76_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A610	Miscellaneous Register (CCM_MISC76)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A614	Miscellaneous Register (CCM_MISC_ROOT76_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A618	Miscellaneous Register (CCM_MISC_ROOT76_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A61C	Miscellaneous Register (CCM_MISC_ROOT76_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A620	Post Divider Register (CCM_POST76)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A624	Post Divider Register (CCM_POST_ROOT76_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A628	Post Divider Register (CCM_POST_ROOT76_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A62C	Post Divider Register (CCM_POST_ROOT76_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A630	Pre Divider Register (CCM_PRE76)	32	R/W	1000_0000h	5.1.7.22/ 492

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A634	Pre Divider Register (CCM_PRE_ROOT76_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A638	Pre Divider Register (CCM_PRE_ROOT76_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A63C	Pre Divider Register (CCM_PRE_ROOT76_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A670	Access Control Register (CCM_ACCESS_CTRL76)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A674	Access Control Register (CCM_ACCESS_CTRL_ROOT76_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A678	Access Control Register (CCM_ACCESS_CTRL_ROOT76_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A67C	Access Control Register (CCM_ACCESS_CTRL_ROOT76_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A680	Target Register (CCM_TARGET_ROOT77)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A684	Target Register (CCM_TARGET_ROOT77_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A688	Target Register (CCM_TARGET_ROOT77_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A68C	Target Register (CCM_TARGET_ROOT77_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A690	Miscellaneous Register (CCM_MISC77)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A694	Miscellaneous Register (CCM_MISC_ROOT77_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A698	Miscellaneous Register (CCM_MISC_ROOT77_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A69C	Miscellaneous Register (CCM_MISC_ROOT77_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A6A0	Post Divider Register (CCM_POST77)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A6A4	Post Divider Register (CCM_POST_ROOT77_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A6A8	Post Divider Register (CCM_POST_ROOT77_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A6AC	Post Divider Register (CCM_POST_ROOT77_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A6B0	Pre Divider Register (CCM_PRE77)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A6B4	Pre Divider Register (CCM_PRE_ROOT77_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A6B8	Pre Divider Register (CCM_PRE_ROOT77_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A6BC	Pre Divider Register (CCM_PRE_ROOT77_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A6F0	Access Control Register (CCM_ACCESS_CTRL77)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A6F4	Access Control Register (CCM_ACCESS_CTRL_ROOT77_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A6F8	Access Control Register (CCM_ACCESS_CTRL_ROOT77_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A6FC	Access Control Register (CCM_ACCESS_CTRL_ROOT77_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A700	Target Register (CCM_TARGET_ROOT78)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A704	Target Register (CCM_TARGET_ROOT78_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A708	Target Register (CCM_TARGET_ROOT78_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A70C	Target Register (CCM_TARGET_ROOT78_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A710	Miscellaneous Register (CCM_MISC78)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A714	Miscellaneous Register (CCM_MISC_ROOT78_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A718	Miscellaneous Register (CCM_MISC_ROOT78_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A71C	Miscellaneous Register (CCM_MISC_ROOT78_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A720	Post Divider Register (CCM_POST78)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A724	Post Divider Register (CCM_POST_ROOT78_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A728	Post Divider Register (CCM_POST_ROOT78_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A72C	Post Divider Register (CCM_POST_ROOT78_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A730	Pre Divider Register (CCM_PRE78)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A734	Pre Divider Register (CCM_PRE_ROOT78_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A738	Pre Divider Register (CCM_PRE_ROOT78_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A73C	Pre Divider Register (CCM_PRE_ROOT78_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A770	Access Control Register (CCM_ACCESS_CTRL78)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A774	Access Control Register (CCM_ACCESS_CTRL_ROOT78_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A778	Access Control Register (CCM_ACCESS_CTRL_ROOT78_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A77C	Access Control Register (CCM_ACCESS_CTRL_ROOT78_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A780	Target Register (CCM_TARGET_ROOT79)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A784	Target Register (CCM_TARGET_ROOT79_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A788	Target Register (CCM_TARGET_ROOT79_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A78C	Target Register (CCM_TARGET_ROOT79_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A790	Miscellaneous Register (CCM_MISC79)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A794	Miscellaneous Register (CCM_MISC_ROOT79_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A798	Miscellaneous Register (CCM_MISC_ROOT79_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A79C	Miscellaneous Register (CCM_MISC_ROOT79_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A7A0	Post Divider Register (CCM_POST79)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A7A4	Post Divider Register (CCM_POST_ROOT79_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A7A8	Post Divider Register (CCM_POST_ROOT79_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A7AC	Post Divider Register (CCM_POST_ROOT79_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A7B0	Pre Divider Register (CCM_PRE79)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A7B4	Pre Divider Register (CCM_PRE_ROOT79_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A7B8	Pre Divider Register (CCM_PRE_ROOT79_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A7BC	Pre Divider Register (CCM_PRE_ROOT79_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A7F0	Access Control Register (CCM_ACCESS_CTRL79)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A7F4	Access Control Register (CCM_ACCESS_CTRL_ROOT79_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A7F8	Access Control Register (CCM_ACCESS_CTRL_ROOT79_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A7FC	Access Control Register (CCM_ACCESS_CTRL_ROOT79_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A800	Target Register (CCM_TARGET_ROOT80)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A804	Target Register (CCM_TARGET_ROOT80_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A808	Target Register (CCM_TARGET_ROOT80_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A80C	Target Register (CCM_TARGET_ROOT80_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A810	Miscellaneous Register (CCM_MISC80)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A814	Miscellaneous Register (CCM_MISC_ROOT80_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A818	Miscellaneous Register (CCM_MISC_ROOT80_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A81C	Miscellaneous Register (CCM_MISC_ROOT80_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A820	Post Divider Register (CCM_POST80)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A824	Post Divider Register (CCM_POST_ROOT80_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A828	Post Divider Register (CCM_POST_ROOT80_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A82C	Post Divider Register (CCM_POST_ROOT80_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A830	Pre Divider Register (CCM_PRE80)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A834	Pre Divider Register (CCM_PRE_ROOT80_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A838	Pre Divider Register (CCM_PRE_ROOT80_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A83C	Pre Divider Register (CCM_PRE_ROOT80_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A870	Access Control Register (CCM_ACCESS_CTRL80)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A874	Access Control Register (CCM_ACCESS_CTRL_ROOT80_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A878	Access Control Register (CCM_ACCESS_CTRL_ROOT80_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A87C	Access Control Register (CCM_ACCESS_CTRL_ROOT80_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A880	Target Register (CCM_TARGET_ROOT81)	32	R/W	1000_0000h	5.1.7.10/ 468

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A884	Target Register (CCM_TARGET_ROOT81_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A888	Target Register (CCM_TARGET_ROOT81_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A88C	Target Register (CCM_TARGET_ROOT81_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A890	Miscellaneous Register (CCM_MISC81)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A894	Miscellaneous Register (CCM_MISC_ROOT81_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A898	Miscellaneous Register (CCM_MISC_ROOT81_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A89C	Miscellaneous Register (CCM_MISC_ROOT81_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A8A0	Post Divider Register (CCM_POST81)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A8A4	Post Divider Register (CCM_POST_ROOT81_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A8A8	Post Divider Register (CCM_POST_ROOT81_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A8AC	Post Divider Register (CCM_POST_ROOT81_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A8B0	Pre Divider Register (CCM_PRE81)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A8B4	Pre Divider Register (CCM_PRE_ROOT81_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A8B8	Pre Divider Register (CCM_PRE_ROOT81_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A8BC	Pre Divider Register (CCM_PRE_ROOT81_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A8F0	Access Control Register (CCM_ACCESS_CTRL81)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A8F4	Access Control Register (CCM_ACCESS_CTRL_ROOT81_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A8F8	Access Control Register (CCM_ACCESS_CTRL_ROOT81_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A8FC	Access Control Register (CCM_ACCESS_CTRL_ROOT81_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A900	Target Register (CCM_TARGET_ROOT82)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A904	Target Register (CCM_TARGET_ROOT82_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A908	Target Register (CCM_TARGET_ROOT82_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A90C	Target Register (CCM_TARGET_ROOT82_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A910	Miscellaneous Register (CCM_MISC82)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_A914	Miscellaneous Register (CCM_MISC_ROOT82_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A918	Miscellaneous Register (CCM_MISC_ROOT82_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A91C	Miscellaneous Register (CCM_MISC_ROOT82_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A920	Post Divider Register (CCM_POST82)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A924	Post Divider Register (CCM_POST_ROOT82_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A928	Post Divider Register (CCM_POST_ROOT82_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A92C	Post Divider Register (CCM_POST_ROOT82_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A930	Pre Divider Register (CCM_PRE82)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A934	Pre Divider Register (CCM_PRE_ROOT82_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A938	Pre Divider Register (CCM_PRE_ROOT82_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A93C	Pre Divider Register (CCM_PRE_ROOT82_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A970	Access Control Register (CCM_ACCESS_CTRL82)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A974	Access Control Register (CCM_ACCESS_CTRL_ROOT82_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A978	Access Control Register (CCM_ACCESS_CTRL_ROOT82_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A97C	Access Control Register (CCM_ACCESS_CTRL_ROOT82_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_A980	Target Register (CCM_TARGET_ROOT83)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_A984	Target Register (CCM_TARGET_ROOT83_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_A988	Target Register (CCM_TARGET_ROOT83_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_A98C	Target Register (CCM_TARGET_ROOT83_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_A990	Miscellaneous Register (CCM_MISC83)	32	R/W	0000_0000h	5.1.7.14/ 476

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_A994	Miscellaneous Register (CCM_MISC_ROOT83_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_A998	Miscellaneous Register (CCM_MISC_ROOT83_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_A99C	Miscellaneous Register (CCM_MISC_ROOT83_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_A9A0	Post Divider Register (CCM_POST83)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_A9A4	Post Divider Register (CCM_POST_ROOT83_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_A9A8	Post Divider Register (CCM_POST_ROOT83_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_A9AC	Post Divider Register (CCM_POST_ROOT83_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_A9B0	Pre Divider Register (CCM_PRE83)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_A9B4	Pre Divider Register (CCM_PRE_ROOT83_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_A9B8	Pre Divider Register (CCM_PRE_ROOT83_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_A9BC	Pre Divider Register (CCM_PRE_ROOT83_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_A9F0	Access Control Register (CCM_ACCESS_CTRL83)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_A9F4	Access Control Register (CCM_ACCESS_CTRL_ROOT83_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_A9F8	Access Control Register (CCM_ACCESS_CTRL_ROOT83_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_A9FC	Access Control Register (CCM_ACCESS_CTRL_ROOT83_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_AA00	Target Register (CCM_TARGET_ROOT84)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_AA04	Target Register (CCM_TARGET_ROOT84_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_AA08	Target Register (CCM_TARGET_ROOT84_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_AA0C	Target Register (CCM_TARGET_ROOT84_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_AA10	Miscellaneous Register (CCM_MISC84)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_AA14	Miscellaneous Register (CCM_MISC_ROOT84_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_AA18	Miscellaneous Register (CCM_MISC_ROOT84_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_AA1C	Miscellaneous Register (CCM_MISC_ROOT84_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_AA20	Post Divider Register (CCM_POST84)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_AA24	Post Divider Register (CCM_POST_ROOT84_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_AA28	Post Divider Register (CCM_POST_ROOT84_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_AA2C	Post Divider Register (CCM_POST_ROOT84_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_AA30	Pre Divider Register (CCM_PRE84)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_AA34	Pre Divider Register (CCM_PRE_ROOT84_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_AA38	Pre Divider Register (CCM_PRE_ROOT84_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_AA3C	Pre Divider Register (CCM_PRE_ROOT84_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_AA70	Access Control Register (CCM_ACCESS_CTRL84)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_AA74	Access Control Register (CCM_ACCESS_CTRL_ROOT84_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_AA78	Access Control Register (CCM_ACCESS_CTRL_ROOT84_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_AA7C	Access Control Register (CCM_ACCESS_CTRL_ROOT84_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_AA80	Target Register (CCM_TARGET_ROOT85)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_AA84	Target Register (CCM_TARGET_ROOT85_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_AA88	Target Register (CCM_TARGET_ROOT85_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_AA8C	Target Register (CCM_TARGET_ROOT85_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_AA90	Miscellaneous Register (CCM_MISC85)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_AA94	Miscellaneous Register (CCM_MISC_ROOT85_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_AA98	Miscellaneous Register (CCM_MISC_ROOT85_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_AA9C	Miscellaneous Register (CCM_MISC_ROOT85_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_AAA0	Post Divider Register (CCM_POST85)	32	R/W	0000_0000h	5.1.7.18/ 480

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_AAA4	Post Divider Register (CCM_POST_ROOT85_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_AAA8	Post Divider Register (CCM_POST_ROOT85_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_AAAC	Post Divider Register (CCM_POST_ROOT85_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_AAB0	Pre Divider Register (CCM_PRE85)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_AAB4	Pre Divider Register (CCM_PRE_ROOT85_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_AAB8	Pre Divider Register (CCM_PRE_ROOT85_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_AABC	Pre Divider Register (CCM_PRE_ROOT85_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_AAF0	Access Control Register (CCM_ACCESS_CTRL85)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_AAF4	Access Control Register (CCM_ACCESS_CTRL_ROOT85_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_AAF8	Access Control Register (CCM_ACCESS_CTRL_ROOT85_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_AAFC	Access Control Register (CCM_ACCESS_CTRL_ROOT85_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_AB00	Target Register (CCM_TARGET_ROOT86)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_AB04	Target Register (CCM_TARGET_ROOT86_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_AB08	Target Register (CCM_TARGET_ROOT86_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_AB0C	Target Register (CCM_TARGET_ROOT86_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_AB10	Miscellaneous Register (CCM_MISC86)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_AB14	Miscellaneous Register (CCM_MISC_ROOT86_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_AB18	Miscellaneous Register (CCM_MISC_ROOT86_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_AB1C	Miscellaneous Register (CCM_MISC_ROOT86_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_AB20	Post Divider Register (CCM_POST86)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_AB24	Post Divider Register (CCM_POST_ROOT86_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_AB28	Post Divider Register (CCM_POST_ROOT86_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_AB2C	Post Divider Register (CCM_POST_ROOT86_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_AB30	Pre Divider Register (CCM_PRE86)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_AB34	Pre Divider Register (CCM_PRE_ROOT86_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_AB38	Pre Divider Register (CCM_PRE_ROOT86_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_AB3C	Pre Divider Register (CCM_PRE_ROOT86_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_AB70	Access Control Register (CCM_ACCESS_CTRL86)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_AB74	Access Control Register (CCM_ACCESS_CTRL_ROOT86_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_AB78	Access Control Register (CCM_ACCESS_CTRL_ROOT86_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_AB7C	Access Control Register (CCM_ACCESS_CTRL_ROOT86_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_AB80	Target Register (CCM_TARGET_ROOT87)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_AB84	Target Register (CCM_TARGET_ROOT87_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_AB88	Target Register (CCM_TARGET_ROOT87_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_AB8C	Target Register (CCM_TARGET_ROOT87_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_AB90	Miscellaneous Register (CCM_MISC87)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_AB94	Miscellaneous Register (CCM_MISC_ROOT87_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_AB98	Miscellaneous Register (CCM_MISC_ROOT87_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_AB9C	Miscellaneous Register (CCM_MISC_ROOT87_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_ABA0	Post Divider Register (CCM_POST87)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_ABA4	Post Divider Register (CCM_POST_ROOT87_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_ABA8	Post Divider Register (CCM_POST_ROOT87_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_ABAC	Post Divider Register (CCM_POST_ROOT87_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_ABB0	Pre Divider Register (CCM_PRE87)	32	R/W	1000_0000h	5.1.7.22/ 492

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_ABB4	Pre Divider Register (CCM_PRE_ROOT87_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_ABB8	Pre Divider Register (CCM_PRE_ROOT87_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_ABBC	Pre Divider Register (CCM_PRE_ROOT87_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_ABF0	Access Control Register (CCM_ACCESS_CTRL87)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_ABF4	Access Control Register (CCM_ACCESS_CTRL_ROOT87_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_ABF8	Access Control Register (CCM_ACCESS_CTRL_ROOT87_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_ABFC	Access Control Register (CCM_ACCESS_CTRL_ROOT87_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_AC00	Target Register (CCM_TARGET_ROOT88)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_AC04	Target Register (CCM_TARGET_ROOT88_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_AC08	Target Register (CCM_TARGET_ROOT88_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_AC0C	Target Register (CCM_TARGET_ROOT88_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_AC10	Miscellaneous Register (CCM_MISC88)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_AC14	Miscellaneous Register (CCM_MISC_ROOT88_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_AC18	Miscellaneous Register (CCM_MISC_ROOT88_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_AC1C	Miscellaneous Register (CCM_MISC_ROOT88_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_AC20	Post Divider Register (CCM_POST88)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_AC24	Post Divider Register (CCM_POST_ROOT88_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_AC28	Post Divider Register (CCM_POST_ROOT88_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_AC2C	Post Divider Register (CCM_POST_ROOT88_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_AC30	Pre Divider Register (CCM_PRE88)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_AC34	Pre Divider Register (CCM_PRE_ROOT88_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_AC38	Pre Divider Register (CCM_PRE_ROOT88_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_AC3C	Pre Divider Register (CCM_PRE_ROOT88_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_AC70	Access Control Register (CCM_ACCESS_CTRL88)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_AC74	Access Control Register (CCM_ACCESS_CTRL_ROOT88_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_AC78	Access Control Register (CCM_ACCESS_CTRL_ROOT88_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_AC7C	Access Control Register (CCM_ACCESS_CTRL_ROOT88_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_AC80	Target Register (CCM_TARGET_ROOT89)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_AC84	Target Register (CCM_TARGET_ROOT89_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_AC88	Target Register (CCM_TARGET_ROOT89_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_AC8C	Target Register (CCM_TARGET_ROOT89_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_AC90	Miscellaneous Register (CCM_MISC89)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_AC94	Miscellaneous Register (CCM_MISC_ROOT89_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_AC98	Miscellaneous Register (CCM_MISC_ROOT89_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_AC9C	Miscellaneous Register (CCM_MISC_ROOT89_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_ACA0	Post Divider Register (CCM_POST89)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_ACA4	Post Divider Register (CCM_POST_ROOT89_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_ACA8	Post Divider Register (CCM_POST_ROOT89_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_ACAC	Post Divider Register (CCM_POST_ROOT89_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_ACB0	Pre Divider Register (CCM_PRE89)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_ACB4	Pre Divider Register (CCM_PRE_ROOT89_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_ACB8	Pre Divider Register (CCM_PRE_ROOT89_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_ACBC	Pre Divider Register (CCM_PRE_ROOT89_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_ACF0	Access Control Register (CCM_ACCESS_CTRL89)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_ACF4	Access Control Register (CCM_ACCESS_CTRL_ROOT89_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_ACF8	Access Control Register (CCM_ACCESS_CTRL_ROOT89_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_ACFC	Access Control Register (CCM_ACCESS_CTRL_ROOT89_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_AD00	Target Register (CCM_TARGET_ROOT90)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_AD04	Target Register (CCM_TARGET_ROOT90_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_AD08	Target Register (CCM_TARGET_ROOT90_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_AD0C	Target Register (CCM_TARGET_ROOT90_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_AD10	Miscellaneous Register (CCM_MISC90)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_AD14	Miscellaneous Register (CCM_MISC_ROOT90_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_AD18	Miscellaneous Register (CCM_MISC_ROOT90_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_AD1C	Miscellaneous Register (CCM_MISC_ROOT90_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_AD20	Post Divider Register (CCM_POST90)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_AD24	Post Divider Register (CCM_POST_ROOT90_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_AD28	Post Divider Register (CCM_POST_ROOT90_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_AD2C	Post Divider Register (CCM_POST_ROOT90_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_AD30	Pre Divider Register (CCM_PRE90)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_AD34	Pre Divider Register (CCM_PRE_ROOT90_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_AD38	Pre Divider Register (CCM_PRE_ROOT90_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_AD3C	Pre Divider Register (CCM_PRE_ROOT90_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_AD70	Access Control Register (CCM_ACCESS_CTRL90)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_AD74	Access Control Register (CCM_ACCESS_CTRL_ROOT90_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_AD78	Access Control Register (CCM_ACCESS_CTRL_ROOT90_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_AD7C	Access Control Register (CCM_ACCESS_CTRL_ROOT90_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_AD80	Target Register (CCM_TARGET_ROOT91)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_AD84	Target Register (CCM_TARGET_ROOT91_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_AD88	Target Register (CCM_TARGET_ROOT91_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_AD8C	Target Register (CCM_TARGET_ROOT91_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_AD90	Miscellaneous Register (CCM_MISC91)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_AD94	Miscellaneous Register (CCM_MISC_ROOT91_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_AD98	Miscellaneous Register (CCM_MISC_ROOT91_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_AD9C	Miscellaneous Register (CCM_MISC_ROOT91_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_ADA0	Post Divider Register (CCM_POST91)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_ADA4	Post Divider Register (CCM_POST_ROOT91_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_ADA8	Post Divider Register (CCM_POST_ROOT91_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_ADAC	Post Divider Register (CCM_POST_ROOT91_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_ADB0	Pre Divider Register (CCM_PRE91)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_ADB4	Pre Divider Register (CCM_PRE_ROOT91_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_ADB8	Pre Divider Register (CCM_PRE_ROOT91_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_ADBC	Pre Divider Register (CCM_PRE_ROOT91_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_ADF0	Access Control Register (CCM_ACCESS_CTRL91)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_ADF4	Access Control Register (CCM_ACCESS_CTRL_ROOT91_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_ADF8	Access Control Register (CCM_ACCESS_CTRL_ROOT91_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_ADFC	Access Control Register (CCM_ACCESS_CTRL_ROOT91_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_AE00	Target Register (CCM_TARGET_ROOT92)	32	R/W	1000_0000h	5.1.7.10/ 468

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_AE04	Target Register (CCM_TARGET_ROOT92_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_AE08	Target Register (CCM_TARGET_ROOT92_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_AE0C	Target Register (CCM_TARGET_ROOT92_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_AE10	Miscellaneous Register (CCM_MISC92)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_AE14	Miscellaneous Register (CCM_MISC_ROOT92_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_AE18	Miscellaneous Register (CCM_MISC_ROOT92_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_AE1C	Miscellaneous Register (CCM_MISC_ROOT92_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_AE20	Post Divider Register (CCM_POST92)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_AE24	Post Divider Register (CCM_POST_ROOT92_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_AE28	Post Divider Register (CCM_POST_ROOT92_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_AE2C	Post Divider Register (CCM_POST_ROOT92_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_AE30	Pre Divider Register (CCM_PRE92)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_AE34	Pre Divider Register (CCM_PRE_ROOT92_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_AE38	Pre Divider Register (CCM_PRE_ROOT92_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_AE3C	Pre Divider Register (CCM_PRE_ROOT92_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_AE70	Access Control Register (CCM_ACCESS_CTRL92)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_AE74	Access Control Register (CCM_ACCESS_CTRL_ROOT92_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_AE78	Access Control Register (CCM_ACCESS_CTRL_ROOT92_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_AE7C	Access Control Register (CCM_ACCESS_CTRL_ROOT92_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_AE80	Target Register (CCM_TARGET_ROOT93)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_AE84	Target Register (CCM_TARGET_ROOT93_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_AE88	Target Register (CCM_TARGET_ROOT93_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_AE8C	Target Register (CCM_TARGET_ROOT93_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_AE90	Miscellaneous Register (CCM_MISC93)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_AE94	Miscellaneous Register (CCM_MISC_ROOT93_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_AE98	Miscellaneous Register (CCM_MISC_ROOT93_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_AE9C	Miscellaneous Register (CCM_MISC_ROOT93_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_AEA0	Post Divider Register (CCM_POST93)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_AEA4	Post Divider Register (CCM_POST_ROOT93_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_AEA8	Post Divider Register (CCM_POST_ROOT93_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_AEAC	Post Divider Register (CCM_POST_ROOT93_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_AEB0	Pre Divider Register (CCM_PRE93)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_AEB4	Pre Divider Register (CCM_PRE_ROOT93_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_AEB8	Pre Divider Register (CCM_PRE_ROOT93_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_AEBC	Pre Divider Register (CCM_PRE_ROOT93_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_AEF0	Access Control Register (CCM_ACCESS_CTRL93)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_AEF4	Access Control Register (CCM_ACCESS_CTRL_ROOT93_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_AEF8	Access Control Register (CCM_ACCESS_CTRL_ROOT93_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_AEFC	Access Control Register (CCM_ACCESS_CTRL_ROOT93_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_AF00	Target Register (CCM_TARGET_ROOT94)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_AF04	Target Register (CCM_TARGET_ROOT94_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_AF08	Target Register (CCM_TARGET_ROOT94_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_AF0C	Target Register (CCM_TARGET_ROOT94_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_AF10	Miscellaneous Register (CCM_MISC94)	32	R/W	0000_0000h	5.1.7.14/ 476

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_AF14	Miscellaneous Register (CCM_MISC_ROOT94_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_AF18	Miscellaneous Register (CCM_MISC_ROOT94_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_AF1C	Miscellaneous Register (CCM_MISC_ROOT94_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_AF20	Post Divider Register (CCM_POST94)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_AF24	Post Divider Register (CCM_POST_ROOT94_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_AF28	Post Divider Register (CCM_POST_ROOT94_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_AF2C	Post Divider Register (CCM_POST_ROOT94_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_AF30	Pre Divider Register (CCM_PRE94)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_AF34	Pre Divider Register (CCM_PRE_ROOT94_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_AF38	Pre Divider Register (CCM_PRE_ROOT94_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_AF3C	Pre Divider Register (CCM_PRE_ROOT94_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_AF70	Access Control Register (CCM_ACCESS_CTRL94)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_AF74	Access Control Register (CCM_ACCESS_CTRL_ROOT94_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_AF78	Access Control Register (CCM_ACCESS_CTRL_ROOT94_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_AF7C	Access Control Register (CCM_ACCESS_CTRL_ROOT94_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_AF80	Target Register (CCM_TARGET_ROOT95)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_AF84	Target Register (CCM_TARGET_ROOT95_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_AF88	Target Register (CCM_TARGET_ROOT95_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_AF8C	Target Register (CCM_TARGET_ROOT95_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_AF90	Miscellaneous Register (CCM_MISC95)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_AF94	Miscellaneous Register (CCM_MISC_ROOT95_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_AF98	Miscellaneous Register (CCM_MISC_ROOT95_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_AF9C	Miscellaneous Register (CCM_MISC_ROOT95_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_AFA0	Post Divider Register (CCM_POST95)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_AFA4	Post Divider Register (CCM_POST_ROOT95_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_AFA8	Post Divider Register (CCM_POST_ROOT95_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_AFAC	Post Divider Register (CCM_POST_ROOT95_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_AFB0	Pre Divider Register (CCM_PRE95)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_AFB4	Pre Divider Register (CCM_PRE_ROOT95_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_AFB8	Pre Divider Register (CCM_PRE_ROOT95_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_AFBC	Pre Divider Register (CCM_PRE_ROOT95_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_AFF0	Access Control Register (CCM_ACCESS_CTRL95)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_AFF4	Access Control Register (CCM_ACCESS_CTRL_ROOT95_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_AFF8	Access Control Register (CCM_ACCESS_CTRL_ROOT95_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_AFFC	Access Control Register (CCM_ACCESS_CTRL_ROOT95_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B000	Target Register (CCM_TARGET_ROOT96)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B004	Target Register (CCM_TARGET_ROOT96_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B008	Target Register (CCM_TARGET_ROOT96_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B00C	Target Register (CCM_TARGET_ROOT96_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B010	Miscellaneous Register (CCM_MISC96)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B014	Miscellaneous Register (CCM_MISC_ROOT96_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B018	Miscellaneous Register (CCM_MISC_ROOT96_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B01C	Miscellaneous Register (CCM_MISC_ROOT96_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B020	Post Divider Register (CCM_POST96)	32	R/W	0000_0000h	5.1.7.18/ 480

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B024	Post Divider Register (CCM_POST_ROOT96_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B028	Post Divider Register (CCM_POST_ROOT96_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B02C	Post Divider Register (CCM_POST_ROOT96_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B030	Pre Divider Register (CCM_PRE96)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B034	Pre Divider Register (CCM_PRE_ROOT96_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B038	Pre Divider Register (CCM_PRE_ROOT96_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B03C	Pre Divider Register (CCM_PRE_ROOT96_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B070	Access Control Register (CCM_ACCESS_CTRL96)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B074	Access Control Register (CCM_ACCESS_CTRL_ROOT96_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B078	Access Control Register (CCM_ACCESS_CTRL_ROOT96_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B07C	Access Control Register (CCM_ACCESS_CTRL_ROOT96_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B080	Target Register (CCM_TARGET_ROOT97)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B084	Target Register (CCM_TARGET_ROOT97_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B088	Target Register (CCM_TARGET_ROOT97_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B08C	Target Register (CCM_TARGET_ROOT97_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B090	Miscellaneous Register (CCM_MISC97)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B094	Miscellaneous Register (CCM_MISC_ROOT97_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B098	Miscellaneous Register (CCM_MISC_ROOT97_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B09C	Miscellaneous Register (CCM_MISC_ROOT97_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B0A0	Post Divider Register (CCM_POST97)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B0A4	Post Divider Register (CCM_POST_ROOT97_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B0A8	Post Divider Register (CCM_POST_ROOT97_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B0AC	Post Divider Register (CCM_POST_ROOT97_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B0B0	Pre Divider Register (CCM_PRE97)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B0B4	Pre Divider Register (CCM_PRE_ROOT97_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B0B8	Pre Divider Register (CCM_PRE_ROOT97_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B0BC	Pre Divider Register (CCM_PRE_ROOT97_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B0F0	Access Control Register (CCM_ACCESS_CTRL97)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B0F4	Access Control Register (CCM_ACCESS_CTRL_ROOT97_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B0F8	Access Control Register (CCM_ACCESS_CTRL_ROOT97_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B0FC	Access Control Register (CCM_ACCESS_CTRL_ROOT97_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B100	Target Register (CCM_TARGET_ROOT98)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B104	Target Register (CCM_TARGET_ROOT98_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B108	Target Register (CCM_TARGET_ROOT98_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B10C	Target Register (CCM_TARGET_ROOT98_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B110	Miscellaneous Register (CCM_MISC98)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B114	Miscellaneous Register (CCM_MISC_ROOT98_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B118	Miscellaneous Register (CCM_MISC_ROOT98_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B11C	Miscellaneous Register (CCM_MISC_ROOT98_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B120	Post Divider Register (CCM_POST98)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B124	Post Divider Register (CCM_POST_ROOT98_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B128	Post Divider Register (CCM_POST_ROOT98_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B12C	Post Divider Register (CCM_POST_ROOT98_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B130	Pre Divider Register (CCM_PRE98)	32	R/W	1000_0000h	5.1.7.22/ 492

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B134	Pre Divider Register (CCM_PRE_ROOT98_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B138	Pre Divider Register (CCM_PRE_ROOT98_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B13C	Pre Divider Register (CCM_PRE_ROOT98_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B170	Access Control Register (CCM_ACCESS_CTRL98)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B174	Access Control Register (CCM_ACCESS_CTRL_ROOT98_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B178	Access Control Register (CCM_ACCESS_CTRL_ROOT98_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B17C	Access Control Register (CCM_ACCESS_CTRL_ROOT98_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B180	Target Register (CCM_TARGET_ROOT99)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B184	Target Register (CCM_TARGET_ROOT99_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B188	Target Register (CCM_TARGET_ROOT99_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B18C	Target Register (CCM_TARGET_ROOT99_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B190	Miscellaneous Register (CCM_MISC99)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B194	Miscellaneous Register (CCM_MISC_ROOT99_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B198	Miscellaneous Register (CCM_MISC_ROOT99_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B19C	Miscellaneous Register (CCM_MISC_ROOT99_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B1A0	Post Divider Register (CCM_POST99)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B1A4	Post Divider Register (CCM_POST_ROOT99_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B1A8	Post Divider Register (CCM_POST_ROOT99_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B1AC	Post Divider Register (CCM_POST_ROOT99_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B1B0	Pre Divider Register (CCM_PRE99)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B1B4	Pre Divider Register (CCM_PRE_ROOT99_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B1B8	Pre Divider Register (CCM_PRE_ROOT99_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B1BC	Pre Divider Register (CCM_PRE_ROOT99_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B1F0	Access Control Register (CCM_ACCESS_CTRL99)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B1F4	Access Control Register (CCM_ACCESS_CTRL_ROOT99_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B1F8	Access Control Register (CCM_ACCESS_CTRL_ROOT99_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B1FC	Access Control Register (CCM_ACCESS_CTRL_ROOT99_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B200	Target Register (CCM_TARGET_ROOT100)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B204	Target Register (CCM_TARGET_ROOT100_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B208	Target Register (CCM_TARGET_ROOT100_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B20C	Target Register (CCM_TARGET_ROOT100_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B210	Miscellaneous Register (CCM_MISC100)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B214	Miscellaneous Register (CCM_MISC_ROOT100_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B218	Miscellaneous Register (CCM_MISC_ROOT100_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B21C	Miscellaneous Register (CCM_MISC_ROOT100_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B220	Post Divider Register (CCM_POST100)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B224	Post Divider Register (CCM_POST_ROOT100_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B228	Post Divider Register (CCM_POST_ROOT100_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B22C	Post Divider Register (CCM_POST_ROOT100_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B230	Pre Divider Register (CCM_PRE100)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B234	Pre Divider Register (CCM_PRE_ROOT100_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B238	Pre Divider Register (CCM_PRE_ROOT100_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B23C	Pre Divider Register (CCM_PRE_ROOT100_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B270	Access Control Register (CCM_ACCESS_CTRL100)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B274	Access Control Register (CCM_ACCESS_CTRL_ROOT100_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B278	Access Control Register (CCM_ACCESS_CTRL_ROOT100_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B27C	Access Control Register (CCM_ACCESS_CTRL_ROOT100_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B280	Target Register (CCM_TARGET_ROOT101)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B284	Target Register (CCM_TARGET_ROOT101_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B288	Target Register (CCM_TARGET_ROOT101_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B28C	Target Register (CCM_TARGET_ROOT101_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B290	Miscellaneous Register (CCM_MISC101)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B294	Miscellaneous Register (CCM_MISC_ROOT101_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B298	Miscellaneous Register (CCM_MISC_ROOT101_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B29C	Miscellaneous Register (CCM_MISC_ROOT101_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B2A0	Post Divider Register (CCM_POST101)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B2A4	Post Divider Register (CCM_POST_ROOT101_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B2A8	Post Divider Register (CCM_POST_ROOT101_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B2AC	Post Divider Register (CCM_POST_ROOT101_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B2B0	Pre Divider Register (CCM_PRE101)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B2B4	Pre Divider Register (CCM_PRE_ROOT101_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B2B8	Pre Divider Register (CCM_PRE_ROOT101_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B2BC	Pre Divider Register (CCM_PRE_ROOT101_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B2F0	Access Control Register (CCM_ACCESS_CTRL101)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B2F4	Access Control Register (CCM_ACCESS_CTRL_ROOT101_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B2F8	Access Control Register (CCM_ACCESS_CTRL_ROOT101_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B2FC	Access Control Register (CCM_ACCESS_CTRL_ROOT101_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B300	Target Register (CCM_TARGET_ROOT102)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B304	Target Register (CCM_TARGET_ROOT102_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B308	Target Register (CCM_TARGET_ROOT102_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B30C	Target Register (CCM_TARGET_ROOT102_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B310	Miscellaneous Register (CCM_MISC102)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B314	Miscellaneous Register (CCM_MISC_ROOT102_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B318	Miscellaneous Register (CCM_MISC_ROOT102_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B31C	Miscellaneous Register (CCM_MISC_ROOT102_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B320	Post Divider Register (CCM_POST102)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B324	Post Divider Register (CCM_POST_ROOT102_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B328	Post Divider Register (CCM_POST_ROOT102_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B32C	Post Divider Register (CCM_POST_ROOT102_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B330	Pre Divider Register (CCM_PRE102)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B334	Pre Divider Register (CCM_PRE_ROOT102_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B338	Pre Divider Register (CCM_PRE_ROOT102_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B33C	Pre Divider Register (CCM_PRE_ROOT102_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B370	Access Control Register (CCM_ACCESS_CTRL102)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B374	Access Control Register (CCM_ACCESS_CTRL_ROOT102_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B378	Access Control Register (CCM_ACCESS_CTRL_ROOT102_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B37C	Access Control Register (CCM_ACCESS_CTRL_ROOT102_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B380	Target Register (CCM_TARGET_ROOT103)	32	R/W	1000_0000h	5.1.7.10/ 468

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B384	Target Register (CCM_TARGET_ROOT103_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B388	Target Register (CCM_TARGET_ROOT103_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B38C	Target Register (CCM_TARGET_ROOT103_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B390	Miscellaneous Register (CCM_MISC103)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B394	Miscellaneous Register (CCM_MISC_ROOT103_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B398	Miscellaneous Register (CCM_MISC_ROOT103_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B39C	Miscellaneous Register (CCM_MISC_ROOT103_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B3A0	Post Divider Register (CCM_POST103)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B3A4	Post Divider Register (CCM_POST_ROOT103_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B3A8	Post Divider Register (CCM_POST_ROOT103_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B3AC	Post Divider Register (CCM_POST_ROOT103_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B3B0	Pre Divider Register (CCM_PRE103)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B3B4	Pre Divider Register (CCM_PRE_ROOT103_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B3B8	Pre Divider Register (CCM_PRE_ROOT103_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B3BC	Pre Divider Register (CCM_PRE_ROOT103_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B3F0	Access Control Register (CCM_ACCESS_CTRL103)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B3F4	Access Control Register (CCM_ACCESS_CTRL_ROOT103_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B3F8	Access Control Register (CCM_ACCESS_CTRL_ROOT103_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B3FC	Access Control Register (CCM_ACCESS_CTRL_ROOT103_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B400	Target Register (CCM_TARGET_ROOT104)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B404	Target Register (CCM_TARGET_ROOT104_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B408	Target Register (CCM_TARGET_ROOT104_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B40C	Target Register (CCM_TARGET_ROOT104_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B410	Miscellaneous Register (CCM_MISC104)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B414	Miscellaneous Register (CCM_MISC_ROOT104_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B418	Miscellaneous Register (CCM_MISC_ROOT104_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B41C	Miscellaneous Register (CCM_MISC_ROOT104_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B420	Post Divider Register (CCM_POST104)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B424	Post Divider Register (CCM_POST_ROOT104_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B428	Post Divider Register (CCM_POST_ROOT104_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B42C	Post Divider Register (CCM_POST_ROOT104_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B430	Pre Divider Register (CCM_PRE104)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B434	Pre Divider Register (CCM_PRE_ROOT104_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B438	Pre Divider Register (CCM_PRE_ROOT104_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B43C	Pre Divider Register (CCM_PRE_ROOT104_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B470	Access Control Register (CCM_ACCESS_CTRL104)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B474	Access Control Register (CCM_ACCESS_CTRL_ROOT104_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B478	Access Control Register (CCM_ACCESS_CTRL_ROOT104_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B47C	Access Control Register (CCM_ACCESS_CTRL_ROOT104_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B480	Target Register (CCM_TARGET_ROOT105)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B484	Target Register (CCM_TARGET_ROOT105_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B488	Target Register (CCM_TARGET_ROOT105_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B48C	Target Register (CCM_TARGET_ROOT105_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B490	Miscellaneous Register (CCM_MISC105)	32	R/W	0000_0000h	5.1.7.14/ 476

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B494	Miscellaneous Register (CCM_MISC_ROOT105_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B498	Miscellaneous Register (CCM_MISC_ROOT105_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B49C	Miscellaneous Register (CCM_MISC_ROOT105_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B4A0	Post Divider Register (CCM_POST105)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B4A4	Post Divider Register (CCM_POST_ROOT105_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B4A8	Post Divider Register (CCM_POST_ROOT105_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B4AC	Post Divider Register (CCM_POST_ROOT105_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B4B0	Pre Divider Register (CCM_PRE105)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B4B4	Pre Divider Register (CCM_PRE_ROOT105_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B4B8	Pre Divider Register (CCM_PRE_ROOT105_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B4BC	Pre Divider Register (CCM_PRE_ROOT105_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B4F0	Access Control Register (CCM_ACCESS_CTRL105)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B4F4	Access Control Register (CCM_ACCESS_CTRL_ROOT105_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B4F8	Access Control Register (CCM_ACCESS_CTRL_ROOT105_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B4FC	Access Control Register (CCM_ACCESS_CTRL_ROOT105_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B500	Target Register (CCM_TARGET_ROOT106)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B504	Target Register (CCM_TARGET_ROOT106_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B508	Target Register (CCM_TARGET_ROOT106_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B50C	Target Register (CCM_TARGET_ROOT106_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B510	Miscellaneous Register (CCM_MISC106)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B514	Miscellaneous Register (CCM_MISC_ROOT106_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B518	Miscellaneous Register (CCM_MISC_ROOT106_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B51C	Miscellaneous Register (CCM_MISC_ROOT106_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B520	Post Divider Register (CCM_POST106)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B524	Post Divider Register (CCM_POST_ROOT106_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B528	Post Divider Register (CCM_POST_ROOT106_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B52C	Post Divider Register (CCM_POST_ROOT106_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B530	Pre Divider Register (CCM_PRE106)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B534	Pre Divider Register (CCM_PRE_ROOT106_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B538	Pre Divider Register (CCM_PRE_ROOT106_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B53C	Pre Divider Register (CCM_PRE_ROOT106_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B570	Access Control Register (CCM_ACCESS_CTRL106)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B574	Access Control Register (CCM_ACCESS_CTRL_ROOT106_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B578	Access Control Register (CCM_ACCESS_CTRL_ROOT106_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B57C	Access Control Register (CCM_ACCESS_CTRL_ROOT106_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B580	Target Register (CCM_TARGET_ROOT107)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B584	Target Register (CCM_TARGET_ROOT107_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B588	Target Register (CCM_TARGET_ROOT107_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B58C	Target Register (CCM_TARGET_ROOT107_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B590	Miscellaneous Register (CCM_MISC107)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B594	Miscellaneous Register (CCM_MISC_ROOT107_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B598	Miscellaneous Register (CCM_MISC_ROOT107_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B59C	Miscellaneous Register (CCM_MISC_ROOT107_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B5A0	Post Divider Register (CCM_POST107)	32	R/W	0000_0000h	5.1.7.18/ 480

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B5A4	Post Divider Register (CCM_POST_ROOT107_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B5A8	Post Divider Register (CCM_POST_ROOT107_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B5AC	Post Divider Register (CCM_POST_ROOT107_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B5B0	Pre Divider Register (CCM_PRE107)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B5B4	Pre Divider Register (CCM_PRE_ROOT107_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B5B8	Pre Divider Register (CCM_PRE_ROOT107_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B5BC	Pre Divider Register (CCM_PRE_ROOT107_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B5F0	Access Control Register (CCM_ACCESS_CTRL107)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B5F4	Access Control Register (CCM_ACCESS_CTRL_ROOT107_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B5F8	Access Control Register (CCM_ACCESS_CTRL_ROOT107_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B5FC	Access Control Register (CCM_ACCESS_CTRL_ROOT107_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B600	Target Register (CCM_TARGET_ROOT108)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B604	Target Register (CCM_TARGET_ROOT108_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B608	Target Register (CCM_TARGET_ROOT108_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B60C	Target Register (CCM_TARGET_ROOT108_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B610	Miscellaneous Register (CCM_MISC108)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B614	Miscellaneous Register (CCM_MISC_ROOT108_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B618	Miscellaneous Register (CCM_MISC_ROOT108_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B61C	Miscellaneous Register (CCM_MISC_ROOT108_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B620	Post Divider Register (CCM_POST108)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B624	Post Divider Register (CCM_POST_ROOT108_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B628	Post Divider Register (CCM_POST_ROOT108_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B62C	Post Divider Register (CCM_POST_ROOT108_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B630	Pre Divider Register (CCM_PRE108)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B634	Pre Divider Register (CCM_PRE_ROOT108_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B638	Pre Divider Register (CCM_PRE_ROOT108_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B63C	Pre Divider Register (CCM_PRE_ROOT108_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B670	Access Control Register (CCM_ACCESS_CTRL108)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B674	Access Control Register (CCM_ACCESS_CTRL_ROOT108_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B678	Access Control Register (CCM_ACCESS_CTRL_ROOT108_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B67C	Access Control Register (CCM_ACCESS_CTRL_ROOT108_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B680	Target Register (CCM_TARGET_ROOT109)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B684	Target Register (CCM_TARGET_ROOT109_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B688	Target Register (CCM_TARGET_ROOT109_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B68C	Target Register (CCM_TARGET_ROOT109_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B690	Miscellaneous Register (CCM_MISC109)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B694	Miscellaneous Register (CCM_MISC_ROOT109_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B698	Miscellaneous Register (CCM_MISC_ROOT109_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B69C	Miscellaneous Register (CCM_MISC_ROOT109_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B6A0	Post Divider Register (CCM_POST109)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B6A4	Post Divider Register (CCM_POST_ROOT109_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B6A8	Post Divider Register (CCM_POST_ROOT109_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B6AC	Post Divider Register (CCM_POST_ROOT109_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B6B0	Pre Divider Register (CCM_PRE109)	32	R/W	1000_0000h	5.1.7.22/ 492

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B6B4	Pre Divider Register (CCM_PRE_ROOT109_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B6B8	Pre Divider Register (CCM_PRE_ROOT109_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B6BC	Pre Divider Register (CCM_PRE_ROOT109_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B6F0	Access Control Register (CCM_ACCESS_CTRL109)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B6F4	Access Control Register (CCM_ACCESS_CTRL_ROOT109_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B6F8	Access Control Register (CCM_ACCESS_CTRL_ROOT109_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B6FC	Access Control Register (CCM_ACCESS_CTRL_ROOT109_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B700	Target Register (CCM_TARGET_ROOT110)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B704	Target Register (CCM_TARGET_ROOT110_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B708	Target Register (CCM_TARGET_ROOT110_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B70C	Target Register (CCM_TARGET_ROOT110_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B710	Miscellaneous Register (CCM_MISC110)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B714	Miscellaneous Register (CCM_MISC_ROOT110_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B718	Miscellaneous Register (CCM_MISC_ROOT110_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B71C	Miscellaneous Register (CCM_MISC_ROOT110_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B720	Post Divider Register (CCM_POST110)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B724	Post Divider Register (CCM_POST_ROOT110_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B728	Post Divider Register (CCM_POST_ROOT110_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B72C	Post Divider Register (CCM_POST_ROOT110_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B730	Pre Divider Register (CCM_PRE110)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B734	Pre Divider Register (CCM_PRE_ROOT110_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B738	Pre Divider Register (CCM_PRE_ROOT110_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B73C	Pre Divider Register (CCM_PRE_ROOT110_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B770	Access Control Register (CCM_ACCESS_CTRL110)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B774	Access Control Register (CCM_ACCESS_CTRL_ROOT110_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B778	Access Control Register (CCM_ACCESS_CTRL_ROOT110_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B77C	Access Control Register (CCM_ACCESS_CTRL_ROOT110_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B780	Target Register (CCM_TARGET_ROOT111)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B784	Target Register (CCM_TARGET_ROOT111_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B788	Target Register (CCM_TARGET_ROOT111_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B78C	Target Register (CCM_TARGET_ROOT111_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B790	Miscellaneous Register (CCM_MISC111)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B794	Miscellaneous Register (CCM_MISC_ROOT111_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B798	Miscellaneous Register (CCM_MISC_ROOT111_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B79C	Miscellaneous Register (CCM_MISC_ROOT111_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B7A0	Post Divider Register (CCM_POST111)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B7A4	Post Divider Register (CCM_POST_ROOT111_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B7A8	Post Divider Register (CCM_POST_ROOT111_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B7AC	Post Divider Register (CCM_POST_ROOT111_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B7B0	Pre Divider Register (CCM_PRE111)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B7B4	Pre Divider Register (CCM_PRE_ROOT111_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B7B8	Pre Divider Register (CCM_PRE_ROOT111_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B7BC	Pre Divider Register (CCM_PRE_ROOT111_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B7F0	Access Control Register (CCM_ACCESS_CTRL111)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B7F4	Access Control Register (CCM_ACCESS_CTRL_ROOT111_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B7F8	Access Control Register (CCM_ACCESS_CTRL_ROOT111_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B7FC	Access Control Register (CCM_ACCESS_CTRL_ROOT111_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B800	Target Register (CCM_TARGET_ROOT112)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B804	Target Register (CCM_TARGET_ROOT112_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B808	Target Register (CCM_TARGET_ROOT112_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B80C	Target Register (CCM_TARGET_ROOT112_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B810	Miscellaneous Register (CCM_MISC112)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B814	Miscellaneous Register (CCM_MISC_ROOT112_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B818	Miscellaneous Register (CCM_MISC_ROOT112_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B81C	Miscellaneous Register (CCM_MISC_ROOT112_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B820	Post Divider Register (CCM_POST112)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B824	Post Divider Register (CCM_POST_ROOT112_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B828	Post Divider Register (CCM_POST_ROOT112_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B82C	Post Divider Register (CCM_POST_ROOT112_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B830	Pre Divider Register (CCM_PRE112)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B834	Pre Divider Register (CCM_PRE_ROOT112_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B838	Pre Divider Register (CCM_PRE_ROOT112_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B83C	Pre Divider Register (CCM_PRE_ROOT112_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B870	Access Control Register (CCM_ACCESS_CTRL112)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B874	Access Control Register (CCM_ACCESS_CTRL_ROOT112_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B878	Access Control Register (CCM_ACCESS_CTRL_ROOT112_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B87C	Access Control Register (CCM_ACCESS_CTRL_ROOT112_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B880	Target Register (CCM_TARGET_ROOT113)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B884	Target Register (CCM_TARGET_ROOT113_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B888	Target Register (CCM_TARGET_ROOT113_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B88C	Target Register (CCM_TARGET_ROOT113_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B890	Miscellaneous Register (CCM_MISC113)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B894	Miscellaneous Register (CCM_MISC_ROOT113_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B898	Miscellaneous Register (CCM_MISC_ROOT113_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B89C	Miscellaneous Register (CCM_MISC_ROOT113_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B8A0	Post Divider Register (CCM_POST113)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B8A4	Post Divider Register (CCM_POST_ROOT113_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B8A8	Post Divider Register (CCM_POST_ROOT113_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B8AC	Post Divider Register (CCM_POST_ROOT113_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B8B0	Pre Divider Register (CCM_PRE113)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B8B4	Pre Divider Register (CCM_PRE_ROOT113_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B8B8	Pre Divider Register (CCM_PRE_ROOT113_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B8BC	Pre Divider Register (CCM_PRE_ROOT113_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B8F0	Access Control Register (CCM_ACCESS_CTRL113)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B8F4	Access Control Register (CCM_ACCESS_CTRL_ROOT113_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B8F8	Access Control Register (CCM_ACCESS_CTRL_ROOT113_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B8FC	Access Control Register (CCM_ACCESS_CTRL_ROOT113_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B900	Target Register (CCM_TARGET_ROOT114)	32	R/W	1000_0000h	5.1.7.10/ 468

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B904	Target Register (CCM_TARGET_ROOT114_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B908	Target Register (CCM_TARGET_ROOT114_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_B90C	Target Register (CCM_TARGET_ROOT114_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B910	Miscellaneous Register (CCM_MISC114)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B914	Miscellaneous Register (CCM_MISC_ROOT114_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B918	Miscellaneous Register (CCM_MISC_ROOT114_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B91C	Miscellaneous Register (CCM_MISC_ROOT114_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B920	Post Divider Register (CCM_POST114)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B924	Post Divider Register (CCM_POST_ROOT114_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B928	Post Divider Register (CCM_POST_ROOT114_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B92C	Post Divider Register (CCM_POST_ROOT114_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B930	Pre Divider Register (CCM_PRE114)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B934	Pre Divider Register (CCM_PRE_ROOT114_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B938	Pre Divider Register (CCM_PRE_ROOT114_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B93C	Pre Divider Register (CCM_PRE_ROOT114_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B970	Access Control Register (CCM_ACCESS_CTRL114)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B974	Access Control Register (CCM_ACCESS_CTRL_ROOT114_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B978	Access Control Register (CCM_ACCESS_CTRL_ROOT114_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B97C	Access Control Register (CCM_ACCESS_CTRL_ROOT114_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_B980	Target Register (CCM_TARGET_ROOT115)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_B984	Target Register (CCM_TARGET_ROOT115_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_B988	Target Register (CCM_TARGET_ROOT115_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_B98C	Target Register (CCM_TARGET_ROOT115_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_B990	Miscellaneous Register (CCM_MISC115)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_B994	Miscellaneous Register (CCM_MISC_ROOT115_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_B998	Miscellaneous Register (CCM_MISC_ROOT115_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_B99C	Miscellaneous Register (CCM_MISC_ROOT115_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_B9A0	Post Divider Register (CCM_POST115)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_B9A4	Post Divider Register (CCM_POST_ROOT115_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_B9A8	Post Divider Register (CCM_POST_ROOT115_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_B9AC	Post Divider Register (CCM_POST_ROOT115_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_B9B0	Pre Divider Register (CCM_PRE115)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_B9B4	Pre Divider Register (CCM_PRE_ROOT115_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_B9B8	Pre Divider Register (CCM_PRE_ROOT115_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_B9BC	Pre Divider Register (CCM_PRE_ROOT115_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_B9F0	Access Control Register (CCM_ACCESS_CTRL115)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_B9F4	Access Control Register (CCM_ACCESS_CTRL_ROOT115_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_B9F8	Access Control Register (CCM_ACCESS_CTRL_ROOT115_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_B9FC	Access Control Register (CCM_ACCESS_CTRL_ROOT115_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_BA00	Target Register (CCM_TARGET_ROOT116)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_BA04	Target Register (CCM_TARGET_ROOT116_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_BA08	Target Register (CCM_TARGET_ROOT116_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_BA0C	Target Register (CCM_TARGET_ROOT116_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_BA10	Miscellaneous Register (CCM_MISC116)	32	R/W	0000_0000h	5.1.7.14/ 476

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_BA14	Miscellaneous Register (CCM_MISC_ROOT116_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_BA18	Miscellaneous Register (CCM_MISC_ROOT116_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_BA1C	Miscellaneous Register (CCM_MISC_ROOT116_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_BA20	Post Divider Register (CCM_POST116)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_BA24	Post Divider Register (CCM_POST_ROOT116_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_BA28	Post Divider Register (CCM_POST_ROOT116_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_BA2C	Post Divider Register (CCM_POST_ROOT116_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_BA30	Pre Divider Register (CCM_PRE116)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_BA34	Pre Divider Register (CCM_PRE_ROOT116_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_BA38	Pre Divider Register (CCM_PRE_ROOT116_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_BA3C	Pre Divider Register (CCM_PRE_ROOT116_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_BA70	Access Control Register (CCM_ACCESS_CTRL116)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_BA74	Access Control Register (CCM_ACCESS_CTRL_ROOT116_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_BA78	Access Control Register (CCM_ACCESS_CTRL_ROOT116_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_BA7C	Access Control Register (CCM_ACCESS_CTRL_ROOT116_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_BA80	Target Register (CCM_TARGET_ROOT117)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_BA84	Target Register (CCM_TARGET_ROOT117_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_BA88	Target Register (CCM_TARGET_ROOT117_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_BA8C	Target Register (CCM_TARGET_ROOT117_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_BA90	Miscellaneous Register (CCM_MISC117)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_BA94	Miscellaneous Register (CCM_MISC_ROOT117_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_BA98	Miscellaneous Register (CCM_MISC_ROOT117_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_BA9C	Miscellaneous Register (CCM_MISC_ROOT117_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_BAA0	Post Divider Register (CCM_POST117)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_BAA4	Post Divider Register (CCM_POST_ROOT117_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_BAA8	Post Divider Register (CCM_POST_ROOT117_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_BAAC	Post Divider Register (CCM_POST_ROOT117_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_BAB0	Pre Divider Register (CCM_PRE117)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_BAB4	Pre Divider Register (CCM_PRE_ROOT117_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_BAB8	Pre Divider Register (CCM_PRE_ROOT117_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_BABC	Pre Divider Register (CCM_PRE_ROOT117_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_BAF0	Access Control Register (CCM_ACCESS_CTRL117)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_BAF4	Access Control Register (CCM_ACCESS_CTRL_ROOT117_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_BAF8	Access Control Register (CCM_ACCESS_CTRL_ROOT117_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_BAFC	Access Control Register (CCM_ACCESS_CTRL_ROOT117_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_BB00	Target Register (CCM_TARGET_ROOT118)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_BB04	Target Register (CCM_TARGET_ROOT118_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_BB08	Target Register (CCM_TARGET_ROOT118_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_BB0C	Target Register (CCM_TARGET_ROOT118_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_BB10	Miscellaneous Register (CCM_MISC118)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_BB14	Miscellaneous Register (CCM_MISC_ROOT118_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_BB18	Miscellaneous Register (CCM_MISC_ROOT118_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_BB1C	Miscellaneous Register (CCM_MISC_ROOT118_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_BB20	Post Divider Register (CCM_POST118)	32	R/W	0000_0000h	5.1.7.18/ 480

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_BB24	Post Divider Register (CCM_POST_ROOT118_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_BB28	Post Divider Register (CCM_POST_ROOT118_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_BB2C	Post Divider Register (CCM_POST_ROOT118_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_BB30	Pre Divider Register (CCM_PRE118)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_BB34	Pre Divider Register (CCM_PRE_ROOT118_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_BB38	Pre Divider Register (CCM_PRE_ROOT118_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_BB3C	Pre Divider Register (CCM_PRE_ROOT118_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_BB70	Access Control Register (CCM_ACCESS_CTRL118)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_BB74	Access Control Register (CCM_ACCESS_CTRL_ROOT118_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_BB78	Access Control Register (CCM_ACCESS_CTRL_ROOT118_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_BB7C	Access Control Register (CCM_ACCESS_CTRL_ROOT118_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_BB80	Target Register (CCM_TARGET_ROOT119)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_BB84	Target Register (CCM_TARGET_ROOT119_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_BB88	Target Register (CCM_TARGET_ROOT119_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_BB8C	Target Register (CCM_TARGET_ROOT119_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_BB90	Miscellaneous Register (CCM_MISC119)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_BB94	Miscellaneous Register (CCM_MISC_ROOT119_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_BB98	Miscellaneous Register (CCM_MISC_ROOT119_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_BB9C	Miscellaneous Register (CCM_MISC_ROOT119_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_BBA0	Post Divider Register (CCM_POST119)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_BBA4	Post Divider Register (CCM_POST_ROOT119_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_BBA8	Post Divider Register (CCM_POST_ROOT119_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_BBAC	Post Divider Register (CCM_POST_ROOT119_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_BBB0	Pre Divider Register (CCM_PRE119)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_BBB4	Pre Divider Register (CCM_PRE_ROOT119_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_BBB8	Pre Divider Register (CCM_PRE_ROOT119_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_BBBC	Pre Divider Register (CCM_PRE_ROOT119_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_BBF0	Access Control Register (CCM_ACCESS_CTRL119)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_BBF4	Access Control Register (CCM_ACCESS_CTRL_ROOT119_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_BBF8	Access Control Register (CCM_ACCESS_CTRL_ROOT119_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_BBFC	Access Control Register (CCM_ACCESS_CTRL_ROOT119_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_BC00	Target Register (CCM_TARGET_ROOT120)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_BC04	Target Register (CCM_TARGET_ROOT120_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_BC08	Target Register (CCM_TARGET_ROOT120_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_BC0C	Target Register (CCM_TARGET_ROOT120_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_BC10	Miscellaneous Register (CCM_MISC120)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_BC14	Miscellaneous Register (CCM_MISC_ROOT120_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_BC18	Miscellaneous Register (CCM_MISC_ROOT120_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_BC1C	Miscellaneous Register (CCM_MISC_ROOT120_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_BC20	Post Divider Register (CCM_POST120)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_BC24	Post Divider Register (CCM_POST_ROOT120_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_BC28	Post Divider Register (CCM_POST_ROOT120_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_BC2C	Post Divider Register (CCM_POST_ROOT120_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_BC30	Pre Divider Register (CCM_PRE120)	32	R/W	1000_0000h	5.1.7.22/ 492

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_BC34	Pre Divider Register (CCM_PRE_ROOT120_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_BC38	Pre Divider Register (CCM_PRE_ROOT120_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_BC3C	Pre Divider Register (CCM_PRE_ROOT120_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_BC70	Access Control Register (CCM_ACCESS_CTRL120)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_BC74	Access Control Register (CCM_ACCESS_CTRL_ROOT120_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_BC78	Access Control Register (CCM_ACCESS_CTRL_ROOT120_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_BC7C	Access Control Register (CCM_ACCESS_CTRL_ROOT120_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_BC80	Target Register (CCM_TARGET_ROOT121)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_BC84	Target Register (CCM_TARGET_ROOT121_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_BC88	Target Register (CCM_TARGET_ROOT121_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_BC8C	Target Register (CCM_TARGET_ROOT121_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_BC90	Miscellaneous Register (CCM_MISC121)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_BC94	Miscellaneous Register (CCM_MISC_ROOT121_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_BC98	Miscellaneous Register (CCM_MISC_ROOT121_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_BC9C	Miscellaneous Register (CCM_MISC_ROOT121_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_BCA0	Post Divider Register (CCM_POST121)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_BCA4	Post Divider Register (CCM_POST_ROOT121_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_BCA8	Post Divider Register (CCM_POST_ROOT121_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_BCAC	Post Divider Register (CCM_POST_ROOT121_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_BCB0	Pre Divider Register (CCM_PRE121)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_BCB4	Pre Divider Register (CCM_PRE_ROOT121_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_BCB8	Pre Divider Register (CCM_PRE_ROOT121_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_BCBC	Pre Divider Register (CCM_PRE_ROOT121_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_BCF0	Access Control Register (CCM_ACCESS_CTRL121)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_BCF4	Access Control Register (CCM_ACCESS_CTRL_ROOT121_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_BCF8	Access Control Register (CCM_ACCESS_CTRL_ROOT121_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_BCFC	Access Control Register (CCM_ACCESS_CTRL_ROOT121_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_BD00	Target Register (CCM_TARGET_ROOT122)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_BD04	Target Register (CCM_TARGET_ROOT122_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_BD08	Target Register (CCM_TARGET_ROOT122_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_BD0C	Target Register (CCM_TARGET_ROOT122_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_BD10	Miscellaneous Register (CCM_MISC122)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_BD14	Miscellaneous Register (CCM_MISC_ROOT122_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_BD18	Miscellaneous Register (CCM_MISC_ROOT122_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_BD1C	Miscellaneous Register (CCM_MISC_ROOT122_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_BD20	Post Divider Register (CCM_POST122)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_BD24	Post Divider Register (CCM_POST_ROOT122_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_BD28	Post Divider Register (CCM_POST_ROOT122_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_BD2C	Post Divider Register (CCM_POST_ROOT122_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_BD30	Pre Divider Register (CCM_PRE122)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_BD34	Pre Divider Register (CCM_PRE_ROOT122_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_BD38	Pre Divider Register (CCM_PRE_ROOT122_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_BD3C	Pre Divider Register (CCM_PRE_ROOT122_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_BD70	Access Control Register (CCM_ACCESS_CTRL122)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_BD74	Access Control Register (CCM_ACCESS_CTRL_ROOT122_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_BD78	Access Control Register (CCM_ACCESS_CTRL_ROOT122_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_BD7C	Access Control Register (CCM_ACCESS_CTRL_ROOT122_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_BD80	Target Register (CCM_TARGET_ROOT123)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_BD84	Target Register (CCM_TARGET_ROOT123_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_BD88	Target Register (CCM_TARGET_ROOT123_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_BD8C	Target Register (CCM_TARGET_ROOT123_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_BD90	Miscellaneous Register (CCM_MISC123)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_BD94	Miscellaneous Register (CCM_MISC_ROOT123_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_BD98	Miscellaneous Register (CCM_MISC_ROOT123_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_BD9C	Miscellaneous Register (CCM_MISC_ROOT123_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_BDA0	Post Divider Register (CCM_POST123)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_BDA4	Post Divider Register (CCM_POST_ROOT123_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_BDA8	Post Divider Register (CCM_POST_ROOT123_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_BDAC	Post Divider Register (CCM_POST_ROOT123_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_BDB0	Pre Divider Register (CCM_PRE123)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_BDB4	Pre Divider Register (CCM_PRE_ROOT123_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_BDB8	Pre Divider Register (CCM_PRE_ROOT123_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_BDBC	Pre Divider Register (CCM_PRE_ROOT123_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_BDF0	Access Control Register (CCM_ACCESS_CTRL123)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_BDF4	Access Control Register (CCM_ACCESS_CTRL_ROOT123_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_BDF8	Access Control Register (CCM_ACCESS_CTRL_ROOT123_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_BDFC	Access Control Register (CCM_ACCESS_CTRL_ROOT123_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_BE00	Target Register (CCM_TARGET_ROOT124)	32	R/W	1000_0000h	5.1.7.10/ 468
3038_BE04	Target Register (CCM_TARGET_ROOT124_SET)	32	R/W	0000_0000h	5.1.7.11/ 470
3038_BE08	Target Register (CCM_TARGET_ROOT124_CLR)	32	R/W	0000_0000h	5.1.7.12/ 472
3038_BE0C	Target Register (CCM_TARGET_ROOT124_TOG)	32	R/W	0000_0000h	5.1.7.13/ 474
3038_BE10	Miscellaneous Register (CCM_MISC124)	32	R/W	0000_0000h	5.1.7.14/ 476
3038_BE14	Miscellaneous Register (CCM_MISC_ROOT124_SET)	32	R/W	0000_0000h	5.1.7.15/ 477
3038_BE18	Miscellaneous Register (CCM_MISC_ROOT124_CLR)	32	R/W	0000_0000h	5.1.7.16/ 478
3038_BE1C	Miscellaneous Register (CCM_MISC_ROOT124_TOG)	32	R/W	0000_0000h	5.1.7.17/ 479
3038_BE20	Post Divider Register (CCM_POST124)	32	R/W	0000_0000h	5.1.7.18/ 480
3038_BE24	Post Divider Register (CCM_POST_ROOT124_SET)	32	R/W	0000_0000h	5.1.7.19/ 483
3038_BE28	Post Divider Register (CCM_POST_ROOT124_CLR)	32	R/W	0000_0000h	5.1.7.20/ 486
3038_BE2C	Post Divider Register (CCM_POST_ROOT124_TOG)	32	R/W	0000_0000h	5.1.7.21/ 489
3038_BE30	Pre Divider Register (CCM_PRE124)	32	R/W	1000_0000h	5.1.7.22/ 492
3038_BE34	Pre Divider Register (CCM_PRE_ROOT124_SET)	32	R/W	0000_0000h	5.1.7.23/ 495
3038_BE38	Pre Divider Register (CCM_PRE_ROOT124_CLR)	32	R/W	0000_0000h	5.1.7.24/ 498
3038_BE3C	Pre Divider Register (CCM_PRE_ROOT124_TOG)	32	R/W	0000_0000h	5.1.7.25/ 501
3038_BE70	Access Control Register (CCM_ACCESS_CTRL124)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_BE74	Access Control Register (CCM_ACCESS_CTRL_ROOT124_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_BE78	Access Control Register (CCM_ACCESS_CTRL_ROOT124_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_BE7C	Access Control Register (CCM_ACCESS_CTRL_ROOT124_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_BEF0	Access Control Register (CCM_ACCESS_CTRL125)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_BEF4	Access Control Register (CCM_ACCESS_CTRL_ROOT125_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_BEF8	Access Control Register (CCM_ACCESS_CTRL_ROOT125_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_BEFC	Access Control Register (CCM_ACCESS_CTRL_ROOT125_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_BF70	Access Control Register (CCM_ACCESS_CTRL126)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_BF74	Access Control Register (CCM_ACCESS_CTRL_ROOT126_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_BF78	Access Control Register (CCM_ACCESS_CTRL_ROOT126_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_BF7C	Access Control Register (CCM_ACCESS_CTRL_ROOT126_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_BFF0	Access Control Register (CCM_ACCESS_CTRL127)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_BFF4	Access Control Register (CCM_ACCESS_CTRL_ROOT127_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_BFF8	Access Control Register (CCM_ACCESS_CTRL_ROOT127_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_BFFC	Access Control Register (CCM_ACCESS_CTRL_ROOT127_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C070	Access Control Register (CCM_ACCESS_CTRL128)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C074	Access Control Register (CCM_ACCESS_CTRL_ROOT128_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C078	Access Control Register (CCM_ACCESS_CTRL_ROOT128_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_C07C	Access Control Register (CCM_ACCESS_CTRL_ROOT128_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C0F0	Access Control Register (CCM_ACCESS_CTRL129)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C0F4	Access Control Register (CCM_ACCESS_CTRL_ROOT129_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C0F8	Access Control Register (CCM_ACCESS_CTRL_ROOT129_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_C0FC	Access Control Register (CCM_ACCESS_CTRL_ROOT129_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C170	Access Control Register (CCM_ACCESS_CTRL130)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C174	Access Control Register (CCM_ACCESS_CTRL_ROOT130_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C178	Access Control Register (CCM_ACCESS_CTRL_ROOT130_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_C17C	Access Control Register (CCM_ACCESS_CTRL_ROOT130_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C1F0	Access Control Register (CCM_ACCESS_CTRL131)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C1F4	Access Control Register (CCM_ACCESS_CTRL_ROOT131_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C1F8	Access Control Register (CCM_ACCESS_CTRL_ROOT131_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_C1FC	Access Control Register (CCM_ACCESS_CTRL_ROOT131_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C270	Access Control Register (CCM_ACCESS_CTRL132)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C274	Access Control Register (CCM_ACCESS_CTRL_ROOT132_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C278	Access Control Register (CCM_ACCESS_CTRL_ROOT132_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_C27C	Access Control Register (CCM_ACCESS_CTRL_ROOT132_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C2F0	Access Control Register (CCM_ACCESS_CTRL133)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C2F4	Access Control Register (CCM_ACCESS_CTRL_ROOT133_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C2F8	Access Control Register (CCM_ACCESS_CTRL_ROOT133_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_C2FC	Access Control Register (CCM_ACCESS_CTRL_ROOT133_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C370	Access Control Register (CCM_ACCESS_CTRL134)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C374	Access Control Register (CCM_ACCESS_CTRL_ROOT134_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C378	Access Control Register (CCM_ACCESS_CTRL_ROOT134_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_C37C	Access Control Register (CCM_ACCESS_CTRL_ROOT134_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C3F0	Access Control Register (CCM_ACCESS_CTRL135)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C3F4	Access Control Register (CCM_ACCESS_CTRL_ROOT135_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C3F8	Access Control Register (CCM_ACCESS_CTRL_ROOT135_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_C3FC	Access Control Register (CCM_ACCESS_CTRL_ROOT135_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C470	Access Control Register (CCM_ACCESS_CTRL136)	32	R/W	0000_0000h	5.1.7.26/ 504

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_C474	Access Control Register (CCM_ACCESS_CTRL_ROOT136_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C478	Access Control Register (CCM_ACCESS_CTRL_ROOT136_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_C47C	Access Control Register (CCM_ACCESS_CTRL_ROOT136_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C4F0	Access Control Register (CCM_ACCESS_CTRL137)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C4F4	Access Control Register (CCM_ACCESS_CTRL_ROOT137_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C4F8	Access Control Register (CCM_ACCESS_CTRL_ROOT137_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_C4FC	Access Control Register (CCM_ACCESS_CTRL_ROOT137_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C570	Access Control Register (CCM_ACCESS_CTRL138)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C574	Access Control Register (CCM_ACCESS_CTRL_ROOT138_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C578	Access Control Register (CCM_ACCESS_CTRL_ROOT138_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_C57C	Access Control Register (CCM_ACCESS_CTRL_ROOT138_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C5F0	Access Control Register (CCM_ACCESS_CTRL139)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C5F4	Access Control Register (CCM_ACCESS_CTRL_ROOT139_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C5F8	Access Control Register (CCM_ACCESS_CTRL_ROOT139_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_C5FC	Access Control Register (CCM_ACCESS_CTRL_ROOT139_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C670	Access Control Register (CCM_ACCESS_CTRL140)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C674	Access Control Register (CCM_ACCESS_CTRL_ROOT140_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C678	Access Control Register (CCM_ACCESS_CTRL_ROOT140_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509
3038_C67C	Access Control Register (CCM_ACCESS_CTRL_ROOT140_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511
3038_C6F0	Access Control Register (CCM_ACCESS_CTRL141)	32	R/W	0000_0000h	5.1.7.26/ 504
3038_C6F4	Access Control Register (CCM_ACCESS_CTRL_ROOT141_SET)	32	R/W	0000_0000h	5.1.7.27/ 506
3038_C6F8	Access Control Register (CCM_ACCESS_CTRL_ROOT141_CLR)	32	R/W	0000_0000h	5.1.7.28/ 509

Table continues on the next page...

CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3038_C6FC	Access Control Register (CCM_ACCESS_CTRL_ROOT141_TOG)	32	R/W	0000_0000h	5.1.7.29/ 511

5.1.7.1 General Purpose Register (CCM_GPR0*n*)

GPR0

Address: 3038_0000h base + 0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GP0																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CCM_GPR0*n* field descriptions

Field	Description
GP0	Timeout cycle count of ipg_clk, when perform read and write.

5.1.7.2 CCM PLL Control Register (CCM_PLL_CTRL*n*)

See Input Clocks for PLL control mapping.

NOTE

For the SoC to correctly power up after entering DSM, CCM_PLL_CTRLx must not be set to 0x0 or 0x3 for any domain in use.

Address: 3038_0000h base + 800h offset + (16d × i), where i=0d to 38d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	SETTING3		Reserved	Reserved	SETTING2		Reserved	Reserved	SETTING1		Reserved	Reserved	SETTING0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

CCM_PLL_CTRL*n* field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 -	This field is reserved. Reserved
14 -	This field is reserved. Reserved
13–12 SETTING3	Clock gate control setting for domain 3. This field can only be written by domain 3 00 Domain clocks not needed

Table continues on the next page...

CCM_PLL_CTRL*n* field descriptions (continued)

Field	Description
	01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
11 -	This field is reserved. Reserved
10 -	This field is reserved. Reserved
9–8 SETTING2	Clock gate control setting for domain 2. This field can only be written by domain 2 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
7 -	This field is reserved. Reserved
6 -	This field is reserved. Reserved
5–4 SETTING1	Clock gate control setting for domain 1. This field can only be written by domain 1. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
3 -	This field is reserved. Reserved
2 -	This field is reserved. Reserved
SETTING0	Clock gate control setting for domain 0. This field can only be written by domain 0. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time

5.1.7.3 CCM PLL Control Register (CCM_PLL_CTRL*n*_SET)

See Input Clocks for PLL control mapping.

NOTE

For the SoC to correctly power up after entering DSM, CCM_PLL_CTRLx must not be set to 0x0 or 0x3 for any domain in use.

Address: 3038_0000h base + 804h offset + (16d × i), where i=0d to 38d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	SETTING3		Reserved	Reserved	SETTING2		Reserved	Reserved	SETTING1		Reserved	Reserved	SETTING0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

CCM_PLL_CTRL*n*_SET field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 -	This field is reserved. Reserved
14 -	This field is reserved. Reserved
13–12 SETTING3	Clock gate control setting for domain 3. This field can only be written by domain 3 00 Domain clocks not needed

Table continues on the next page...

CCM_PLL_CTRL*n*_SET field descriptions (continued)

Field	Description
	01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
11 -	This field is reserved. Reserved
10 -	This field is reserved. Reserved
9–8 SETTING2	Clock gate control setting for domain 2. This field can only be written by domain 2 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
7 -	This field is reserved. Reserved
6 -	This field is reserved. Reserved
5–4 SETTING1	Clock gate control setting for domain 1. This field can only be written by domain 1. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
3 -	This field is reserved. Reserved
2 -	This field is reserved. Reserved
SETTING0	Clock gate control setting for domain 0. This field can only be written by domain 0. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time

5.1.7.4 CCM PLL Control Register (CCM_PLL_CTRL*n*_CLR)

See Input Clocks for PLL control mapping.

NOTE

For the SoC to correctly power up after entering DSM, CCM_PLL_CTRLx must not be set to 0x0 or 0x3 for any domain in use.

Address: 3038_0000h base + 808h offset + (16d × i), where i=0d to 38d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	SETTING3		Reserved	Reserved	SETTING2		Reserved	Reserved	SETTING1		Reserved	Reserved	SETTING0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

CCM_PLL_CTRL*n*_CLR field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 -	This field is reserved. Reserved
14 -	This field is reserved. Reserved
13–12 SETTING3	Clock gate control setting for domain 3. This field can only be written by domain 3 00 Domain clocks not needed

Table continues on the next page...

CCM_PLL_CTRL*n*_CLR field descriptions (continued)

Field	Description
	01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
11 -	This field is reserved. Reserved
10 -	This field is reserved. Reserved
9–8 SETTING2	Clock gate control setting for domain 2. This field can only be written by domain 2 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
7 -	This field is reserved. Reserved
6 -	This field is reserved. Reserved
5–4 SETTING1	Clock gate control setting for domain 1. This field can only be written by domain 1. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
3 -	This field is reserved. Reserved
2 -	This field is reserved. Reserved
SETTING0	Clock gate control setting for domain 0. This field can only be written by domain 0. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time

5.1.7.5 CCM PLL Control Register (CCM_PLL_CTRL*n*_TOG)

See Input Clocks for PLL control mapping.

NOTE

For the SoC to correctly power up after entering DSM, CCM_PLL_CTRLx must not be set to 0x0 or 0x3 for any domain in use.

Address: 3038_0000h base + 80Ch offset + (16d × i), where i=0d to 38d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	SETTING3		Reserved	Reserved	SETTING2		Reserved	Reserved	SETTING1		Reserved	Reserved	SETTING0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

CCM_PLL_CTRL*n*_TOG field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 -	This field is reserved. Reserved
14 -	This field is reserved. Reserved
13–12 SETTING3	Clock gate control setting for domain 3. This field can only be written by domain 3 00 Domain clocks not needed

Table continues on the next page...

CCM_PLL_CTRL*n*_TOG field descriptions (continued)

Field	Description
	01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
11 -	This field is reserved. Reserved
10 -	This field is reserved. Reserved
9–8 SETTING2	Clock gate control setting for domain 2. This field can only be written by domain 2 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
7 -	This field is reserved. Reserved
6 -	This field is reserved. Reserved
5–4 SETTING1	Clock gate control setting for domain 1. This field can only be written by domain 1. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
3 -	This field is reserved. Reserved
2 -	This field is reserved. Reserved
SETTING0	Clock gate control setting for domain 0. This field can only be written by domain 0. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time

5.1.7.6 CCM Clock Gating Register (CCM_CCGR*n*)

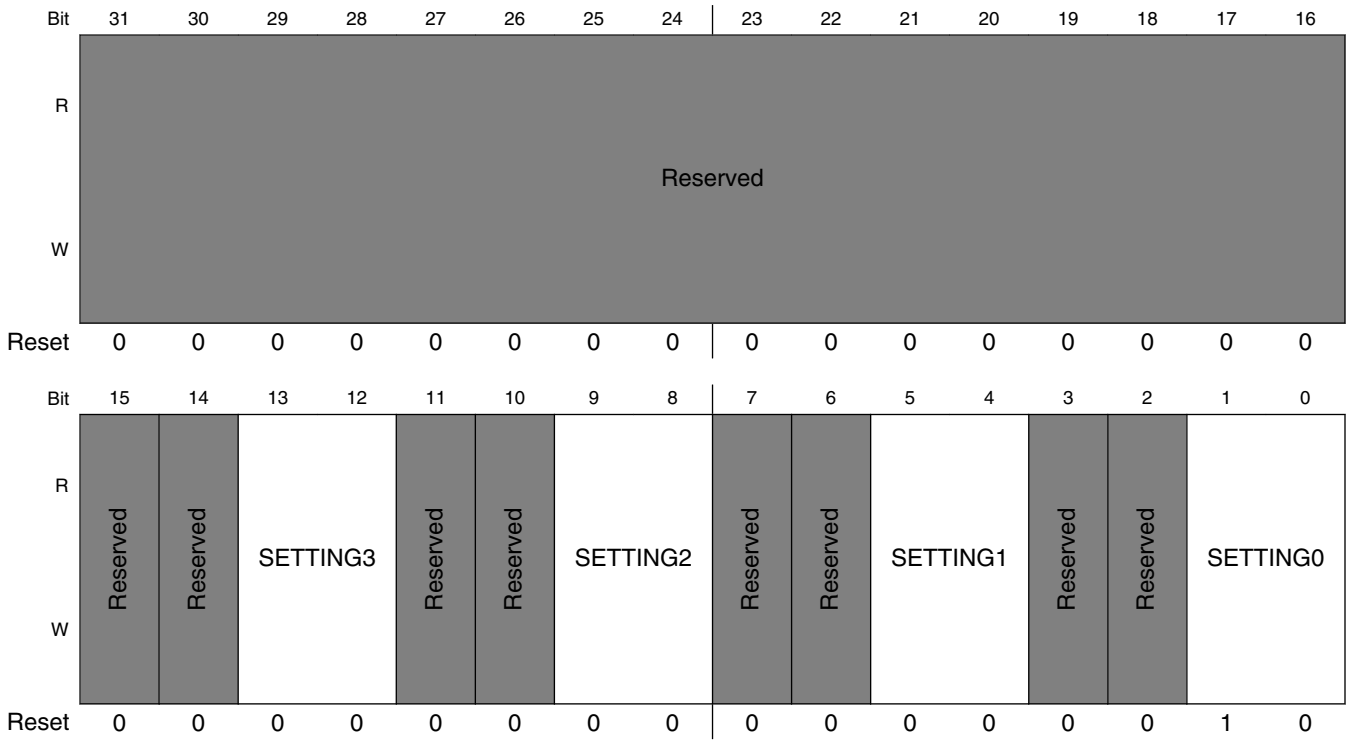
NOTE

Not all CCGRs are mapped. See CCGR Interface for CCGR mapping and clock gating information.

NOTE

Sec_debug clock gating (CCGR60) must be active in low power mode. DO NOT gate this clock in low power mode to guarantee the low power mode functions such as stop WDOG counting.

Address: 3038_0000h base + 4000h offset + (16d × i), where i=0d to 190d



CCM_CCGR*n* field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 -	This field is reserved. Reserved
14 -	This field is reserved. Reserved
13–12 SETTING3	Clock gate control setting for domain 3. This field can only be written by domain 3 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
11 -	This field is reserved. Reserved

Table continues on the next page...

CCM_CCGR*n* field descriptions (continued)

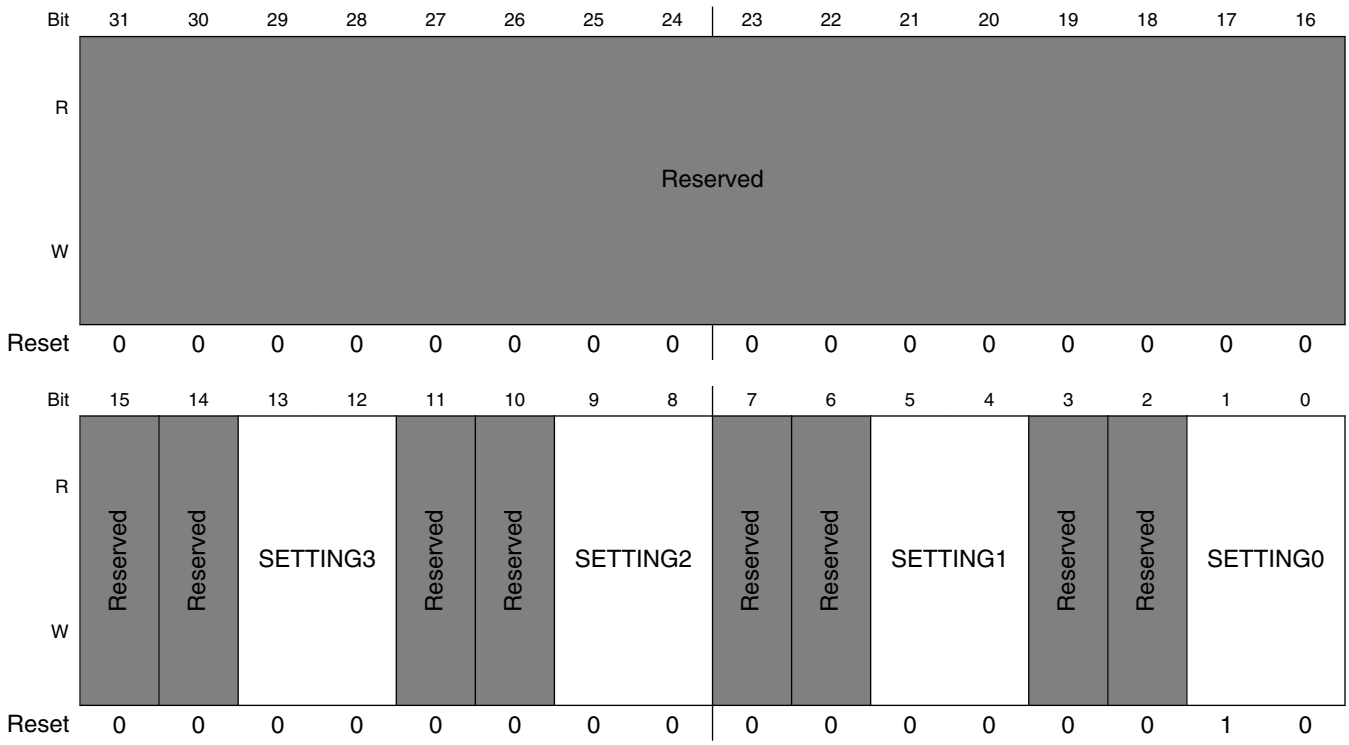
Field	Description
10 -	This field is reserved. Reserved
9–8 SETTING2	Clock gate control setting for domain 2. This field can only be written by domain 2 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
7 -	This field is reserved. Reserved
6 -	This field is reserved. Reserved
5–4 SETTING1	Clock gate control setting for domain 1. This field can only be written by domain 1. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
3 -	This field is reserved. Reserved
2 -	This field is reserved. Reserved
SETTING0	Clock gate control setting for domain 0. This field can only be written by domain 0. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time

5.1.7.7 CCM Clock Gating Register (CCM_CCGR*n*_SET)

NOTE

Not all CCGRs are mapped. See CCGR Interface for CCGR mapping and clock gating information.

Address: 3038_0000h base + 4004h offset + (16d × i), where i=0d to 190d



CCM_CCGR*n*_SET field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 -	This field is reserved. Reserved
14 -	This field is reserved. Reserved
13–12 SETTING3	Clock gate control setting for domain 3. This field can only be written by domain 3 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time

Table continues on the next page...

CCM_CCGR*n*_SET field descriptions (continued)

Field	Description
11 -	This field is reserved. Reserved
10 -	This field is reserved. Reserved
9–8 SETTING2	Clock gate control setting for domain 2. This field can only be written by domain 2 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
7 -	This field is reserved. Reserved
6 -	This field is reserved. Reserved
5–4 SETTING1	Clock gate control setting for domain 1. This field can only be written by domain 1. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
3 -	This field is reserved. Reserved
2 -	This field is reserved. Reserved
SETTING0	Clock gate control setting for domain 0. This field can only be written by domain 0. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time

5.1.7.8 CCM Clock Gating Register (CCM_CCGR*n*_CLR)

NOTE

Not all CCGRs are mapped. See CCGR Interface for CCGR mapping and clock gating information.

Address: 3038_0000h base + 4008h offset + (16d × i), where i=0d to 190d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	SETTING3		Reserved	Reserved	SETTING2		Reserved	Reserved	SETTING1		Reserved	Reserved	SETTING0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

CCM_CCGR*n*_CLR field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 -	This field is reserved. Reserved
14 -	This field is reserved. Reserved
13–12 SETTING3	Clock gate control setting for domain 3. This field can only be written by domain 3 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time

Table continues on the next page...

CCM_CCGR*n*_CLR field descriptions (continued)

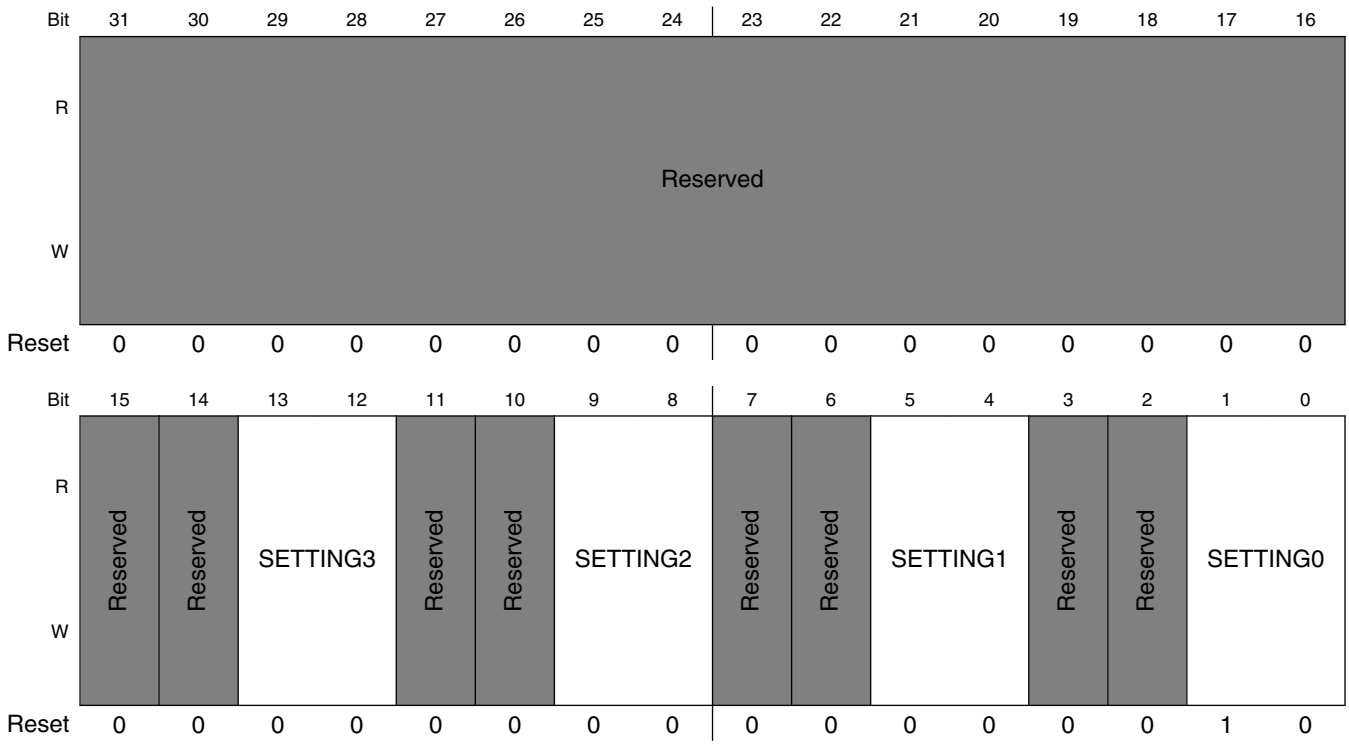
Field	Description
11 -	This field is reserved. Reserved
10 -	This field is reserved. Reserved
9–8 SETTING2	Clock gate control setting for domain 2. This field can only be written by domain 2 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
7 -	This field is reserved. Reserved
6 -	This field is reserved. Reserved
5–4 SETTING1	Clock gate control setting for domain 1. This field can only be written by domain 1. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
3 -	This field is reserved. Reserved
2 -	This field is reserved. Reserved
SETTING0	Clock gate control setting for domain 0. This field can only be written by domain 0. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time

5.1.7.9 CCM Clock Gating Register (CCM_CCGR*n*_TOG)

NOTE

Not all CCGRs are mapped. See CCGR Interface for CCGR mapping and clock gating information.

Address: 3038_0000h base + 400Ch offset + (16d × i), where i=0d to 190d



CCM_CCGR*n*_TOG field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 -	This field is reserved. Reserved
14 -	This field is reserved. Reserved
13–12 SETTING3	Clock gate control setting for domain 3. This field can only be written by domain 3 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time

Table continues on the next page...

CCM_CCGR*n*_TOG field descriptions (continued)

Field	Description
11 -	This field is reserved. Reserved
10 -	This field is reserved. Reserved
9–8 SETTING2	Clock gate control setting for domain 2. This field can only be written by domain 2 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
7 -	This field is reserved. Reserved
6 -	This field is reserved. Reserved
5–4 SETTING1	Clock gate control setting for domain 1. This field can only be written by domain 1. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time
3 -	This field is reserved. Reserved
2 -	This field is reserved. Reserved
SETTING0	Clock gate control setting for domain 0. This field can only be written by domain 0. 00 Domain clocks not needed 01 Domain clocks needed when in RUN 10 Domain clocks needed when in RUN and WAIT 11 Domain clocks needed all the time

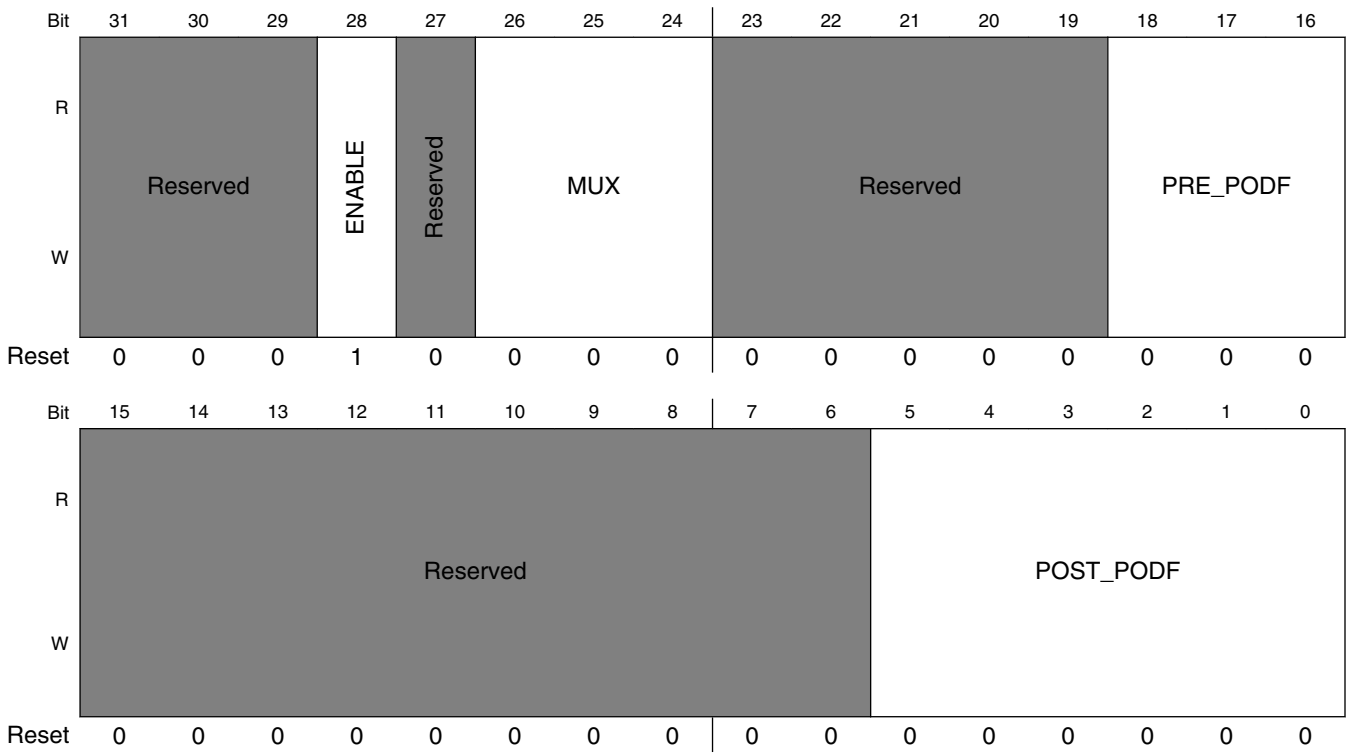
5.1.7.10 Target Register (CCM_TARGET_ROOT*n*)

See Target Interface for more information.

NOTE

See Clock Root Selects for clock root offsets and muxing information.

Address: 3038_0000h base + 8000h offset + (128d × i), where i=0d to 124d



CCM_TARGET_ROOT*n* field descriptions

Field	Description
31–29 -	This field is reserved. Reserved
28 ENABLE	Enable this clock 0 clock root is OFF 1 clock root is ON
27 -	This field is reserved. Reserved
26–24 MUX	Selection of clock sources This field is 1 bit long for DRAM and CORE

Table continues on the next page...

CCM_TARGET_ROOT*n* field descriptions (continued)

Field	Description
23–19 -	This field is reserved. Reserved
18–16 PRE_PODF	Pre divider divide the number Divider value is n+1 This field does not apply for CORE, DRAM, DRAM_PHYM 000 Divide by 1 001 Divide by 2 010 Divide by 3 011 Divide by 4 100 Divide by 5 101 Divide by 6 110 Divide by 7 111 Divide by 8
15–6 -	This field is reserved. Reserved
POST_PODF	Post divider divide number Divider value is n + 1. For CORE, this field is 3 bit long. For IPG, this field is 1 bit long. This field does not apply to DRAM_PHYM 000000 Divide by 1 000001 Divide by 2 000010 Divide by 3 000011 Divide by 4 000100 Divide by 5 000101 Divide by 6 : 111111 Divide by 64

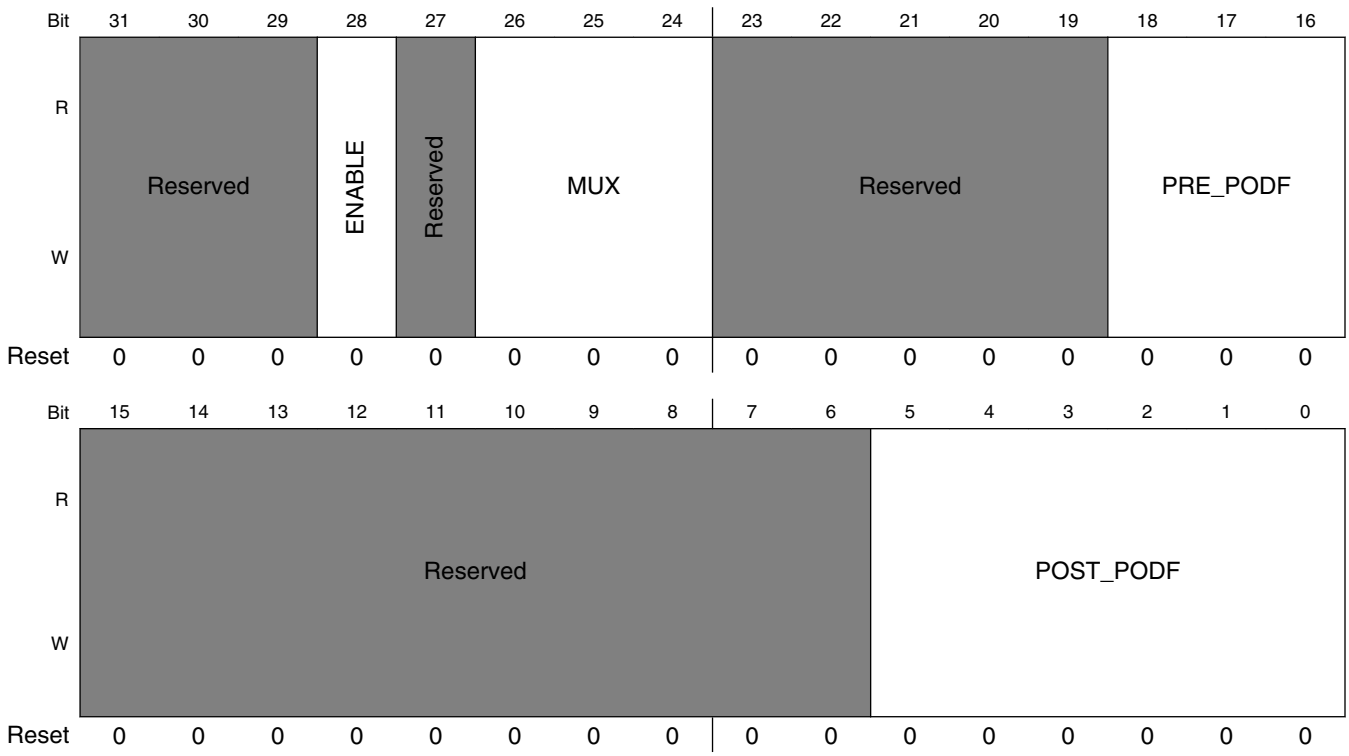
5.1.7.11 Target Register (CCM_TARGET_ROOT*n*_SET)

See Target Interface for more information.

NOTE

See Clock Root Selects for clock root offsets and muxing information.

Address: 3038_0000h base + 8004h offset + (128d × i), where i=0d to 124d



CCM_TARGET_ROOT*n*_SET field descriptions

Field	Description
31–29 -	This field is reserved. Reserved
28 ENABLE	Enable this clock 0 clock root is OFF 1 clock root is ON
27 -	This field is reserved. Reserved
26–24 MUX	Selection of clock sources This field is 1 bit long for DRAM and CORE

Table continues on the next page...

CCM_TARGET_ROOT*n*_SET field descriptions (continued)

Field	Description
23–19 -	This field is reserved. Reserved
18–16 PRE_PODF	Pre divider divide the number Divider value is n+1 This field does not apply for CORE, DRAM, DRAM_PHYM 000 Divide by 1 001 Divide by 2 010 Divide by 3 011 Divide by 4 100 Divide by 5 101 Divide by 6 110 Divide by 7 111 Divide by 8
15–6 -	This field is reserved. Reserved
POST_PODF	Post divider divide the number Divider value is n + 1. For CORE, this field is 3 bit long. For IPG, this field is 1 bit long. This field does not apply to DRAM_PHYM 000000 Divide by 1 000001 Divide by 2 000010 Divide by 3 000011 Divide by 4 000100 Divide by 5 000101 Divide by 6 : 111111 Divide by 64

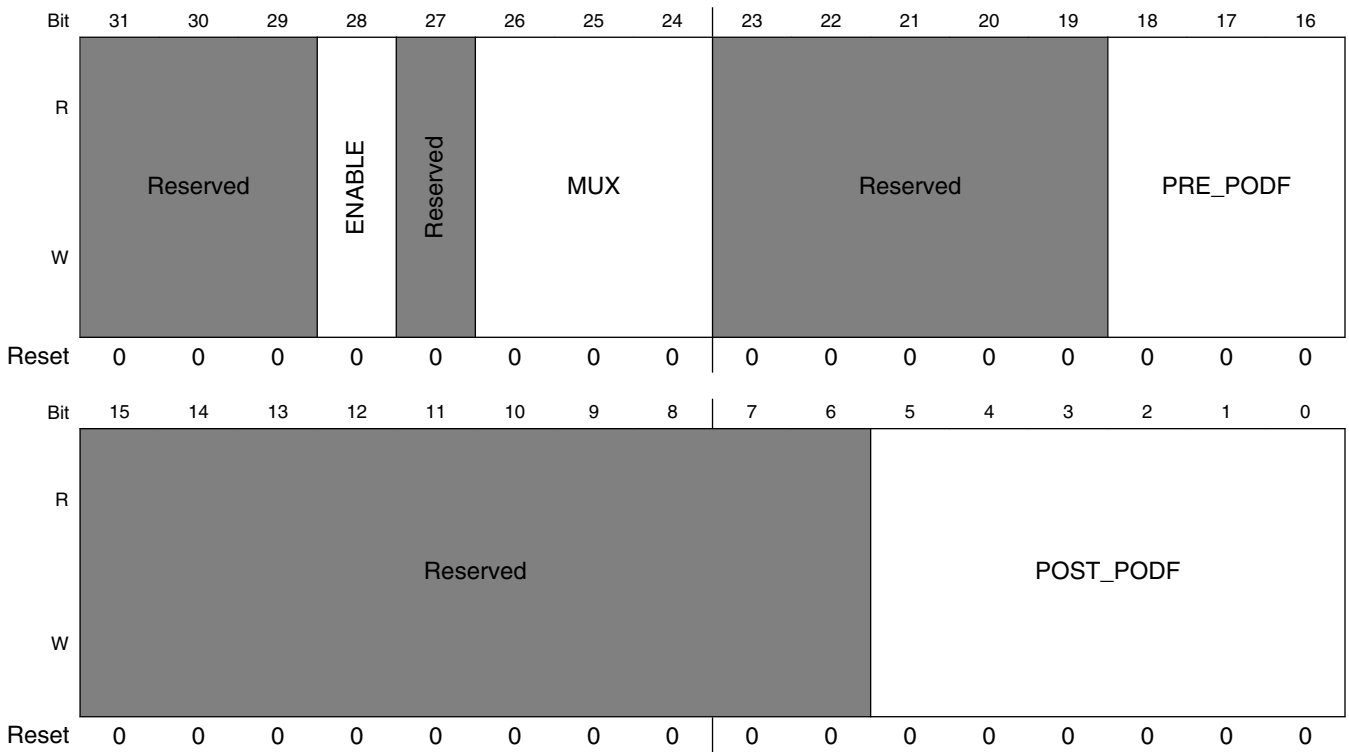
5.1.7.12 Target Register (CCM_TARGET_ROOT*n*_CLR)

See Target Interface for more information.

NOTE

See Clock Root Selects for clock root offsets and muxing information.

Address: 3038_0000h base + 8008h offset + (128d × i), where i=0d to 124d



CCM_TARGET_ROOT*n*_CLR field descriptions

Field	Description
31–29 -	This field is reserved. Reserved
28 ENABLE	Enable this clock 0 clock root is OFF 1 clock root is ON
27 -	This field is reserved. Reserved
26–24 MUX	Selection of clock sources This field is 1 bit long for DRAM and CORE

Table continues on the next page...

CCM_TARGET_ROOT*n*_CLR field descriptions (continued)

Field	Description
23–19 -	This field is reserved. Reserved
18–16 PRE_PODF	Pre divider divide the number Divider value is n+1 This field does not apply for CORE, DRAM, DRAM_PHYM 000 Divide by 1 001 Divide by 2 010 Divide by 3 011 Divide by 4 100 Divide by 5 101 Divide by 6 110 Divide by 7 111 Divide by 8
15–6 -	This field is reserved. Reserved
POST_PODF	Post divider divide the number Divider value is n + 1. For CORE, this field is 3 bit long. For IPG, this field is 1 bit long. This field does not apply to DRAM_PHYM 000000 Divide by 1 000001 Divide by 2 000010 Divide by 3 000011 Divide by 4 000100 Divide by 5 000101 Divide by 6 : 111111 Divide by 64

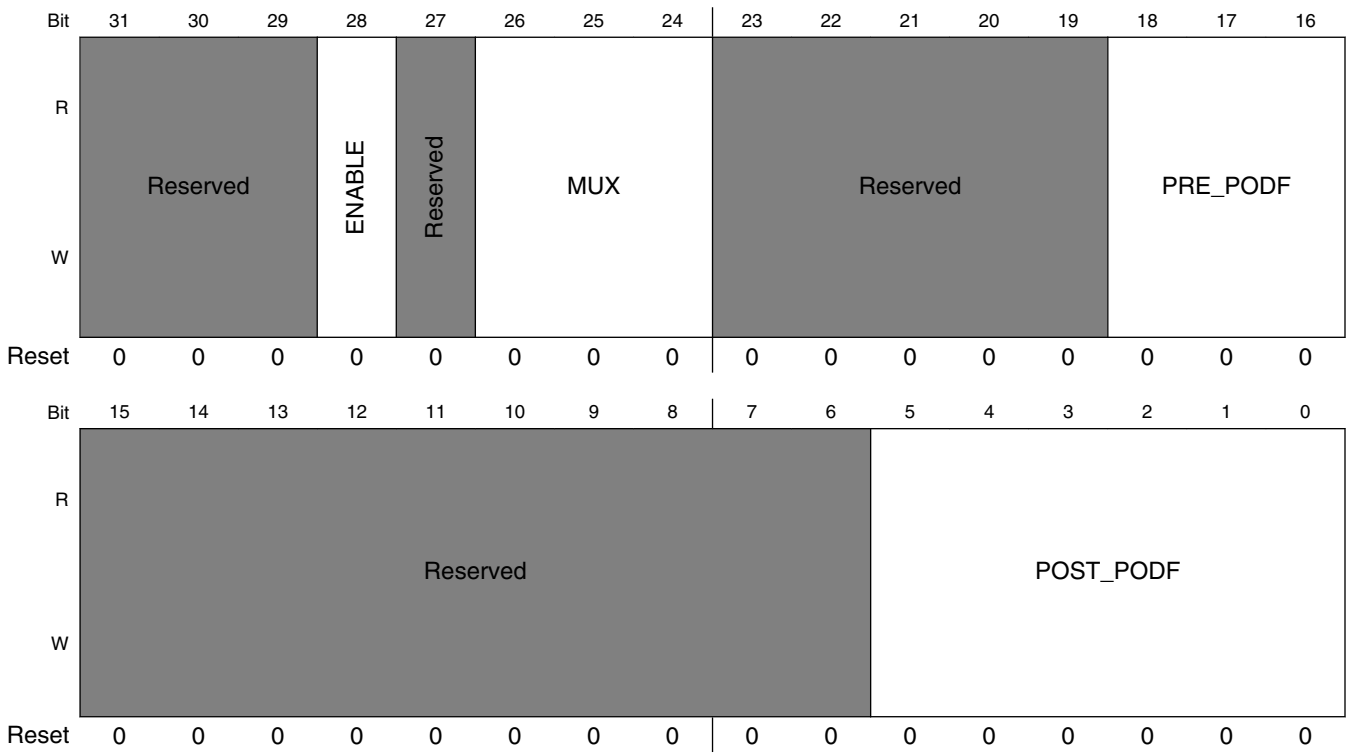
5.1.7.13 Target Register (CCM_TARGET_ROOT*n*_TOG)

See Target Interface for more information.

NOTE

See Clock Root Selects for clock root offsets and muxing information.

Address: 3038_0000h base + 800Ch offset + (128d × i), where i=0d to 124d



CCM_TARGET_ROOT*n*_TOG field descriptions

Field	Description
31–29 -	This field is reserved. Reserved
28 ENABLE	Enable this clock 0 clock root is OFF 1 clock root is ON
27 -	This field is reserved. Reserved
26–24 MUX	Selection of clock sources This field is 1 bit long for DRAM and CORE

Table continues on the next page...

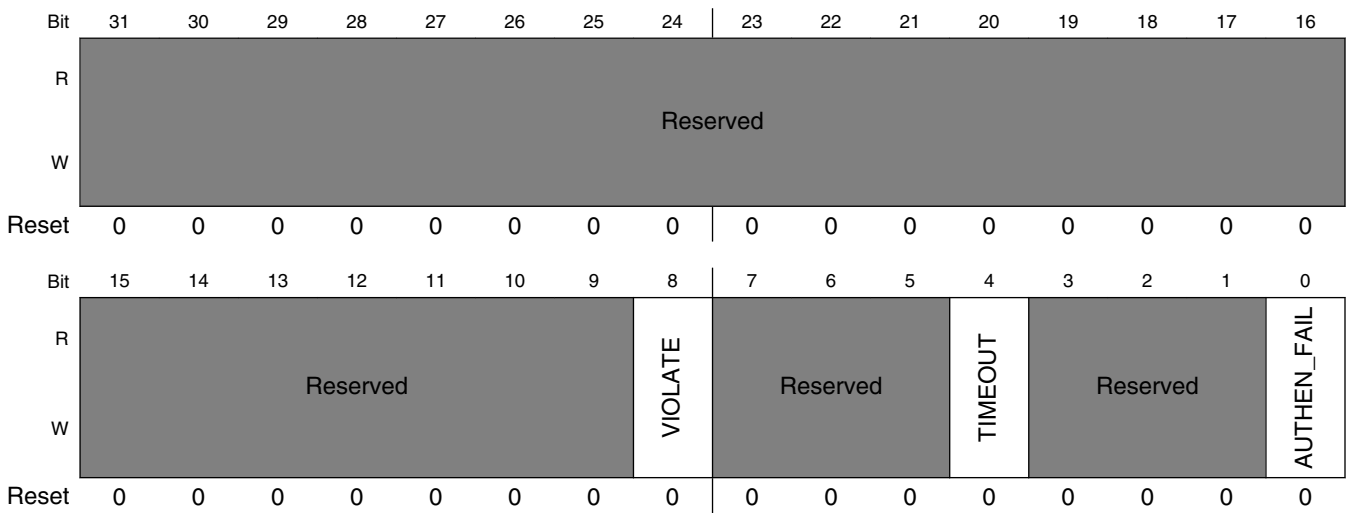
CCM_TARGET_ROOT*n*_TOG field descriptions (continued)

Field	Description
23–19 -	This field is reserved. Reserved
18–16 PRE_PODF	Pre divide divide number Divider value is n+1 This field does not apply for CORE, DRAM, DRAM_PHYM 000 Divide by 1 001 Divide by 2 010 Divide by 3 011 Divide by 4 100 Divide by 5 101 Divide by 6 110 Divide by 7 111 Divide by 8
15–6 -	This field is reserved. Reserved
POST_PODF	Post divider divide number Divider value is n + 1. For CORE, this field is 3 bit long. For IPG, this field is 1 bit long. This field does not apply to DRAM_PHYM 000000 Divide by 1 000001 Divide by 2 000010 Divide by 3 000011 Divide by 4 000100 Divide by 5 000101 Divide by 6 : 111111 Divide by 64

5.1.7.14 Miscellaneous Register (CCM_MISC*n*)

MISC

Address: 3038_0000h base + 8010h offset + (128d × i), where i=0d to 124d



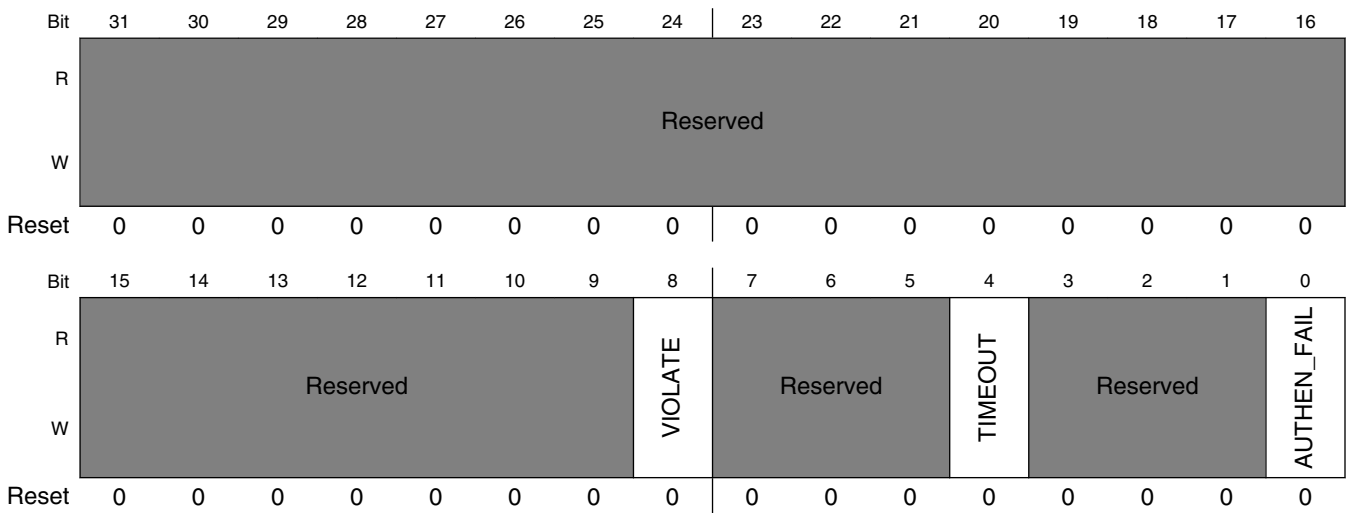
CCM_MISC*n* field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 VIOLATE	This sticky bit reflects access violation in normal interface of this clock. This bit has internal 4 bits, one for each domain. Violation from other domain is not visible or clearable. This file is cleared to 0 while write 1.
7–5 -	This field is reserved. Reserved
4 TIMEOUT	This sticky bit reflects time out happened during accessing this clock. This bit has internal 4 bits, one for each domain. Timeout from other domain is not visible or clearable. This file is cleared to 0 while write 1.
3–1 -	This field is reserved. Reserved
0 AUTHEN_FAIL	This sticky bit reflects access restricted by access control of this clock. This bit has internal 4 bits, one for each domain. Authentic fail from other domain is not visible or clearable. This file is cleared to 0 while write 1

5.1.7.15 Miscellaneous Register (CCM_MISC_ROOT*n*_SET)

Misc

Address: 3038_0000h base + 8014h offset + (128d × i), where i=0d to 124d



CCM_MISC_ROOT*n*_SET field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 VIOLATE	This sticky bit reflects access violation in normal interface of this clock. This bit has internal 4 bits, one for each domain. Violation from other domain is not visible or clearable. This file is cleared to 0 while write 1.
7–5 -	This field is reserved. Reserved
4 TIMEOUT	This sticky bit reflects time out happened during accessing this clock. This bit has internal 4 bits, one for each domain. Timeout from other domain is not visible or clearable. This file is cleared to 0 while write 1.
3–1 -	This field is reserved. Reserved
0 AUTHEN_FAIL	This sticky bit reflects access restricted by access control of this clock. This bit has internal 4 bits, one for each domain. Authentic fail from other domain is not visible or clearable. This file is cleared to 0 while write 1

5.1.7.16 Miscellaneous Register (CCM_MISC_ROOT*n*_CLR)

MISC

Address: 3038_0000h base + 8018h offset + (128d × i), where i=0d to 124d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							VIOLATE	Reserved			TIMEOUT	Reserved			AUTHEN_FAIL
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

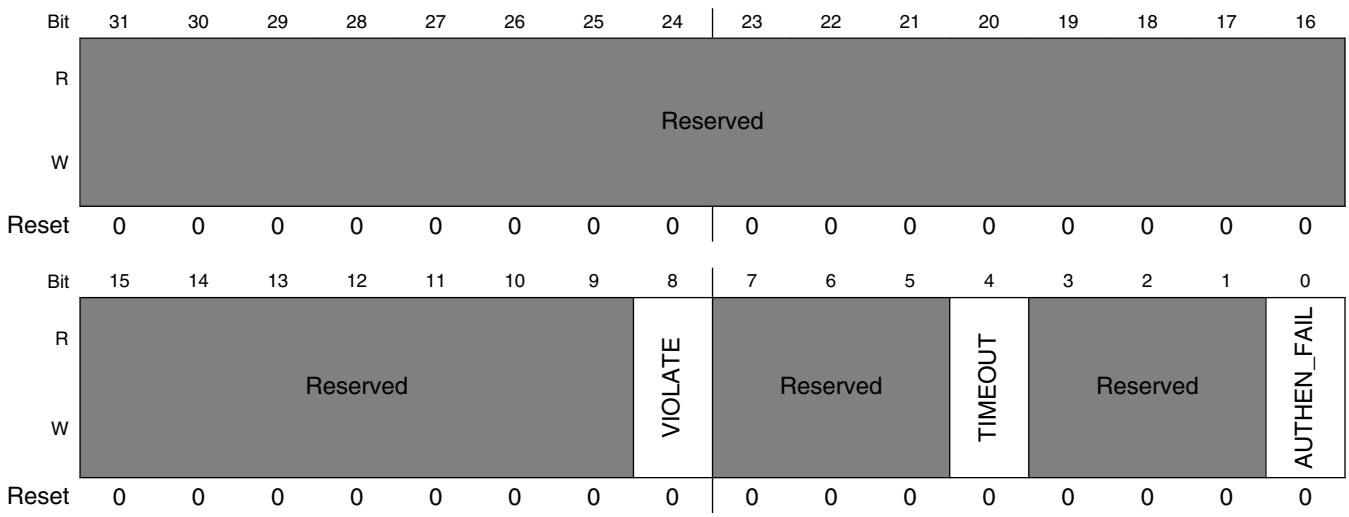
CCM_MISC_ROOT*n*_CLR field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 VIOLATE	This sticky bit reflects access violation in normal interface of this clock. This bit has internal 4 bits, one for each domain. Violation from other domain is not visible or clearable. This file is cleared to 0 while write 1.
7–5 -	This field is reserved. Reserved
4 TIMEOUT	This sticky bit reflects time out happened during accessing this clock. This bit has internal 4 bits, one for each domain. Timeout from other domain is not visible or clearable. This file is cleared to 0 while write 1.
3–1 -	This field is reserved. Reserved
0 AUTHEN_FAIL	This sticky bit reflects access restricted by access control of this clock. This bit has internal 4 bits, one for each domain. Authentic fail from other domain is not visible or clearable. This file is cleared to 0 while write 1

5.1.7.17 Miscellaneous Register (CCM_MISC_ROOT*n*_TOG)

MISC

Address: 3038_0000h base + 801Ch offset + (128d × i), where i=0d to 124d



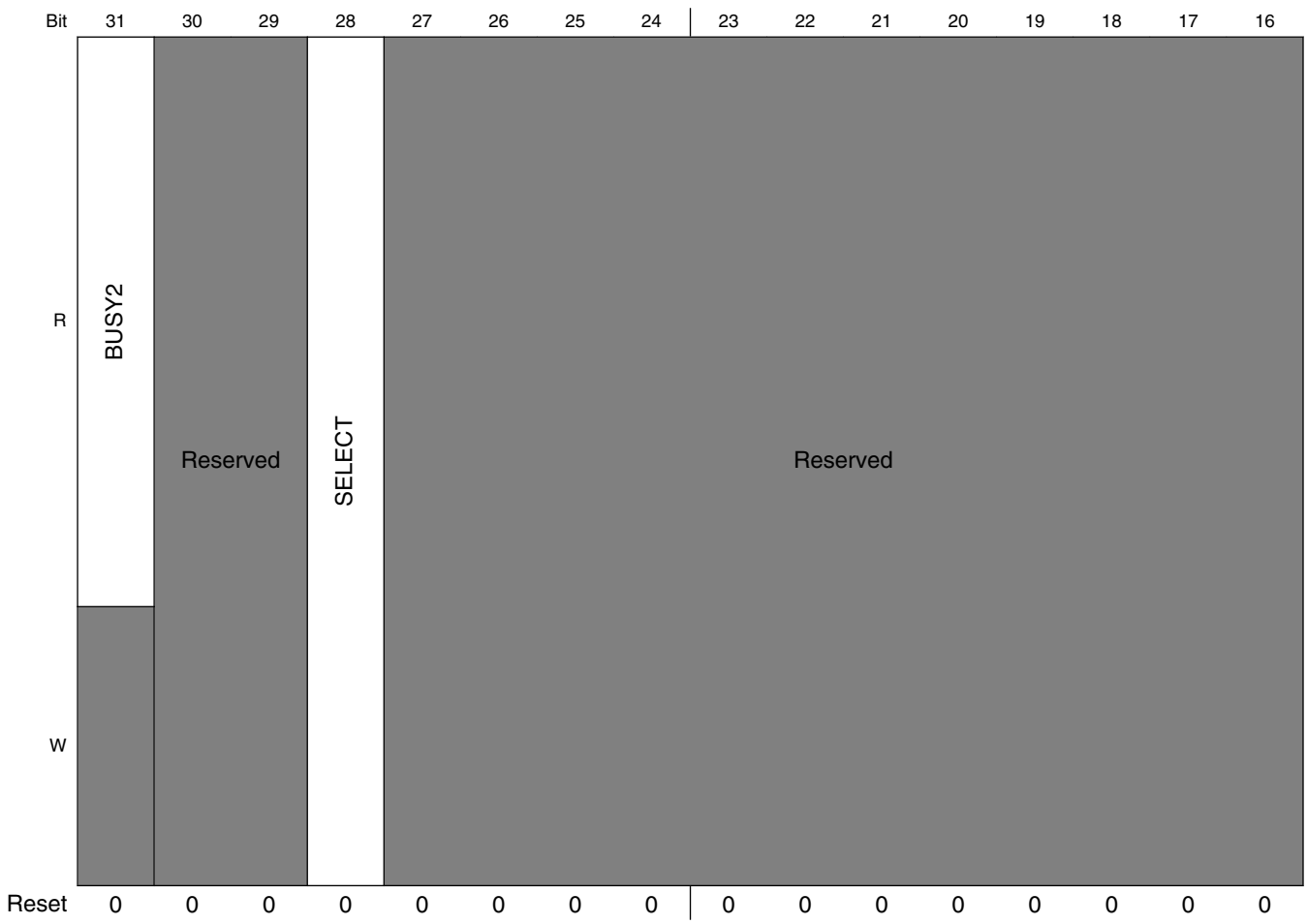
CCM_MISC_ROOT*n*_TOG field descriptions

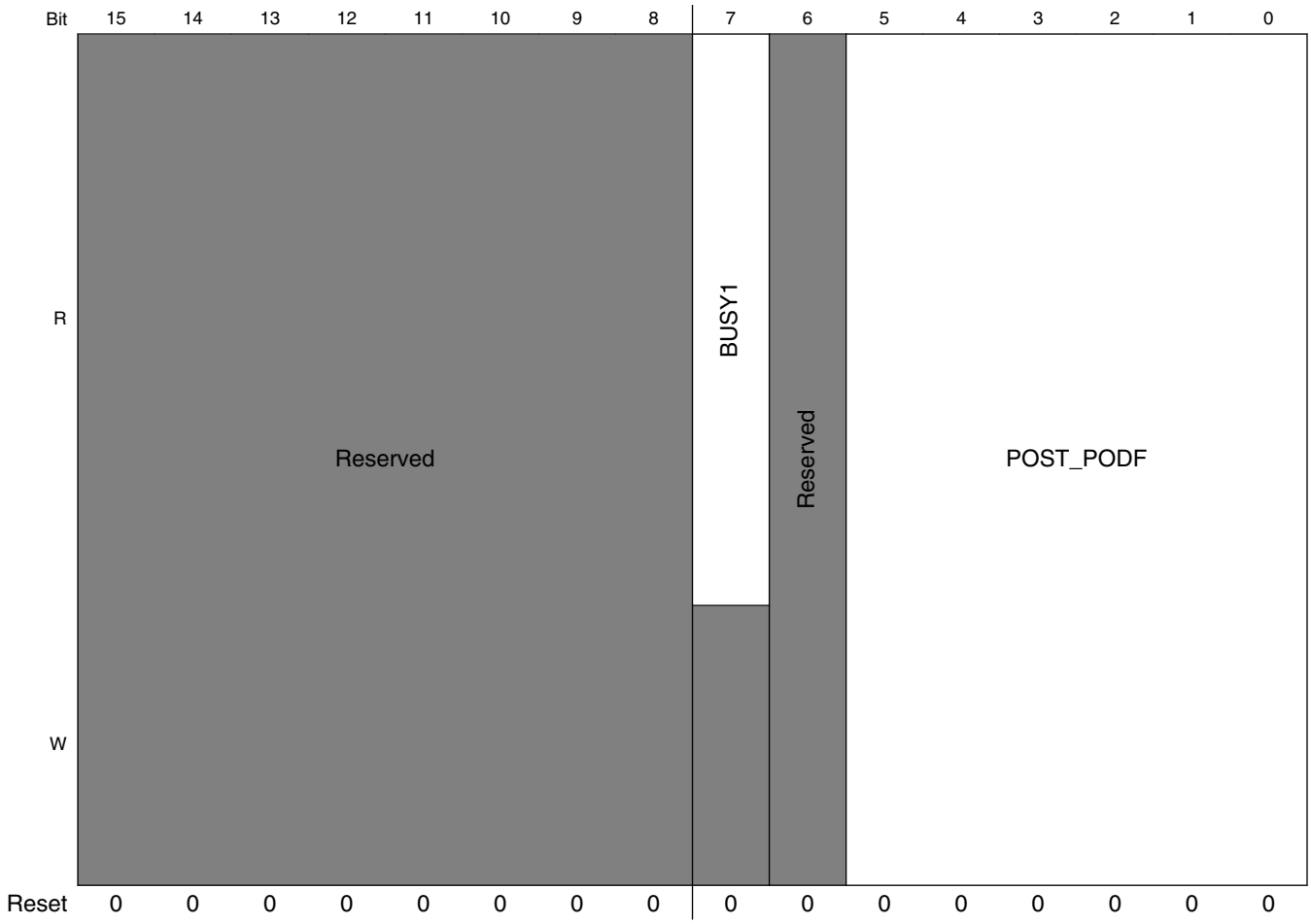
Field	Description
31–9 -	This field is reserved. Reserved
8 VIOLATE	This sticky bit reflects access violation in normal interface of this clock. This bit has internal 4 bits, one for each domain. Violation from other domain is not visible or clearable. This file is cleared to 0 while write 1.
7–5 -	This field is reserved. Reserved
4 TIMEOUT	This sticky bit reflects time out happened during accessing this clock. This bit has internal 4 bits, one for each domain. Timeout from other domain is not visible or clearable. This file is cleared to 0 while write 1.
3–1 -	This field is reserved. Reserved
0 AUTHEN_FAIL	This sticky bit reflects access restricted by access control of this clock. This bit has internal 4 bits, one for each domain. Authentic fail from other domain is not visible or clearable. This file is cleared to 0 while write 1

5.1.7.18 Post Divider Register (CCM_POST*n*)

Post Register

Address: 3038_0000h base + 8020h offset + (128d × i), where i=0d to 124d





CCM_POST*n* field descriptions

Field	Description
31 BUSY2	Clock switching multiplexer is applying new setting
30–29 -	This field is reserved. Reserved
28 SELECT	Selection of pre clock branches This field is not applied in IP 0 select branch A 1 select branch B
27–8 -	This field is reserved. Reserved
7 BUSY1	Post divider is applying new set value
6 -	This field is reserved. Reserved
POST_PODF	Post divider divide the number Divider value is n + 1

Table continues on the next page...

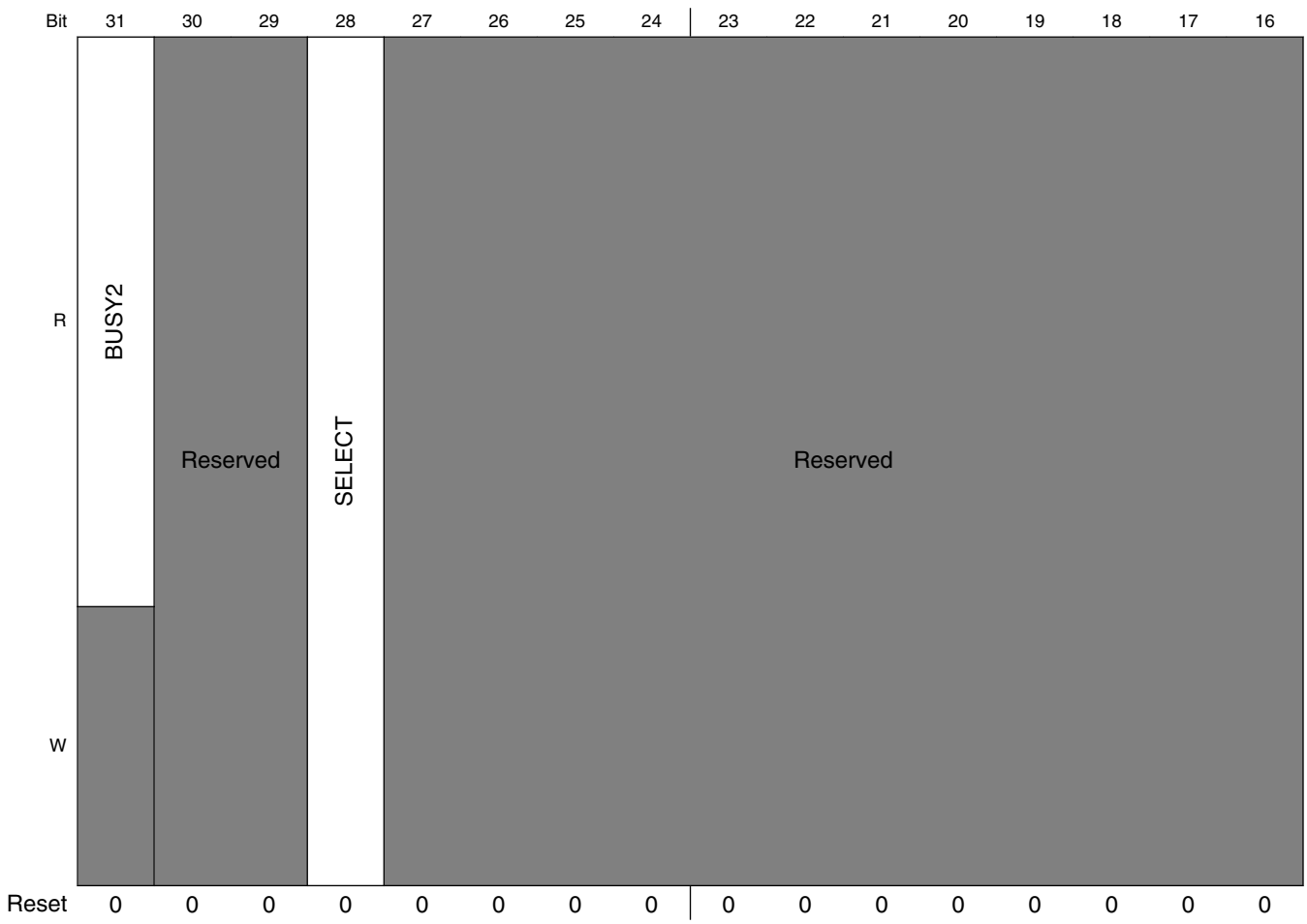
CCM_POST*n* field descriptions (continued)

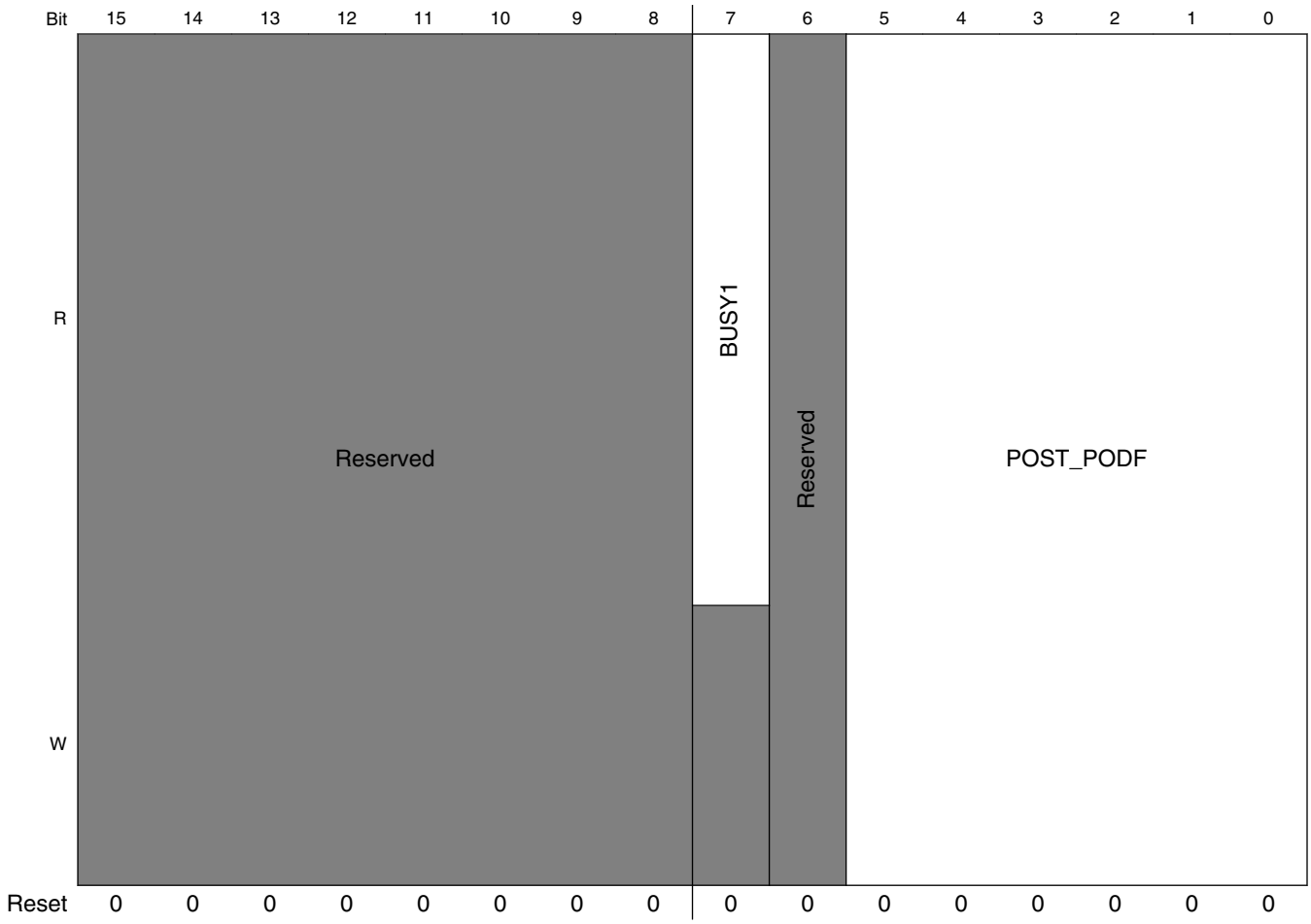
Field	Description
	For CORE, this field is 3 bit long. For IPG, this field is 2 bit long. This field does not apply to DRAM_PHYM 000000 Divide by 1 000001 Divide by 2 000010 Divide by 3 000011 Divide by 4 000100 Divide by 5 000101 Divide by 6 : 111111 Divide by 64

5.1.7.19 Post Divider Register (CCM_POST_ROOT*n*_SET)

Post Divider Register

Address: 3038_0000h base + 8024h offset + (128d × i), where i=0d to 124d





CCM_POST_ROOT*n*_SET field descriptions

Field	Description
31 BUSY2	Clock switching multiplexer is applying new setting
30–29 -	This field is reserved. Reserved
28 SELECT	Selection of pre clock branches This field is not applied in IP 0 select branch A 1 select branch B
27–8 -	This field is reserved. Reserved
7 BUSY1	Post divider is applying new set value
6 -	This field is reserved. Reserved
POST_PODF	Post divider divide number Divider value is n + 1

Table continues on the next page...

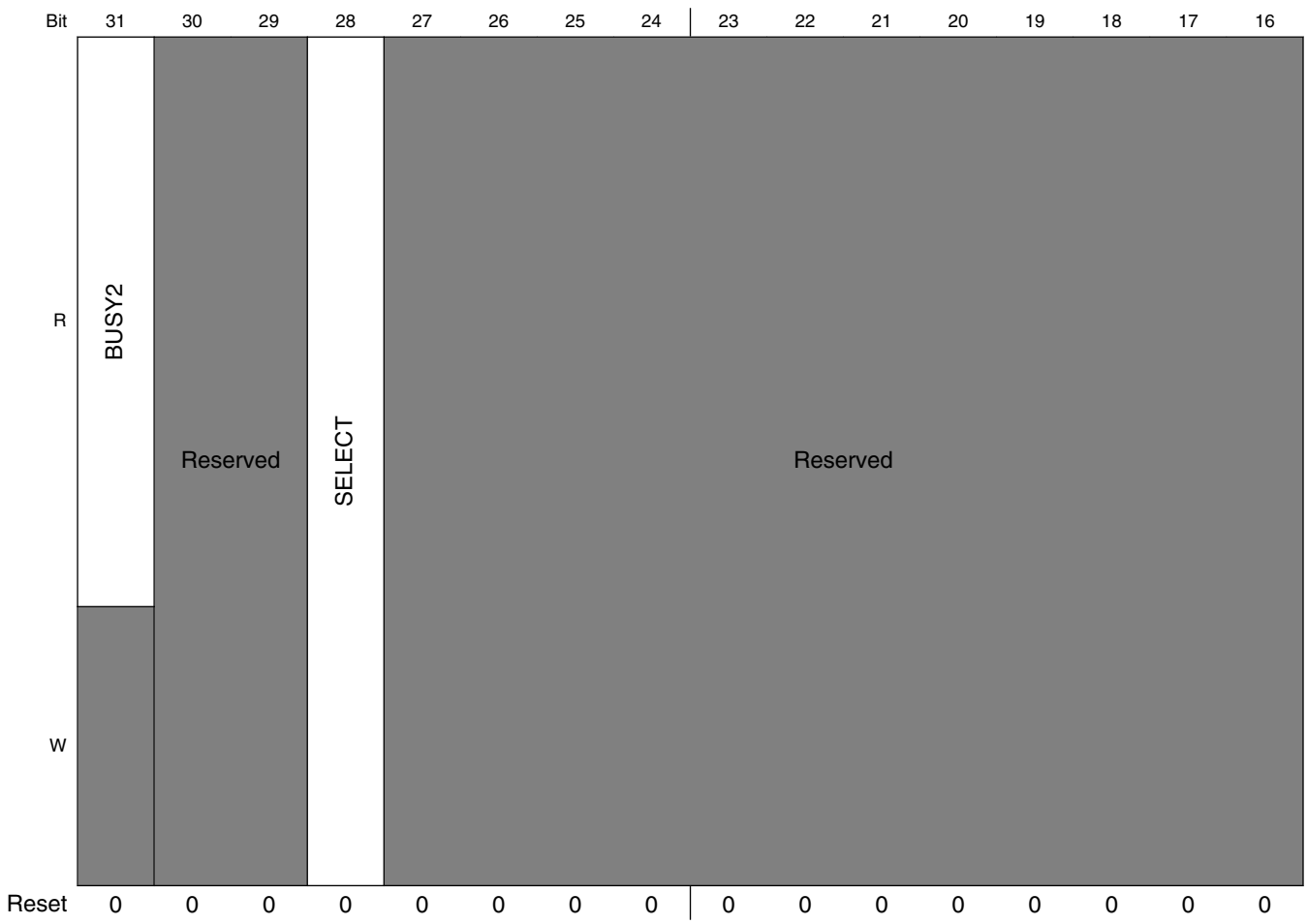
CCM_POST_ROOT*n*_SET field descriptions (continued)

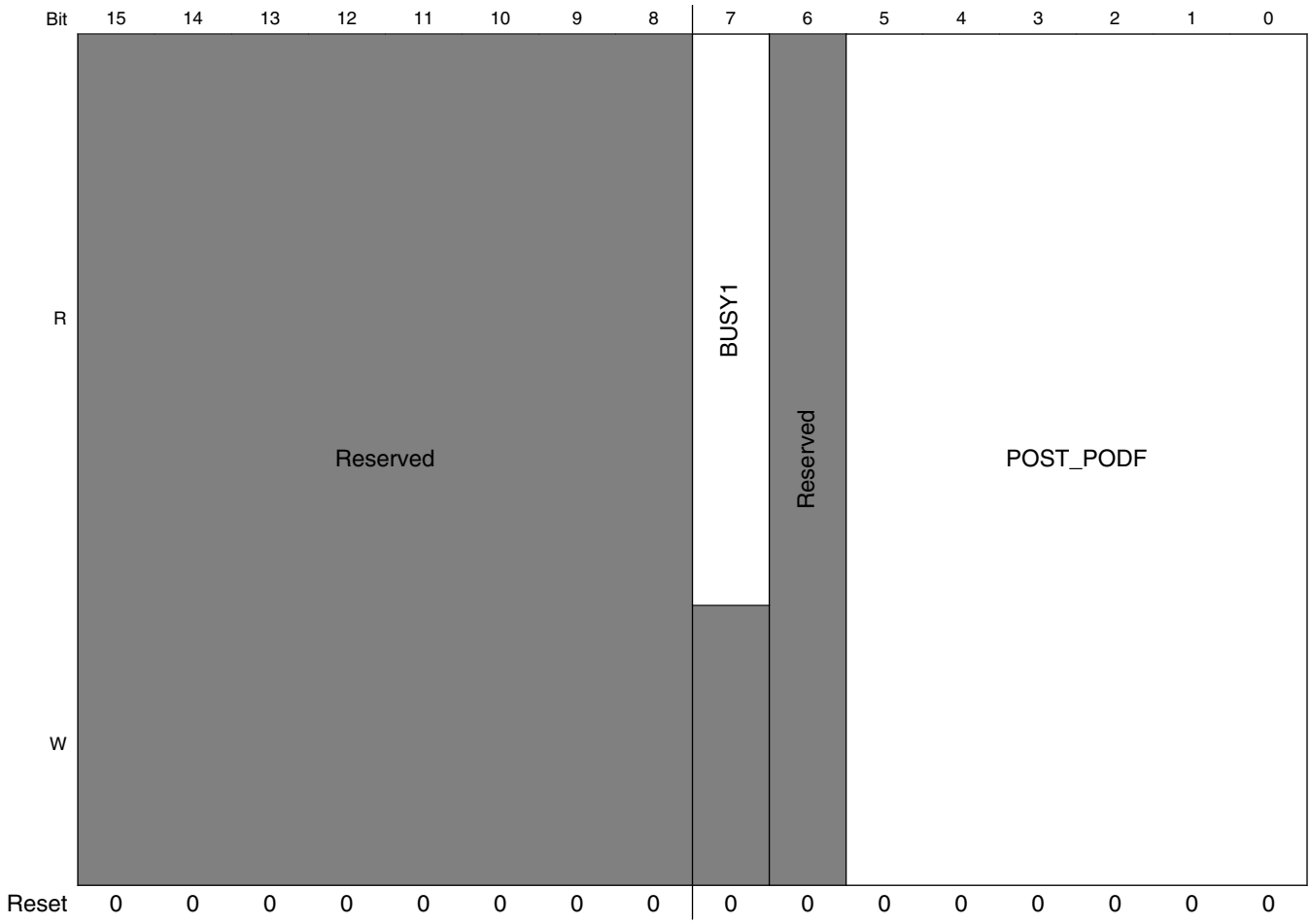
Field	Description
	For CORE, this field is 3 bit long. For IPG, this field is 2 bit long. This field does not apply to DRAM_PHYM 000000 Divide by 1 000001 Divide by 2 000010 Divide by 3 000011 Divide by 4 000100 Divide by 5 000101 Divide by 6 : 111111 Divide by 64

5.1.7.20 Post Divider Register (CCM_POST_ROOT*n*_CLR)

Post Root Register

Address: 3038_0000h base + 8028h offset + (128d × i), where i=0d to 124d





CCM_POST_ROOT*n*_CLR field descriptions

Field	Description
31 BUSY2	Clock switching multiplexer is applying new setting
30–29 -	This field is reserved. Reserved
28 SELECT	Selection of pre clock branches This field is not applied in IP 0 select branch A 1 select branch B
27–8 -	This field is reserved. Reserved
7 BUSY1	Post divider is applying new set value
6 -	This field is reserved. Reserved
POST_PODF	Post divider divide the number Divider value is n + 1

Table continues on the next page...

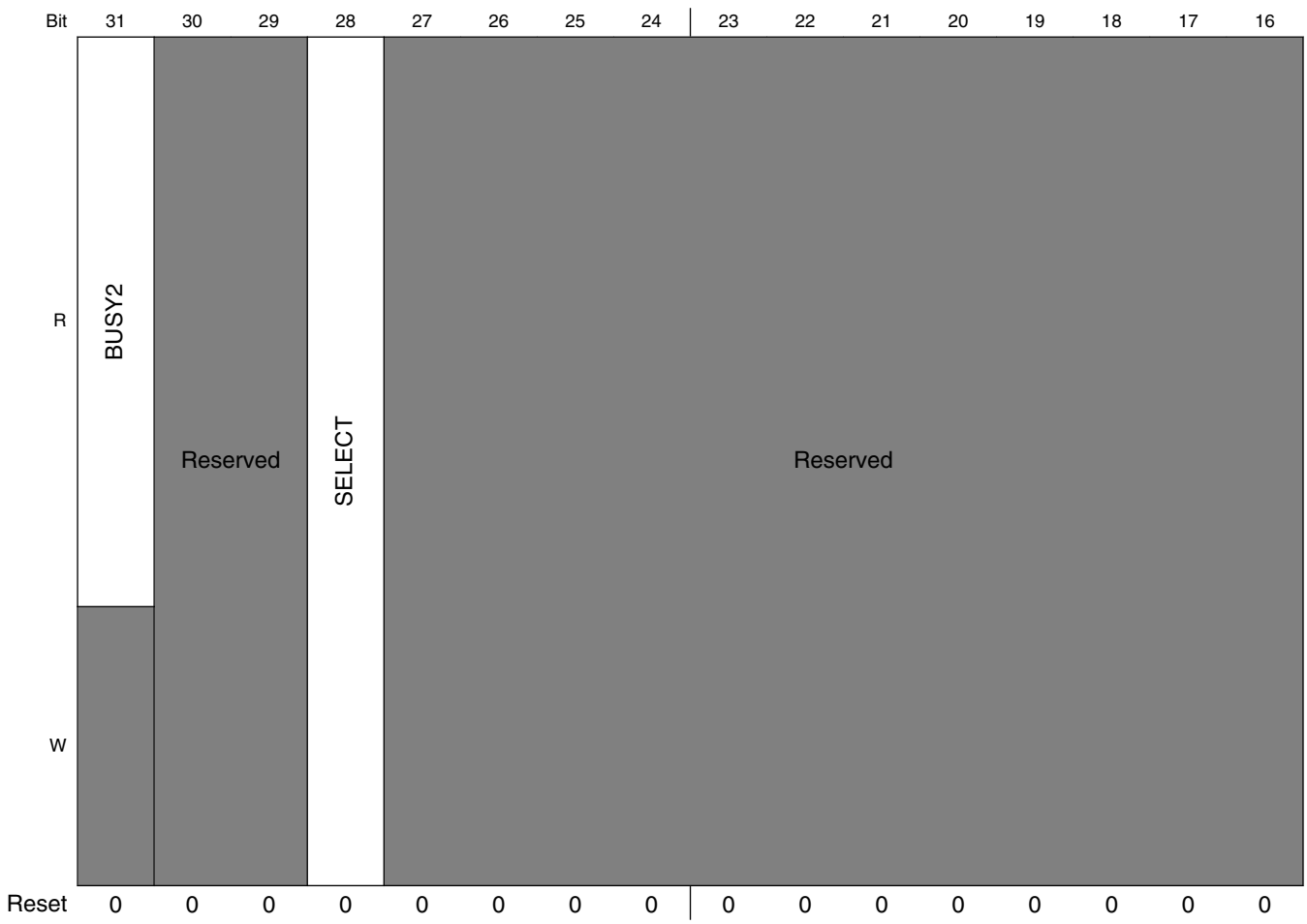
CCM_POST_ROOT*n*_CLR field descriptions (continued)

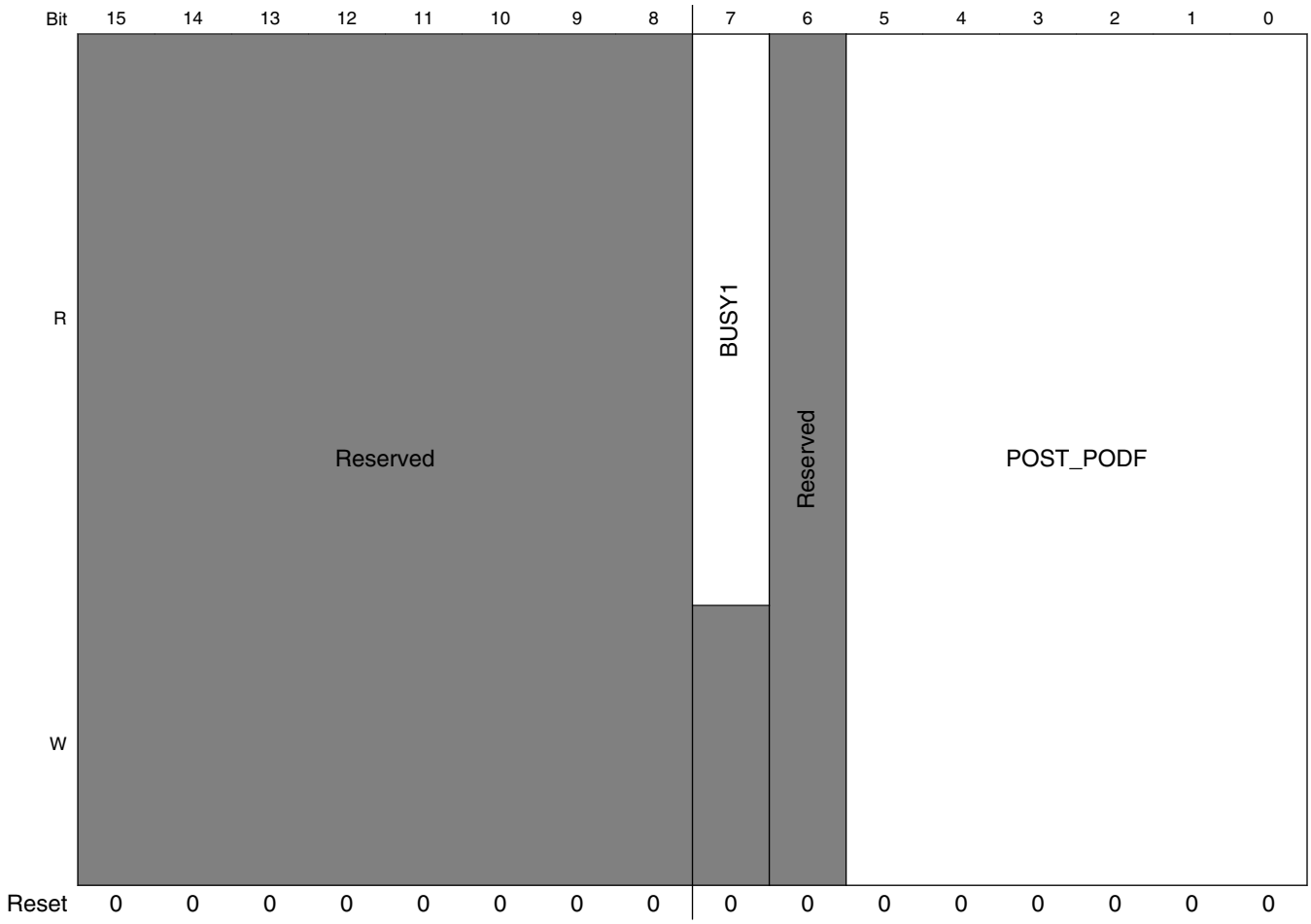
Field	Description
	For CORE, this field is 3 bit long. For IPG, this field is 2 bit long. This field does not apply to DRAM_PHYM 000000 Divide by 1 000001 Divide by 2 000010 Divide by 3 000011 Divide by 4 000100 Divide by 5 000101 Divide by 6 : 111111 Divide by 64

5.1.7.21 Post Divider Register (CCM_POST_ROOT*n*_TOG)

Post Root Register

Address: 3038_0000h base + 802Ch offset + (128d × i), where i=0d to 124d





CCM_POST_ROOT*n*_TOG field descriptions

Field	Description
31 BUSY2	Clock switching multiplexer is applying new setting
30–29 -	This field is reserved. Reserved
28 SELECT	Selection of pre clock branches This field is not applied in IP 0 select branch A 1 select branch B
27–8 -	This field is reserved. Reserved
7 BUSY1	Post divider is applying new set value
6 -	This field is reserved. Reserved
POST_PODF	Post divider divide number Divider value is n + 1

Table continues on the next page...

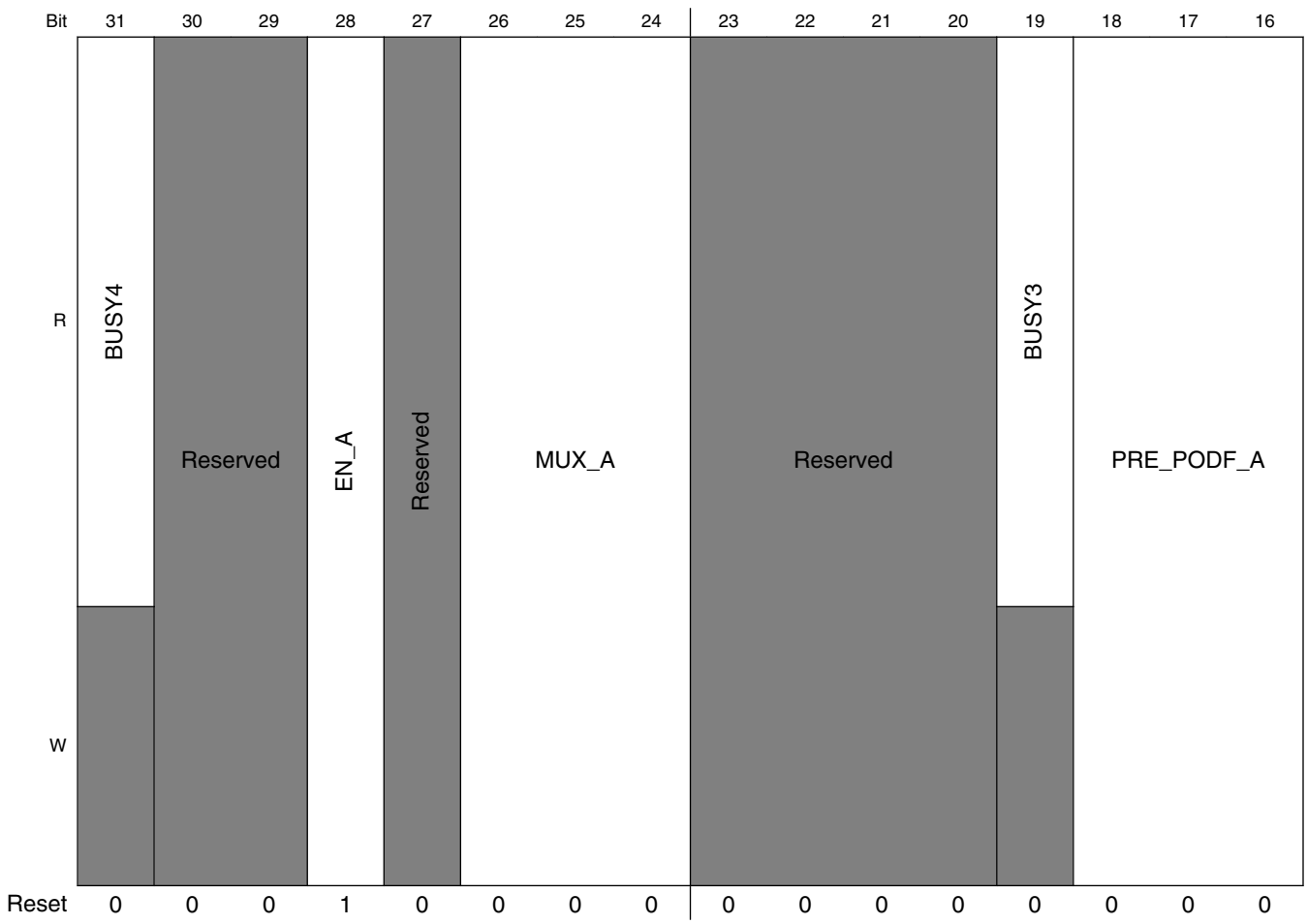
CCM_POST_ROOT*n*_TOG field descriptions (continued)

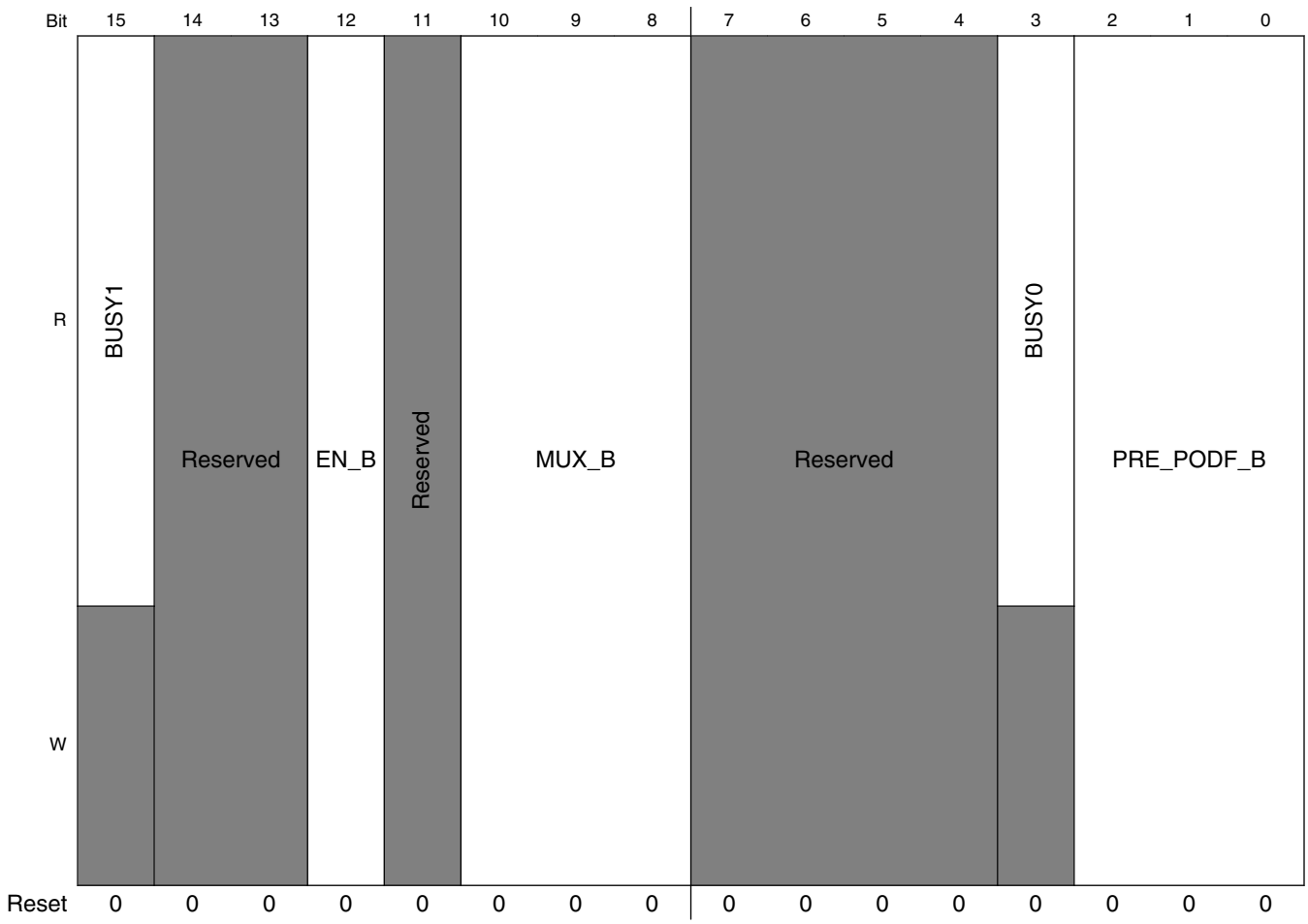
Field	Description
	For CORE, this field is 3 bit long. For IPG, this field is 2 bit long. This field does not apply to DRAM_PHYM 000000 Divide by 1 000001 Divide by 2 000010 Divide by 3 000011 Divide by 4 000100 Divide by 5 000101 Divide by 6 : 111111 Divide by 64

5.1.7.22 Pre Divider Register (CCM_PRE*n*)

Pre Register

Address: 3038_0000h base + 8030h offset + (128d × i), where i=0d to 124d





CCM_PRE*n* field descriptions

Field	Description
31 BUSY4	EN_A field is applied to field This field applies to DRAM and DRAM_PHYM
30–29 -	This field is reserved. Reserved
28 EN_A	Branch A clock gate control This field applies to DRAM and DRAM_PHYM 0 Clock shutdown 1 clock ON
27 -	This field is reserved. Reserved
26–24 MUX_A	Selection control of multiplexer of branch A This field applies to DRAM and DRAM_PHYM
23–20 -	This field is reserved. Reserved
19 BUSY3	Pre divider value for branch A is applied This field applies to DRAM and DRAM_PHYM

Table continues on the next page...

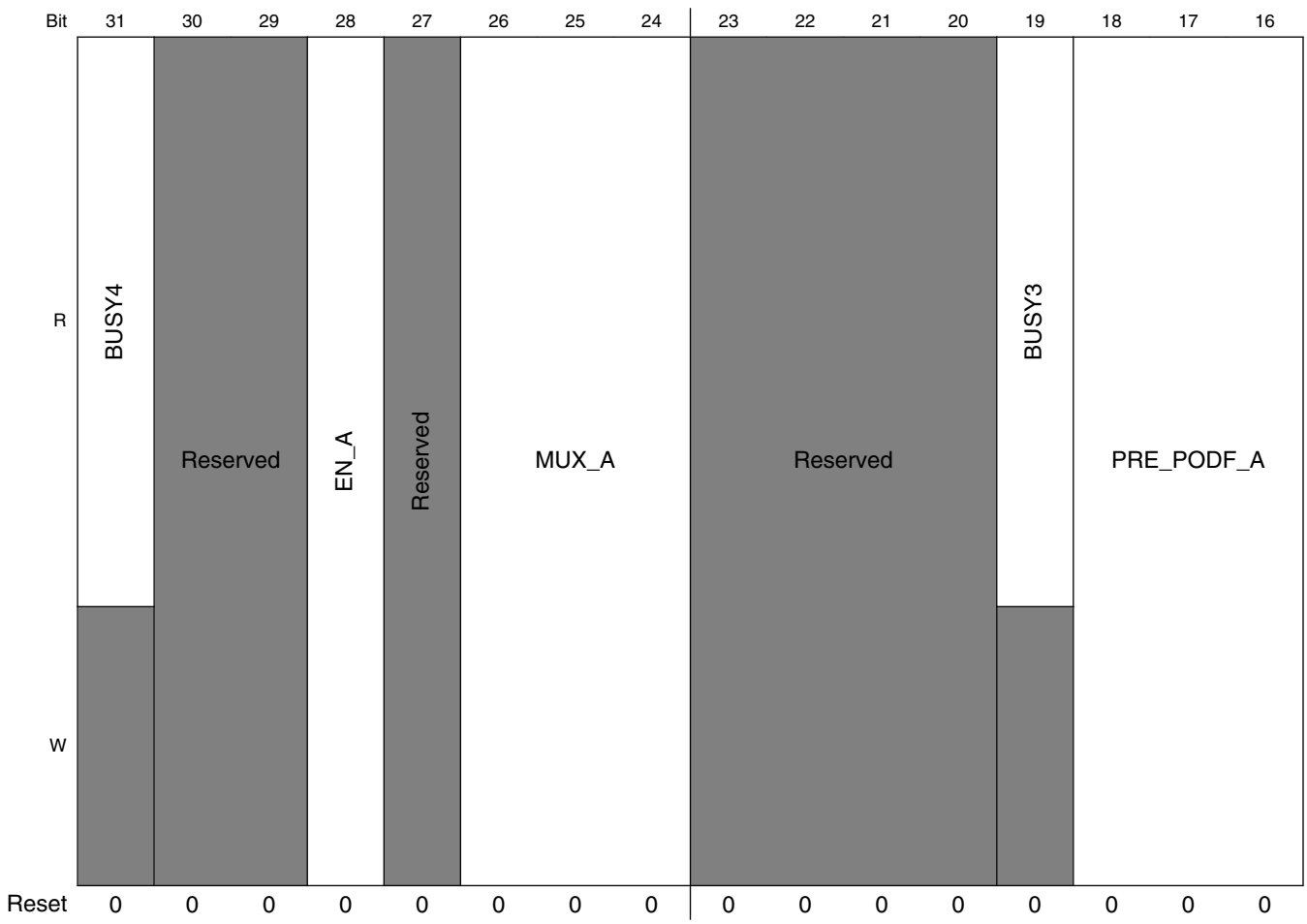
CCM_PRE*n* field descriptions (continued)

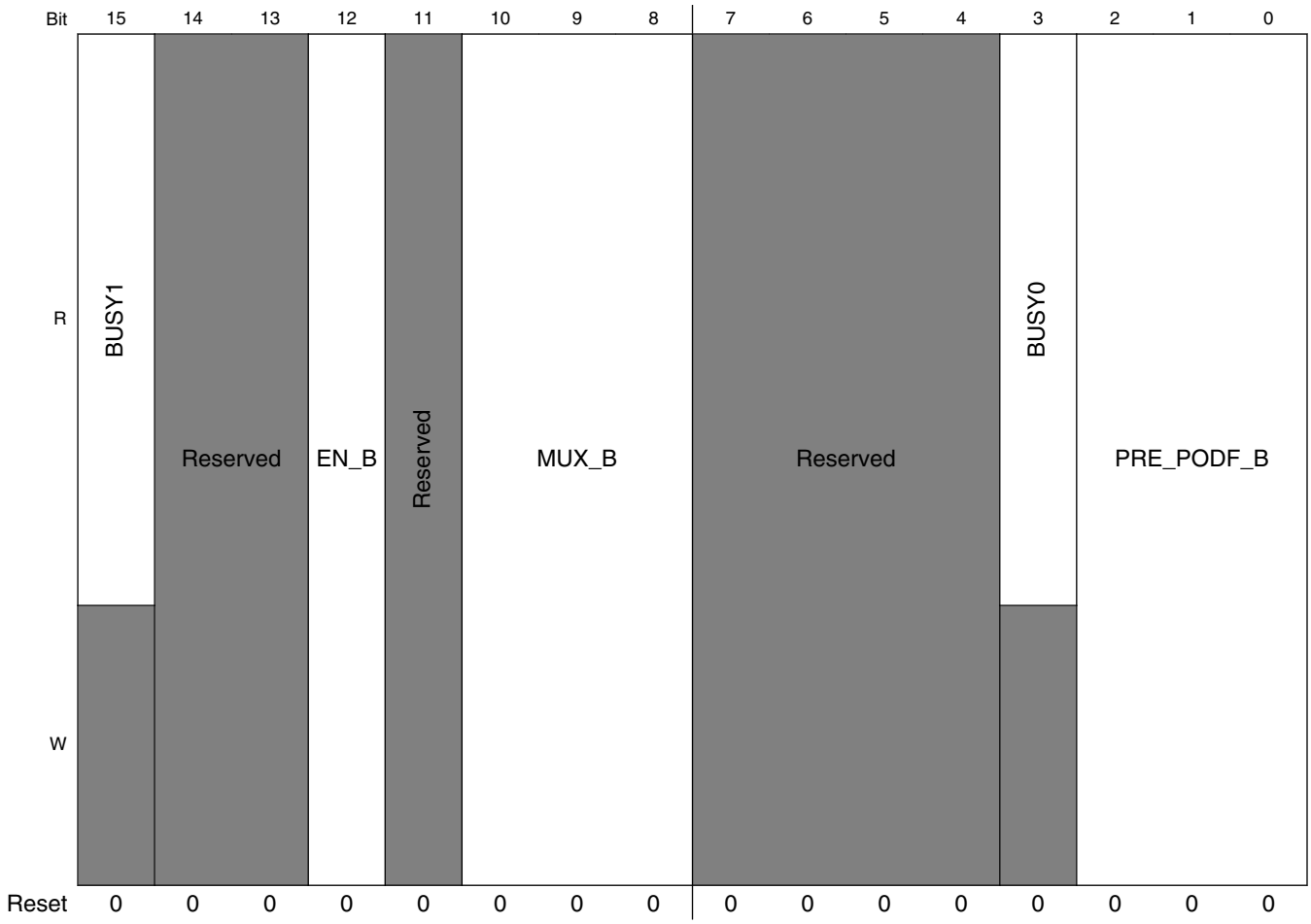
Field	Description
18–16 PRE_PODF_A	Pre divider divide number for branch A Divider value is n + 1. This field does not apply for CORE, DRAM, DRAM_PHYM 000 Divide by 1 001 Divide by 2 010 Divide by 3 011 Divide by 4 100 Divide by 5 101 Divide by 6 110 Divide by 7 111 Divide by 8
15 BUSY1	EN_B is applied to field This field does not apply for CORE, IP,DRAM, DRAM_PHYM
14–13 -	This field is reserved. Reserved
12 EN_B	Branch B clock gate control This field does not apply for CORE, IP,DRAM, DRAM_PHYM 0 Clock shutdown 1 Clock ON
11 -	This field is reserved. Reserved
10–8 MUX_B	Selection control of multiplexer of branch B This field does not apply for CORE, IP,DRAM, DRAM_PHYM
7–4 -	This field is reserved. Reserved
3 BUSY0	Pre divider value for branch a is applying field does not apply for CORE, IP,DRAM, DRAM_PHYM
PRE_PODF_B	Pre divider divide number for branch B Divider value is n + 1. This field does not apply for CORE, IP, DRAM, DRAM_PHYM 000 Divide by 1 001 Divide by 2 010 Divide by 3 011 Divide by 4 100 Divide by 5 101 Divide by 6 110 Divide by 7 111 Divide by 8

5.1.7.23 Pre Divider Register (CCM_PRE_ROOT*n*_SET)

Pre Divider Register

Address: 3038_0000h base + 8034h offset + (128d × i), where i=0d to 124d





CCM_PRE_ROOT*n*_SET field descriptions

Field	Description
31 BUSY4	EN_A field is applied to field This field applies to DRAM and DRAM_PHYM
30–29 -	This field is reserved. Reserved
28 EN_A	Branch A clock gate control This field applies to DRAM and DRAM_PHYM 0 Clock shutdown 1 clock ON
27 -	This field is reserved. Reserved
26–24 MUX_A	Selection control of multiplexer of branch A This field applies to DRAM and DRAM_PHYM
23–20 -	This field is reserved. Reserved
19 BUSY3	Pre divider value for branch A is applied This field applies to DRAM and DRAM_PHYM

Table continues on the next page...

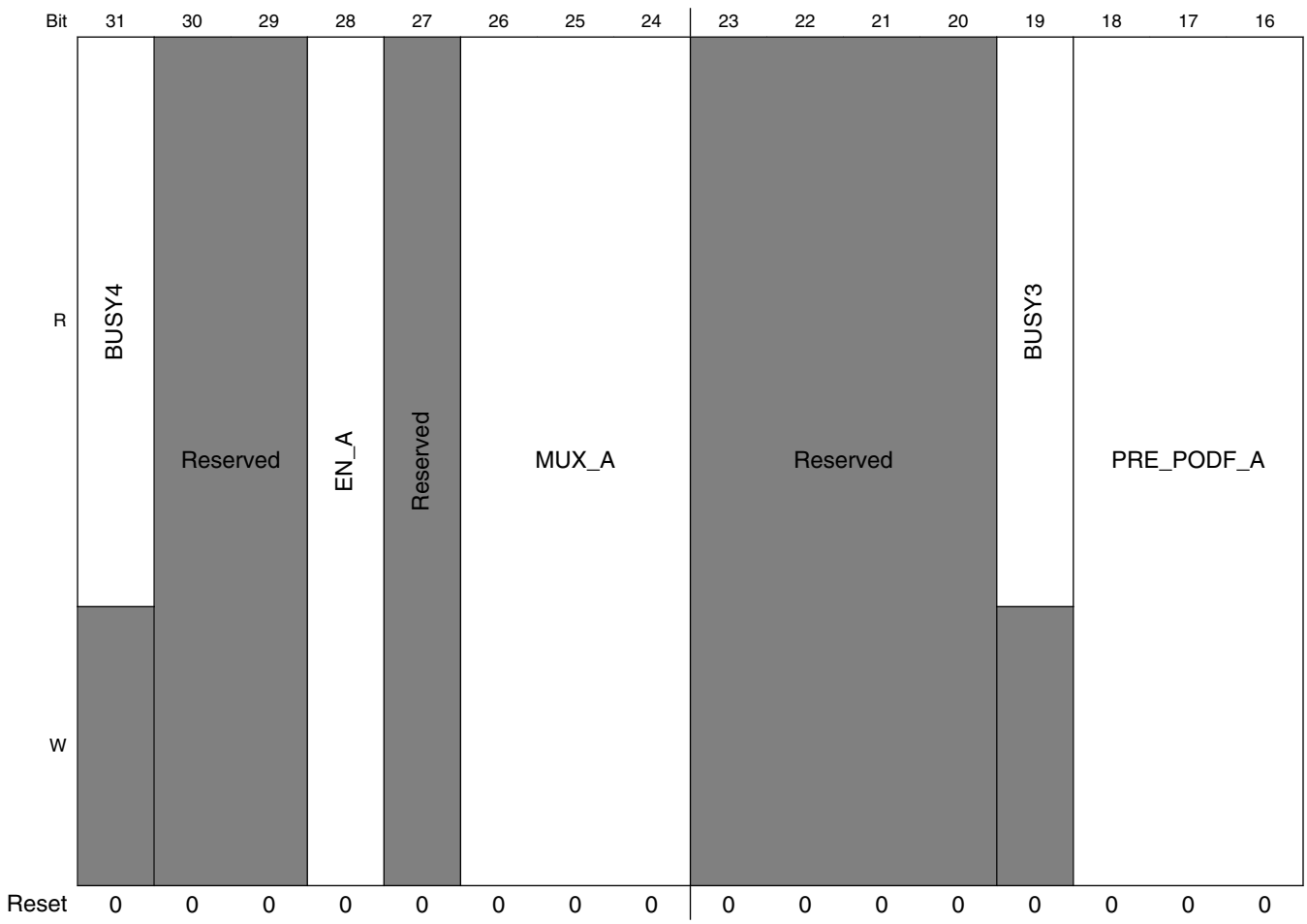
CCM_PRE_ROOT*n*_SET field descriptions (continued)

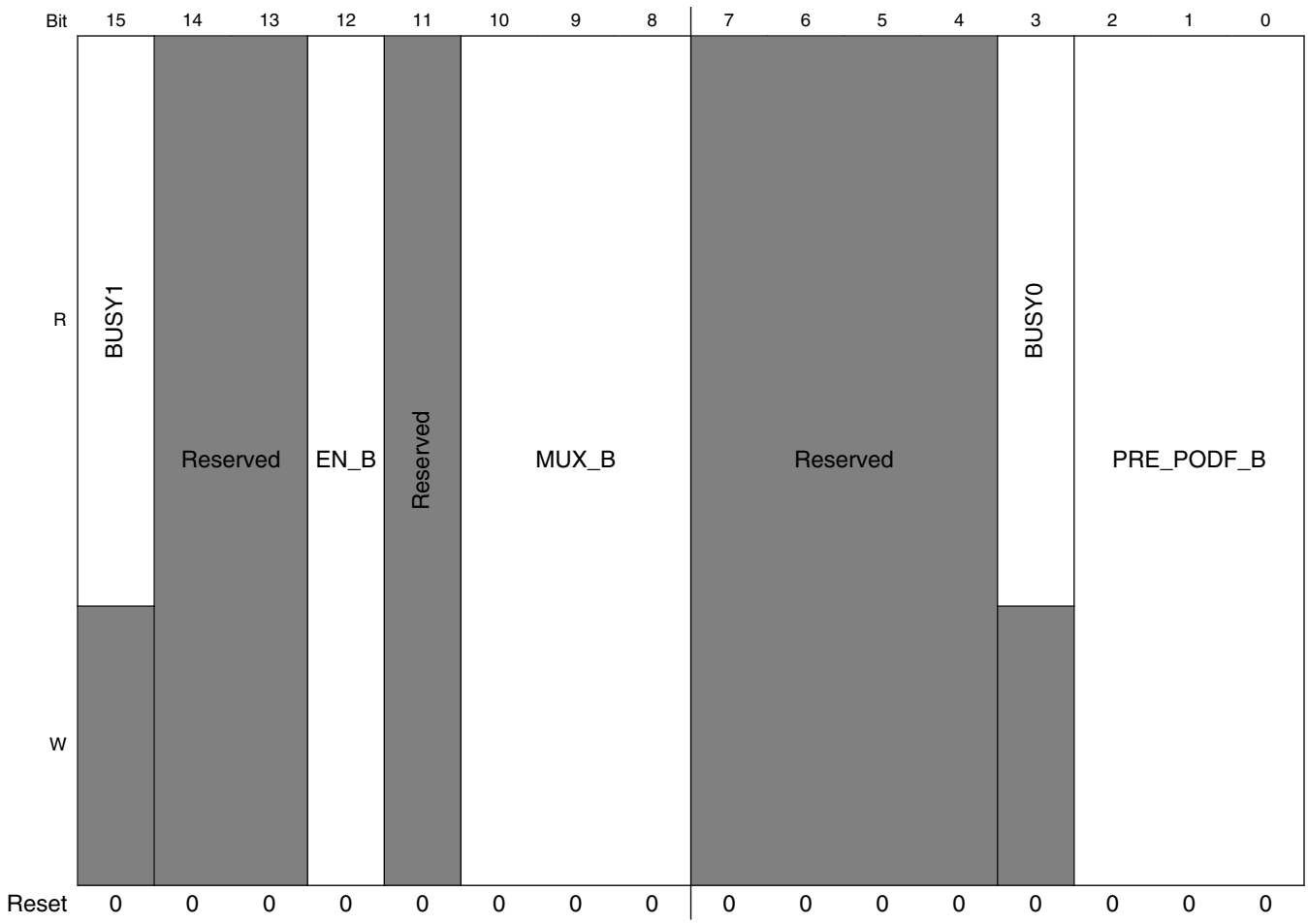
Field	Description
18–16 PRE_PODF_A	Pre divider divide number for branch A Divider value is n + 1. This field does not apply for CORE, DRAM, DRAM_PHYM 000 Divide by 1 001 Divide by 2 010 Divide by 3 011 Divide by 4 100 Divide by 5 101 Divide by 6 110 Divide by 7 111 Divide by 8
15 BUSY1	EN_B is applied to field This field does not apply for CORE, IP,DRAM, DRAM_PHYM
14–13 -	This field is reserved. Reserved
12 EN_B	Branch B clock gate control This field does not apply for CORE, IP,DRAM, DRAM_PHYM 0 Clock shutdown 1 Clock ON
11 -	This field is reserved. Reserved
10–8 MUX_B	Selection control of multiplexer of branch B This field does not apply for CORE, IP,DRAM, DRAM_PHYM
7–4 -	This field is reserved. Reserved
3 BUSY0	Pre divider value for branch A is applying field does not apply for CORE, IP,DRAM, DRAM_PHYM
PRE_PODF_B	Pre divider divide number for branch B Divider value is n + 1. This field does not apply for CORE, IP, DRAM, DRAM_PHYM 000 Divide by 1 001 Divide by 2 010 Divide by 3 011 Divide by 4 100 Divide by 5 101 Divide by 6 110 Divide by 7 111 Divide by 8

5.1.7.24 Pre Divider Register (CCM_PRE_ROOT*n*_CLR)

Pree Root Register

Address: 3038_0000h base + 8038h offset + (128d × i), where i=0d to 124d





CCM_PRE_ROOT*n*_CLR field descriptions

Field	Description
31 BUSY4	EN_A field is applied to field This field applies to DRAM and DRAM_PHYM
30–29 -	This field is reserved. Reserved
28 EN_A	Branch A clock gate control This field applies to DRAM and DRAM_PHYM 0 Clock shutdown 1 clock ON
27 -	This field is reserved. Reserved
26–24 MUX_A	Selection control of multiplexer of branch A This field applies to DRAM and DRAM_PHYM
23–20 -	This field is reserved. Reserved
19 BUSY3	Pre divider value for branch A is applied This field applies to DRAM and DRAM_PHYM

Table continues on the next page...

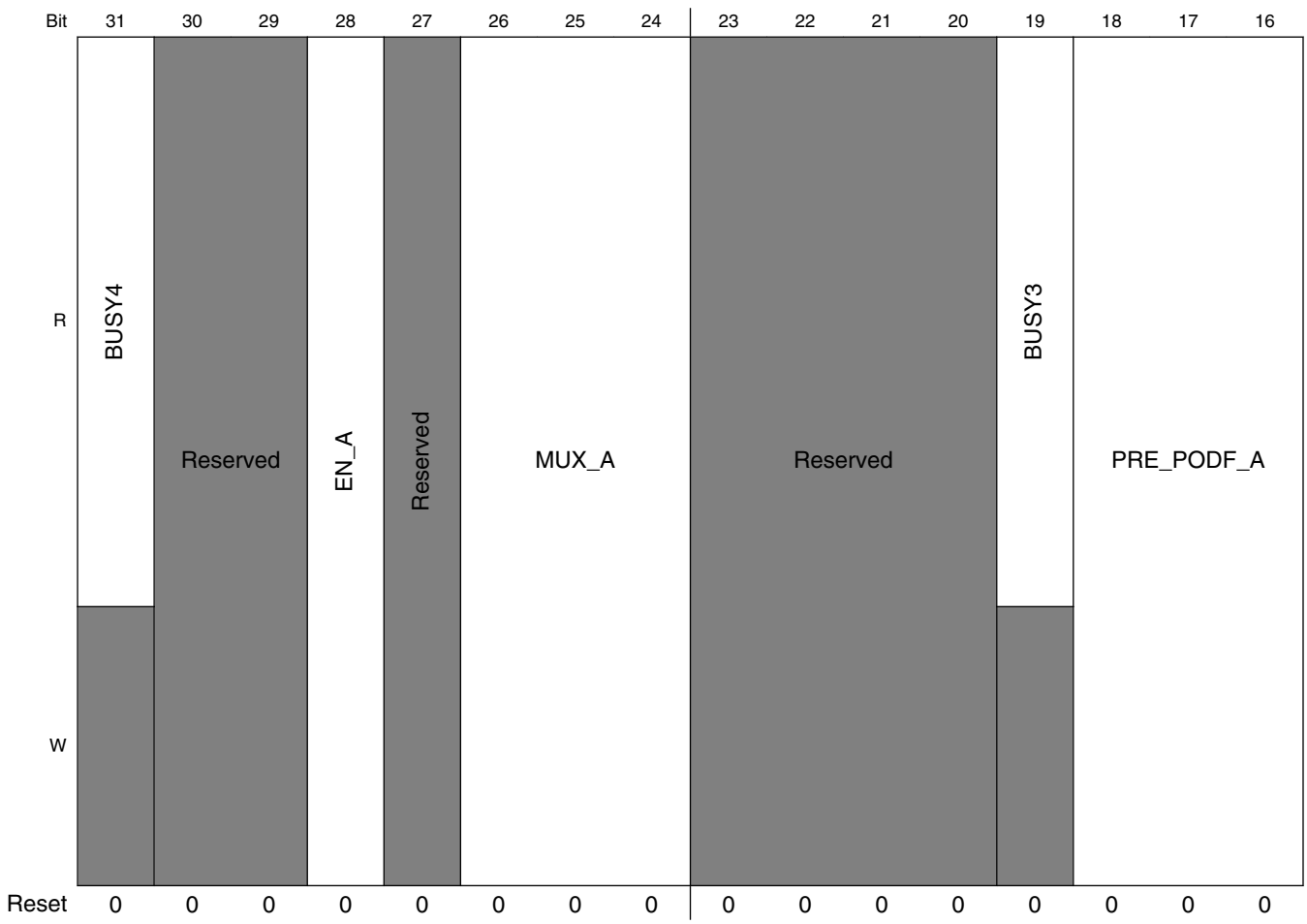
CCM_PRE_ROOT*n*_CLR field descriptions (continued)

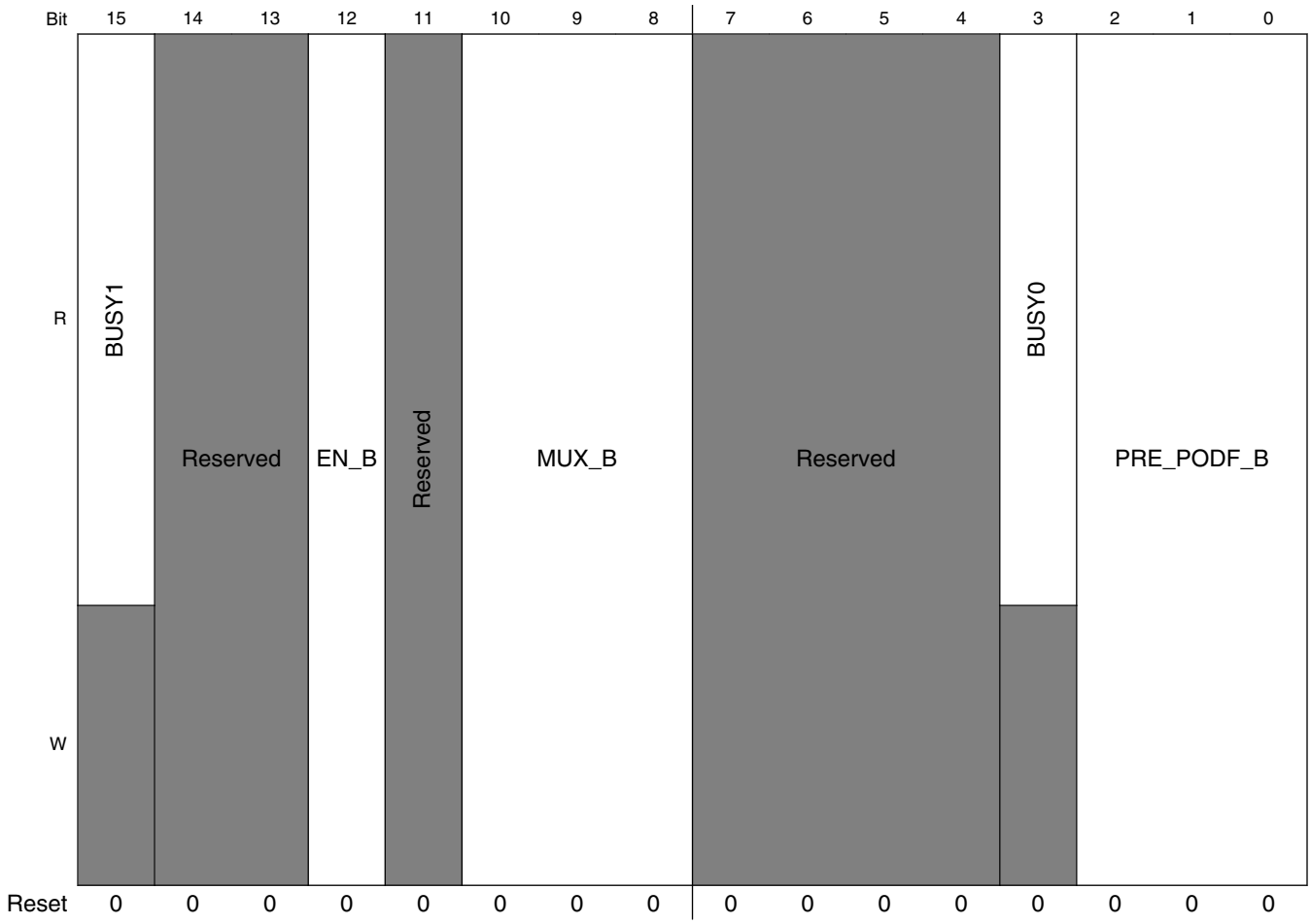
Field	Description
18–16 PRE_PODF_A	Pre divider divide number for branch A Divider value is n + 1. This field does not apply for CORE, DRAM, DRAM_PHYM 000 Divide by 1 001 Divide by 2 010 Divide by 3 011 Divide by 4 100 Divide by 5 101 Divide by 6 110 Divide by 7 111 Divide by 8
15 BUSY1	EN_B is applied to field This field does not apply for CORE, IP,DRAM, DRAM_PHYM
14–13 -	This field is reserved. Reserved
12 EN_B	Branch B clock gate control This field does not apply for CORE, IP,DRAM, DRAM_PHYM 0 Clock shutdown 1 Clock ON
11 -	This field is reserved. Reserved
10–8 MUX_B	Selection control of multiplexer of branch B This field does not apply for CORE, IP,DRAM, DRAM_PHYM
7–4 -	This field is reserved. Reserved
3 BUSY0	Pre divider value for branch A is applied This field does not apply for CORE, IP,DRAM, DRAM_PHYM
PRE_PODF_B	Pre divider divide number for branch B Divider value is n + 1. This field does not apply for CORE, IP, DRAM, DRAM_PHYM 000 Divide by 1 001 Divide by 2 010 Divide by 3 011 Divide by 4 100 Divide by 5 101 Divide by 6 110 Divide by 7 111 Divide by 8

5.1.7.25 Pre Divider Register (CCM_PRE_ROOT*n*_TOG)

Pre Root Register

Address: 3038_0000h base + 803Ch offset + (128d × i), where i=0d to 124d





CCM_PRE_ROOT*n*_TOG field descriptions

Field	Description
31 BUSY4	EN_A field is applied to field This field applies to DRAM and DRAM_PHYM
30–29 -	This field is reserved. Reserved
28 EN_A	Branch A clock gate control This field applies to DRAM and DRAM_PHYM 0 Clock shutdown 1 clock ON
27 -	This field is reserved. Reserved
26–24 MUX_A	Selection control of multiplexer of branch A This field applies to DRAM and DRAM_PHYM
23–20 -	This field is reserved. Reserved
19 BUSY3	Pre divider value for branch A is applied This field applies to DRAM and DRAM_PHYM

Table continues on the next page...

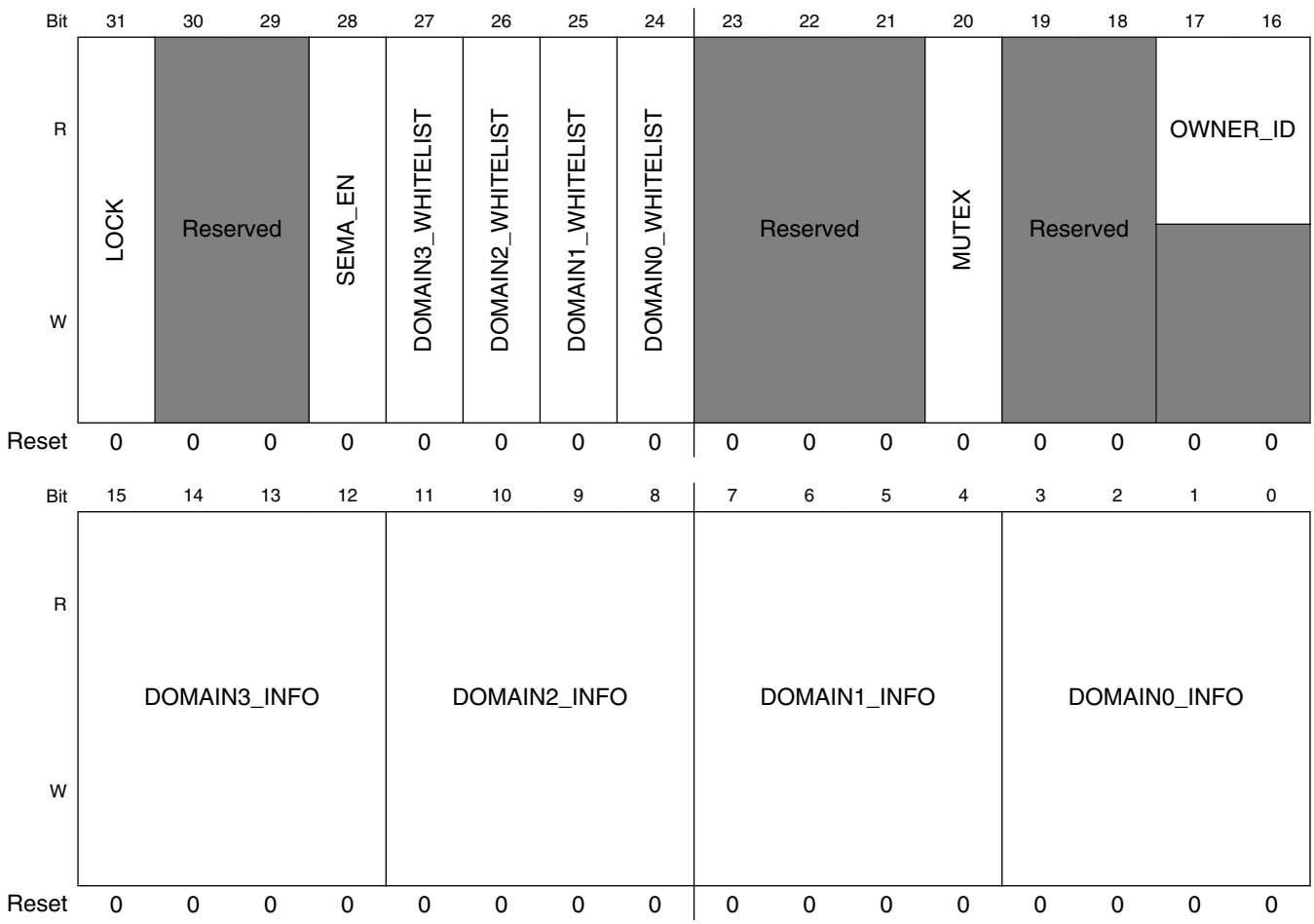
CCM_PRE_ROOT*n*_TOG field descriptions (continued)

Field	Description
18–16 PRE_PODF_A	Pre divider divide number for branch A Divider value is n + 1. This field does not apply for CORE, DRAM, DRAM_PHYM 000 Divide by 1 001 Divide by 2 010 Divide by 3 011 Divide by 4 100 Divide by 5 101 Divide by 6 110 Divide by 7 111 Divide by 8
15 BUSY1	EN_B is applied to field This field does not apply for CORE, IP,DRAM, DRAM_PHYM
14–13 -	This field is reserved. Reserved
12 EN_B	Branch B clock gate control This field does not apply for CORE, IP,DRAM, DRAM_PHYM 0 Clock shutdown 1 Clock ON
11 -	This field is reserved. Reserved
10–8 MUX_B	Selection control of multiplexer of branch B This field does not apply for CORE, IP,DRAM, DRAM_PHYM
7–4 -	This field is reserved. Reserved
3 BUSY0	Pre divider value for branch a is applied field does not apply for CORE, IP,DRAM, DRAM_PHYM
PRE_PODF_B	Pre divider divide number for branch B Divider value is n + 1. This field does not apply for CORE, IP, DRAM, DRAM_PHYM 000 Divide by 1 001 Divide by 2 010 Divide by 3 011 Divide by 4 100 Divide by 5 101 Divide by 6 110 Divide by 7 111 Divide by 8

5.1.7.26 Access Control Register (CCM_ACCESS_CTRL*n*)

Access Control Register

Address: 3038_0000h base + 8070h offset + (128d × i), where i=0d to 141d



CCM_ACCESS_CTRL*n* field descriptions

Field	Description
31 LOCK	Lock this clock root to use access control This bit can be set to 1 by software, and can be cleared only by system reset. 0 Access control inactive 1 Access control active
30–29 -	This field is reserved. Reserved
28 SEMA_EN	Enable internal semaphore This field cannot be changed when lock bit is 1

Table continues on the next page...

CCM_ACCESS_CTRL*n* field descriptions (continued)

Field	Description
	0 Disable 1 Enable
27 DOMAIN3_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
26 DOMAIN2_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
25 DOMAIN1_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
24 DOMAIN0_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
23–21 -	This field is reserved. Reserved
20 MUTEX	Semaphore to control access 0 Semaphore is free to take 1 Semaphore is taken Write 0 Release semaphore Write 1 Acquire semaphore
19–18 -	This field is reserved. Reserved
17–16 OWNER_ID	Current domain that owns semaphore This field is meaningless when MUTEX is 0 0 domaino 1 domain1 2 domain2 3 domain3
15–12 DOMAIN3_INFO	Information from domain 3 to pass to others This field can only be changed by domain 3
11–8 DOMAIN2_INFO	Information from domain 2 to pass to others This field can only be changed by domain 2
7–4 DOMAIN1_INFO	Information from domain 1 to pass to others This field can only be changed by domain 1

Table continues on the next page...

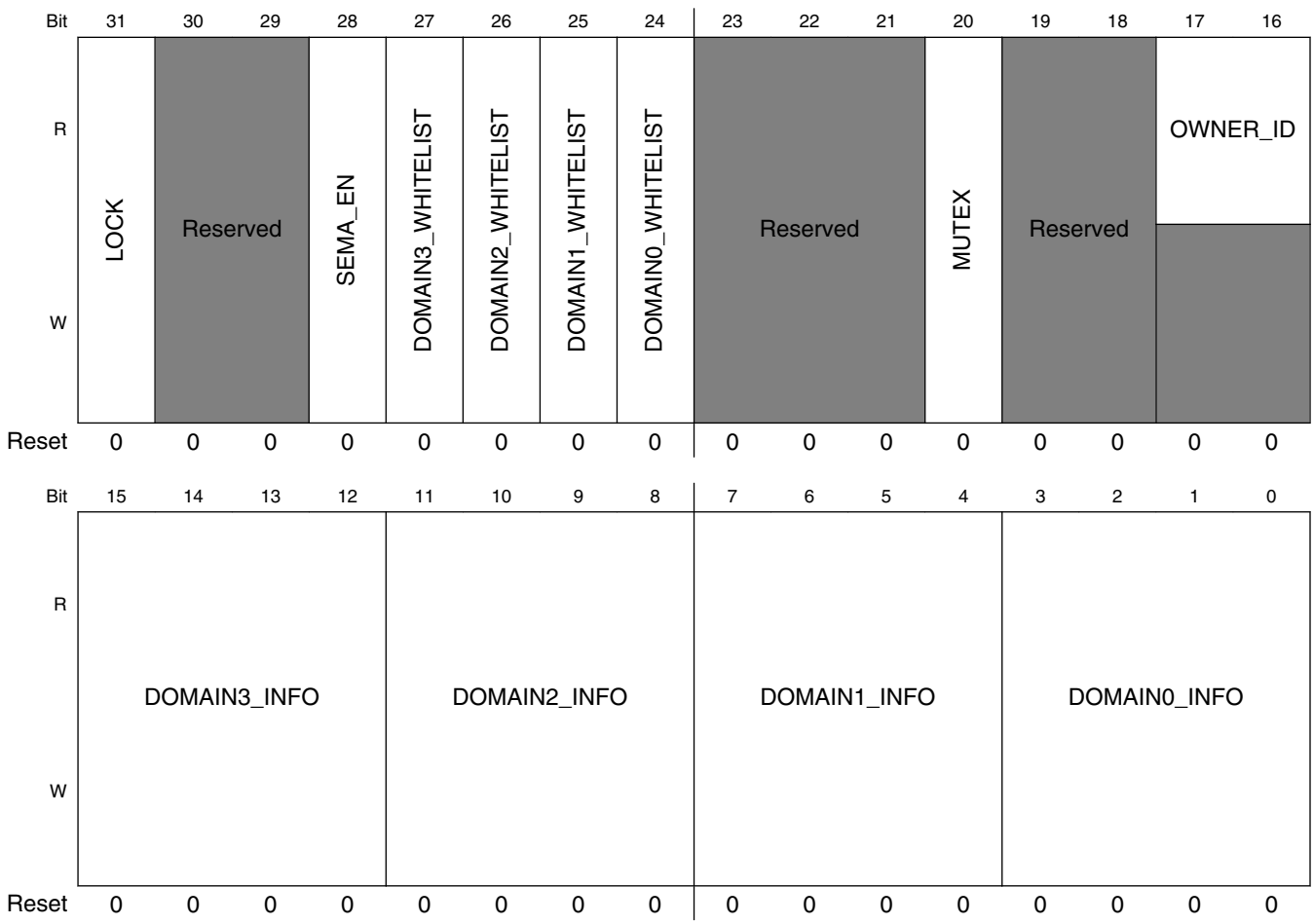
CCM_ACCESS_CTRL*n* field descriptions (continued)

Field	Description
DOMAIN0_INFO	Information from domain 0 to pass to others This field can only be changed by domain 0

5.1.7.27 Access Control Register (CCM_ACCESS_CTRL_ROOT*n*_SET)

Access Control Register

Address: 3038_0000h base + 8074h offset + (128d × i), where i=0d to 141d



CCM_ACCESS_CTRL_ROOT*n*_SET field descriptions

Field	Description
31 LOCK	Lock this clock root to use access control This bit can be set to 1 by software, and can be cleared only by system reset.

Table continues on the next page...

CCM_ACCESS_CTRL_ROOT*n*_SET field descriptions (continued)

Field	Description
	0 Access control inactive 1 Access control active
30–29 -	This field is reserved. Reserved
28 SEMA_EN	Enable internal semaphore This field cannot be changed when lock bit is 1 0 Disable 1 Enable
27 DOMAIN3_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
26 DOMAIN2_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
25 DOMAIN1_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
24 DOMAIN0_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
23–21 -	This field is reserved. Reserved
20 MUTEX	Semaphore to control access 0 Semaphore is free to take 1 Semaphore is taken Write 0 Release semaphore Write 1 Acquire semaphore
19–18 -	This field is reserved. Reserved
17–16 OWNER_ID	Current domain that owns semaphore This field is meaningless when MUTEX is 0 0 domaino 1 domain1 2 domain2 3 domain3

Table continues on the next page...

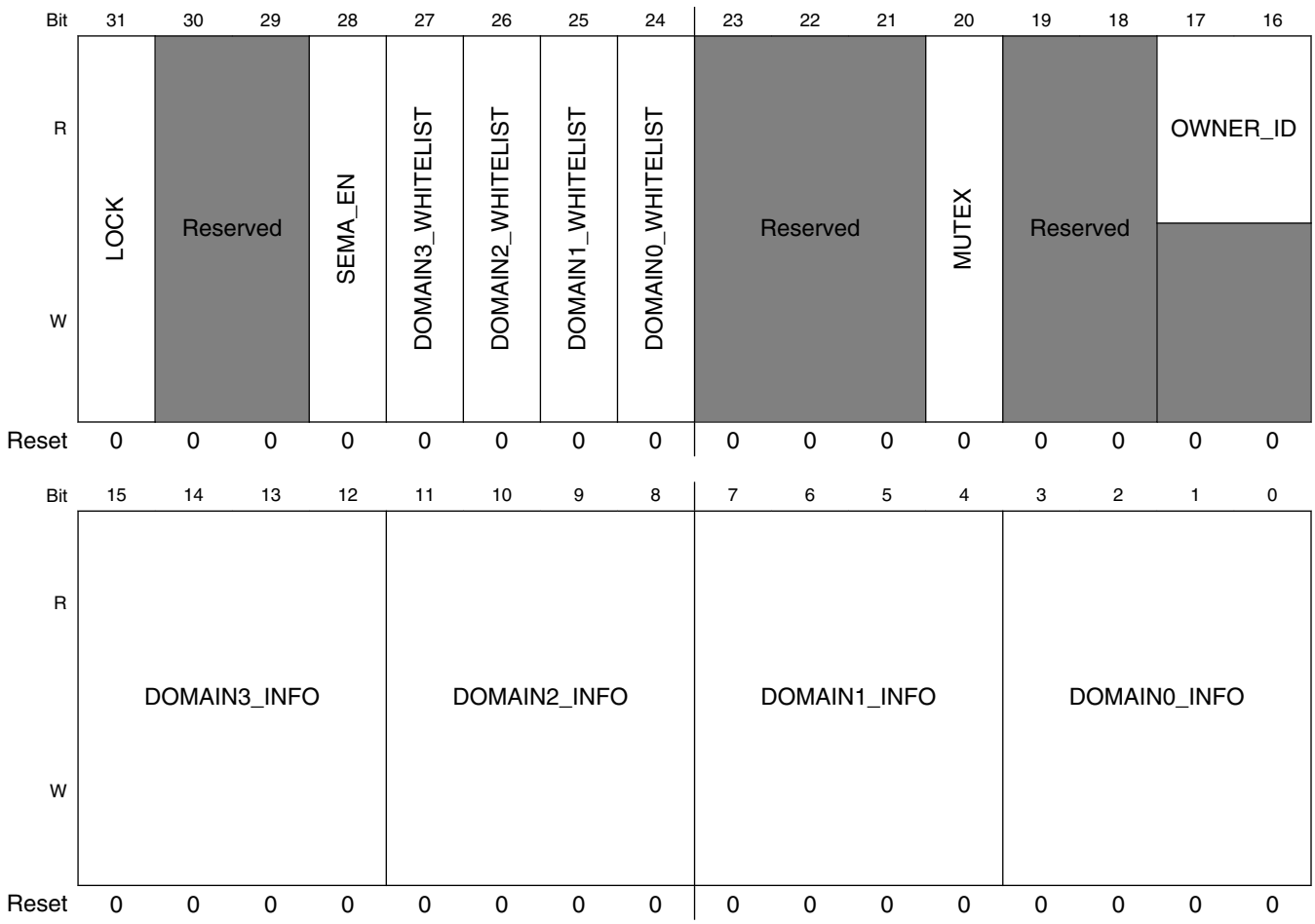
CCM_ACCESS_CTRL_ROOT*n*_SET field descriptions (continued)

Field	Description
15–12 DOMAIN3_INFO	Information from domain 3 to pass to others This field can only be changed by domain 3
11–8 DOMAIN2_INFO	Information from domain 2 to pass to others This field can only be changed by domain 2
7–4 DOMAIN1_INFO	Information from domain 1 to pass to others This field can only be changed by domain 1
DOMAIN0_INFO	Information from domain 0 to pass to others This field can only be changed by domain 0

5.1.7.28 Access Control Register (CCM_ACCESS_CTRL_ROOT*n*_CLR)

Access Control Register

Address: 3038_0000h base + 8078h offset + (128d × i), where i=0d to 141d



CCM_ACCESS_CTRL_ROOT*n*_CLR field descriptions

Field	Description
31 LOCK	Lock this clock root to use access control This bit can be set to 1 by software, and can be cleared only by system reset. 0 Access control inactive 1 Access control active
30–29 -	This field is reserved. Reserved

Table continues on the next page...

CCM_ACCESS_CTRL_ROOT*n*_CLR field descriptions (continued)

Field	Description
28 SEMA_EN	Enable internal semaphore This field cannot be changed when lock bit is 1 0 Disable 1 Enable
27 DOMAIN3_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
26 DOMAIN2_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
25 DOMAIN1_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
24 DOMAIN0_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
23–21 -	This field is reserved. Reserved
20 MUTEX	Semaphore to control access 0 Semaphore is free to take 1 Semaphore is taken Write 0 Release semaphore Write 1 Acquire semaphore
19–18 -	This field is reserved. Reserved
17–16 OWNER_ID	Current domain that owns semaphore This field is meaningless when MUTEX is 0 0 domaino 1 domain1 2 domain2 3 domain3
15–12 DOMAIN3_INFO	Information from domain 3 to pass to others This field can only be changed by domain 3
11–8 DOMAIN2_INFO	Information from domain 2 to pass to others This field can only be changed by domain 2

Table continues on the next page...

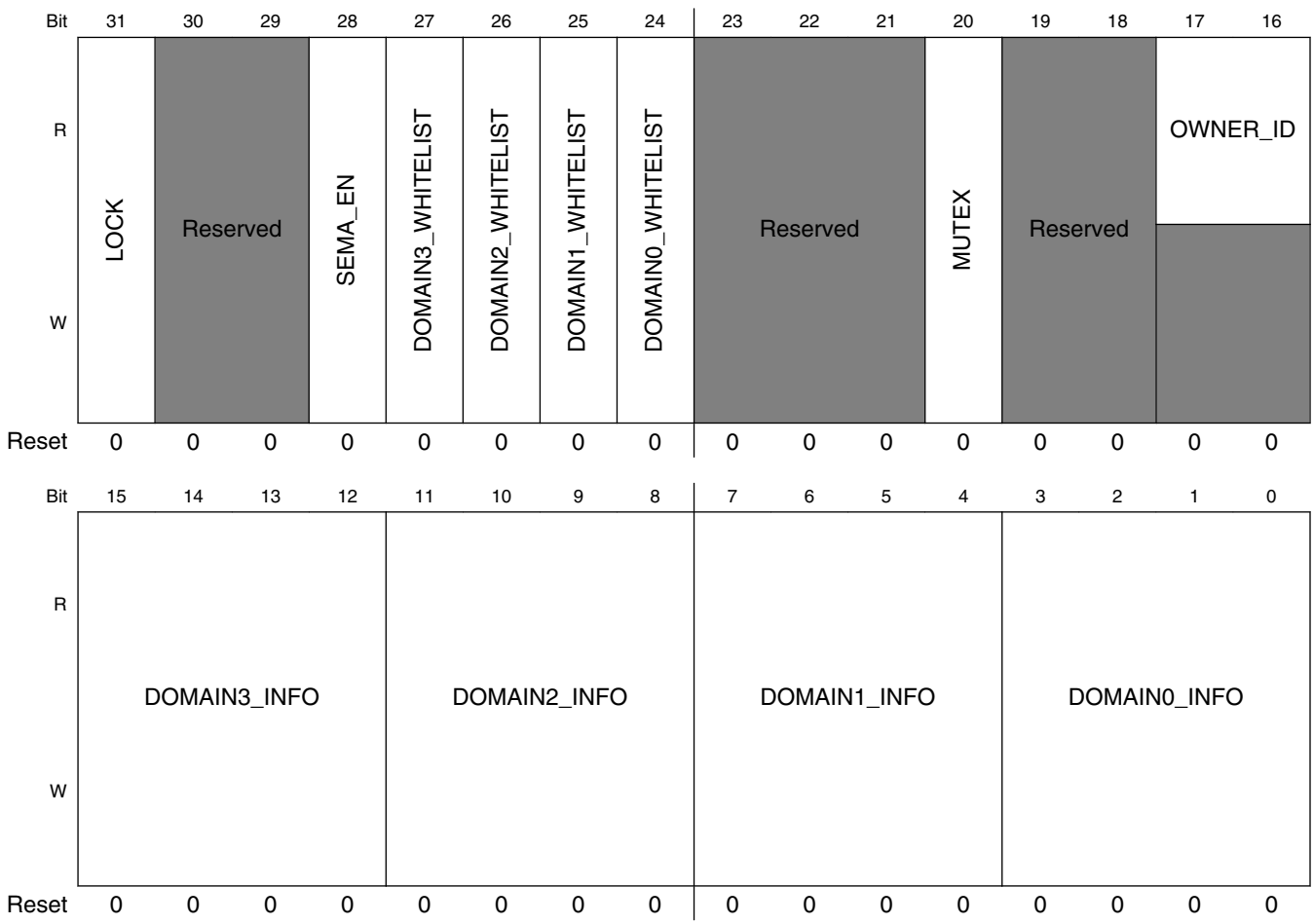
CCM_ACCESS_CTRL_ROOT*n*_CLR field descriptions (continued)

Field	Description
7–4 DOMAIN1_INFO	Information from domain 1 to pass to others This field can only be changed by domain 1
DOMAIN0_INFO	Information from domain 0 to pass to others This field can only be changed by domain 0

5.1.7.29 Access Control Register (CCM_ACCESS_CTRL_ROOT*n*_TOG)

Access Control Register

Address: 3038_0000h base + 807Ch offset + (128d × i), where i=0d to 141d



CCM_ACCESS_CTRL_ROOT*n*_TOG field descriptions

Field	Description
31 LOCK	Lock this clock root to use access control This bit can be set to 1 by software, and can be cleared only by system reset. 0 Access control inactive 1 Access control active
30–29 -	This field is reserved. Reserved
28 SEMA_EN	Enable internal semaphore This field cannot be changed when lock bit is 1 0 Disable 1 Enable
27 DOMAIN3_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
26 DOMAIN2_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
25 DOMAIN1_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
24 DOMAIN0_ WHITELIST	White list of domains that can change setting of this clock root. Each domain has a corresponding bit. 0 Domain cannot change the setting 1 Domain can change the setting
23–21 -	This field is reserved. Reserved
20 MUTEX	Semaphore to control access 0 Semaphore is free to take 1 Semaphore is taken Write 0 Release semaphore Write 1 Acquire semaphore
19–18 -	This field is reserved. Reserved
17–16 OWNER_ID	Current domain that owns semaphore This field is meaningless when MUTEX is 0 0 domaino 1 domain1

Table continues on the next page...

CCM_ACCESS_CTRL_ROOT*n*_TOG field descriptions (continued)

Field	Description
	2 domain2 3 domain3
15–12 DOMAIN3_INFO	Information from domain 3 to pass to others This field can only be changed by domain 3
11–8 DOMAIN2_INFO	Information from domain 2 to pass to others This field can only be changed by domain 2
7–4 DOMAIN1_INFO	Information from domain 1 to pass to others This field can only be changed by domain 1
DOMAIN0_INFO	Information from domain 0 to pass to others This field can only be changed by domain 0

5.1.8 CCM Analog Memory Map/Register Definition

CCM_ANALOG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3036_0000	AUDIO PLL1 General Function Control Register (CCM_ANALOG_AUDIO_PLL1_GEN_CTRL)	32	R/W	0000_2010h	5.1.8.1/ 516
3036_0004	AUDIO PLL1 Divide and Fraction Data Control 0 Register (CCM_ANALOG_AUDIO_PLL1_FDIV_CTL0)	32	R/W	0014_5032h	5.1.8.2/ 518
3036_0008	AUDIO PLL1 Divide and Fraction Data Control 1 Register (CCM_ANALOG_AUDIO_PLL1_FDIV_CTL1)	32	R/W	0000_0000h	5.1.8.3/ 519
3036_000C	AUDIO PLL1 PLL SSCG Control Register (CCM_ANALOG_AUDIO_PLL1_SSCG_CTRL)	32	R/W	0000_0000h	5.1.8.4/ 519
3036_0010	AUDIO PLL1 PLL Monitoring Control Register (CCM_ANALOG_AUDIO_PLL1_MNIT_CTRL)	32	R/W	0010_0103h	5.1.8.5/ 521
3036_0014	AUDIO PLL2 General Function Control Register (CCM_ANALOG_AUDIO_PLL2_GEN_CTRL)	32	R/W	0000_2010h	5.1.8.6/ 523
3036_0018	AUDIO PLL2 Divide and Fraction Data Control 0 Register (CCM_ANALOG_AUDIO_PLL2_FDIV_CTL0)	32	R/W	0014_5032h	5.1.8.7/ 525
3036_001C	AUDIO PLL2 Divide and Fraction Data Control 1 Register (CCM_ANALOG_AUDIO_PLL2_FDIV_CTL1)	32	R/W	0000_0000h	5.1.8.8/ 526
3036_0020	AUDIO PLL2 PLL SSCG Control Register (CCM_ANALOG_AUDIO_PLL2_SSCG_CTRL)	32	R/W	0000_0000h	5.1.8.9/ 526
3036_0024	AUDIO PLL2 PLL Monitoring Control Register (CCM_ANALOG_AUDIO_PLL2_MNIT_CTRL)	32	R/W	0010_0103h	5.1.8.10/ 528
3036_0028	VIDEO PLL1 General Function Control Register (CCM_ANALOG_VIDEO_PLL1_GEN_CTRL)	32	R/W	0000_2010h	5.1.8.11/ 530
3036_002C	VIDEO PLL1 Divide and Fraction Data Control 0 Register (CCM_ANALOG_VIDEO_PLL1_FDIV_CTL0)	32	R/W	0014_5032h	5.1.8.12/ 532

Table continues on the next page...

CCM_ANALOG memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3036_0030	VIDEO PLL1 Divide and Fraction Data Control 1 Register (CCM_ANALOG_VIDEO_PLL1_FDIV_CTL1)	32	R/W	0000_0000h	5.1.8.13/ 533
3036_0034	VIDEO PLL1 PLL SSCG Control Register (CCM_ANALOG_VIDEO_PLL1_SSCG_CTRL)	32	R/W	0000_0000h	5.1.8.14/ 533
3036_0038	VIDEO PLL1 PLL Monitoring Control Register (CCM_ANALOG_VIDEO_PLL1_MNIT_CTRL)	32	R/W	0010_0103h	5.1.8.15/ 535
3036_0050	DRAM PLL General Function Control Register (CCM_ANALOG_DRAM_PLL_GEN_CTRL)	32	R/W	0000_2010h	5.1.8.16/ 537
3036_0054	DRAM PLL Divide and Fraction Data Control 0 Register (CCM_ANALOG_DRAM_PLL_FDIV_CTL0)	32	R/W	0012_C032h	5.1.8.17/ 539
3036_0058	DRAM PLL Divide and Fraction Data Control 1 Register (CCM_ANALOG_DRAM_PLL_FDIV_CTL1)	32	R/W	0000_0000h	5.1.8.18/ 540
3036_005C	DRAM PLL PLL SSCG Control Register (CCM_ANALOG_DRAM_PLL_SSCG_CTRL)	32	R/W	0000_0000h	5.1.8.19/ 540
3036_0060	DRAM PLL PLL Monitoring Control Register (CCM_ANALOG_DRAM_PLL_MNIT_CTRL)	32	R/W	0010_0103h	5.1.8.20/ 542
3036_0064	GPU PLL General Function Control Register (CCM_ANALOG_GPU_PLL_GEN_CTRL)	32	R/W	0000_0810h	5.1.8.21/ 544
3036_0068	GPU PLL Divide and Fraction Data Control 0 Register (CCM_ANALOG_GPU_PLL_FDIV_CTL0)	32	R/W	000C_8031h	5.1.8.22/ 546
3036_006C	PLL Lock Detector Control Register (CCM_ANALOG_GPU_PLL_LOCKD_CTRL)	32	R/W	0010_003Fh	5.1.8.23/ 547
3036_0070	PLL Monitoring Control Register (CCM_ANALOG_GPU_PLL_MNIT_CTRL)	32	R/W	0028_0081h	5.1.8.24/ 548
3036_0074	VPU PLL General Function Control Register (CCM_ANALOG_VPU_PLL_GEN_CTRL)	32	R/W	0000_0810h	5.1.8.25/ 550
3036_0078	VPU PLL Divide and Fraction Data Control 0 Register (CCM_ANALOG_VPU_PLL_FDIV_CTL0)	32	R/W	0012_C032h	5.1.8.26/ 552
3036_007C	PLL Lock Detector Control Register (CCM_ANALOG_VPU_PLL_LOCKD_CTRL)	32	R/W	0010_003Fh	5.1.8.23/ 547
3036_0080	PLL Monitoring Control Register (CCM_ANALOG_VPU_PLL_MNIT_CTRL)	32	R/W	0028_0081h	5.1.8.24/ 548
3036_0084	ARM PLL General Function Control Register (CCM_ANALOG_ARM_PLL_GEN_CTRL)	32	R/W	0000_0810h	5.1.8.27/ 554
3036_0088	ARM PLL Divide and Fraction Data Control 0 Register (CCM_ANALOG_ARM_PLL_FDIV_CTL0)	32	R/W	000F_A030h	5.1.8.28/ 556
3036_008C	PLL Lock Detector Control Register (CCM_ANALOG_ARM_PLL_LOCKD_CTRL)	32	R/W	0010_003Fh	5.1.8.23/ 547
3036_0090	PLL Monitoring Control Register (CCM_ANALOG_ARM_PLL_MNIT_CTRL)	32	R/W	0028_0081h	5.1.8.24/ 548
3036_0094	SYS PLL1 General Function Control Register (CCM_ANALOG_SYS_PLL1_GEN_CTRL)	32	R/W	0AAA_A810h	5.1.8.29/ 558
3036_0098	SYS PLL1 Divide and Fraction Data Control 0 Register (CCM_ANALOG_SYS_PLL1_FDIV_CTL0)	32	R/W	0019_0032h	5.1.8.30/ 561

Table continues on the next page...

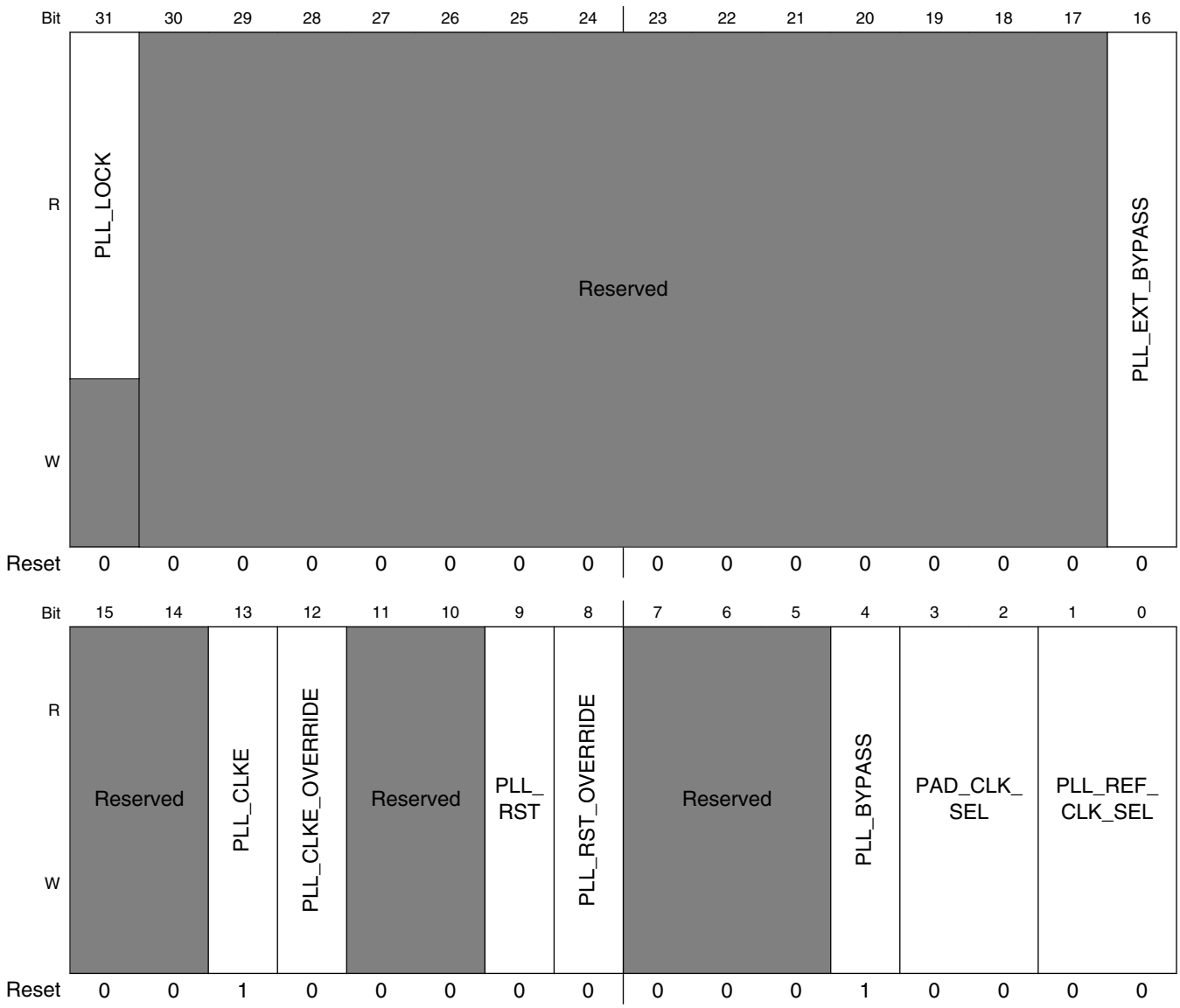
CCM_ANALOG memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3036_009C	PLL Lock Detector Control Register (CCM_ANALOG_SYS_PLL1_LOCKD_CTRL)	32	R/W	0010_003Fh	5.1.8.23/ 547
3036_0100	PLL Monitoring Control Register (CCM_ANALOG_SYS_PLL1_MNIT_CTRL)	32	R/W	0028_0081h	5.1.8.31/ 562
3036_0104	SYS PLL2 General Function Control Register (CCM_ANALOG_SYS_PLL2_GEN_CTRL)	32	R/W	0AAA_A810h	5.1.8.32/ 564
3036_0108	SYS PLL2 Divide and Fraction Data Control 0 Register (CCM_ANALOG_SYS_PLL2_FDIV_CTL0)	32	R/W	000F_A031h	5.1.8.33/ 567
3036_010C	PLL Lock Detector Control Register (CCM_ANALOG_SYS_PLL2_LOCKD_CTRL)	32	R/W	0010_003Fh	5.1.8.34/ 568
3036_0110	PLL Monitoring Control Register (CCM_ANALOG_SYS_PLL2_MNIT_CTRL)	32	R/W	0028_0081h	5.1.8.31/ 562
3036_0114	SYS PLL3 General Function Control Register (CCM_ANALOG_SYS_PLL3_GEN_CTRL)	32	R/W	0000_0810h	5.1.8.35/ 569
3036_0118	SYS PLL3 Divide and Fraction Data Control 0 Register (CCM_ANALOG_SYS_PLL3_FDIV_CTL0)	32	R/W	000F_A031h	5.1.8.36/ 571
3036_011C	PLL Lock Detector Control Register (CCM_ANALOG_SYS_PLL3_LOCKD_CTRL)	32	R/W	0010_003Fh	5.1.8.34/ 568
3036_0120	PLL Monitoring Control Register (CCM_ANALOG_SYS_PLL3_MNIT_CTRL)	32	R/W	0028_0081h	5.1.8.31/ 562
3036_0124	Osc Misc Configuration Register (CCM_ANALOG_OSC_MISC_CFG)	32	R/W	0000_0000h	5.1.8.37/ 572
3036_0128	PLL Clock Output for Test Enable and Select Register (CCM_ANALOG_ANAMIX_PLL_MNIT_CTL)	32	R/W	0000_0000h	5.1.8.38/ 573
3036_0800	DIGPROG Register (CCM_ANALOG_DIGPROG)	32	R	0082_4010h	5.1.8.39/ 575

5.1.8.1 AUDIO PLL1 General Function Control Register (CCM_ANALOG_AUDIO_PLL1_GEN_CTRL)

AUDIO PLL1 General Function Control Register

Address: 3036_0000h base + 0h offset = 3036_0000h



CCM_ANALOG_AUDIO_PLL1_GEN_CTRL field descriptions

Field	Description
31 PLL_LOCK	PLL lock signal

Table continues on the next page...

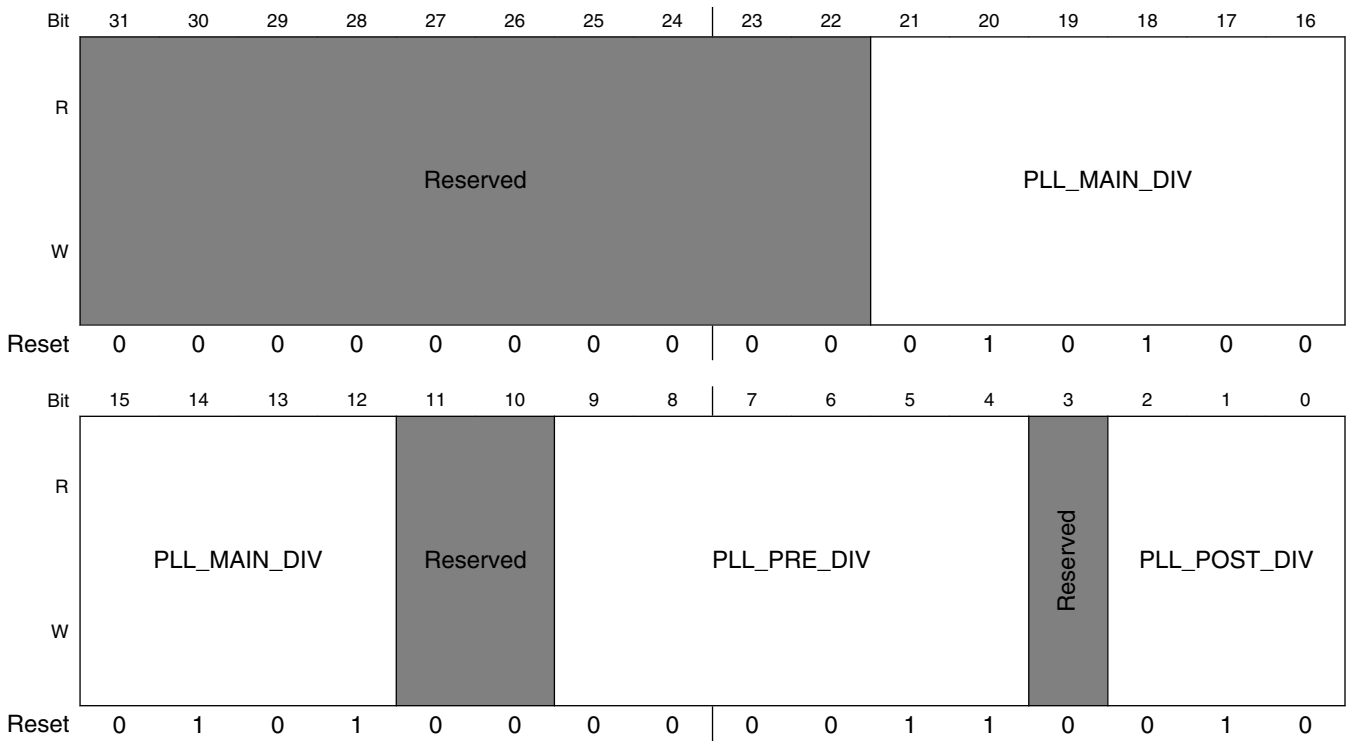
CCM_ANALOG_AUDIO_PLL1_GEN_CTRL field descriptions (continued)

Field	Description
30–17 -	This field is reserved. Reserved
16 PLL_EXT_ BYPASS	PLL analog block bypass, clock output traces to PLL source
15–14 -	This field is reserved. Reserved
13 PLL_CLKE	PLL output clock clock gating enable
12 PLL_CLKE_ OVERRIDE	Override the PLL_CLKE, clock gating enable signal from CCM
11–10 -	This field is reserved. Reserved
9 PLL_RST	PLL reset (active low)
8 PLL_RST_ OVERRIDE	PLL reset overrided by CCM
7–5 -	This field is reserved. Reserved
4 PLL_BYPASS	PLL output clock bypass
3–2 PAD_CLK_SEL	PAD clock select, the output clock is PAD_CLK, PLL reference clock option 00 CLKIN1 XOR CLKIN2 01 CLKIN2 10 CLKIN1 11 Reserved
PLL_REF_CLK_ SEL	PLL reference clock select 00 SYS_XTAL 01 PAD_CLK 10 Reserved 11 Reserved

5.1.8.2 AUDIO PLL1 Divide and Fraction Data Control 0 Register (CCM_ANALOG_AUDIO_PLL1_FDIV_CTL0)

AUDIO PLL1 Divide and Fraction Data Control 0 Register

Address: 3036_0000h base + 4h offset = 3036_0004h



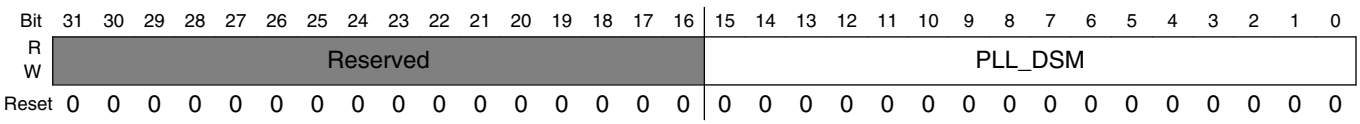
CCM_ANALOG_AUDIO_PLL1_FDIV_CTL0 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–12 PLL_MAIN_DIV	Value of the main-divider
11–10 -	This field is reserved. Reserved
9–4 PLL_PRE_DIV	Value of the pre-divider
3 -	This field is reserved. Reserved
PLL_POST_DIV	Value of the post-divider

5.1.8.3 AUDIO PLL1 Divide and Fraction Data Control 1 Register (CCM_ANALOG_AUDIO_PLL1_FDIV_CTL1)

AUDIO PLL1 Divide and Fraction Data Control 1 Register

Address: 3036_0000h base + 8h offset = 3036_0008h



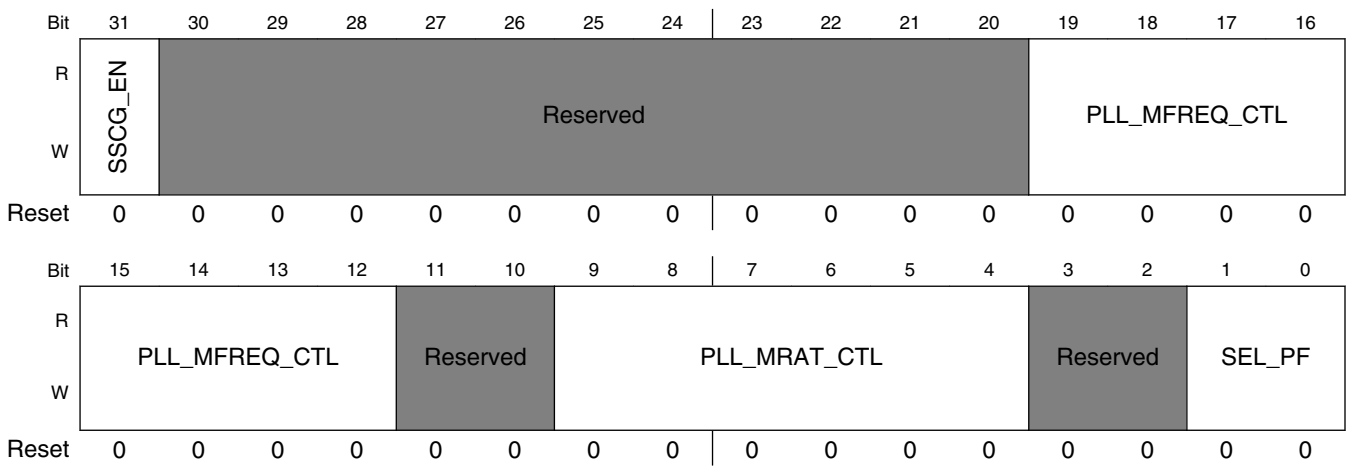
CCM_ANALOG_AUDIO_PLL1_FDIV_CTL1 field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
PLL_DSM	Value of the DSM

5.1.8.4 AUDIO PLL1 PLL SSCG Control Register (CCM_ANALOG_AUDIO_PLL1_SSCG_CTRL)

AUDIO PLL1 PLL SSCG Control Register

Address: 3036_0000h base + Ch offset = 3036_000Ch



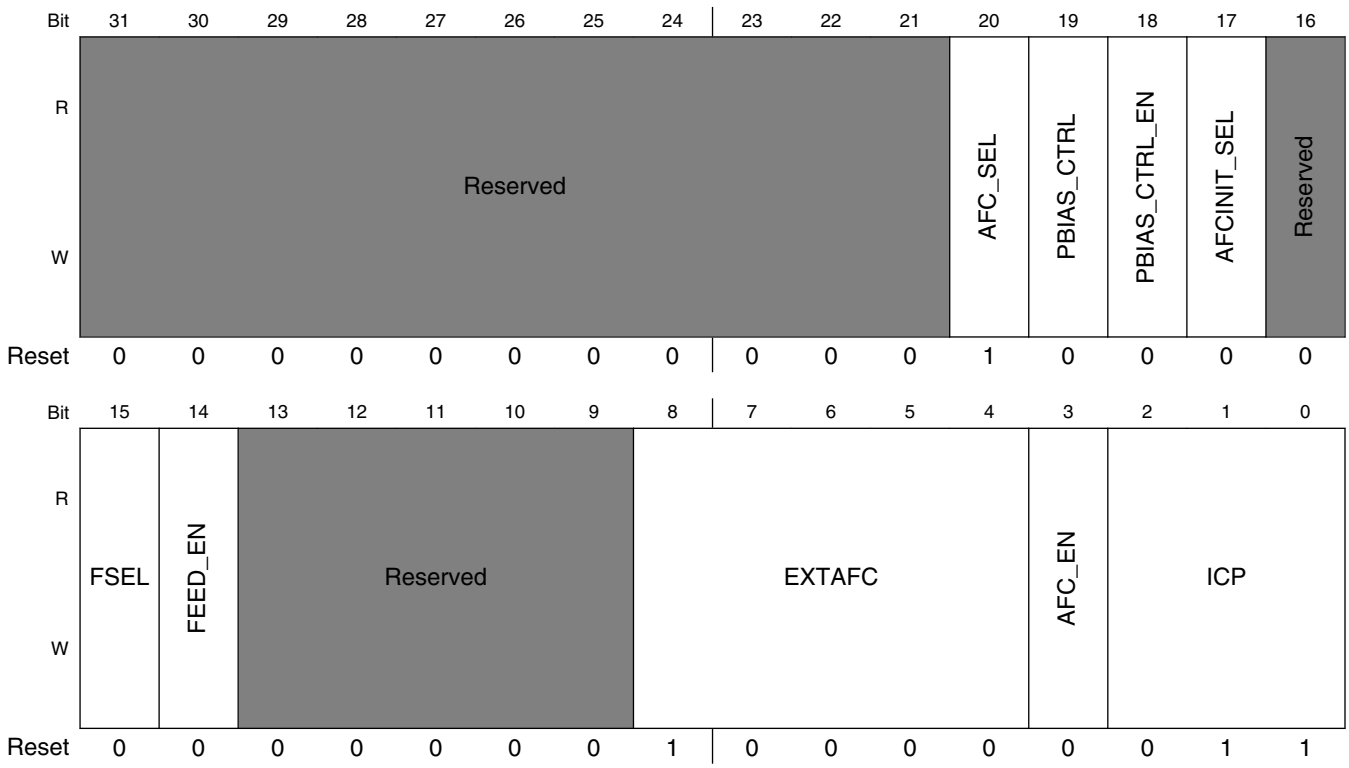
CCM_ANALOG_AUDIO_PLL1_SSCG_CTRL field descriptions

Field	Description
31 SSCG_EN	SSCG Enable
30–20 -	This field is reserved. Reserved
19–12 PLL_MFREQ_CTL	Value of modulation frequency control
11–10 -	This field is reserved. Reserved
9–4 PLL_MRAT_CTL	Value of modulation rate control
3–2 -	This field is reserved. Reserved
SEL_PF	Value of modulation method control 00 Down spread 01 Up spread 1x Center spread

5.1.8.5 AUDIO PLL1 PLL Monitoring Control Register (CCM_ANALOG_AUDIO_PLL1_MNIT_CTRL)

AUDIO PLL1 PLL Monitoring Control Register

Address: 3036_0000h base + 10h offset = 3036_0010h



CCM_ANALOG_AUDIO_PLL1_MNIT_CTRL field descriptions

Field	Description
31–21 -	This field is reserved. Reserved
20 AFC_SEL	AFC Mode select
19 PBIAS_CTRL	PBIAS pull-down initial voltage control pin 0 0.50*VDD 1 0.67*VDD
18 PBIAS_CTRL_EN	PBIAS voltage pull-down enable pin
17 AFCINIT_SEL	AFC initial delay select pin

Table continues on the next page...

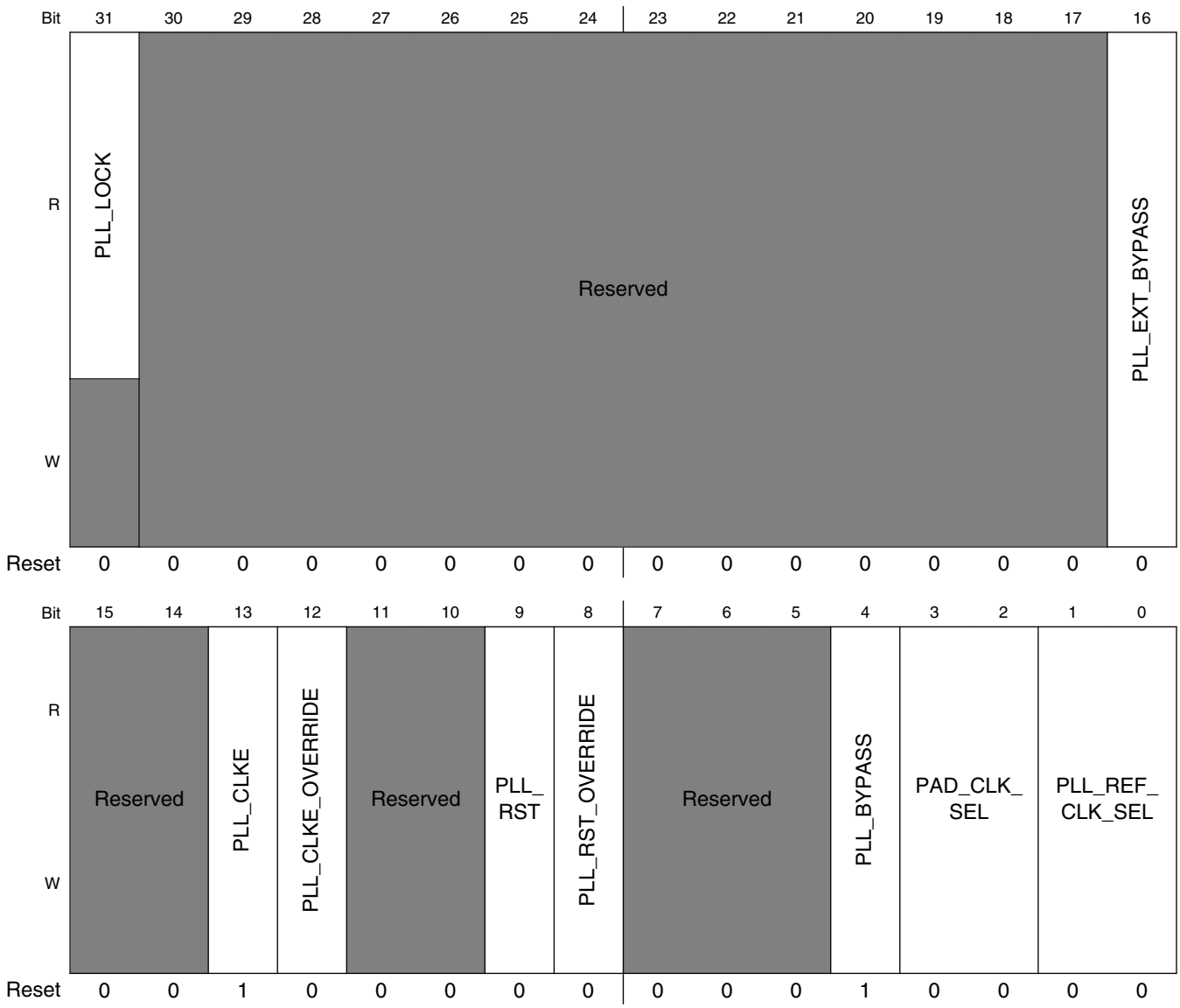
CCM_ANALOG_AUDIO_PLL1_MNIT_CTRL field descriptions (continued)

Field	Description
	0 nominal delay 1 nominal delay * 2
16 -	This field is reserved. Reserved
15 FSEL	Monitoring frequency select pin 0 FEED_OUT = FREF 1 FEED_OUT = FEED
14 FEED_EN	FEED_OUT enable pin
13–9 -	This field is reserved. Reserved
8–4 EXTAFC	Monitoring pin. If AFC_ENB=1, AFC is disabled and VCO is calibrated manually by EXTAFC[4:0] for the test of VCO range
3 AFC_EN	If AFC_ENB=0, AFC is enabled and VCO is calibrated automatically.(AFC_ENB=0 and EXTAFC=0 are mandatory) If AFC_ENB=1, AFC is disabled and VCO is calibrated manually by EXTAFC[4:0] for the test of VCO range
ICP	Controls the charge-pump current

5.1.8.6 AUDIO PLL2 General Function Control Register (CCM_ANALOG_AUDIO_PLL2_GEN_CTRL)

AUDIO PLL2 General Function Control Register

Address: 3036_0000h base + 14h offset = 3036_0014h



CCM_ANALOG_AUDIO_PLL2_GEN_CTRL field descriptions

Field	Description
31 PLL_LOCK	PLL lock signal

Table continues on the next page...

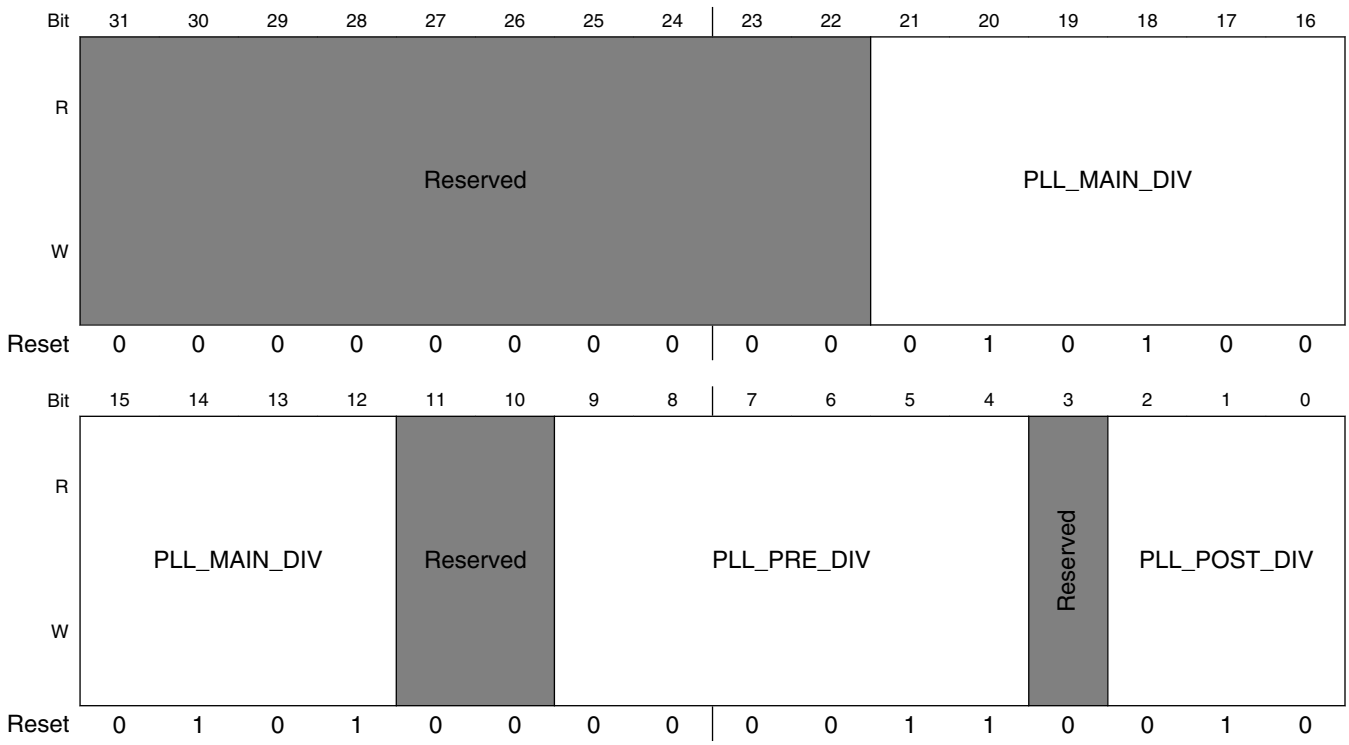
CCM_ANALOG_AUDIO_PLL2_GEN_CTRL field descriptions (continued)

Field	Description
30–17 -	This field is reserved. Reserved
16 PLL_EXT_ BYPASS	PLL analog block bypass, clock output traces to PLL source
15–14 -	This field is reserved. Reserved
13 PLL_CLKE	PLL output clock clock gating enable
12 PLL_CLKE_ OVERRIDE	Override the PLL_CLKE, clock gating enable signal from CCM
11–10 -	This field is reserved. Reserved
9 PLL_RST	PLL reset (active low)
8 PLL_RST_ OVERRIDE	PLL reset overrided by CCM
7–5 -	This field is reserved. Reserved
4 PLL_BYPASS	PLL output clock bypass
3–2 PAD_CLK_SEL	PAD clock select, the output clock is PAD_CLK, PLL reference clock option 00 CLKIN1 XOR CLKIN2 01 CLKIN2 10 CLKIN1 11 Reserved
PLL_REF_CLK_ SEL	PLL reference clock select 00 SYS_XTAL 01 PAD_CLK 10 Reserved 11 Reserved

5.1.8.7 AUDIO PLL2 Divide and Fraction Data Control 0 Register (CCM_ANALOG_AUDIO_PLL2_FDIV_CTL0)

AUDIO PLL2 Divide and Fraction Data Control 0 Register

Address: 3036_0000h base + 18h offset = 3036_0018h



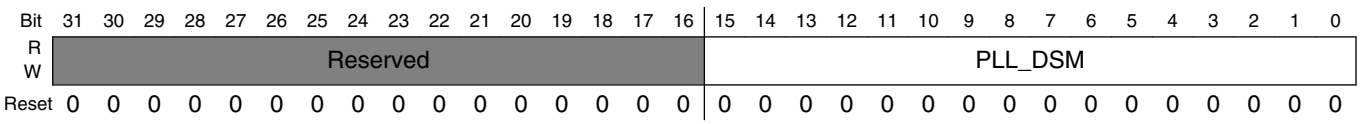
CCM_ANALOG_AUDIO_PLL2_FDIV_CTL0 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–12 PLL_MAIN_DIV	Value of the main-divider
11–10 -	This field is reserved. Reserved
9–4 PLL_PRE_DIV	Value of the pre-divider
3 -	This field is reserved. Reserved
PLL_POST_DIV	Value of the post-divider

5.1.8.8 AUDIO PLL2 Divide and Fraction Data Control 1 Register (CCM_ANALOG_AUDIO_PLL2_FDIV_CTL1)

AUDIO PLL2 Divide and Fraction Data Control 1 Register

Address: 3036_0000h base + 1Ch offset = 3036_001Ch



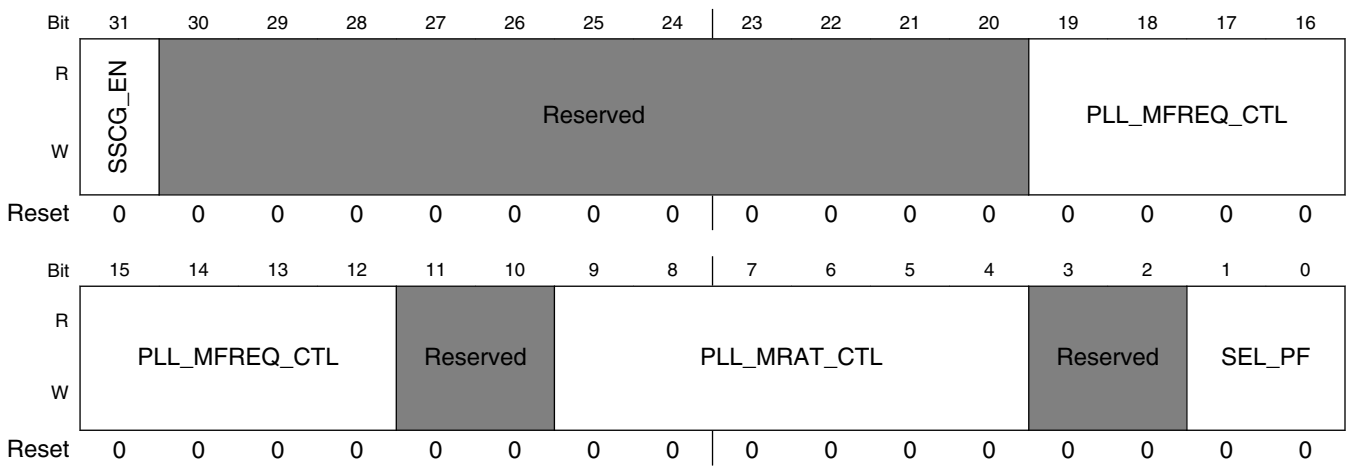
CCM_ANALOG_AUDIO_PLL2_FDIV_CTL1 field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
PLL_DSM	Value of the DSM

5.1.8.9 AUDIO PLL2 PLL SSCG Control Register (CCM_ANALOG_AUDIO_PLL2_SSCG_CTRL)

AUDIO PLL2 PLL SSCG Control Register

Address: 3036_0000h base + 20h offset = 3036_0020h



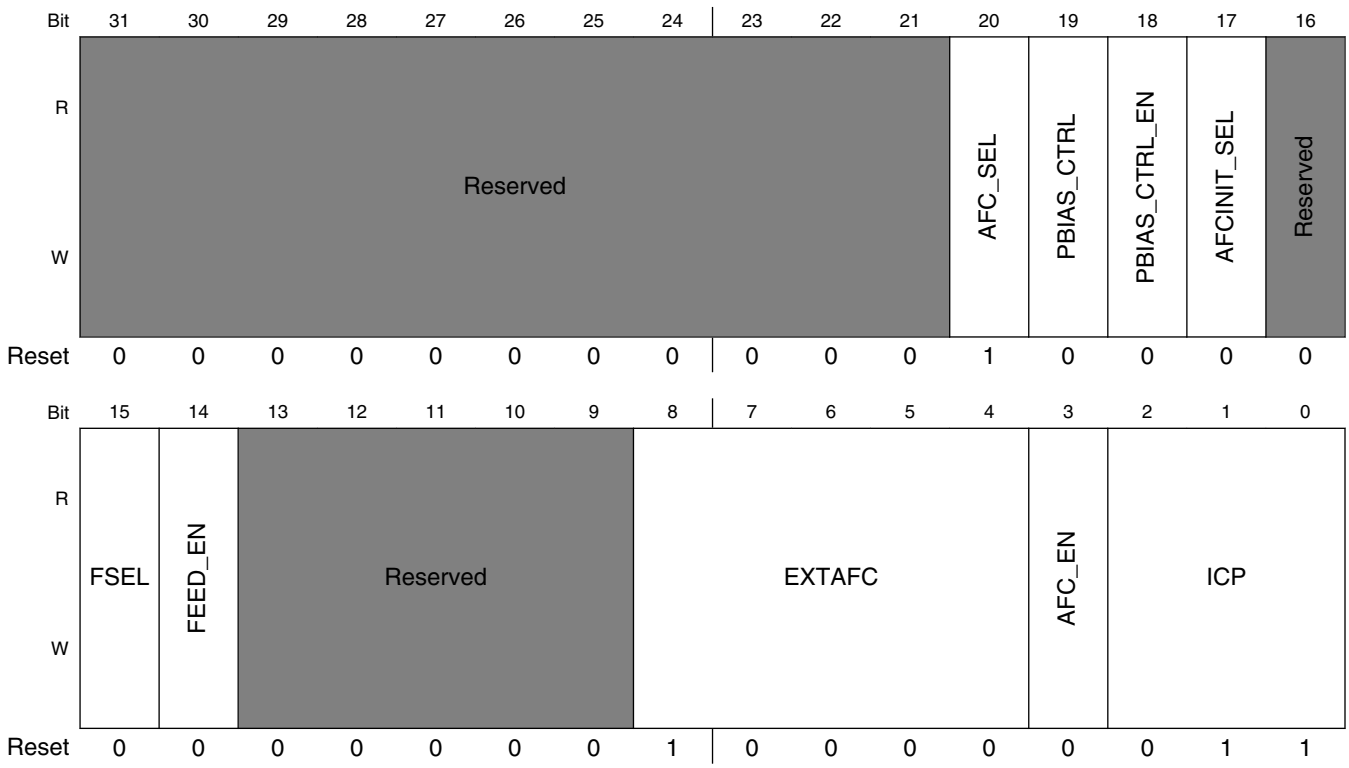
CCM_ANALOG_AUDIO_PLL2_SSCG_CTRL field descriptions

Field	Description
31 SSCG_EN	SSCG Enable
30–20 -	This field is reserved. Reserved
19–12 PLL_MFREQ_CTL	Value of modulation frequency control
11–10 -	This field is reserved. Reserved
9–4 PLL_MRAT_CTL	Value of modulation rate control
3–2 -	This field is reserved. Reserved
SEL_PF	Value of modulation method control 00 Down spread 01 Up spread 1x Center spread

5.1.8.10 AUDIO PLL2 PLL Monitoring Control Register (CCM_ANALOG_AUDIO_PLL2_MNIT_CTRL)

AUDIO PLL2 PLL Monitoring Control Register

Address: 3036_0000h base + 24h offset = 3036_0024h



CCM_ANALOG_AUDIO_PLL2_MNIT_CTRL field descriptions

Field	Description
31–21 -	This field is reserved. Reserved
20 AFC_SEL	AFC Mode select
19 PBIAS_CTRL	PBIAS pull-down initial voltage control pin 0 0.50*VDD 1 0.67*VDD
18 PBIAS_CTRL_EN	PBIAS voltage pull-down enable pin
17 AFCINIT_SEL	AFC initial delay select pin

Table continues on the next page...

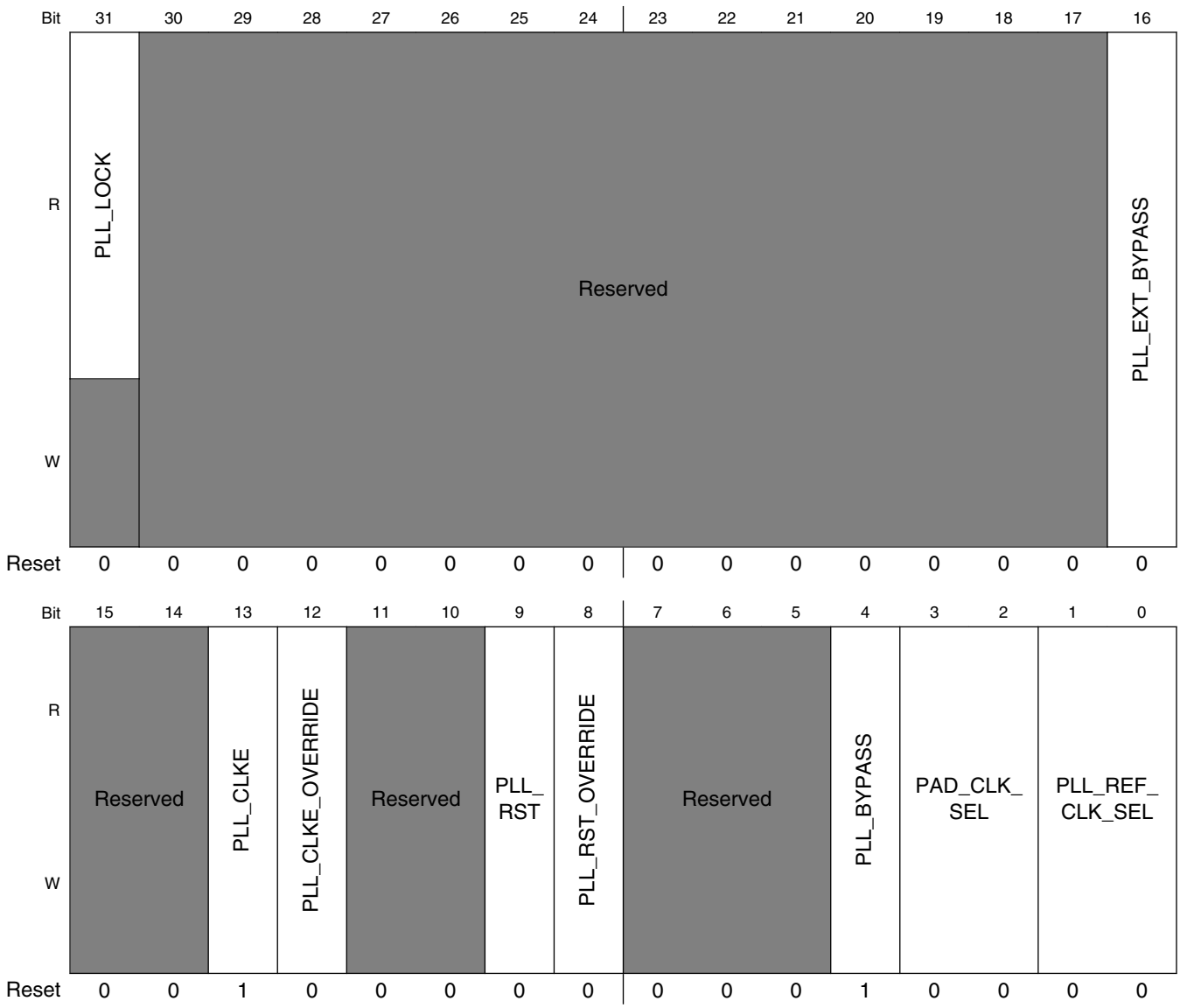
CCM_ANALOG_AUDIO_PLL2_MNIT_CTRL field descriptions (continued)

Field	Description
	0 nominal delay 1 nominal delay * 2
16 -	This field is reserved. Reserved
15 FSEL	Monitoring frequency select pin 0 FEED_OUT = FREF 1 FEED_OUT = FEED
14 FEED_EN	FEED_OUT enable pin
13–9 -	This field is reserved. Reserved
8–4 EXTAFC	Monitoring pin. If AFC_ENB=1, AFC is disabled and VCO is calibrated manually by EXTAFC[4:0] for the test of VCO range
3 AFC_EN	If AFC_ENB=0, AFC is enabled and VCO is calibrated automatically.(AFC_ENB=0 and EXTAFC=0 are mandatory) If AFC_ENB=1, AFC is disabled and VCO is calibrated manually by EXTAFC[4:0] for the test of VCO range
ICP	Controls the charge-pump current

5.1.8.11 VIDEO PLL1 General Function Control Register (CCM_ANALOG_VIDEO_PLL1_GEN_CTRL)

VIDEO PLL1 General Function Control Register

Address: 3036_0000h base + 28h offset = 3036_0028h



CCM_ANALOG_VIDEO_PLL1_GEN_CTRL field descriptions

Field	Description
31 PLL_LOCK	PLL lock signal

Table continues on the next page...

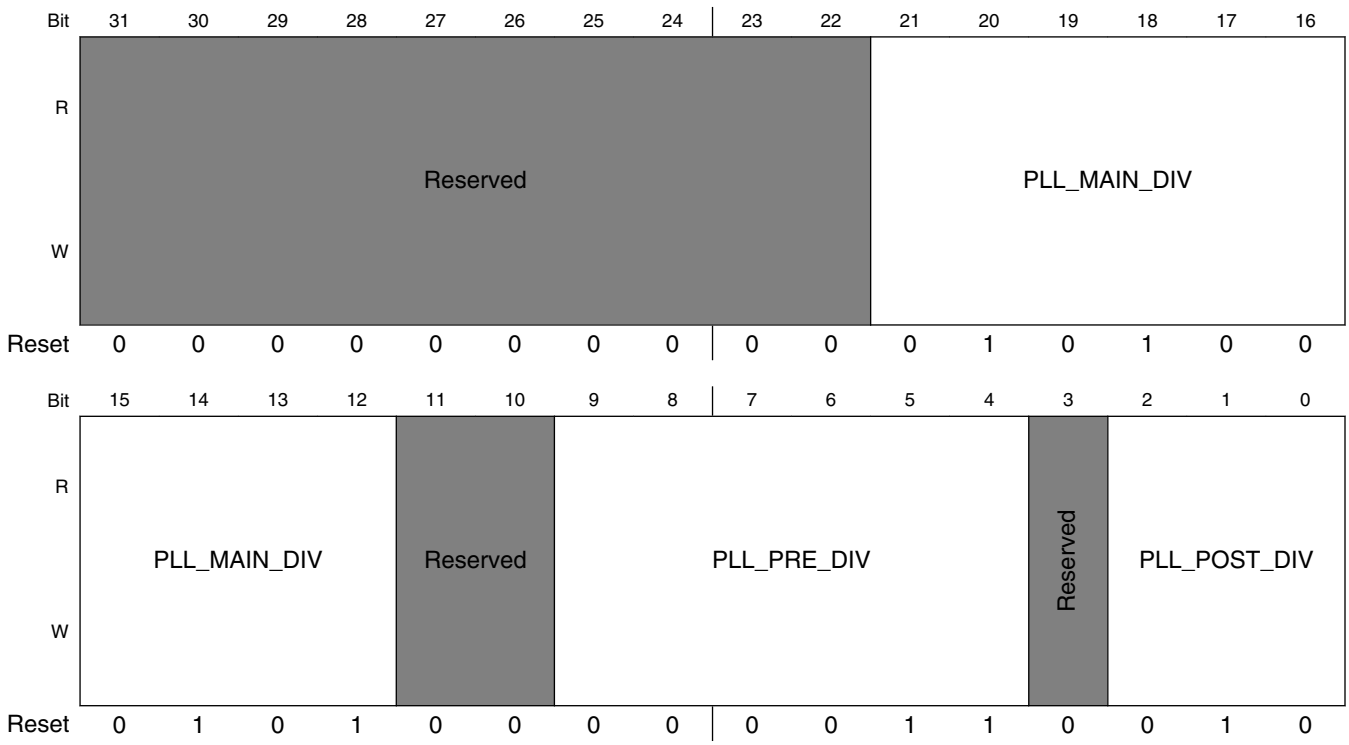
CCM_ANALOG_VIDEO_PLL1_GEN_CTRL field descriptions (continued)

Field	Description
30–17 -	This field is reserved. Reserved
16 PLL_EXT_ BYPASS	PLL analog block bypass, clock output traces to PLL source
15–14 -	This field is reserved. Reserved
13 PLL_CLKE	PLL output clock clock gating enable
12 PLL_CLKE_ OVERRIDE	Override the PLL_CLKE, clock gating enable signal from CCM
11–10 -	This field is reserved. Reserved
9 PLL_RST	PLL reset (active low)
8 PLL_RST_ OVERRIDE	PLL reset overrided by CCM
7–5 -	This field is reserved. Reserved
4 PLL_BYPASS	PLL output clock bypass
3–2 PAD_CLK_SEL	PAD clock select, the output clock is PAD_CLK, PLL reference clock option 00 CLKIN1 XOR CLKIN2 01 CLKIN2 10 CLKIN1 11 Reserved
PLL_REF_CLK_ SEL	PLL reference clock select 00 SYS_XTAL 01 PAD_CLK 10 Reserved 11 Reserved

5.1.8.12 VIDEO PLL1 Divide and Fraction Data Control 0 Register (CCM_ANALOG_VIDEO_PLL1_FDIV_CTL0)

VIDEO PLL1 Divide and Fraction Data Control 0 Register

Address: 3036_0000h base + 2Ch offset = 3036_002Ch



CCM_ANALOG_VIDEO_PLL1_FDIV_CTL0 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–12 PLL_MAIN_DIV	Value of the main-divider
11–10 -	This field is reserved. Reserved
9–4 PLL_PRE_DIV	Value of the pre-divider
3 -	This field is reserved. Reserved
PLL_POST_DIV	Value of the post-divider

5.1.8.13 VIDEO PLL1 Divide and Fraction Data Control 1 Register (CCM_ANALOG_VIDEO_PLL1_FDIV_CTL1)

VIDEO PLL1 Divide and Fraction Data Control 1 Register

Address: 3036_0000h base + 30h offset = 3036_0030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																PLL_DSM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CCM_ANALOG_VIDEO_PLL1_FDIV_CTL1 field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
PLL_DSM	Value of the DSM

5.1.8.14 VIDEO PLL1 PLL SSCG Control Register (CCM_ANALOG_VIDEO_PLL1_SSCG_CTRL)

VIDEO PLL1 PLL SSCG Control Register

Address: 3036_0000h base + 34h offset = 3036_0034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SSCG_EN	Reserved											PLL_MFREQ_CTL			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PLL_MFREQ_CTL				Reserved		PLL_MRAT_CTL						Reserved		SEL_PF	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

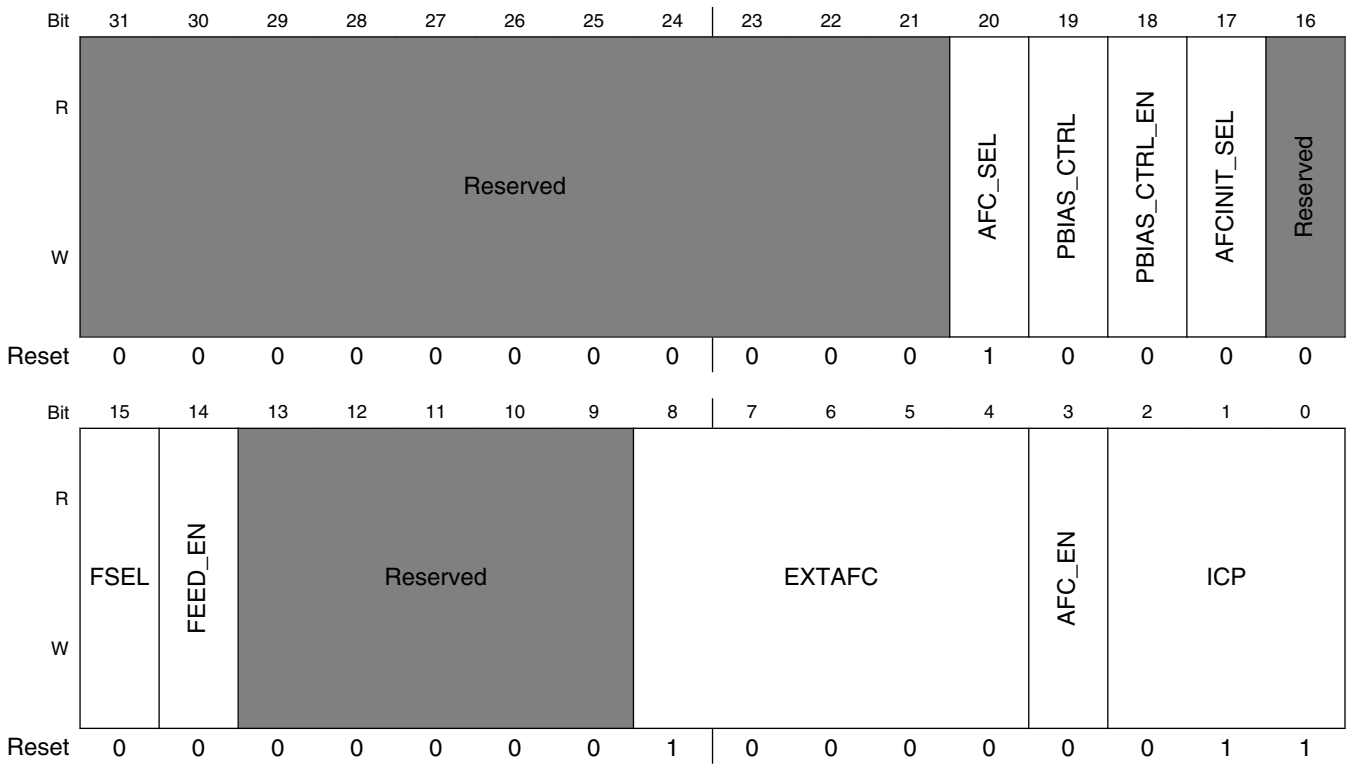
CCM_ANALOG_VIDEO_PLL1_SSCG_CTRL field descriptions

Field	Description
31 SSCG_EN	SSCG Enable
30–20 -	This field is reserved. Reserved
19–12 PLL_MFREQ_CTL	Value of modulation frequency control
11–10 -	This field is reserved. Reserved
9–4 PLL_MRAT_CTL	Value of modulation rate control
3–2 -	This field is reserved. Reserved
SEL_PF	Value of modulation method control 00 Down spread 01 Up spread 1x Center spread

5.1.8.15 VIDEO PLL1 PLL Monitoring Control Register (CCM_ANALOG_VIDEO_PLL1_MNIT_CTRL)

VIDEO PLL1 PLL Monitoring Control Register

Address: 3036_0000h base + 38h offset = 3036_0038h



CCM_ANALOG_VIDEO_PLL1_MNIT_CTRL field descriptions

Field	Description
31–21 -	This field is reserved. Reserved
20 AFC_SEL	AFC Mode select
19 PBIAS_CTRL	PBIAS pull-down initial voltage control pin 0 0.50*VDD 1 0.67*VDD
18 PBIAS_CTRL_EN	PBIAS voltage pull-down enable pin
17 AFCINIT_SEL	AFC initial delay select pin

Table continues on the next page...

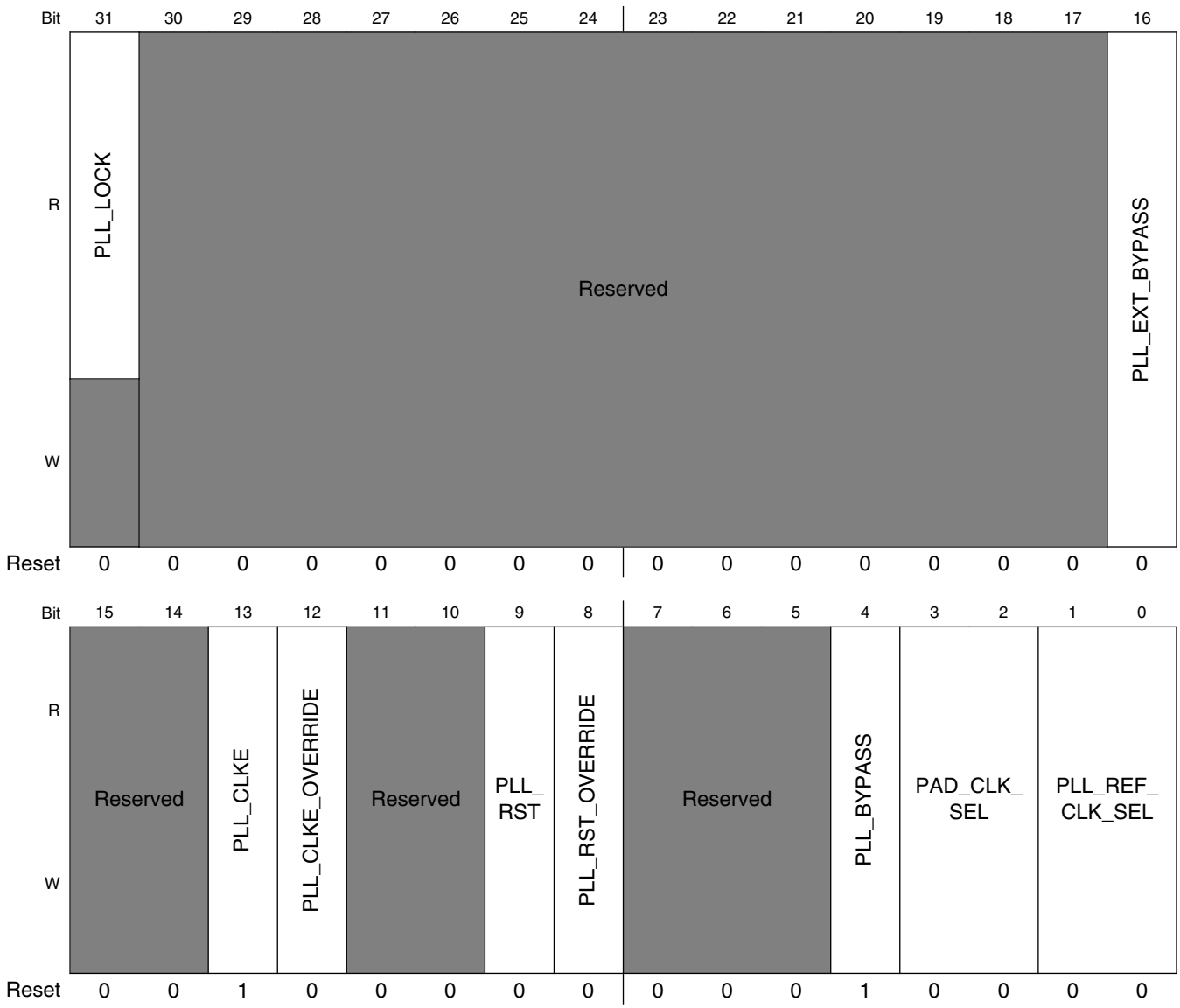
CCM_ANALOG_VIDEO_PLL1_MNIT_CTRL field descriptions (continued)

Field	Description
	0 nominal delay 1 nominal delay * 2
16 -	This field is reserved. Reserved
15 FSEL	Monitoring frequency select pin 0 FEED_OUT = FREF 1 FEED_OUT = FEED
14 FEED_EN	FEED_OUT enable pin
13–9 -	This field is reserved. Reserved
8–4 EXTAFC	Monitoring pin. If AFC_ENB=1, AFC is disabled and VCO is calibrated manually by EXTAFC[4:0] for the test of VCO range
3 AFC_EN	If AFC_ENB=0, AFC is enabled and VCO is calibrated automatically.(AFC_ENB=0 and EXTAFC=0 are mandatory) If AFC_ENB=1, AFC is disabled and VCO is calibrated manually by EXTAFC[4:0] for the test of VCO range
ICP	Controls the charge-pump current

5.1.8.16 DRAM PLL General Function Control Register (CCM_ANALOG_DRAM_PLL_GEN_CTRL)

DRAM PLL General Function Control Register

Address: 3036_0000h base + 50h offset = 3036_0050h



CCM_ANALOG_DRAM_PLL_GEN_CTRL field descriptions

Field	Description
31 PLL_LOCK	PLL lock signal

Table continues on the next page...

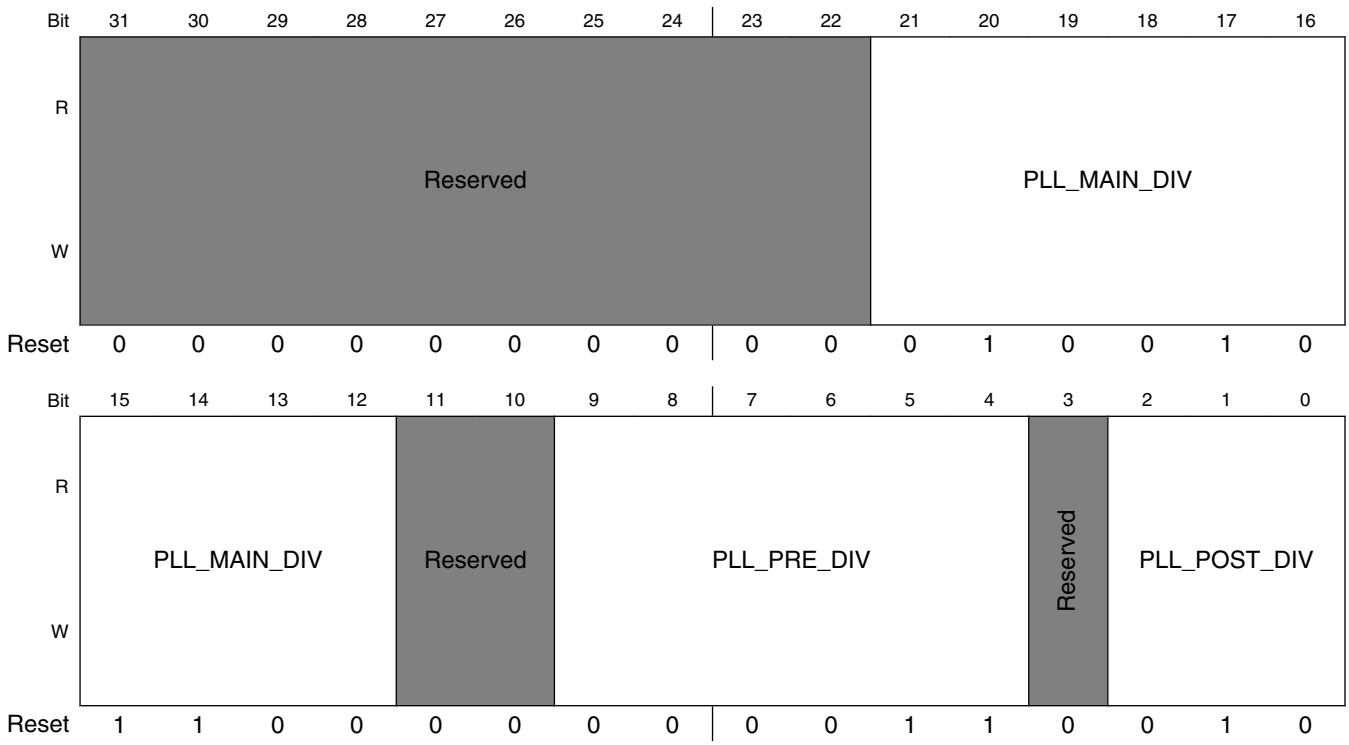
CCM_ANALOG_DRAM_PLL_GEN_CTRL field descriptions (continued)

Field	Description
30–17 -	This field is reserved. Reserved
16 PLL_EXT_ BYPASS	PLL analog block bypass, clock output traces to PLL source
15–14 -	This field is reserved. Reserved
13 PLL_CLKE	PLL output clock clock gating enable
12 PLL_CLKE_ OVERRIDE	Override the PLL_CLKE, clock gating enable signal from CCM
11–10 -	This field is reserved. Reserved
9 PLL_RST	PLL reset (active low)
8 PLL_RST_ OVERRIDE	PLL reset overrided by CCM
7–5 -	This field is reserved. Reserved
4 PLL_BYPASS	PLL output clock bypass
3–2 PAD_CLK_SEL	PAD clock select, the output clock is PAD_CLK, PLL reference clock option 00 CLKIN1 XOR CLKIN2 01 CLKIN2 10 CLKIN1 11 Reserved
PLL_REF_CLK_ SEL	PLL reference clock select 00 SYS_XTAL 01 PAD_CLK 10 Reserved 11 Reserved

5.1.8.17 DRAM PLL Divide and Fraction Data Control 0 Register (CCM_ANALOG_DRAM_PLL_FDIV_CTL0)

DRAM PLL Divide and Fraction Data Control 0 Register

Address: 3036_0000h base + 54h offset = 3036_0054h



CCM_ANALOG_DRAM_PLL_FDIV_CTL0 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–12 PLL_MAIN_DIV	Value of the main-divider
11–10 -	This field is reserved. Reserved
9–4 PLL_PRE_DIV	Value of the pre-divider
3 -	This field is reserved. Reserved
PLL_POST_DIV	Value of the post-divider

5.1.8.18 DRAM PLL Divide and Fraction Data Control 1 Register (CCM_ANALOG_DRAM_PLL_FDIV_CTL1)

DRAM PLL Divide and Fraction Data Control 1 Register

Address: 3036_0000h base + 58h offset = 3036_0058h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																PLL_DSM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CCM_ANALOG_DRAM_PLL_FDIV_CTL1 field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
PLL_DSM	Value of the DSM

5.1.8.19 DRAM PLL PLL SSCG Control Register (CCM_ANALOG_DRAM_PLL_SSCG_CTRL)

DRAM PLL PLL SSCG Control Register

Address: 3036_0000h base + 5Ch offset = 3036_005Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SSCG_EN	Reserved											PLL_MFREQ_CTL			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PLL_MFREQ_CTL				Reserved		PLL_MRAT_CTL						Reserved		SEL_PF	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

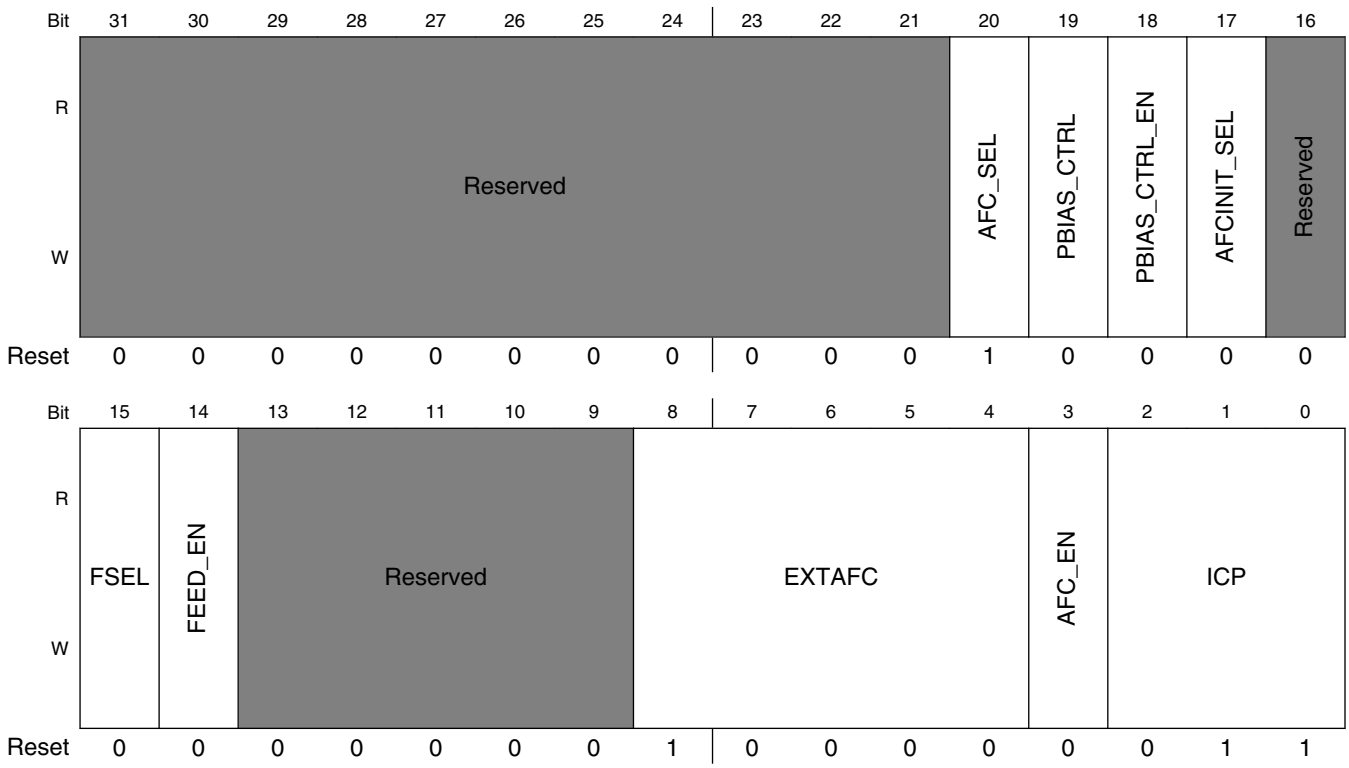
CCM_ANALOG_DRAM_PLL_SSCG_CTRL field descriptions

Field	Description
31 SSCG_EN	SSCG Enable
30–20 -	This field is reserved. Reserved
19–12 PLL_MFREQ_CTL	Value of modulation frequency control
11–10 -	This field is reserved. Reserved
9–4 PLL_MRAT_CTL	Value of modulation rate control
3–2 -	This field is reserved. Reserved
SEL_PF	Value of modulation method control 00 Down spread 01 Up spread 1x Center spread

5.1.8.20 DRAM PLL PLL Monitoring Control Register (CCM_ANALOG_DRAM_PLL_MNIT_CTRL)

DRAM PLL PLL Monitoring Control Register

Address: 3036_0000h base + 60h offset = 3036_0060h



CCM_ANALOG_DRAM_PLL_MNIT_CTRL field descriptions

Field	Description
31–21 -	This field is reserved. Reserved
20 AFC_SEL	AFC Mode select
19 PBIAS_CTRL	PBIAS pull-down initial voltage control pin 0 0.50*VDD 1 0.67*VDD
18 PBIAS_CTRL_EN	PBIAS voltage pull-down enable pin
17 AFCINIT_SEL	AFC initial delay select pin

Table continues on the next page...

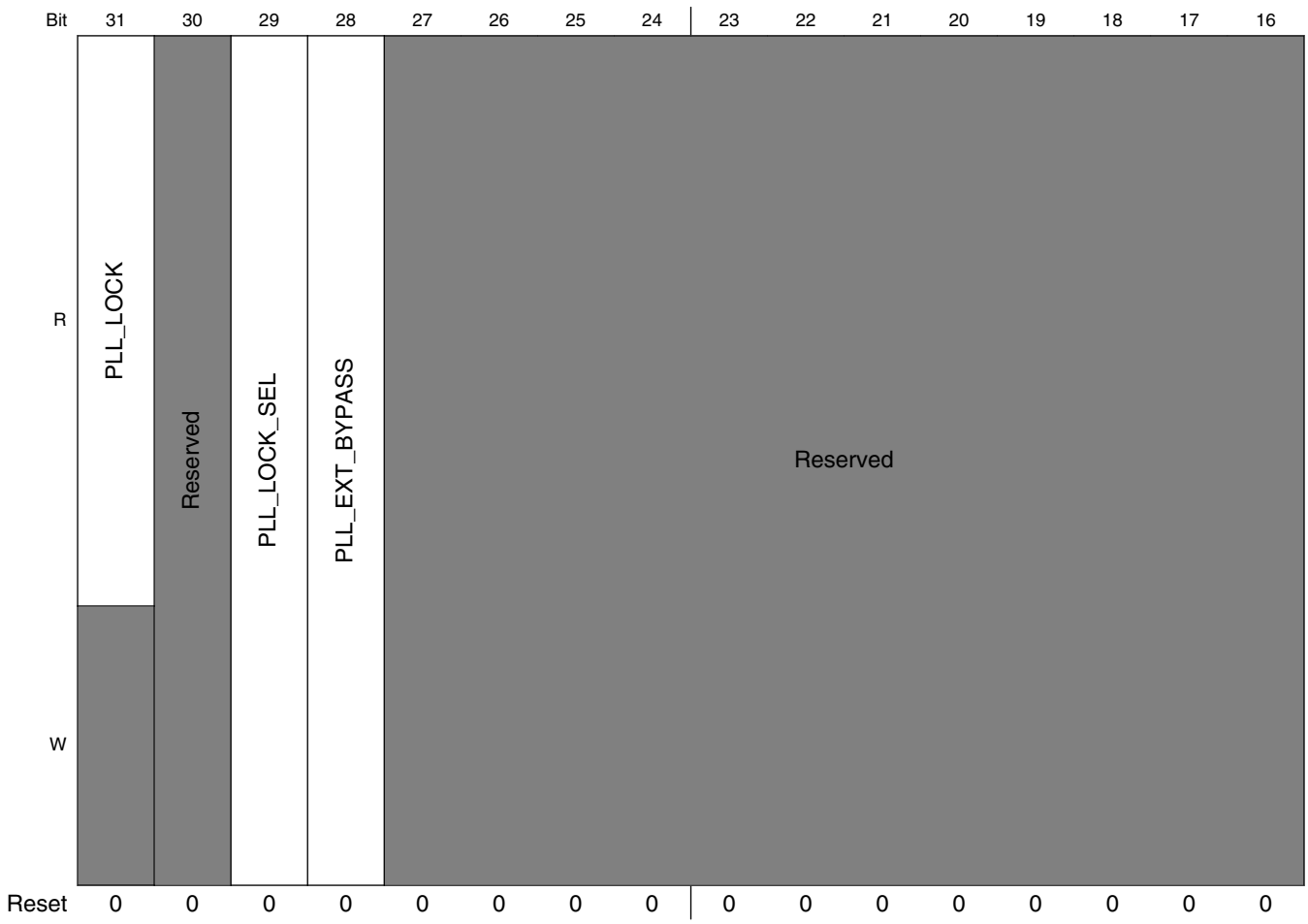
CCM_ANALOG_DRAM_PLL_MNIT_CTRL field descriptions (continued)

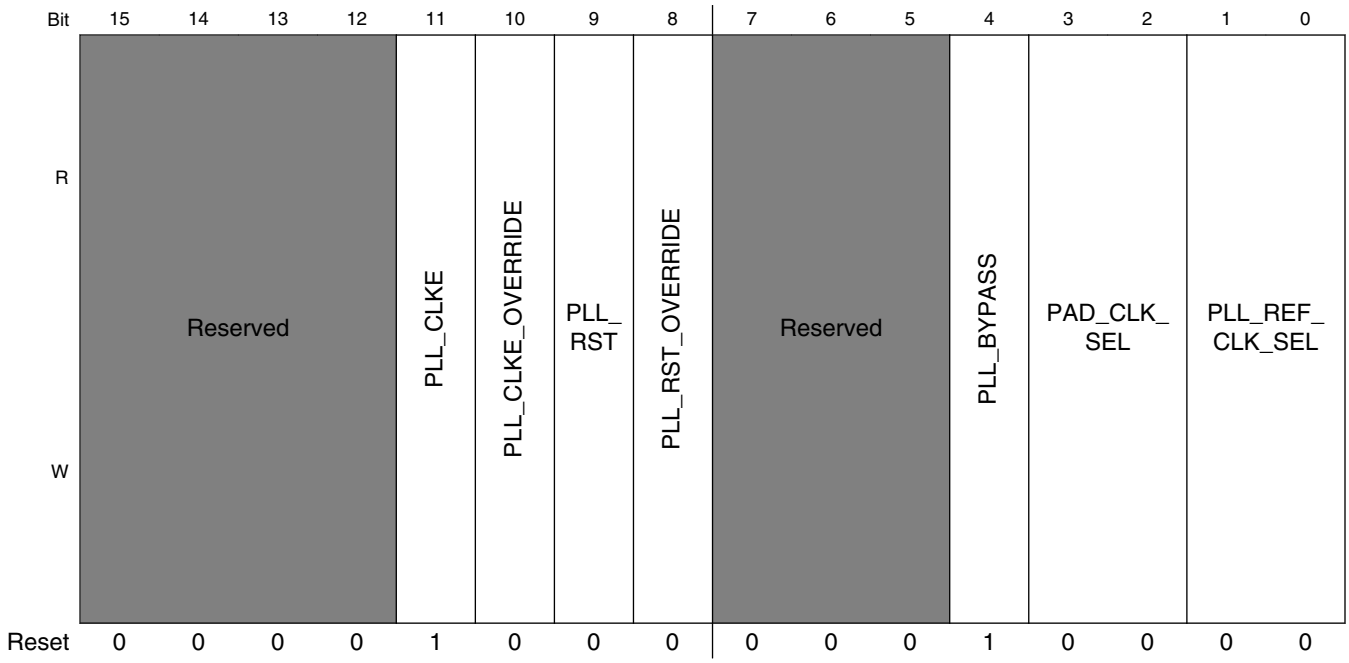
Field	Description
	0 nominal delay 1 nominal delay * 2
16 -	This field is reserved. Reserved
15 FSEL	Monitoring frequency select pin 0 FEED_OUT = FREF 1 FEED_OUT = FEED
14 FEED_EN	FEED_OUT enable pin
13–9 -	This field is reserved. Reserved
8–4 EXTAFC	Monitoring pin. If AFC_ENB=1, AFC is disabled and VCO is calibrated manually by EXTAFC[4:0] for the test of VCO range
3 AFC_EN	If AFC_ENB=0, AFC is enabled and VCO is calibrated automatically.(AFC_ENB=0 and EXTAFC=0 are mandatory) If AFC_ENB=1, AFC is disabled and VCO is calibrated manually by EXTAFC[4:0] for the test of VCO range
ICP	Controls the charge-pump current

5.1.8.21 GPU PLL General Function Control Register (CCM_ANALOG_GPU_PLL_GEN_CTRL)

GPU PLL General Function Control Register

Address: 3036_0000h base + 64h offset = 3036_0064h





CCM_ANALOG_GPU_PLL_GEN_CTRL field descriptions

Field	Description
31 PLL_LOCK	PLL lock signal
30 -	This field is reserved. Reserved
29 PLL_LOCK_SEL	PLL lock select 0 Using PLL maximum lock time 1 Using PLL output lock
28 PLL_EXT_BYPASS	PLL analog block bypass, clock output traces to PLL source
27–12 -	This field is reserved. Reserved
11 PLL_CLKE	PLL output clock clock gating enable
10 PLL_CLKE_OVERRIDE	Override the PLL_CLKE, clock gating enable signal from CCM
9 PLL_RST	PLL reset (active low)
8 PLL_RST_OVERRIDE	PLL reset overrided by CCM
7–5 -	This field is reserved. Reserved
4 PLL_BYPASS	PLL output clock bypass

Table continues on the next page...

CCM_ANALOG_GPU_PLL_GEN_CTRL field descriptions (continued)

Field	Description
3–2 PAD_CLK_SEL	PAD clock select, the output clock is PAD_CLK, PLL reference clock option 00 CLKIN1 XOR CLKIN2 01 CLKIN2 10 CLKIN1 11 Reserved
PLL_REF_CLK_SEL	PLL reference clock select 00 SYS_XTAL 01 PAD_CLK 10 Reserved 11 Reserved

5.1.8.22 GPU PLL Divide and Fraction Data Control 0 Register (CCM_ANALOG_GPU_PLL_FDIV_CTL0)

GPU PLL Divide and Fraction Data Control 0 Register

Address: 3036_0000h base + 68h offset = 3036_0068h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved										PLL_MAIN_DIV					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PLL_MAIN_DIV				Reserved		PLL_PRE_DIV						Reserved	PLL_POST_DIV		
Reset	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1

CCM_ANALOG_GPU_PLL_FDIV_CTL0 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–12 PLL_MAIN_DIV	Value of the main-divider
11–10 -	This field is reserved. Reserved
9–4 PLL_PRE_DIV	Value of the pre-divider
3 -	This field is reserved. Reserved
PLL_POST_DIV	Value of the post-divider

5.1.8.23 PLL Lock Detector Control Register (CCM_ANALOG_*n*LOCKD_CTRL)

PLL Lock Detector Control Register

Address: 3036_0000h base + 6Ch offset + (16d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																										LOCK_CON_DLY		LOCK_CON_OUT		LOCK_CON_IN	
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

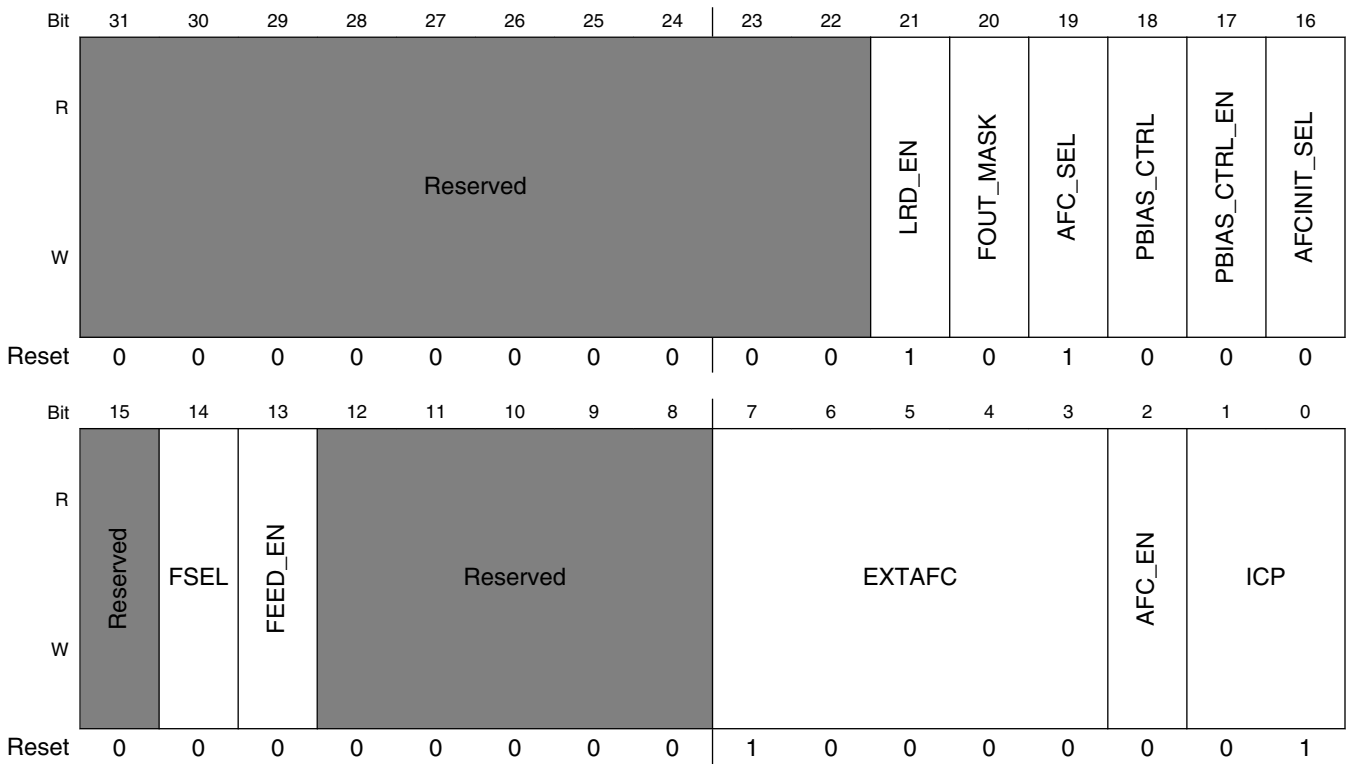
CCM_ANALOG_*n*LOCKD_CTRL field descriptions

Field	Description
31–6 -	This field is reserved. Reserved
5–4 LOCK_CON_DLY	Lock detector setting of the detection resolution
3–2 LOCK_CON_OUT	Lock detector setting of the output margin
LOCK_CON_IN	Lock detector setting of the input margin

5.1.8.24 PLL Monitoring Control Register (CCM_ANALOG_*n*MNIT_CTRL)

PLL Monitoring Control Register

Address: 3036_0000h base + 70h offset + (16d × i), where i=0d to 2d



CCM_ANALOG_*n*MNIT_CTRL field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21 LRD_EN	Monitoring pin. AFC operation mode select pin
20 FOUT_MASK	Scaler's re-initialization time control pin[3]
19 AFC_SEL	AFC Mode select
18 PBIAS_CTRL	PBIAS pull-down initial voltage control pin 0 0.50*VDD 1 0.67*VDD

Table continues on the next page...

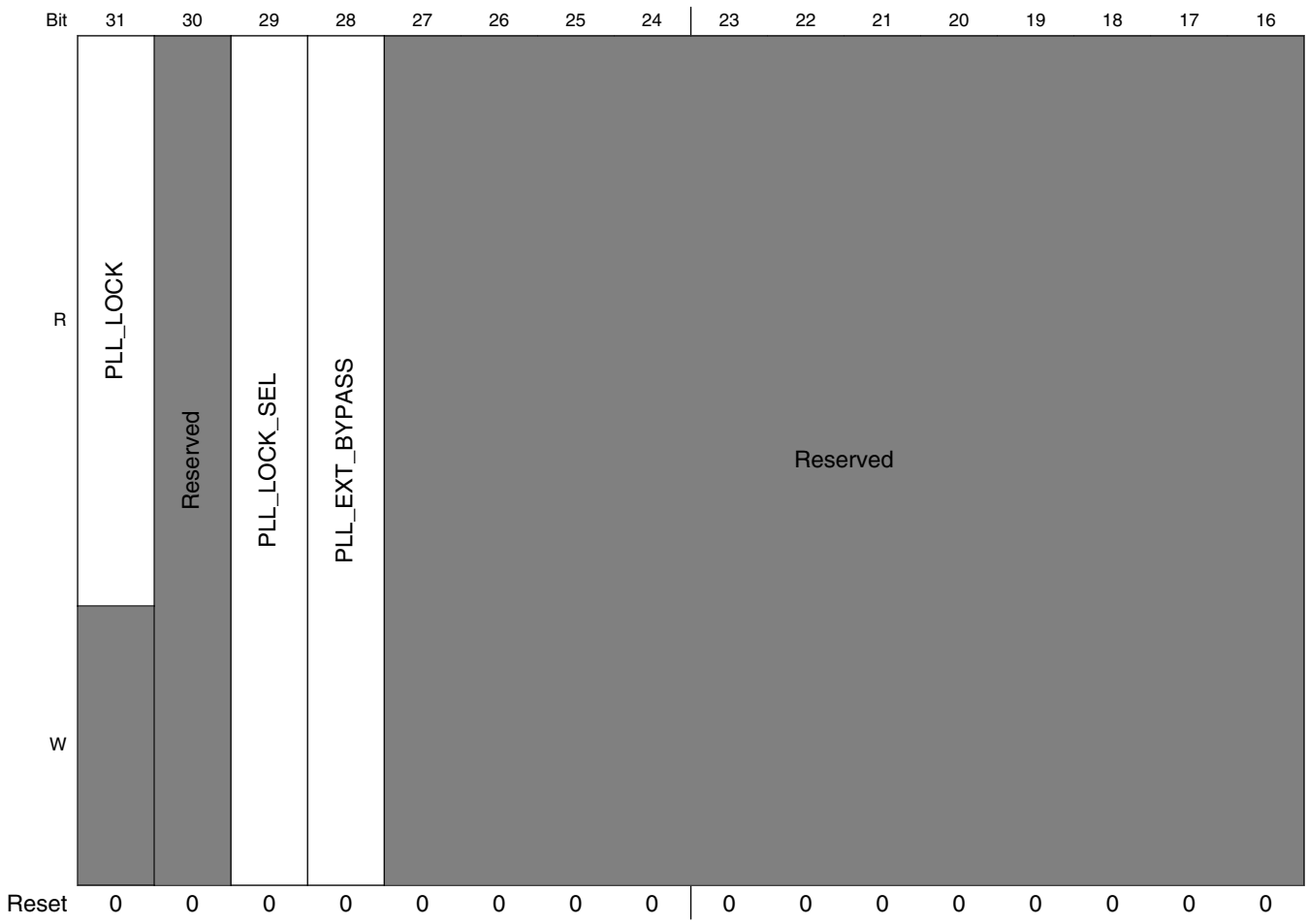
CCM_ANALOG_*n*MNIT_CTRL field descriptions (continued)

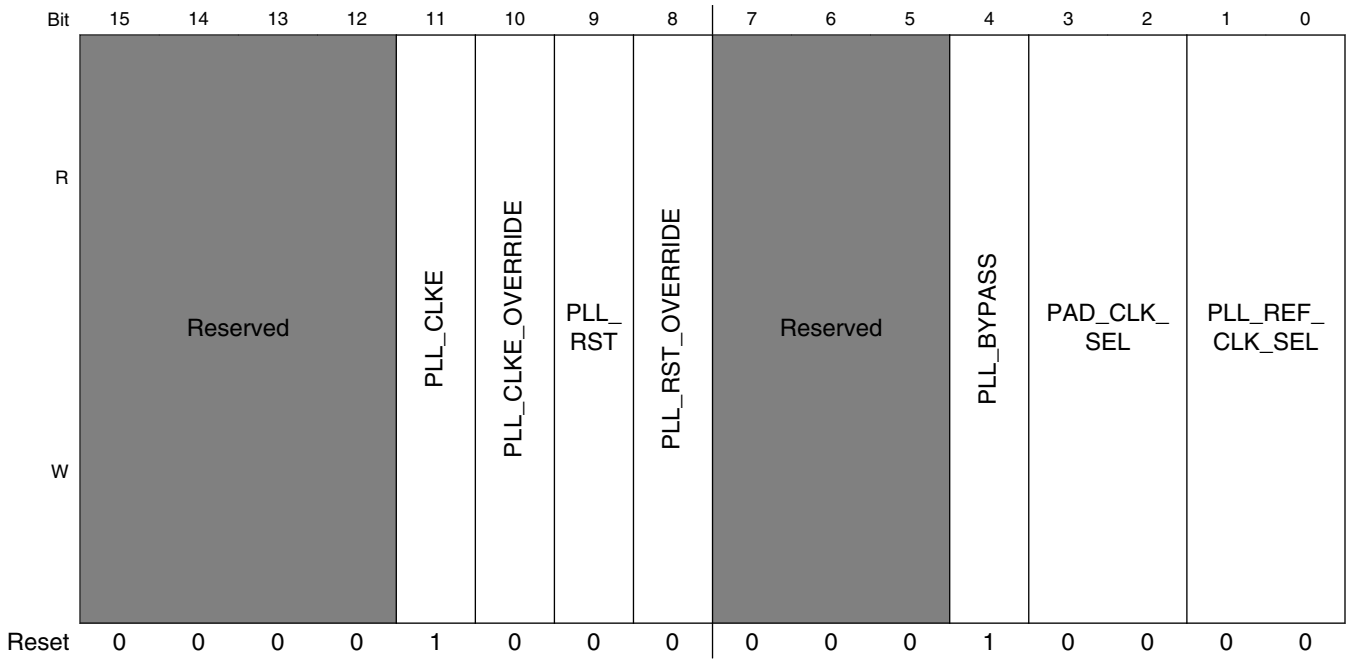
Field	Description
17 PBIAS_CTRL_EN	PBIAS voltage pull-down enable pin
16 AFCINIT_SEL	AFC initial delay select pin 0 nominal delay 1 nominal delay * 2
15 -	This field is reserved. Reserved
14 FSEL	Monitoring frequency select pin 0 FEED_OUT = FREF 1 FEED_OUT = FEED
13 FEED_EN	FEED_OUT enable pin
12–8 -	This field is reserved. Reserved
7–3 EXTAFC	Monitoring pin. If AFC_ENB=1, AFC is disabled and VCO is calibrated manually by EXTAFC[4:0] for the test of VCO range
2 AFC_EN	If AFC_ENB=0, AFC is enabled and VCO is calibrated automatically.(AFC_ENB=0 and EXTAFC=0 are mandatory) If AFC_ENB=1, AFC is disabled and VCO is calibrated manually by EXTAFC[4:0] for the test of VCO range
ICP	Controls the charge-pump current

5.1.8.25 VPU PLL General Function Control Register (CCM_ANALOG_VPU_PLL_GEN_CTRL)

VPU PLL General Function Control Register

Address: 3036_0000h base + 74h offset = 3036_0074h





CCM_ANALOG_VPU_PLL_GEN_CTRL field descriptions

Field	Description
31 PLL_LOCK	PLL lock signal
30 -	This field is reserved. Reserved
29 PLL_LOCK_SEL	PLL lock select 0 Using PLL maximum lock time 1 Using PLL output lock
28 PLL_EXT_BYPASS	PLL analog block bypass, clock output traces to PLL source
27–12 -	This field is reserved. Reserved
11 PLL_CLKE	PLL output clock clock gating enable
10 PLL_CLKE_OVERRIDE	Override the PLL_CLKE, clock gating enable signal from CCM
9 PLL_RST	PLL reset (active low)
8 PLL_RST_OVERRIDE	PLL reset overrided by CCM
7–5 -	This field is reserved. Reserved
4 PLL_BYPASS	PLL output clock bypass

Table continues on the next page...

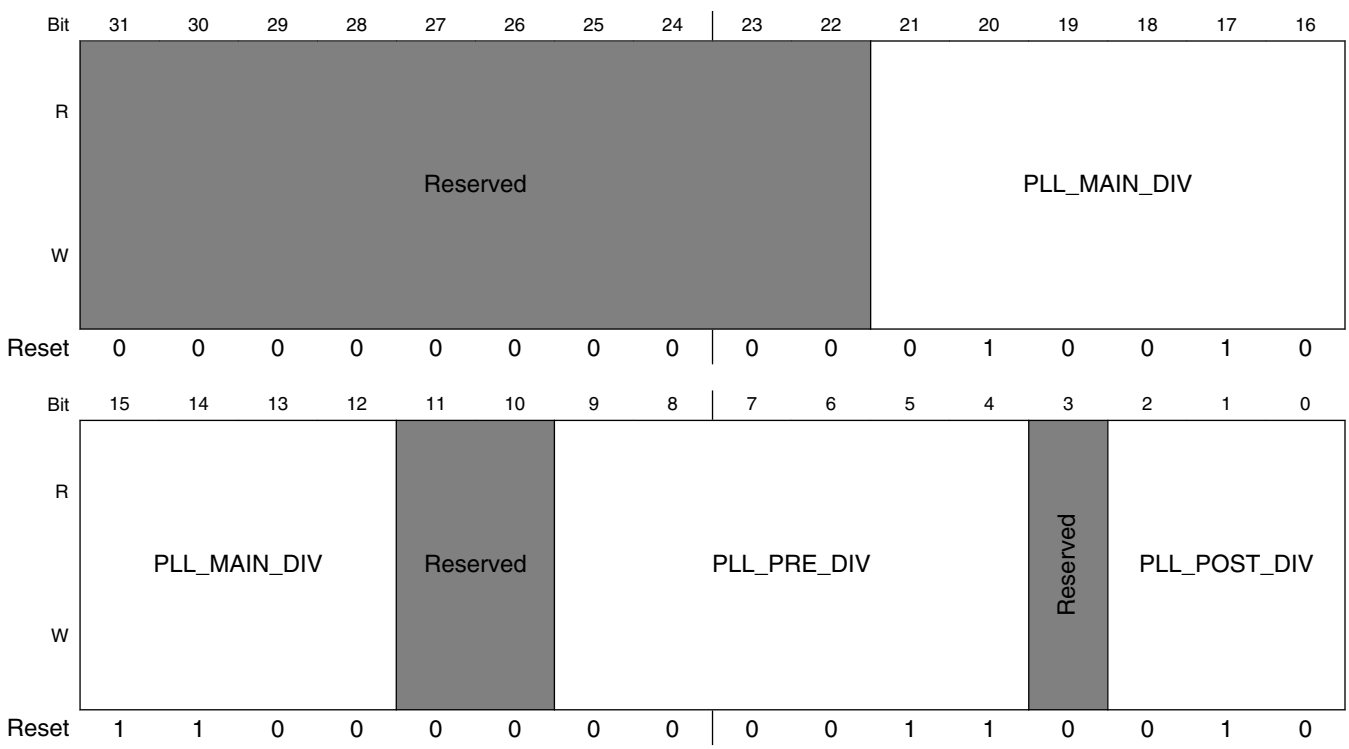
CCM_ANALOG_VPU_PLL_GEN_CTRL field descriptions (continued)

Field	Description
3–2 PAD_CLK_SEL	PAD clock select, the output clock is PAD_CLK, PLL reference clock option 00 CLKIN1 XOR CLKIN2 01 CLKIN2 10 CLKIN1 11 Reserved
PLL_REF_CLK_SEL	PLL reference clock select 00 SYS_XTAL 01 PAD_CLK 10 Reserved 11 Reserved

5.1.8.26 VPU PLL Divide and Fraction Data Control 0 Register (CCM_ANALOG_VPU_PLL_FDIV_CTL0)

VPU PLL Divide and Fraction Data Control 0 Register

Address: 3036_0000h base + 78h offset = 3036_0078h



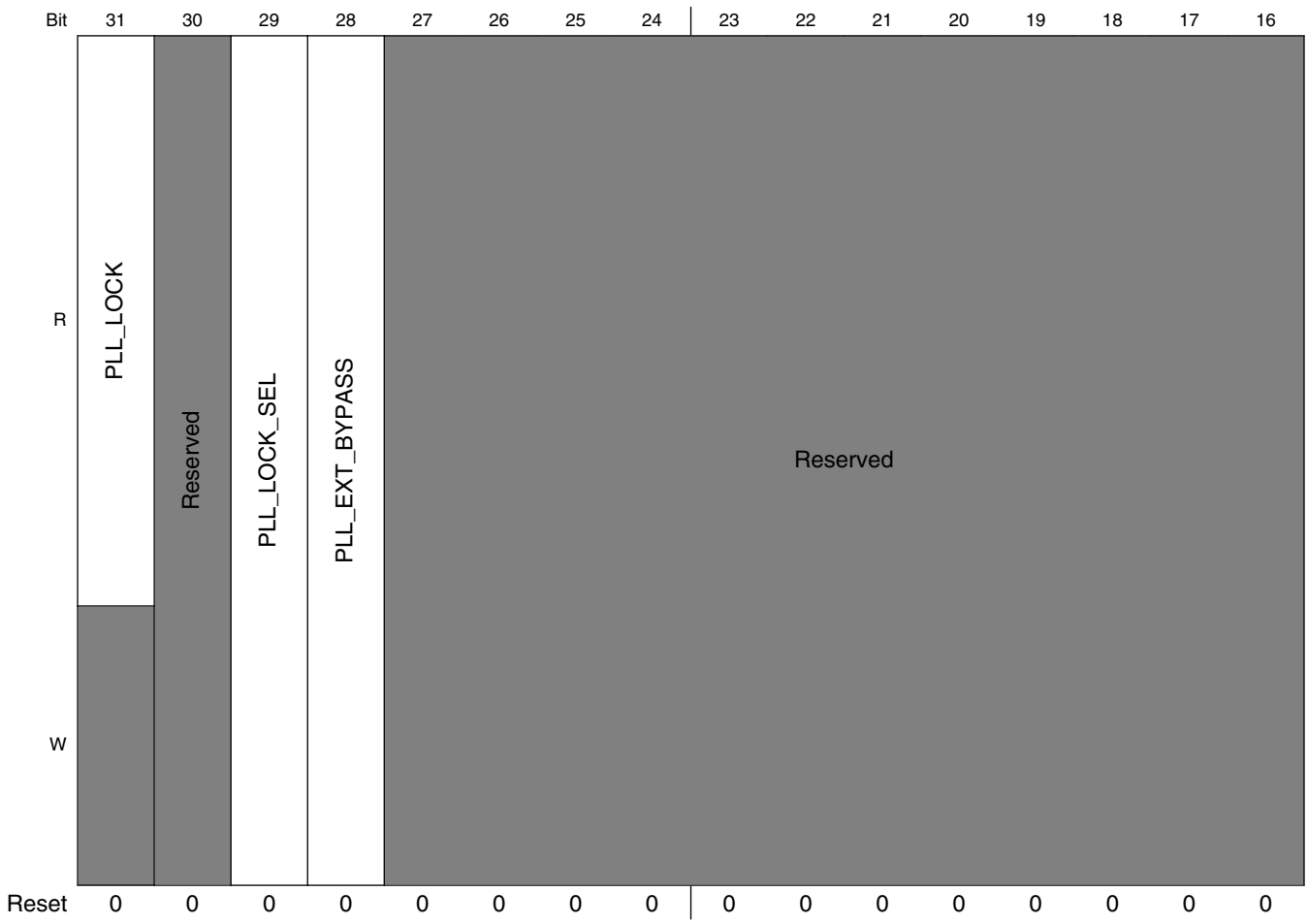
CCM_ANALOG_VPU_PLL_FDIV_CTL0 field descriptions

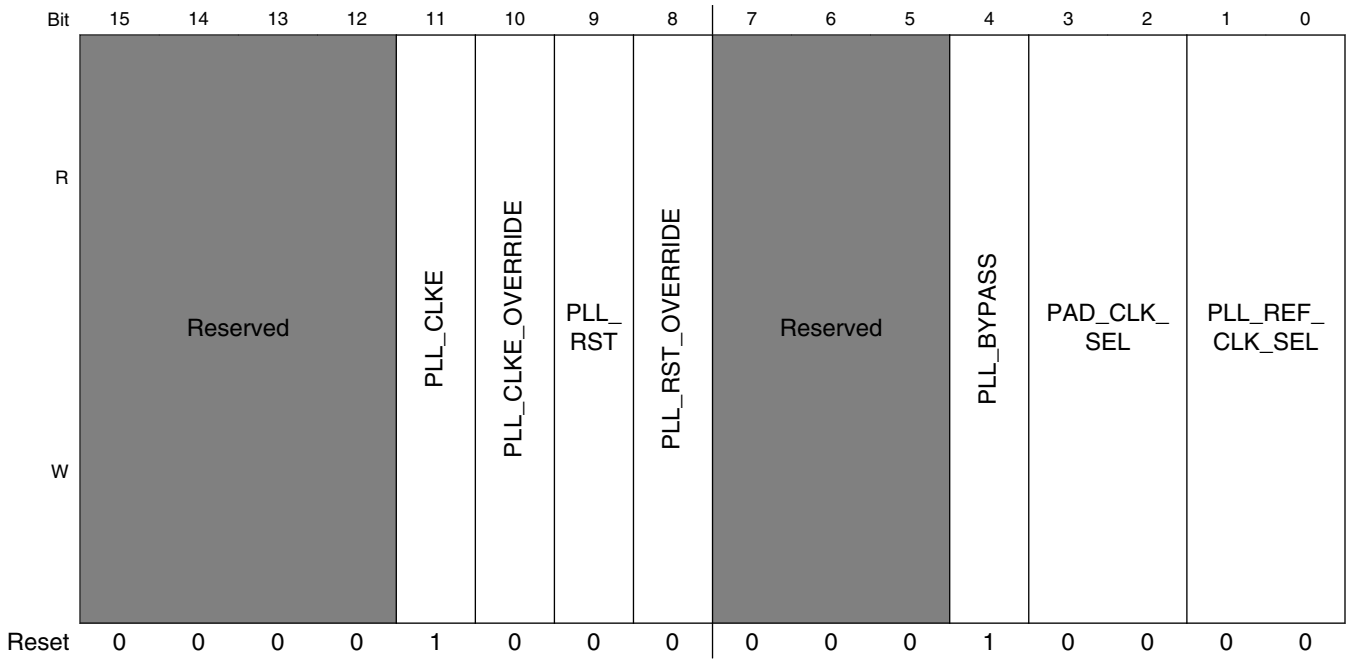
Field	Description
31–22 -	This field is reserved. Reserved
21–12 PLL_MAIN_DIV	Value of the main-divider
11–10 -	This field is reserved. Reserved
9–4 PLL_PRE_DIV	Value of the pre-divider
3 -	This field is reserved. Reserved
PLL_POST_DIV	Value of the post-divider

5.1.8.27 ARM PLL General Function Control Register (CCM_ANALOG_ARM_PLL_GEN_CTRL)

ARM PLL General Function Control Register

Address: 3036_0000h base + 84h offset = 3036_0084h





CCM_ANALOG_ARM_PLL_GEN_CTRL field descriptions

Field	Description
31 PLL_LOCK	PLL lock signal
30 -	This field is reserved. Reserved
29 PLL_LOCK_SEL	PLL lock select 0 Using PLL maximum lock time 1 Using PLL output lock
28 PLL_EXT_BYPASS	PLL analog block bypass, clock output traces to PLL source
27–12 -	This field is reserved. Reserved
11 PLL_CLKE	PLL output clock clock gating enable
10 PLL_CLKE_OVERRIDE	Override the PLL_CLKE, clock gating enable signal from CCM
9 PLL_RST	PLL reset (active low)
8 PLL_RST_OVERRIDE	PLL reset overrided by CCM
7–5 -	This field is reserved. Reserved
4 PLL_BYPASS	PLL output clock bypass

Table continues on the next page...

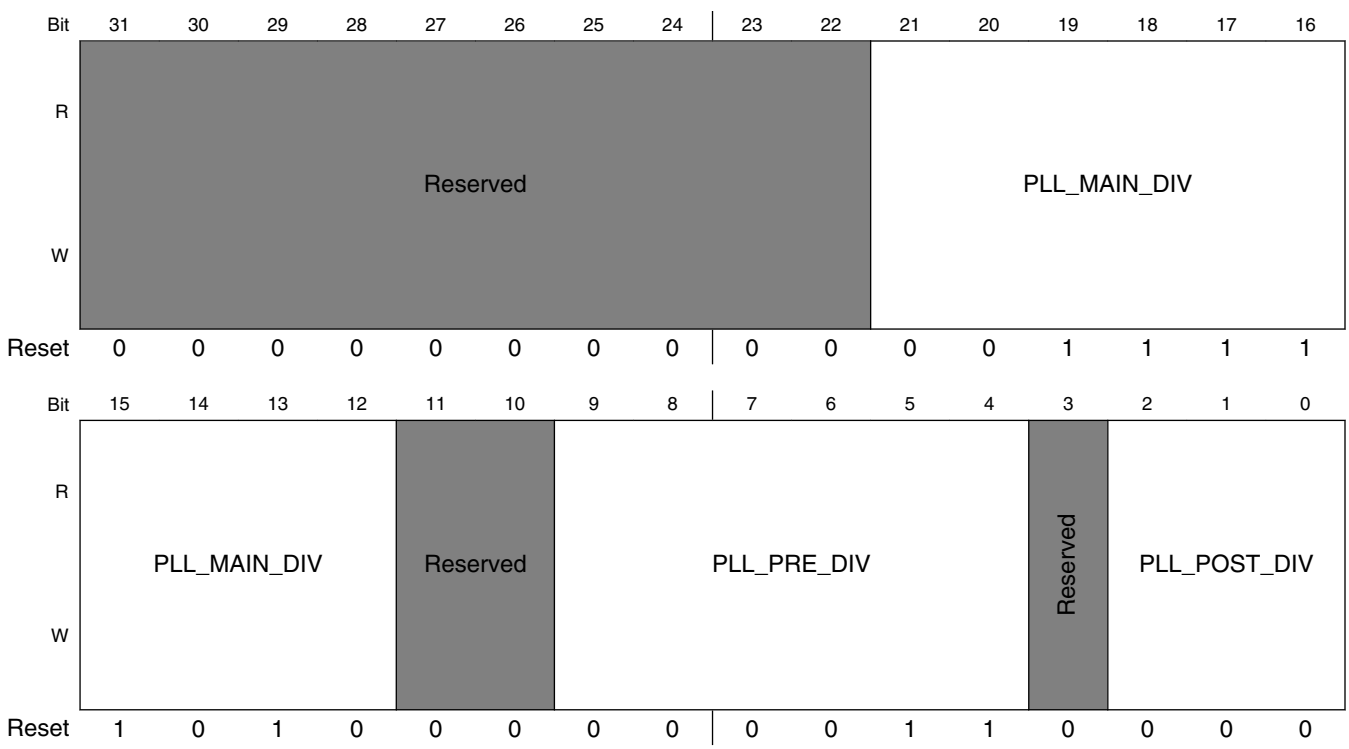
CCM_ANALOG_ARM_PLL_GEN_CTRL field descriptions (continued)

Field	Description
3–2 PAD_CLK_SEL	PAD clock select, the output clock is PAD_CLK, PLL reference clock option 00 CLKIN1 XOR CLKIN2 01 CLKIN2 10 CLKIN1 11 Reserved
PLL_REF_CLK_SEL	PLL reference clock select 00 SYS_XTAL 01 PAD_CLK 10 Reserved 11 Reserved

5.1.8.28 ARM PLL Divide and Fraction Data Control 0 Register (CCM_ANALOG_ARM_PLL_FDIV_CTL0)

ARM PLL Divide and Fraction Data Control 0 Register

Address: 3036_0000h base + 88h offset = 3036_0088h



CCM_ANALOG_ARM_PLL_FDIV_CTL0 field descriptions

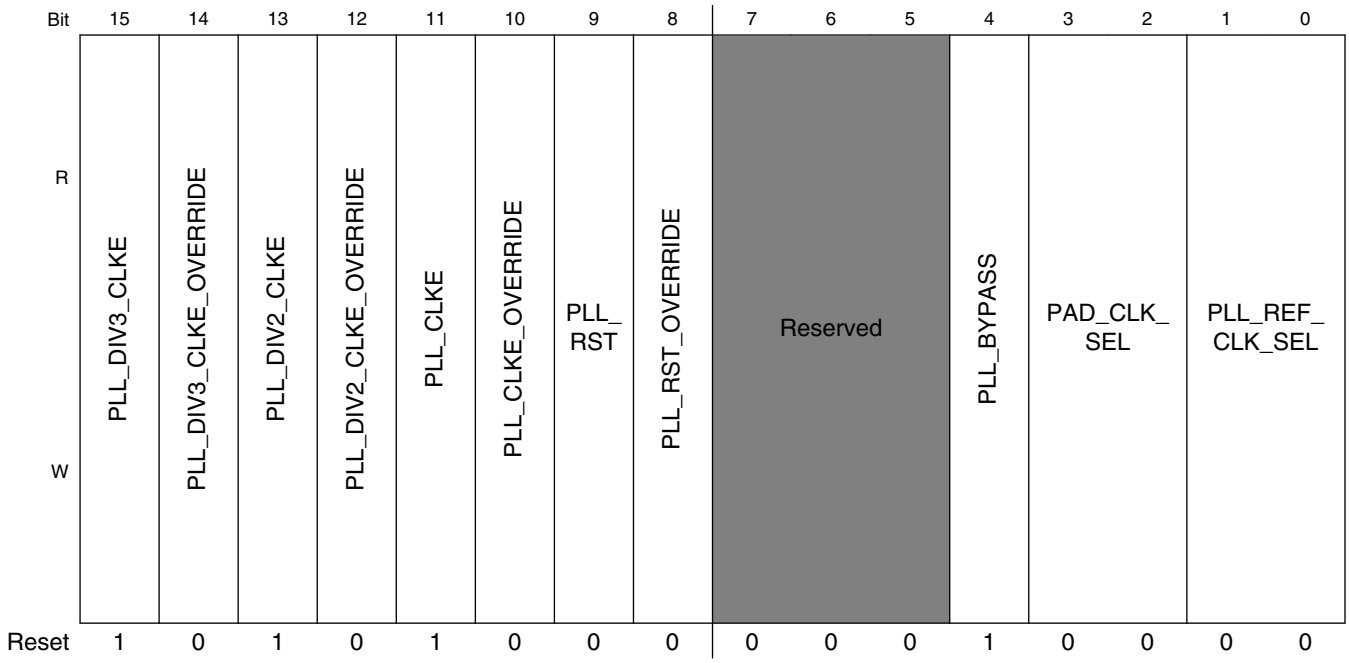
Field	Description
31–22 -	This field is reserved. Reserved
21–12 PLL_MAIN_DIV	Value of the main-divider
11–10 -	This field is reserved. Reserved
9–4 PLL_PRE_DIV	Value of the pre-divider
3 -	This field is reserved. Reserved
PLL_POST_DIV	Value of the post-divider

5.1.8.29 SYS PLL1 General Function Control Register (CCM_ANALOG_SYS_PLL1_GEN_CTRL)

SYS PLL1 General Function Control Register

Address: 3036_0000h base + 94h offset = 3036_0094h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PLL_LOCK	Reserved	PLL_LOCK_SEL	PLL_EXT_BYPASS	PLL_DIV20_CLKE	PLL_DIV20_CLKE_OVERRIDE	PLL_DIV10_CLKE	PLL_DIV10_CLKE_OVERRIDE	PLL_DIV8_CLKE	PLL_DIV8_CLKE_OVERRIDE	PLL_DIV6_CLKE	PLL_DIV6_CLKE_OVERRIDE	PLL_DIV5_CLKE	PLL_DIV5_CLKE_OVERRIDE	PLL_DIV4_CLKE	PLL_DIV4_CLKE_OVERRIDE
W																
Reset	0	0	0	0	1	0	1	0	1	0	1	0	1	0	1	0



CCM_ANALOG_SYS_PLL1_GEN_CTRL field descriptions

Field	Description
31 PLL_LOCK	PLL lock signal
30 -	This field is reserved. Reserved
29 PLL_LOCK_SEL	PLL lock select 0 Using PLL maximum lock time 1 Using PLL output lock
28 PLL_EXT_BYPASS	PLL analog block bypass, clock output traces to PLL source
27 PLL_DIV20_CLKE	PLL clock divided by 20 output gating enable
26 PLL_DIV20_CLKE_OVERRIDE	PLL clock divided by 20 output gating enable overrided by CCM
25 PLL_DIV10_CLKE	PLL clock divided by 10 output gating enable
24 PLL_DIV10_CLKE_OVERRIDE	PLL clock divided by 10 output gating enable overrided by CCM
23 PLL_DIV8_CLKE	PLL clock divided by 8 output gating enable

Table continues on the next page...

CCM_ANALOG_SYS_PLL1_GEN_CTRL field descriptions (continued)

Field	Description
22 PLL_DIV8_ CLKE_ OVERRIDE	PLL clock divided by 8 output gating enable overrided by CCM
21 PLL_DIV6_CLKE	PLL clock divided by 6 output gating enable
20 PLL_DIV6_ CLKE_ OVERRIDE	PLL clock divided by 6 output gating enable overrided by CCM
19 PLL_DIV5_CLKE	PLL clock divided by 5 output gating enable
18 PLL_DIV5_ CLKE_ OVERRIDE	PLL clock divided by 5 output gating enable overrided by CCM
17 PLL_DIV4_CLKE	PLL clock divided by 4 output gating enable
16 PLL_DIV4_ CLKE_ OVERRIDE	PLL clock divided by 4 output gating enable overrided by CCM
15 PLL_DIV3_CLKE	PLL clock divided by 3 output gating enable
14 PLL_DIV3_ CLKE_ OVERRIDE	PLL clock divided by 3 output gating enable overrided by CCM
13 PLL_DIV2_CLKE	PLL clock divided by 2 output gating enable
12 PLL_DIV2_ CLKE_ OVERRIDE	PLL clock divided by 2 output gating enable overrided by CCM
11 PLL_CLKE	PLL output clock clock gating enable
10 PLL_CLKE_ OVERRIDE	Override the PLL_CLKE, clock gating enable signal from CCM
9 PLL_RST	PLL reset (active low)
8 PLL_RST_ OVERRIDE	PLL reset overrided by CCM
7–5 -	This field is reserved. Reserved
4 PLL_BYPASS	PLL output clock bypass

Table continues on the next page...

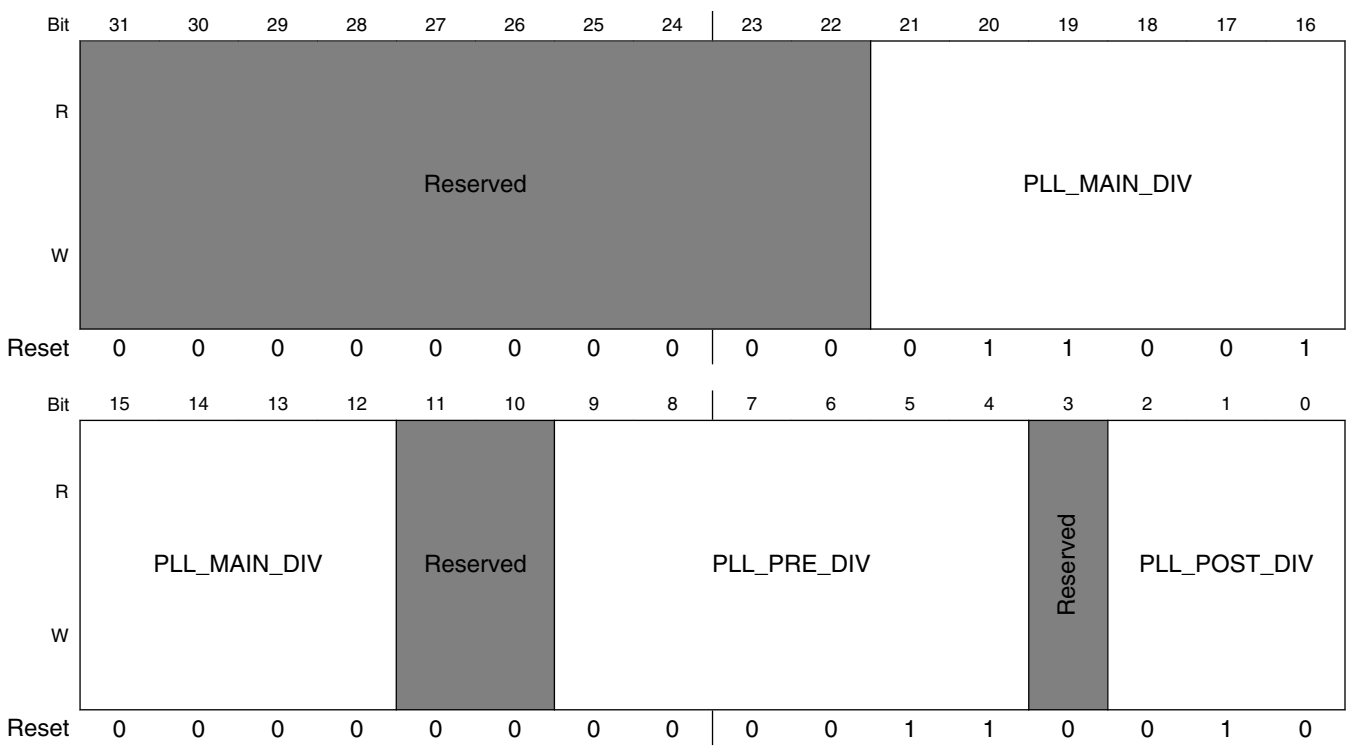
CCM_ANALOG_SYS_PLL1_GEN_CTRL field descriptions (continued)

Field	Description
3–2 PAD_CLK_SEL	PAD clock select, the output clock is PAD_CLK, PLL reference clock option 00 CLKIN1 XOR CLKIN2 01 CLKIN2 10 CLKIN1 11 Reserved
PLL_REF_CLK_SEL	PLL reference clock select 00 SYS_XTAL 01 PAD_CLK 10 Reserved 11 Reserved

5.1.8.30 SYS PLL1 Divide and Fraction Data Control 0 Register (CCM_ANALOG_SYS_PLL1_FDIV_CTL0)

SYS PLL1 Divide and Fraction Data Control 0 Register

Address: 3036_0000h base + 98h offset = 3036_0098h



CCM_ANALOG_SYS_PLL1_FDIV_CTL0 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–12 PLL_MAIN_DIV	Value of the main-divider
11–10 -	This field is reserved. Reserved
9–4 PLL_PRE_DIV	Value of the pre-divider
3 -	This field is reserved. Reserved
PLL_POST_DIV	Value of the post-divider

5.1.8.31 PLL Monitoring Control Register (CCM_ANALOG_*n*MNIT_CTRL)

PLL Monitoring Control Register

Address: 3036_0000h base + 100h offset + (16d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved										LRD_EN	FOUT_MASK	AFC_SEL	PBIAS_CTRL	PBIAS_CTRL_EN	AFCINIT_SEL
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	FSEL	FEED_EN	Reserved					EXTAFC					AFC_EN	ICP	
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1

CCM_ANALOG_*n*MNIT_CTRL field descriptions

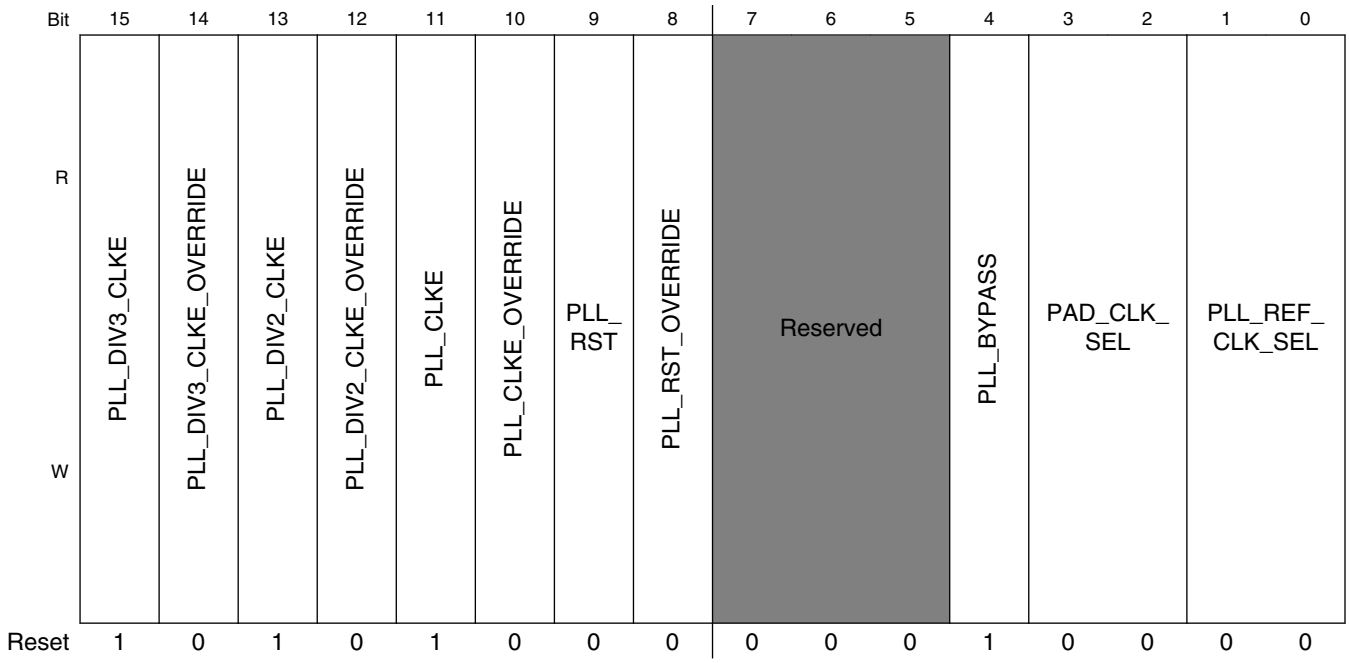
Field	Description
31–22 -	This field is reserved. Reserved
21 LRD_EN	Monitoring pin. AFC operation mode select pin
20 FOUT_MASK	Scaler's re-initialization time control pin[3]
19 AFC_SEL	AFC Mode select
18 PBIAS_CTRL	PBIAS pull-down initial voltage control pin 0 0.50*VDD 1 0.67*VDD
17 PBIAS_CTRL_EN	PBIAS voltage pull-down enable pin
16 AFCINIT_SEL	AFC initial delay select pin 0 nominal delay 1 nominal delay * 2
15 -	This field is reserved. Reserved
14 FSEL	Monitoring frequency select pin 0 FEED_OUT = FREF 1 FEED_OUT = FEED
13 FEED_EN	FEED_OUT enable pin
12–8 -	This field is reserved. Reserved
7–3 EXTAFC	Monitoring pin. If AFC_ENB=1, AFC is disabled and VCO is calibrated manually by EXTAFC[4:0] for the test of VCO range
2 AFC_EN	If AFC_ENB=0, AFC is enabled and VCO is calibrated automatically.(AFC_ENB=0 and EXTAFC=0 are mandatory) If AFC_ENB=1, AFC is disabled and VCO is calibrated manually by EXTAFC[4:0] for the test of VCO range
ICP	Controls the charge-pump current

5.1.8.32 SYS PLL2 General Function Control Register (CCM_ANALOG_SYS_PLL2_GEN_CTRL)

SYS PLL2 General Function Control Register

Address: 3036_0000h base + 104h offset = 3036_0104h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PLL_LOCK	Reserved	PLL_LOCK_SEL	PLL_EXT_BYPASS	PLL_DIV20_CLKE	PLL_DIV20_CLKE_OVERRIDE	PLL_DIV10_CLKE	PLL_DIV10_CLKE_OVERRIDE	PLL_DIV8_CLKE	PLL_DIV8_CLKE_OVERRIDE	PLL_DIV6_CLKE	PLL_DIV6_CLKE_OVERRIDE	PLL_DIV5_CLKE	PLL_DIV5_CLKE_OVERRIDE	PLL_DIV4_CLKE	PLL_DIV4_CLKE_OVERRIDE
W																
Reset	0	0	0	0	1	0	1	0	1	0	1	0	1	0	1	0



CCM_ANALOG_SYS_PLL2_GEN_CTRL field descriptions

Field	Description
31 PLL_LOCK	PLL lock signal
30 -	This field is reserved. Reserved
29 PLL_LOCK_SEL	PLL lock select 0 Using PLL maximum lock time 1 Using PLL output lock
28 PLL_EXT_BYPASS	PLL analog block bypass, clock output traces to PLL source
27 PLL_DIV20_CLKE	PLL clock divided by 20 output gating enable
26 PLL_DIV20_CLKE_OVERRIDE	PLL clock divided by 20 output gating enable overrided by CCM
25 PLL_DIV10_CLKE	PLL clock divided by 10 output gating enable
24 PLL_DIV10_CLKE_OVERRIDE	PLL clock divided by 10 output gating enable overrided by CCM
23 PLL_DIV8_CLKE	PLL clock divided by 8 output gating enable

Table continues on the next page...

CCM_ANALOG_SYS_PLL2_GEN_CTRL field descriptions (continued)

Field	Description
22 PLL_DIV8_ CLKE_ OVERRIDE	PLL clock divided by 8 output gating enable overrided by CCM
21 PLL_DIV6_CLKE	PLL clock divided by 6 output gating enable
20 PLL_DIV6_ CLKE_ OVERRIDE	PLL clock divided by 6 output gating enable overrided by CCM
19 PLL_DIV5_CLKE	PLL clock divided by 5 output gating enable
18 PLL_DIV5_ CLKE_ OVERRIDE	PLL clock divided by 5 output gating enable overrided by CCM
17 PLL_DIV4_CLKE	PLL clock divided by 4 output gating enable
16 PLL_DIV4_ CLKE_ OVERRIDE	PLL clock divided by 4 output gating enable overrided by CCM
15 PLL_DIV3_CLKE	PLL clock divided by 3 output gating enable
14 PLL_DIV3_ CLKE_ OVERRIDE	PLL clock divided by 3 output gating enable overrided by CCM
13 PLL_DIV2_CLKE	PLL clock divided by 2 output gating enable
12 PLL_DIV2_ CLKE_ OVERRIDE	PLL clock divided by 2 output gating enable overrided by CCM
11 PLL_CLKE	PLL output clock clock gating enable
10 PLL_CLKE_ OVERRIDE	Override the PLL_CLKE, clock gating enable signal from CCM
9 PLL_RST	PLL reset (active low)
8 PLL_RST_ OVERRIDE	PLL reset overrided by CCM
7–5 -	This field is reserved. Reserved
4 PLL_BYPASS	PLL output clock bypass

Table continues on the next page...

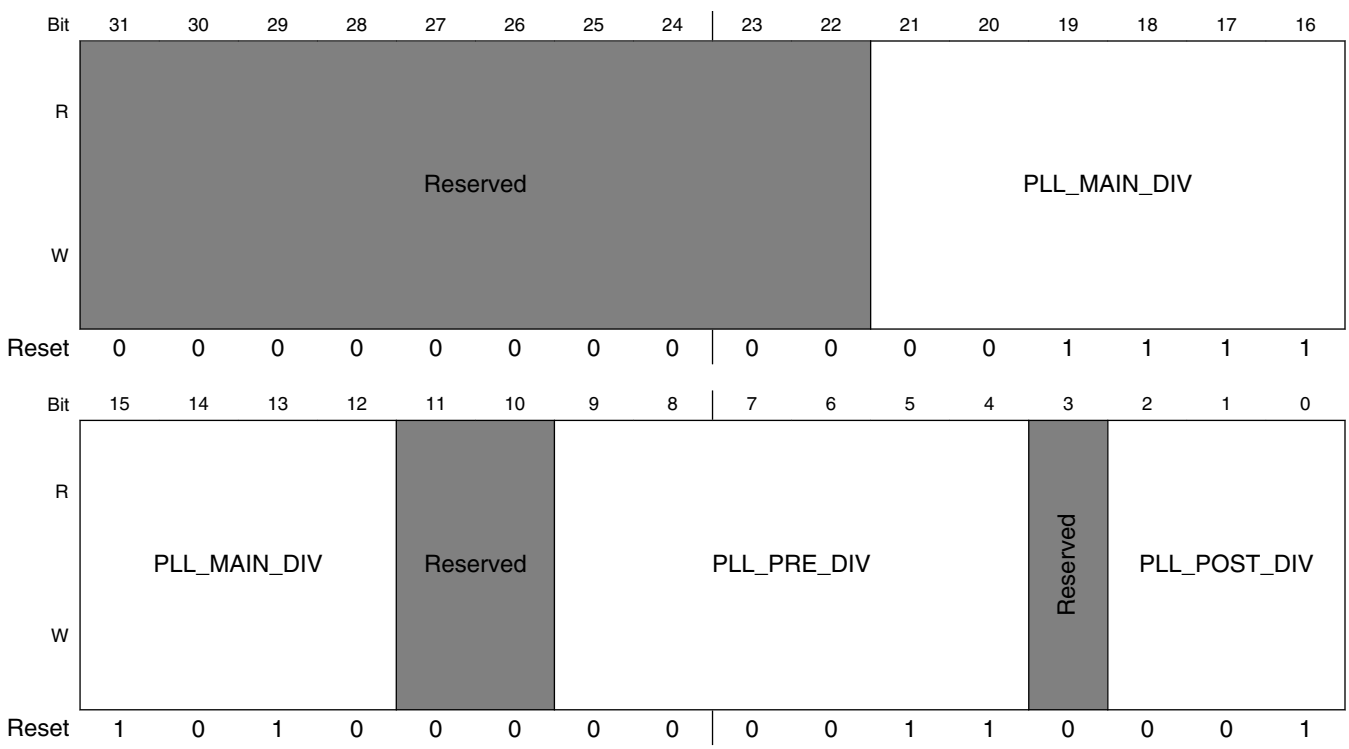
CCM_ANALOG_SYS_PLL2_GEN_CTRL field descriptions (continued)

Field	Description
3–2 PAD_CLK_SEL	PAD clock select, the output clock is PAD_CLK, PLL reference clock option 00 CLKIN1 XOR CLKIN2 01 CLKIN2 10 CLKIN1 11 Reserved
PLL_REF_CLK_SEL	PLL reference clock select 00 SYS_XTAL 01 PAD_CLK 10 Reserved 11 Reserved

5.1.8.33 SYS PLL2 Divide and Fraction Data Control 0 Register (CCM_ANALOG_SYS_PLL2_FDIV_CTL0)

SYS PLL2 Divide and Fraction Data Control 0 Register

Address: 3036_0000h base + 108h offset = 3036_0108h



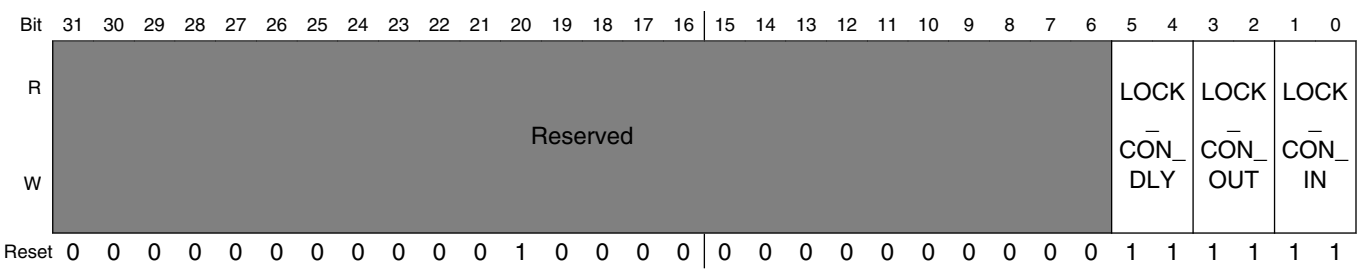
CCM_ANALOG_SYS_PLL2_FDIV_CTL0 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–12 PLL_MAIN_DIV	Value of the main-divider
11–10 -	This field is reserved. Reserved
9–4 PLL_PRE_DIV	Value of the pre-divider
3 -	This field is reserved. Reserved
PLL_POST_DIV	Value of the post-divider

5.1.8.34 PLL Lock Detector Control Register (CCM_ANALOG_*n*LOCKD_CTRL)

PLL Lock Detector Control Register

Address: 3036_0000h base + 10Ch offset + (16d × i), where i=0d to 1d



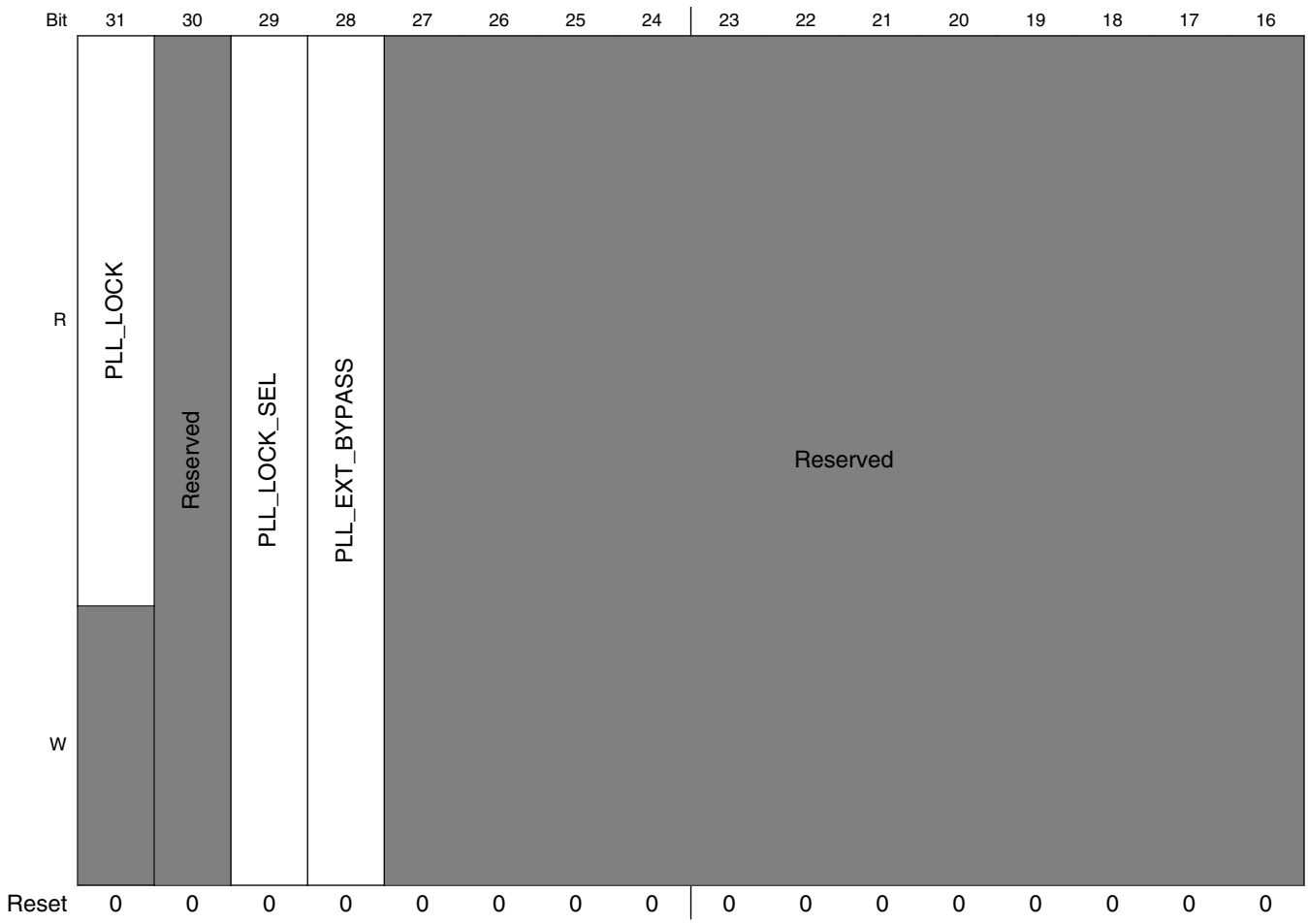
CCM_ANALOG_*n*LOCKD_CTRL field descriptions

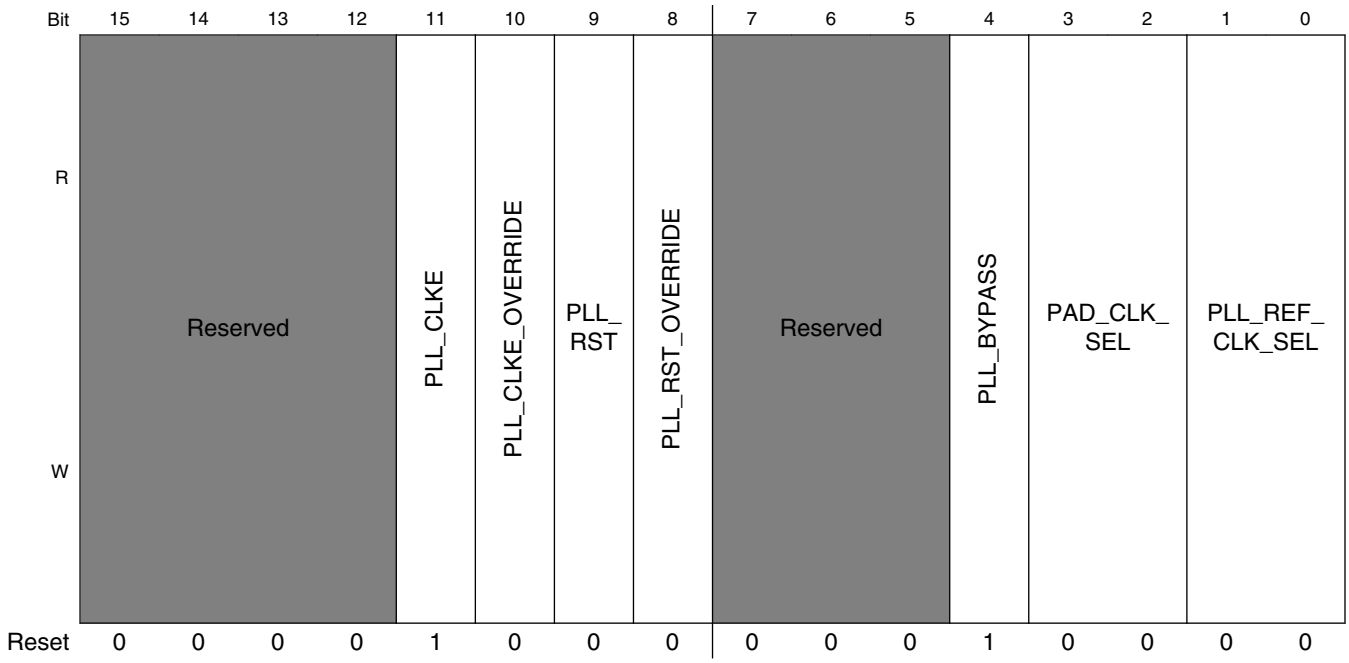
Field	Description
31–6 -	This field is reserved. Reserved
5–4 LOCK_CON_DLY	Lock detector setting of the detection resolution
3–2 LOCK_CON_OUT	Lock detector setting of the output margin
LOCK_CON_IN	Lock detector setting of the input margin

5.1.8.35 SYS PLL3 General Function Control Register (CCM_ANALOG_SYS_PLL3_GEN_CTRL)

SYS PLL3 General Function Control Register

Address: 3036_0000h base + 114h offset = 3036_0114h





CCM_ANALOG_SYS_PLL3_GEN_CTRL field descriptions

Field	Description
31 PLL_LOCK	PLL lock signal
30 -	This field is reserved. Reserved
29 PLL_LOCK_SEL	PLL lock select 0 Using PLL maximum lock time 1 Using PLL output lock
28 PLL_EXT_BYPASS	PLL analog block bypass, clock output traces to PLL source
27–12 -	This field is reserved. Reserved
11 PLL_CLKE	PLL output clock clock gating enable
10 PLL_CLKE_OVERRIDE	Override the PLL_CLKE, clock gating enable signal from CCM
9 PLL_RST	PLL reset (active low)
8 PLL_RST_OVERRIDE	PLL reset overrided by CCM
7–5 -	This field is reserved. Reserved
4 PLL_BYPASS	PLL output clock bypass

Table continues on the next page...

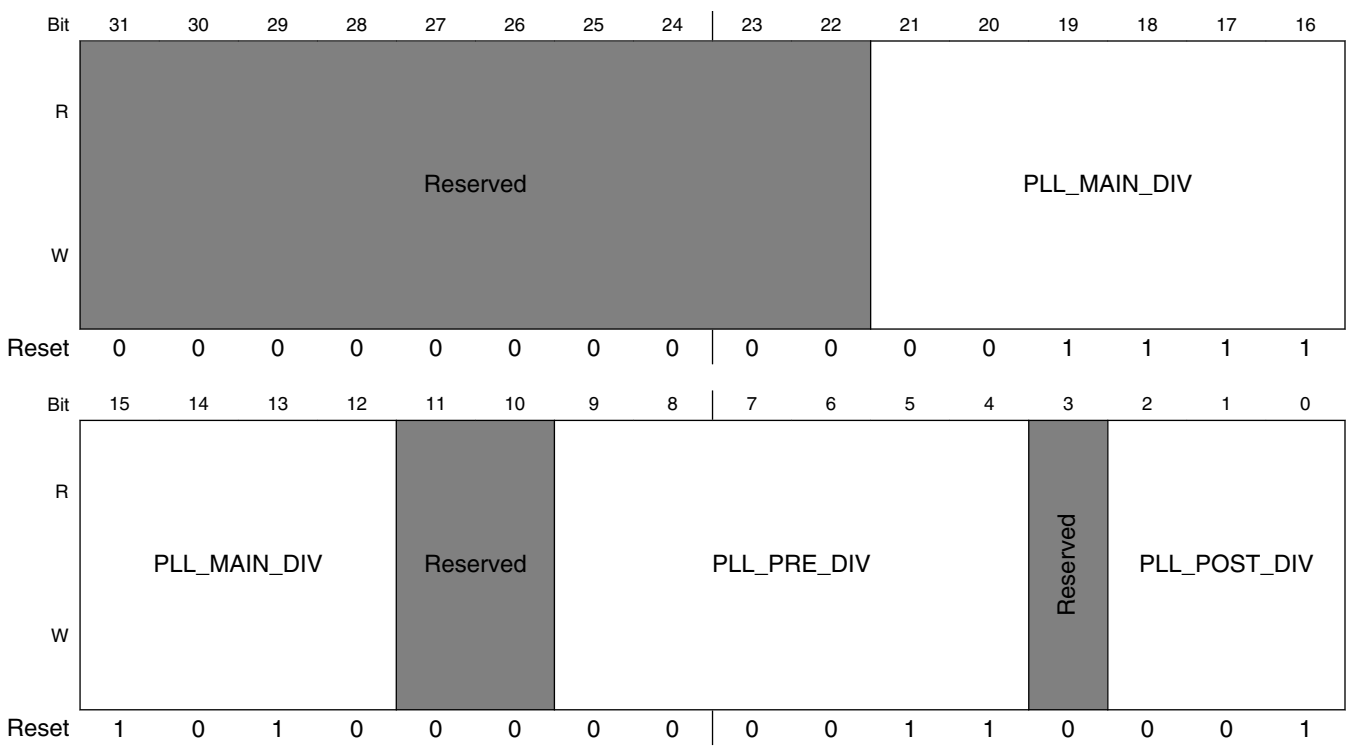
CCM_ANALOG_SYS_PLL3_GEN_CTRL field descriptions (continued)

Field	Description
3–2 PAD_CLK_SEL	PAD clock select, the output clock is PAD_CLK, PLL reference clock option 00 CLKIN1 XOR CLKIN2 01 CLKIN2 10 CLKIN1 11 Reserved
PLL_REF_CLK_SEL	PLL reference clock select 00 SYS_XTAL 01 PAD_CLK 10 Reserved 11 Reserved

5.1.8.36 SYS PLL3 Divide and Fraction Data Control 0 Register (CCM_ANALOG_SYS_PLL3_FDIV_CTL0)

SYS PLL3 Divide and Fraction Data Control 0 Register

Address: 3036_0000h base + 118h offset = 3036_0118h



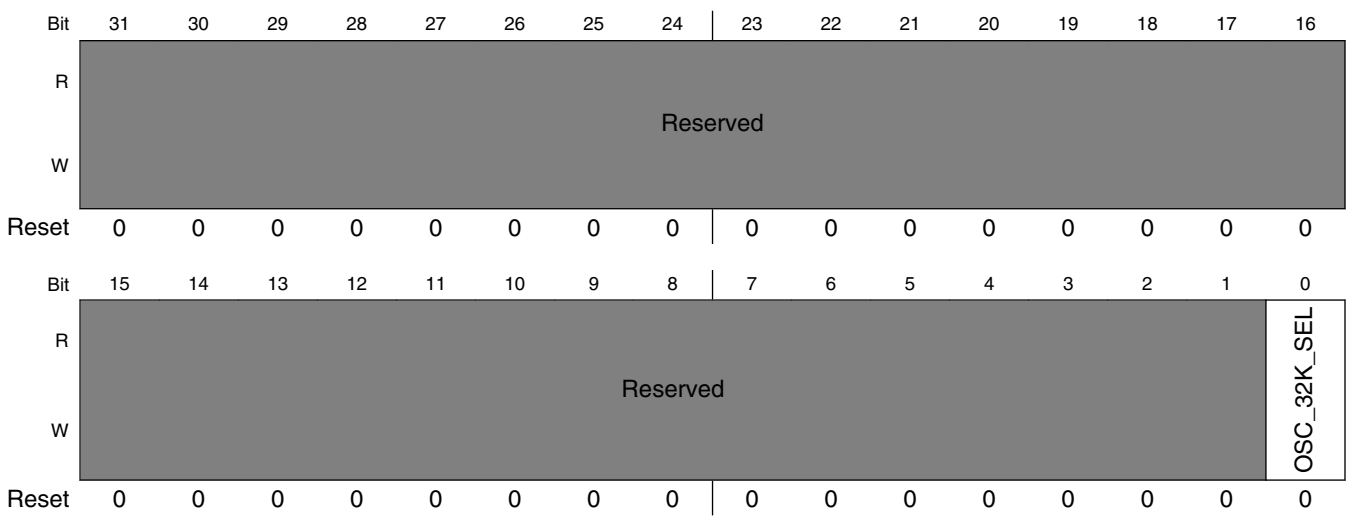
CCM_ANALOG_SYS_PLL3_FDIV_CTL0 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–12 PLL_MAIN_DIV	Value of the main-divider
11–10 -	This field is reserved. Reserved
9–4 PLL_PRE_DIV	Value of the pre-divider
3 -	This field is reserved. Reserved
PLL_POST_DIV	Value of the post-divider

5.1.8.37 Osc Misc Configuration Register (CCM_ANALOG_OSC_MISC_CFG)

Osc Misc Register

Address: 3036_0000h base + 124h offset = 3036_0124h



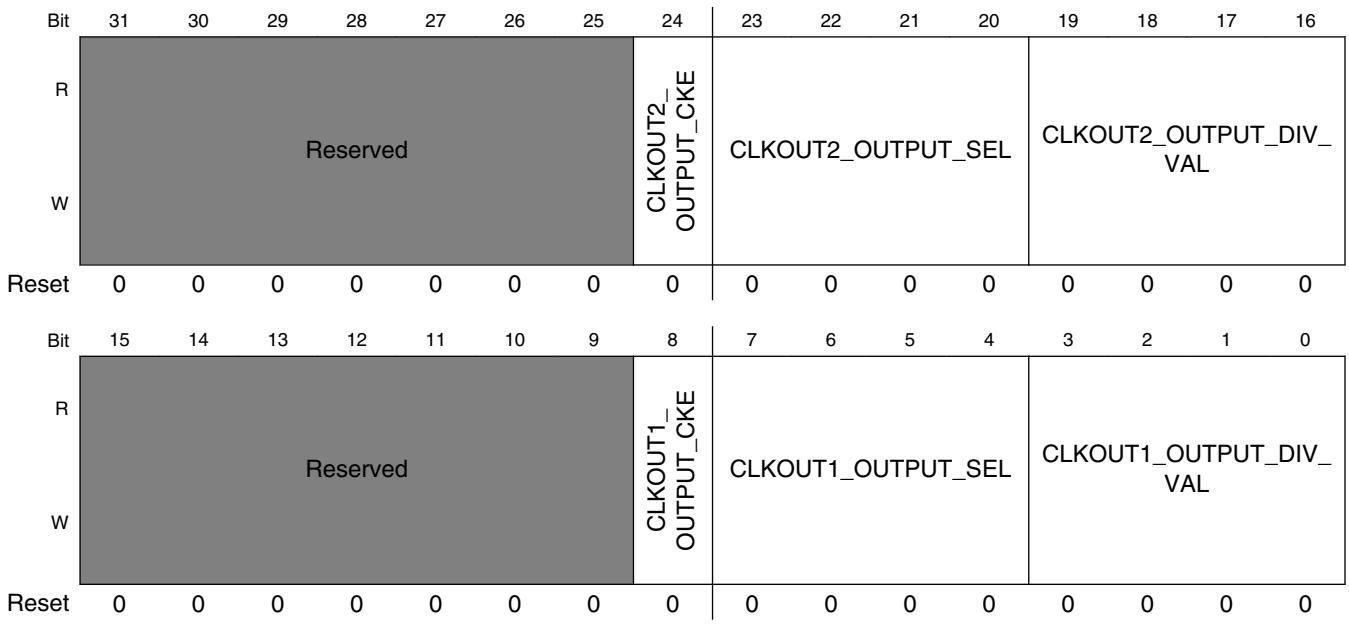
CCM_ANALOG_OSC_MISC_CFG field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 OSC_32K_SEL	32KHz OSC input select 0 Divided by 24M clock 1 32K Oscillator

5.1.8.38 PLL Clock Output for Test Enable and Select Register (CCM_ANALOG_ANAMIX_PLL_MNIT_CTL)

PLL Clock Output for Test Enable and Select Register

Address: 3036_0000h base + 128h offset = 3036_0128h



CCM_ANALOG_ANAMIX_PLL_MNIT_CTL field descriptions

Field	Description
31–25 -	This field is reserved. Reserved
24 CLKOUT2_ OUTPUT_CKE	CLKOUT2 Monitor output enable
23–20 CLKOUT2_ OUTPUT_SEL	CLKOUT2 Monitor output clock select 4'b0000 : audio_pll1_clk 4'b0001 : audio_pll2_clk 4'b0010 : video_pll1_clk 4'b0011 : reserved 4'b0100 : misc_mnit_clk 4'b0101 : gpu_pll_clk 4'b0110 : vpu_pll_clk 4'b0111 : arm_pll_clk 4'b1000 : system_pll1_clk

Table continues on the next page...

CCM_ANALOG_ANAMIX_PLL_MNIT_CTL field descriptions (continued)

Field	Description
	4'b1001 : system_pll2_clk 4'b1010 : system_pll3_clk 4'b1011 : CLKIN1 4'b1100 : CLKIN2 4'b1101 : sysosc_24m_clk 4'b1110 : reserved 4'b1111 : osc_32k_clk
19–16 CLKOUT2_OUTPUT_DIV_VAL	CLKOUT2 output divide value
15–9 -	This field is reserved. Reserved
8 CLKOUT1_OUTPUT_CKE	CLKOUT1 Monitor output enable
7–4 CLKOUT1_OUTPUT_SEL	CLKOUT1 Monitor output clock select 4'b0000 : audio_pll1_clk 4'b0001 : audio_pll2_clk 4'b0010 : video_pll1_clk 4'b0011 : reserved 4'b0100 : misc_mnit_clk 4'b0101 : gpu_pll_clk 4'b0110 : vpu_pll_clk 4'b0111 : arm_pll_clk 4'b1000 : system_pll1_clk 4'b1001 : system_pll2_clk 4'b1010 : system_pll3_clk 4'b1011 : CLKIN1 4'b1100 : CLKIN2 4'b1101 : sysosc_24m_clk 4'b1110 : reserved 4'b1111 : osc_32k_clk
CLKOUT1_OUTPUT_DIV_VAL	CLKOUT1 output divide value

5.1.8.39 DIGPROG Register (CCM_ANALOG_DIGPROG)

DIGPROG Register

Address: 3036_0000h base + 800h offset = 3036_0800h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								DIGPROG_MAJOR_UPPER								DIGPROG_MAJOR_LOWER								DIGPROG_MINOR							
W																																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0

CCM_ANALOG_DIGPROG field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 DIGPROG_ MAJOR_UPPER	Bit[7:4] is 0x8, stands for “i.MX8” Bit[3:0] is 0x2, stands for ”M”
15–8 DIGPROG_ MAJOR_LOWER	Bit[7:4] is 0x4, stands for “Quad” Bit[3:0] is 0x0, stands for “Mini”
DIGPROG_ MINOR	Bit[7:4] is the base layer revision, Bit[3:0] is the metal layer revision 0x10 stands for Tapeout 1.0

5.2 General Power Controller (GPC)

5.2.1 Overview

The General Power Controller (GPC) module controls the following functions:

- Provide low power mode control for A53 and M4 platform
- Provide Power domain management all Arm and SOC power domain
- Provide domain control mechanism based on A53 and M4 CPU domain
- Provide handshake with CCM for clock management in low power mode
- Provide handshake with SRC for power down and power up sequence
- Provide handshake with Analog for Deep Sleep Mode control

5.2.2 Features

The General Power Controller (GPC) module controls the following functions:

- Support programmable feature for WAIT/STOP/DSM low power mode
- Support time slot based power domain control
- Support flexible sleep and wakeup condition
- Support domain control for multi CPU platforms system
- All register accessed by IP bus
- Interface for the following IPs:
    - CCM – clock controller module
    - SRC – system reset controller
    - ANALOG – miscellaneous analog control

5.2.3 Block Diagram

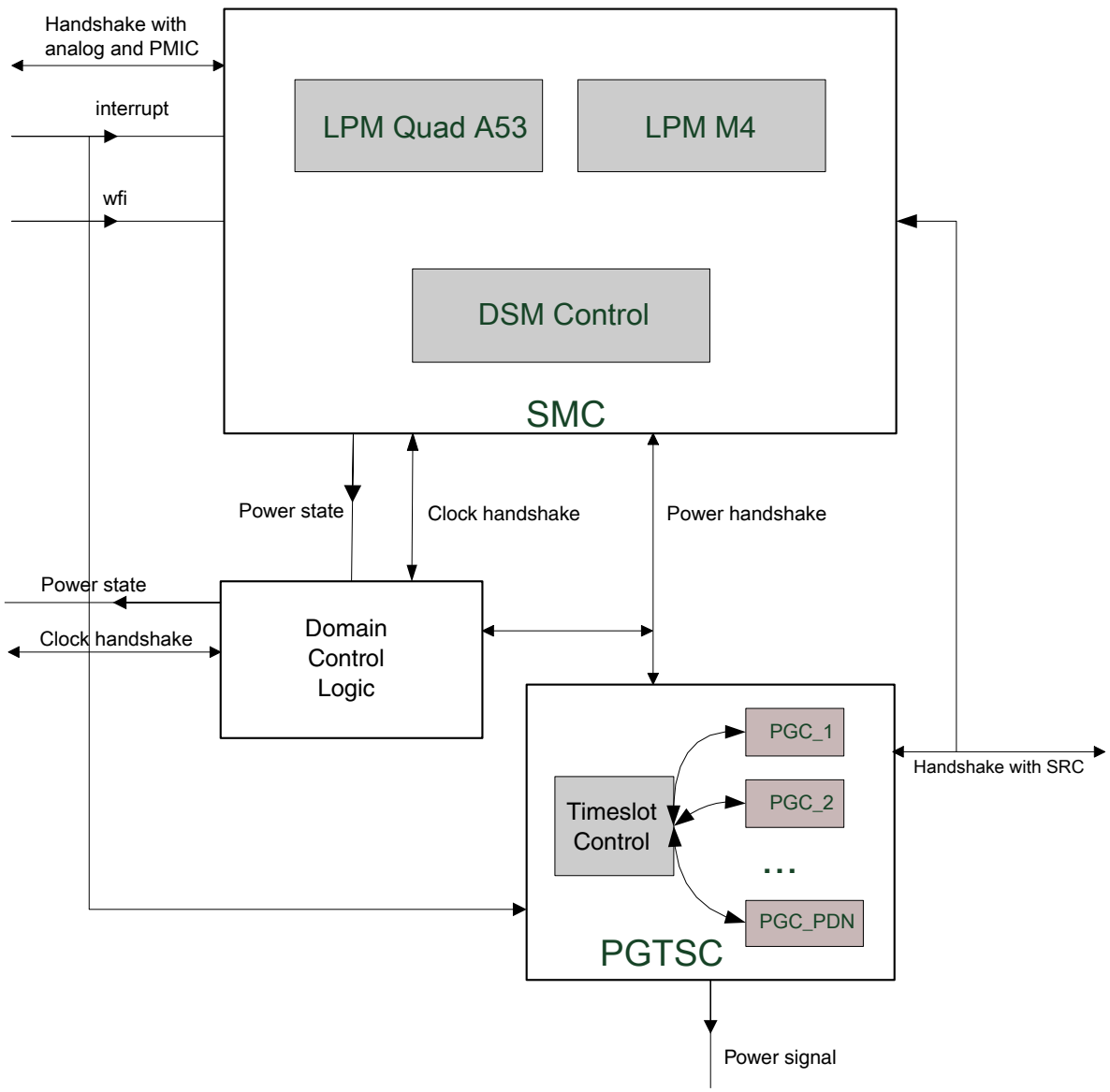


Figure 5-10. GPC Block Diagram

The GPC module contains two sub-modules: System Mode Controller (SMC) and Power Gating Time Slot Control (PGTSC):

- GPC Top: the top level GPC. It also includes the top memory map and registers, domain control information, and memory low power control.
- System Mode Controller (SMC):
  - The SMC supports two low power modes (LPM), WAIT and STOP. Each LPM corresponds to one mode for A53 platform and one mode for M4 platform.

- SMC controls the power sequence in Deep Sleep Mode (DSM)
- SMC support the power up and power down of A53 core0/core1/core2/core3 by IRQ/WFI signals without LPM triggered
- SMC can translate the LPM request for A53 and M4 platform to power up and power down request to PGTSC.
- Power Gating Time Slot Control (PGTSC):
  - The Power Gating Controller (PGC) is a power management component that controls the power-down and power-up sequencing of individual subsystems. For subsystems to be completely powered down in low power modes, a specific sequence of power control signals must be followed. The sequence timing is programmable using the PGC control registers.
  - There are 20 PGCs in the chip, all of them can be power down/up with a software trigger and all of them can be mapped to 20 timing slots and power-up and power down by request from SMC.

5.2.4 Functional Description

5.2.4.1 RUN mode

This is the normal/functional operating mode. In this mode, the CPU runs in its normal operational mode.

5.2.4.2 Low power mode

There are two CPU platforms (each of them represents a CPU domain): Quad core Cortex A53 platform and Cortex M4 platform. Each platform supports two low power modes: WAIT mode and STOP mode.

5.2.4.2.1 WAIT mode

In this low power mode:

- LPCG can be defined to be shut off or not in wait mode for each CPU domain
- PLL can be defined to be shut off or not in wait mode for each CPU domain

NOTE

The PLLs will only been closed in non-fast wake-up mode, relevant bit are

GPC_SLPCR[EN_A53_FASTWUP_WAIT_MODE] and
GPC_SLPCR[EN_M4_FASTWUP_WAIT_MODE]

- CPU clock can be defined been shut off or not in wait for each CPU platform. (GPC_LPCR_A53_BSC[CPU_CLK_ON_LPM] and GPC_LPCR_M4[CPU_CLK_ON_LPM])
- Power of different power domain can be defined be shut off or not in wait mode for each platform domain
- Some peripherals may go to wait mode along with A53 or M4 platform.

5.2.4.2.2 STOP mode

In this low power mode:

- LPCG can be defined been shut off or not in stop mode for each CPU domain
- PLL can be defined been shut off or not in stop mode for each CPU domain

NOTE

The PLLs will only been closed in non-fast wake-up mode, relevant bit are
GPC_SLPCR[EN_A53_FASTWUP_STOP_MODE] and
GPC_SLPCR[EN_M4_FASTWUP_STOP_MODE]

- CPU clock can be defined been shut off or not in stop for each CPU (GPC_LPCR_A53_BSC[CPU_CLK_ON_LPM] and GPC_LPCR_M4[CPU_CLK_ON_LPM])
- Power of different power domain can be defined be shut off or not in stop mode for each platform domain
- Some peripherals may go to stop mode along with A53 or M4 platform.

5.2.4.3 Deep Sleep Mode

The Deep Sleep Mode (DSM) is a system low power mode.

In this mode:

- On-chip OSC can be defined to be shut off or not in DSM (GPC_SLPCR[SBYOS])
- PMIC can be defined to be stand-by mode or not in DSM (GPC_SLPCR[VSTBY])
- Regulator can be defined to be BYPASS mode or not in DSM (GPC_SLPCR[RBC_EN])
- Memory can be defined to go to retention mode or not in DSM(GPC_MLPCR[MEMLP_CTL_DIS])
- A53 platform power (VDD_ARM) can be defined to be shut off or not in DSM.

NOTE

CCM configuration must make sure close all PLLs before system goes to DSM

NOTE

For the SoC to correctly power up after entering DSM, CCM_PLL_CTRLx must not be set to 0x0 or 0x3 for any domain in use.

5.2.4.4 LPM Sleep Process

CPU platform will go to WAIT/STOP under the following conditions:

- LPM registers (GPC_LPCR_A53_BSC[LPM0], GPC_LPCR_A53_BSC[LPM1], GPC_LPCR_A53_BSC2[LPM2], GPC_LPCR_A53_BSC2[LPM3], GPC_LPCR_M4[LPM]) are set to WAIT or STOP.

  NOTE

  Since A53 platform has four cores, the so each core has its own LPM register: GPC_LPCR_A53_BSC[LPM0] and GPC_LPCR_A53_BSC[LPM1], GPC_LPCR_A53_BSC2[LPM2] and GPC_LPCR_A53_BSC2[LPM3]. The unified LPM of A53 will be generated with the lower LPM of the cores.

- Asserting the WFI signal will trigger CPU sleep process. There are five WFIs that come from A53 platform: WFI_core0, WFI_core1, WFI_core2, WFI_core3, and WFI_scu. The A53 platform will go to LPM when all WFIs are asserted. If the GPC_LPCR_A53_AD[EN_C0_WFI_PDN] or GPC_LPCR_A53_AD[EN_C1_WFI_PDN] or GPC_LPCR_A53_AD[EN_C2_WFI_PDN] or GPC_LPCR_A53_AD[EN_C3_WFI_PDN] bit is set, the LPM trigger condition will be a little different. See Power control for A53 Platform for more information. Only one WFI comes from M4 platform. The M4 platform will go to LPM when WFI_m4 asserted.

  NOTE

  WFI condition can be masked by register bits GPC_LPCR_A53_BSC[MASK_n_WFI] and GPC_LPCR_M4[MASK_M4_WFI]

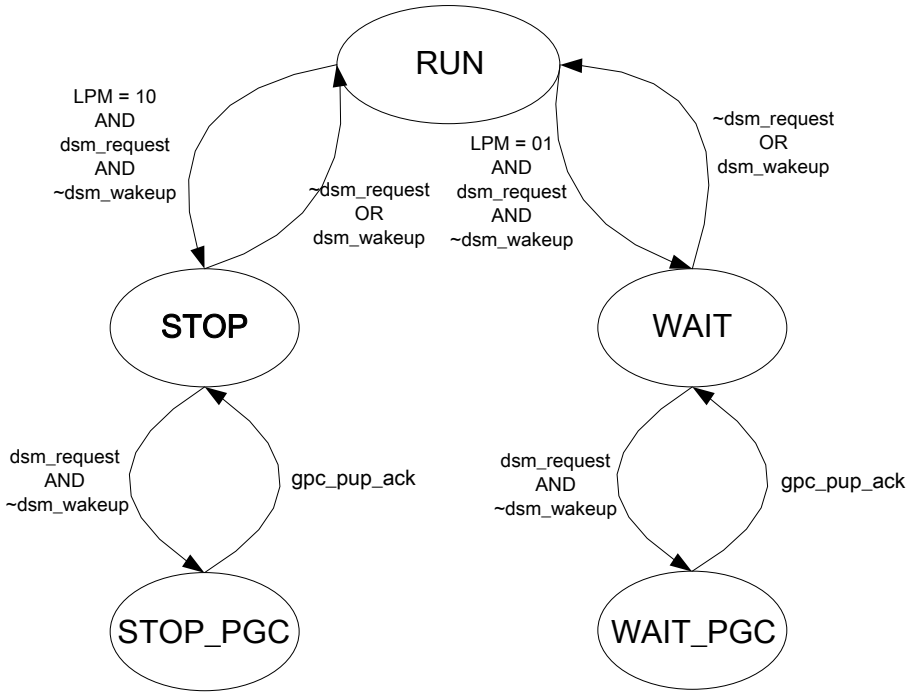


Figure 5-11. LPM transition inside CPU platform

System will go to DSM under the following conditions:

- Both A53 and M4 are STOP mode.
- Both GPC_SLPCR[EN_A53_FASTWUP_STOP_MODE] and GPC_SLPCR[EN_M4_FASTWUP_STOP_MODE] are not set.
- GPC_SLPCR[EN_DSM] is set.

NOTE

If GPC_LPCR_M4[MASK_DSM_TRIGGER] is set, the system will go to DSM when A53 goes STOP mode and GPC_SLPCR[EN_A53_FASTWUP_STOP_MODE] is not set. If GPC_LPCR_A53_BSC[MASK_DSM_TRIGGER] is set, the system will go to DSM when M4 goes to STOP mode and GPC_SLPCR[EN_M4_FASTWUP_STOP_MODE] not set. GPC_LPCR_M4[MASK_DSM_TRIGGER] and GPC_LPCR_A53_BSC[MASK_DSM_TRIGGER] cannot be set at the same time.

5.2.4.5 LPM Wake Up Process

DSM, STOP, WAIT mode will be woken up by interrupts:

- The CPU platforms share the same IRQ sources in this chip. Software can use GPC_IMRn_CORE0_A53, GPC_IMRn_CORE1_A53, GPC_IMRn_CORE2_A53, GPC_IMRn_CORE3_A53, and GPC_IMRn_M4 to separate the 128 bits IRQ sources to A53 core0, core1, core2, and core3, and M4 platform.
- The A53 core0, core1, core2, and core3 IRQ can also be from GIC source (defined by GPC_LPCR_A53_BSC[IRQ_SRC_C3], GPC_LPCR_A53_BSC[IRQ_SRC_C2], GPC_LPCR_A53_BSC[IRQ_SRC_C1], and GPC_LPCR_A53_BSC[IRQ_SRC_C0]) and if it is chosen from GIC source the GPC_IMRn_x_A53 will lose its function. See Power control for A53 Platform for more information.
- Interrupts for both A53 and M4 will cause the system wake up from DSM, A53 interrupt will wake up A53 from LPM, M4 interrupt will wake up M4 from LPM.

5.2.5 Power Gating Controller (PGC) Overview

The Power Gating Controller (PGC) is a power management component that controls the power-down and power-up sequencing of individual subsystems. For subsystems to be completely powered down in low power modes, a specific sequence of power control signals must be followed. The sequence timing is programmable using the PGC control registers.

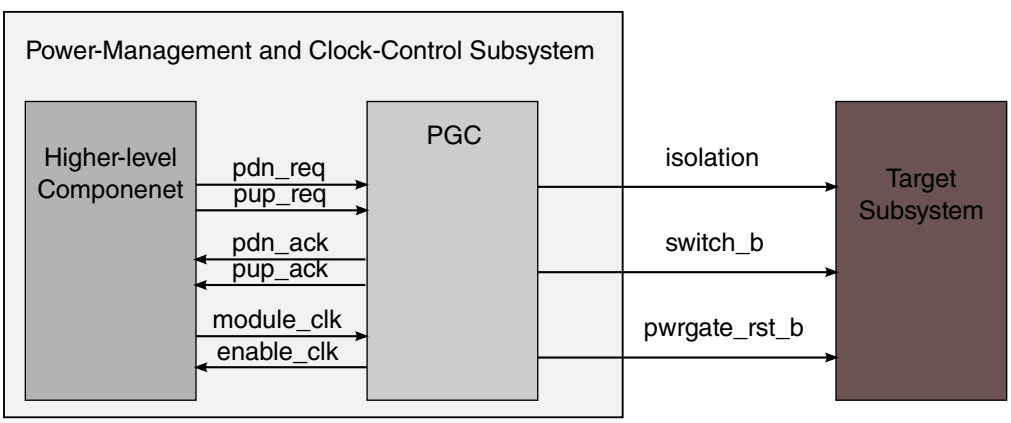


Figure 5-12. Power Gating Controller (PGC)

5.2.5.1 PGC power domains

The following table lists the PGCs in the chip and the corresponding power domain.

PGC type	Power domain
PGC_C0	Core0 of A53 platform
PGC_C1	Core1 of A53 platform
PGC_C2	Core2 of A53 platform
PGC_C3	Core3 of A53 platform
PGC_SCU	SCU/L2 cache RAM of A53 platform
PGC_NOC	NOC
PGC_PCIE	PCIE PHY
PGC_OTG1	USB OTG1 PHY
PGC_OTG2	USB OTG2 PHY
PGC_DDR1	DDR1
PGC_DISPMIX	DISPMIXMIX
GPC_MIPI	MIPI PHY
PGC_GPUMIX	GPUMIX
PGC_GPU_3D	GPU_3D
PGC_GPU_2D	GPU_2D
PGC_VPUMIX	VPUMIX
PGC_VPU_G1	VPU_G1
PGC_VPU_G2	VPU_G2
PGC_VPU_H1	VPU_H1

5.2.5.2 Trigger to PGC: Hardware and Software Requests

All PGCs can be power up/down by hardware or software request.

The LPM controller for each platform can generate hardware power down or power up request. All hardware requests will be mapped to “timeslot controller” before they goes to relevant PGC (see “Time slot control for PGCs” for more information).

The CPU can also generate software power up or power down request to relevant PGCs. The software trigger will not be mapped to timeslot control. If there are PGCs in software PDN/PUP sequence the request from LPM will be masked. The software trigger will also be failed if the timeslot control is in “busy” state.

All power up/down request to PGCs will be mapped to domain control module (see “Domain control for PGCs ”).

PGC_C0, PGC_C1, PGC_C2, and PGC_C3 can be triggered by its own “WFI/IRQ” without LPM trigger and time slot. See Power control for A53 Platform for more information.

5.2.5.3 Time slot control mechanism for PGCs

GPC uses a time slot controller to control the PGC sub-systems, such as the PGC in CPU0, CPU1,CPU2, CPU3, MIX, PCIE PHY, etc. We use it for below reasons when system wakes from low power mode.

1. Support flexible power down/up sequence for different sub-system
2. Sub-system can be power up in different slot to avoid large ramping up current in case they start ramping up at the same time

There are a total of 20 time slots used in the chip, one or more PGCs are used for power up or power down in each of these slots. The time slot controller will sample power up/down requests at slot0. If there are power up/down requests from SMC, it will scan from slot0 to slot19. When the scan process comes to one slot which has a defined power up or power down for one or more PGCs (defined by “SLTn_CFG”), the power up/down sequence of relevant PGCs will happen in that slot. Otherwise, the relevant slot will be skipped.

The next slot will not begin until the all PGCs finish their power up or power down process in current time slot. When all the 20 slots are finished, slot controller will jump to IDLE state and monitor new request from SMC.

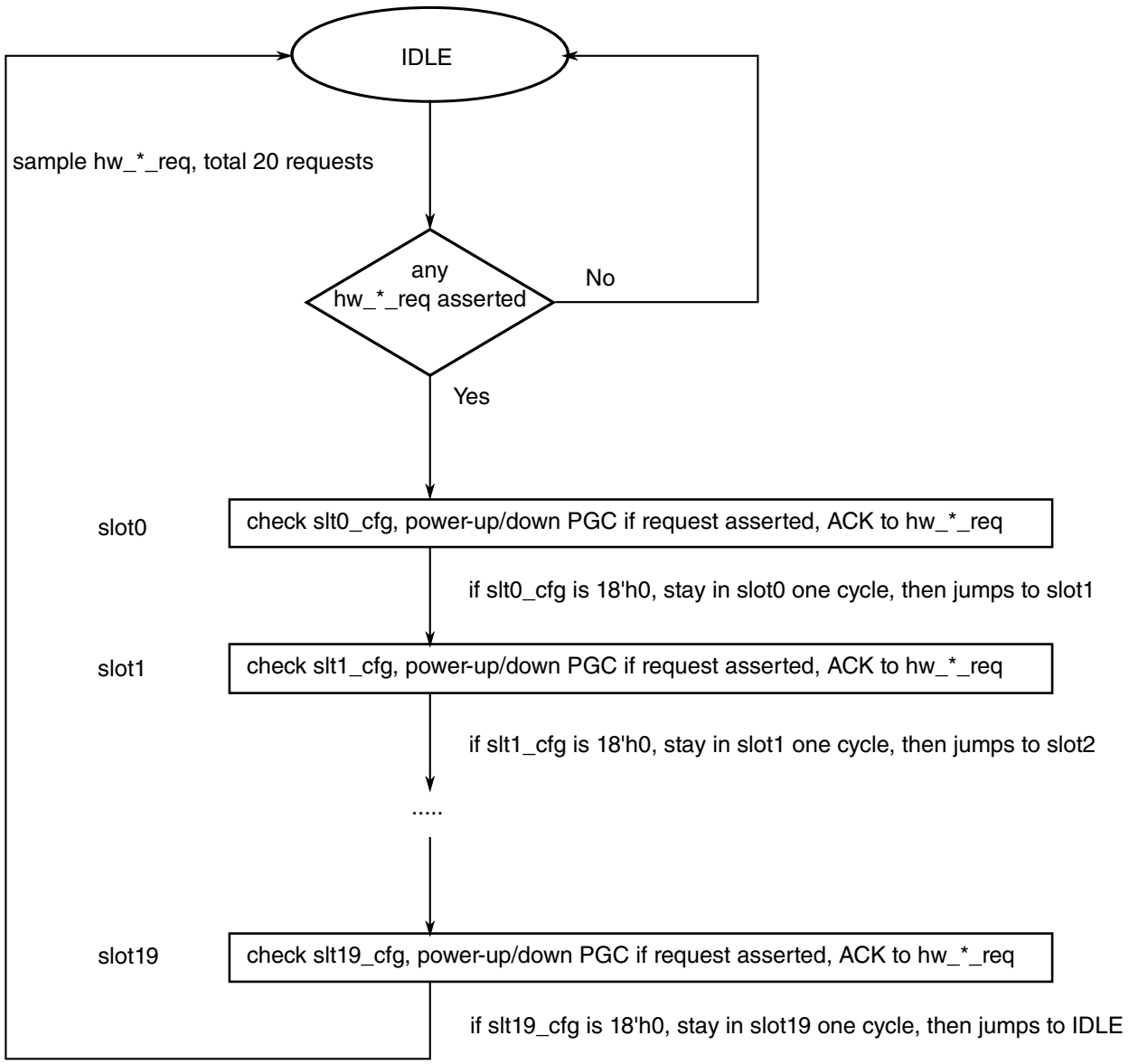


Figure 5-13. Slot controller processing flow

NOTE

PGC_SCU should be “always-on” to PGC_C0 PGC_C1, PGC_C2, and PGC_C3. This means PGC_SCU should be power up earlier than PGC_C0/PGC_C1/PGC_C2/PGC_C3 and should be power down later than PGC_C0/PGC_C1/ PGC_C2/PGC_C3 (see example code 1 and 2). If we arrange A53 Cx/A53 SCU power down/up in same slot, special setting is required (see example code 2).

NOTE

When the system enters/exists ALL_OFF or L2_RETENTION mode, PGC_MF should be power up earlier than PGC_C0/ PGC_C1/PGC_C2/PGC_C3/PGC_SCU. We can arrange MIX PGC power up in earlier slot than A53 Cx/SCU power up slot (See example code 1 and 2).

5.2.5.4 Handshake between LPM controller and time slot controller

The figure below shows an example of A53 “into” LPM sequence. We want to power up A53 SCU and A53 core0 in time slot0 /slot1/slot2 respectively. Request from low power mode controller will be mapped to relevant PGCs according to the rules listed above. We choose acknowledge from PGC_core0 as the acknowledge for A53 LPM power down request (relevant register bits are defined in “PGC_ACK_SEL_A53”). The A53 LPM will regard the three PGCs as a virtual big PGC and it will cancel all power up request when it receive the acknowledge signals from PGC_core0.

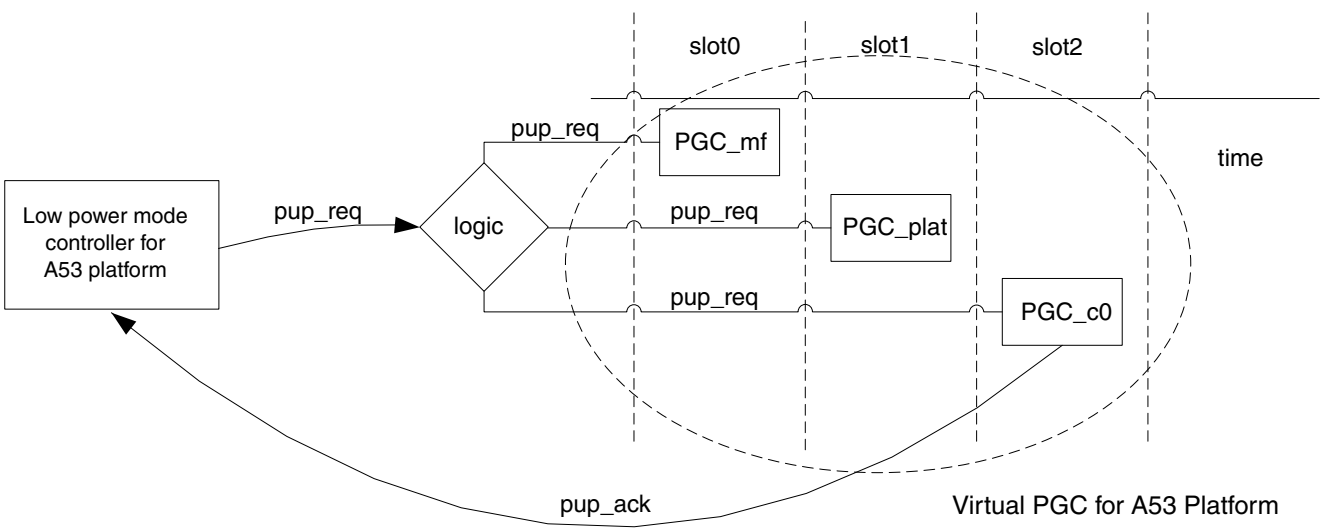


Figure 5-14. A53 into LPM sequence

NOTE

If a PGC is mapped to two CPU domain (refer to “Domain control for PGCs ”for more information), it cannot be selected as the power down acknowledge for both of the CPU platform. “PGC_ACK_SEL_A53”, "PGC_ACK_SEL_A53_PU", and “PGC_ACK_SEL_M4” are should be chosen for the last PGC in power up or power down sequence in the time slot. If there is no PGC be power up/power down with LPM sequence, the “dummy” acknowledge should be selected. Only one PGC

should be selected for power down or power up acknowledge for one CPU platform.

5.2.6 Power control for A53 Platform

5.2.6.1 A53 Platform power domains and power modes

There are four power domains inside SEC dual core Cortex A53 platform: Core0, Core1, Core2, Core3, SCU, and L2 RAM. There are six power states in A53 platform:

Power State	PDCPU0	PDCPU1	PDCPU2	PDCPU3	PDPLAT	PDL2	VDD_ARM
ALL_ON	ON	ON	ON	ON	ON	ON	ON
THREE_CPU_ON	3 CPUs are ON, 1 CPU is OFF				ON	ON	ON
TWO_CPU_ON	2 CPUs are ON, 2 CPUs are OFF				ON	ON	ON
ONE_CPU_ON	1 CPU is ON, 3 CPUs are OFF				ON	ON	ON
ALL_CPU_OFF	OFF	OFF	OFF	OFF	ON	ON	ON
L2_RETENTION	OFF	OFF	OFF	OFF	OFF	RET	ON
ALL_OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON
POWER_DOWN	OFF	OFF	OFF	OFF	OFF	OFF	OFF

NOTE

In all six power states, “ALL_ON”, “THREE_CPU_ON", "TWO_CPU_ON", and "ONE_CPU_ON" can exist in all RUN, WAIT or STOP mode of A53 platform. “L2_RETENTION” and “ALL_OFF” can only exist in WAIT or STOP mode of A53 platform.

5.2.6.2 Power down process for the A53 Platform

5.2.6.2.1 Power down of Core0, Core1, Core2, and Core3 in the A53 Platform

The power of core0, core1, core2, and core3 can be shut off along with the LPM process, as show in the following figure:

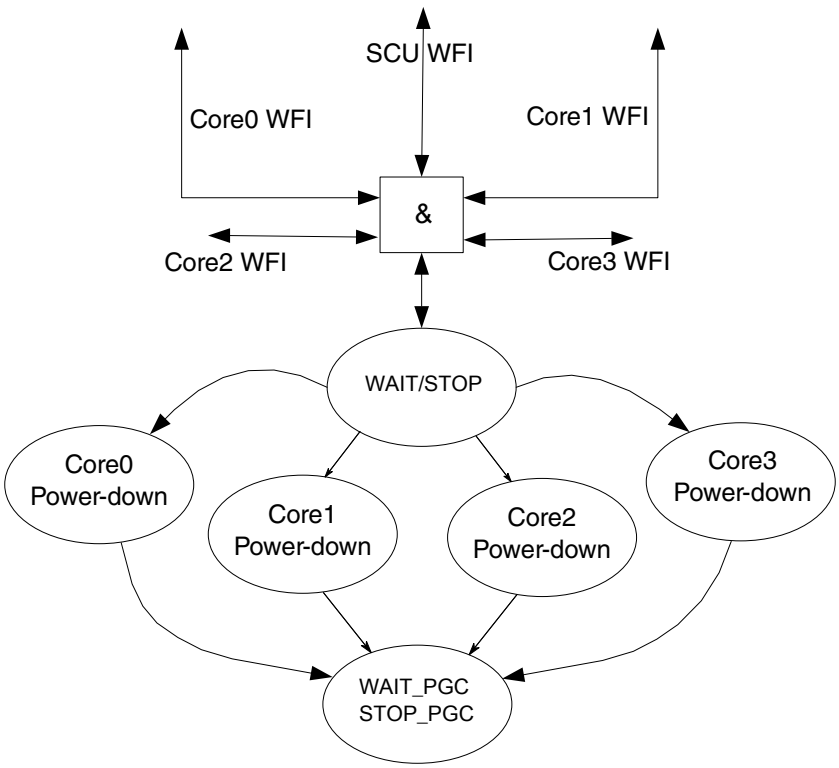


Figure 5-15. Power down of Core0, Core1, Core2, and Core3

WFIs from A53 platform will trigger the A53 platform LPM and the power of core0, core1, core2, or core3 will be shut off when “lpcr_a53_ad.en_c0_pdn”, “lpcr_a53_ad.en_c1_pdn”, “lpcr_a53_ad.en_c2_pdn”, or “lpcr_a53_ad.en_c3_pdn” enabled in this process. This mode should be used when core0 is used as the leading core of A53 platform.

The power of core0, core1, core2, and core3 can also be shut off in RUN mode: in this mode “LPCR_A53_AD.en_c0_wfi_pdn”, “LPCR_A53_AD.en_c1_wfi_pdn”, “LPCR_A53_AD.en_c2_wfi_pdn”, and “LPCR_A53_AD.en_c3_wfi_pdn” should be set and the condition to trigger A53 LPM will be some different:

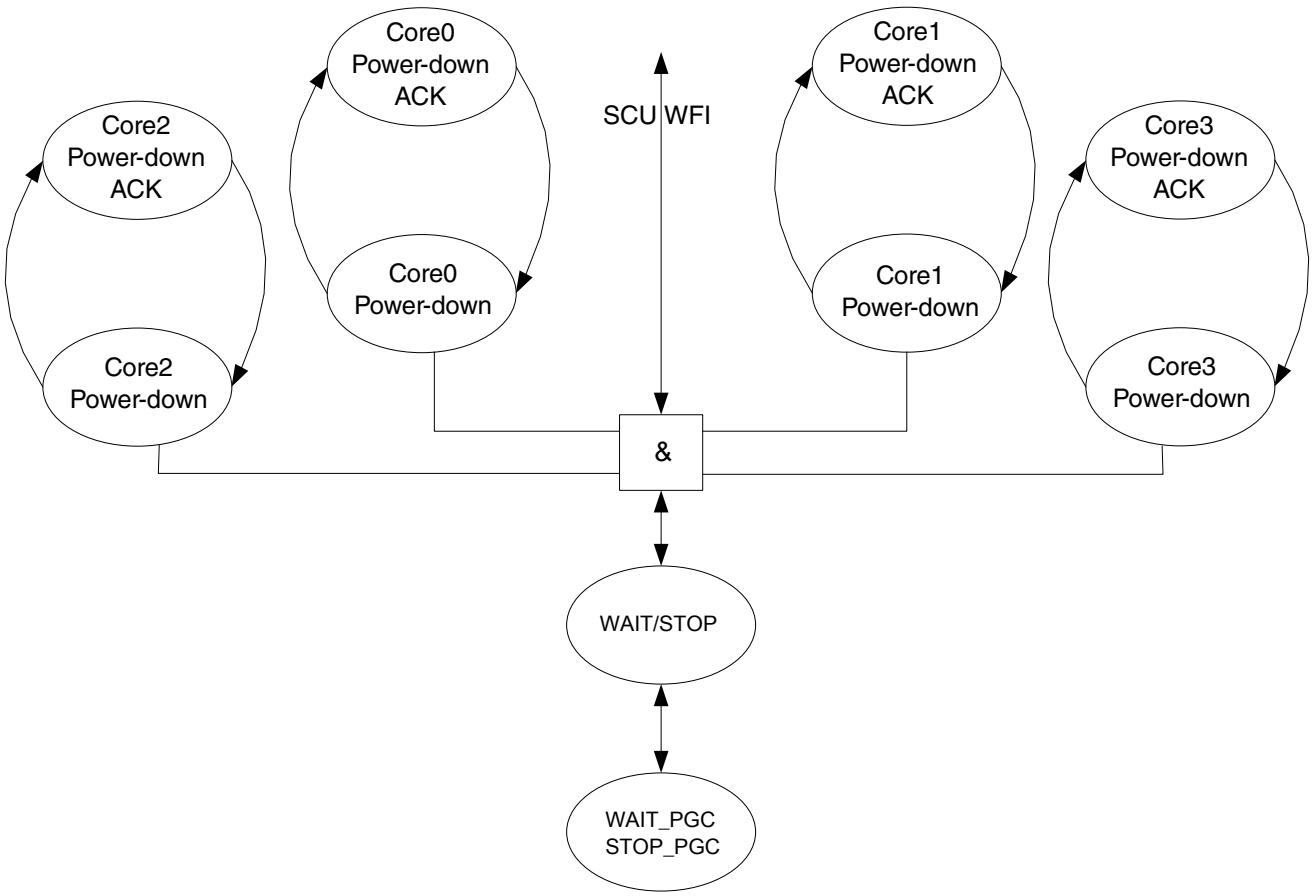


Figure 5-16. Power down of Core0, Core1, Core2, and Core3 in RUN mode

In this mode, core0/core1/core2/core3 power down process should not be disturbed by IRQs.

5.2.6.2.2 Power down of SCU and L2 Cache RAM

Power domain SCU and L2 cache RAM is controlled by one PGC – "PGC_PLAT" and they can only be power down in LPM process (when "LPCR_A53_AD.en_plat_pdn" is set). There is another bit "LPCR_A53_AD.l2pge" which will decide if L2 cache RAM need to be in retention mode when SCU domain is power down.

5.2.6.3 Power up process for the A53 Platform

- The core0, core1, core2, and core3 can be powered up with the exit of A53 LPM. The relevant bits are "LPCR_A53_AD.en_c0_pup", "LPCR_A53_AD.en_c1_pup", "LPCR_A53_AD.en_c2_pup", and "LPCR_A53_AD.en_c3_pup".
- The core0, core1, core2, and core3 can also be powered up by interrupt signal in RUN mode.

- The relevant bit are “LPCR_A53_AD.en_c0_irq_pup”, “LPCR_A53_AD.en_c1_irq_pup”, “LPCR_A53_AD.en_c2_irq_pup”, and “LPCR_A53_AD.en_c3_irq_pup”.
- The interrupt signal can be chosen from GIC or directly from IRQ with mask in GPCv2.
- The relevant select bits are “LPCR_A53_BSC.irq_src_c0”, “LPCR_A53_BSC.irq_src_c1”, “LPCR_A53_BSC.irq_src_c2”, “LPCR_A53_BSC.irq_src_c3”, and “LPCR_A53_BSC.irq_src_a53_wup”. (LPCR_A53_BSC[30],LPCR_A53_BSC[23:22],LPCR_A53_BSC.)

{LPCR_A53_BSC[30],LPCR_A53_BSC[23:22],LPCR_A53_BSC[29:28]}	Usage	Restriction
5'b00000	Use IRQ trigger A53 LPM and use IRQ to power up core0 to core3	None
5'b01111	Use GIC trigger A53 LPM and GIC to power up core0 to core3	SCU cannot power down in LPM, CPU clock cannot stop in LPM
5'b11111	Use IRQ trigger A53 LPM and GIC to power up core0 to core3	SCU cannot power down in LPM

As show in the table above, core0/core1/core2/core3 can only be power up by its own interrupt in RUN mode of A53 platform.

There are three combination of {LPCR_A53_BSC[30],LPCR_A53_BSC[23:22],LPCR_A53_BSC[29:28]}:

1. In the first case, “IMRn_CORE0_A53, IMRn_CORE1_A53, IMRn_CORE2_A53, IMRn_CORE3_A53“ are used to separate the 128 bits interrupts for core0, core1, core2, and core3 of A53 platform and also used as the interrupt mask for A53 LPM.
2. In the second case, “IMRn_CORE0_A53, IMRn_CORE1_A53, IMRn_CORE2_A53, IMRn_CORE3_A53” are not used, GIC setting are used to separate interrupts for core0, core1, core2, and core3 of A53 platform and also used as the interrupt mask for A53 LPM.
3. In the third case, “IMRn_CORE0_A53, IMRn_CORE1_A53, IMRn_CORE2_A53, IMRn_CORE3_A53” is used as the mask for interrupt for A53 LPM, GIC setting are used to separate interrupts for core0, core1, core2, and core3.

5.2.7 Power control for the M4 Platform

M4 LPM is same as A53 LPM. M4 platform doesn’t have its own power domain. There is a virtual PGC reserved for M4, the virtual PGC will do nothing else except generating an acknowledge signal and this signal can be chosen as the acknowledge signal for M4 platform.

5.2.8 Domain control for PGCs

The following rules are used for PGC power up/down with domain mapping control:

1. For PGCs inside CPU platform (since M4 doesn't have its own power domain, only PGCs in A53 platform are referred): for hardware trigger (including both power up and power down) only the LPM request from its own platform will take effect; for software trigger (including both power up and power down) (the relevant register bit are "CPU_PGC_SW_PUP_REQ" and "CPU_PGC_SW_PDN_REQ"), only the software running in its own platform will take effect.
2. For PGCs outside CPU platform(MIX and PU PGCs), register bits "PGC_CPU_MAPPING" will map MIX and PU PGCs to A53 or M4 CPU domain:
   - One PGC can be mapped to one or both of A53 and M4 domain;
   - For hardware power up request, if a PGC is mapped to any CPU domain, the PGC will be powered up when the corresponding CPU platform sends out its power up request.
   - For software power up, if a PGC is mapped any CPU platform, the PGC can be powered up by software running in corresponding CPU.(the relevant register bits are "MIX_PGC_SW_PUP_REQ" and "PU_PGC_SW_PUP_REQ")
   - For hardware power down, if a power domain is mapped to only one CPU domain, the relevant PGC can be powered down when the corresponding CPU platform sends out its power down request; If a power domain is mapped to both two CPU domain, when one CPU platform sends out its hardware power down request, the relevant PGC will power down with any of the following condition satisfied: the other CPU already in LPM; the other CPU want to power down relevant PGC (the relevant register are "A53_MIX_PDN_FLG, M4_MIX_PDN_FLG, A53_PU_PDN_FLG, M4_PU_PDN_FLG")
   - For software power down, if a power domain is mapped to only one CPU domain, the PGC can be powered down by software running in corresponding CPU. (the relevant register bits are "MIX_PGC_SW_PUP_REQ" and "PU_PGC_SW_PUP_REQ"). If a power domain is mapped to both two CPU domain, when one CPU platform sends out its software power down request, the relevant PGC will power down with any of the following condition satisfied: the other CPU already in LPM; the other CPU want to power down relevant PGC (the relevant register are "A53_MIX_PDN_FLG, M4_MIX_PDN_FLG, A53_PU_PDN_FLG, M4_PU_PDN_FLG")
   - The access of "PGC_CPU_MAPPING" is controlled by domain information from RDC

5.2.9 Example Code

Below are code examples for entering specific power scenarios

5.2.9.1 Example Code 1

```
//ARM enters into ALL_OFF(STOP) mode and enable DSM :
//after "wfi", MIX/C0/C1/C2/C3 power down in SLOT0, SCU power down in SLOT1 when
//after "GPT1_INT" arrived, MIX power up in SLOT2, SCU power up in SLOT3, C0 power up in
SLOT4
//IMRx_CORE0_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x30, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x34, 0xFF7FFFFF); //[23] : GPT1 used as wakeup source
reg32_write(GPC_IPS_BASE_ADDR + 0x38, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x3C, 0xFFFFFFFF);
//IMRx_CORE1_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x40, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x44, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x48, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x4C, 0xFFFFFFFF);
//IMRx_CORE2_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x1C0, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1C4, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1C8, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1CC, 0xFFFFFFFF);
//IMRx_CORE3_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x1D0, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1D4, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1D8, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1DC, 0xFFFFFFFF);
//LPCR_A53_BSC
reg32_write(GPC_IPS_BASE_ADDR, 0x0000000A) ;
//[30],[23:22],[29:28] : A53_CO/A53_C1/A53_C2/A53_C3/LPM wakeup from external INT
//[14] : CLOCK OFF during STOP mode
//[3:0] : STOP mode
//LPCR_A53_BSC2
reg32_write(GPC_IPS_BASE_ADDR + 0x108, 0x0000000A) ;
//[3:0] : STOP mode
//LPCR_A53_AD
reg32_write(GPC_IPS_BASE_ADDR + 0x4, 0x0A0A0A1A) ;
//[16] : 1(ALL_OFF mode); 0(L2 retention mode)
//[27]/[25]/[11]/[9]: A53_C0/A53_C1/A53_C2/A53_C3 power up with A53 LPM PUP REQ
//[4] : A53_SCU power down with A53 LPM PDN REQ
//[19]/[17]/[3]/[1] : A53_C0/A53_C1/A53_C2/A53_C3 powr down with A53 LPM PDN REQ
//LPCR_M4
reg32_write(GPC_IPS_BASE_ADDR + 0x8, 0x80000000) ;
//[31] : DSM ingore to check M4 low power state
//[1:0] : M4 LPM run mode
//SLPCR
reg32_write(GPC_IPS_BASE_ADDR + 0x14,0xe000ffA7) ;
//[31] : enable DSM
//[30] : eneable regulator bypass
//[5:3] : wait 64 ckil clock cycles
//[2] : enable PMIC standby
//[1] : enable OSC power down
//[0] : bypass PMIC ready handshake
//PGC_ACK_SEL_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x24,0x00010004) ;
//[2] : A53_SCU PGC as LPM power down ack
//[16] : A53_C0 PGC as LPM power up ack
//SLT_CFG0
```

```
reg32_write(GPC_IPS_BASE_ADDR + 0xB0,0x00000055) ;
//[6]/[4]/[2]/[0] : A53_C0/A53_C1/ A53_C2/A53_C3 power down in SLOT0
//SLT_CFG0_PU
reg32_write(GPC_IPS_BASE_ADDR + 0x200,0x0000001) ;
//[0] : power down in SLOT0
//SLT_CFG1
reg32_write(GPC_IPS_BASE_ADDR + 0xB4,0x00000100) ;
//[8] : A53_SCU power down in SLOT1
//SLT_CFG2_PU
reg32_write(GPC_IPS_BASE_ADDR + 0x208,0x00000002) ;
//[1] : power up in SLOT2
//SLT_CFG3
reg32_write(GPC_IPS_BASE_ADDR + 0xBC,0x00000200) ;
//[9] : A53_SCU power up in SLOT3
//SLT_CFG4
reg32_write(GPC_IPS_BASE_ADDR + 0xC0,0x00000002) ;
//[1] : A53_C0 power up in SLOT4, A53_C1/ A53_C2/A53_C3 not power up
//PGC_CPU_MAPPING
reg32_write(GPC_IPS_BASE_ADDR + 0xEC,0x00000001) ;
//[0] : MIX PGC mapping to A53 LPM
//A53 PGC
reg32_write(GPC_IPS_BASE_ADDR +0x800, reg32_read(GPC_IPS_BASE_ADDR +0x800) | 0x00000001) ;
// enable A53_C0 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x840, reg32_read(GPC_IPS_BASE_ADDR +0x840) | 0x00000001) ;
// enable A53_C1 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x880, reg32_read(GPC_IPS_BASE_ADDR +0x880) | 0x00000001) ;
// enable A53_C2 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x8C0, reg32_read(GPC_IPS_BASE_ADDR +0x8C0) | 0x00000001) ;
// enable A53_C3 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x900, reg32_read(GPC_IPS_BASE_ADDR +0x900) | 0x00000001) ;
// enable A53_SCU PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x910, (0x59 lt:lt: 10) | 0x5B | (0x51 lt:lt: 20) ) ;
// change nL2retn/mempwr/dftram to meet SCU power up timing
// PGC
reg32_write(GPC_IPS_BASE_ADDR +0xA00, reg32_read(GPC_IPS_BASE_ADDR +0xA00) | 0x00000001) ;
// enable MIX PGC power down
```

5.2.9.2 Example Code 2

```
//ARM enters into L2_RETENTION(STOP) mode and enable DSM :
//after "wfi", MIX/C0/C1/C2/C3/SCU power down in SLOT0
//after "GPT1_INT" arrived, MIX power up in SLOT1, SCU/C0 power up in SLOT2
//IMRx_CORE0_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x30, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x34, 0xFF7FFFFF);
//[23] : GPT1 used as wakeup source
reg32_write(GPC_IPS_BASE_ADDR + 0x38, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x3C, 0xFFFFFFFF);
//IMRx_CORE1_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x40, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x44, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x48, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x4C, 0xFFFFFFFF);
//IMRx_CORE2_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x1C0, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1C4, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1C8, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1CC, 0xFFFFFFFF);
//IMRx_CORE3_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x1D0, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1D4, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1D8, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1DC, 0xFFFFFFFF);
```

```
//LPCR_A53_BSC
reg32_write(GPC_IPS_BASE_ADDR, 0x0000000A) ;
//[30],[23:22],[29:28] : A53_C0/A53_C1/A53_C2/A53_C3/LPM wakeup from external INT
//[14] : CLOCK OFF during STOP mode
//[3:0] : STOP mode
//LPCR_A53_BSC2
reg32_write(GPC_IPS_BASE_ADDR + 0x108, 0x0000000A) ;
//[3:0] : STOP mode
//LPCR_A53_AD
reg32_write(GPC_IPS_BASE_ADDR + 0x4, 0x0A0A0A1A) ;
//[16] : 1(ALL_OFF mode); 0(L2 retention mode)
//[27]/[25]/[11]/[9]: A53_C0/A53_C1/A53_C2/A53_C3 power up with A53 LPM PUP REQ
//[4] : A53_SCU power down with A53 LPM PDN REQ
//[19]/[17]/[3]/[1] : A53_C0/A53_C1/A53_C2/A53_C3 powr down with A53 LPM PDN REQ
//LPCR_M4
reg32_write(GPC_IPS_BASE_ADDR + 0x8, 0x80000000) ;
//[31] : DSM ingore to check M4 low power state
//[1:0] : M4 LPM run mode
//SLPCR
reg32_write(GPC_IPS_BASE_ADDR + 0x14,0xe000ffA7) ;
//[31] : enable DSM
//[30] : eneable regulator bypass
//[5:3] : wait 64 ckil clock cycles
//[2] : enable PMIC standby
//[1] : enable OSC power down
//[0] : bypass PMIC ready handshake
//PGC_ACK_SEL_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x24,0x00010004) ;
//[2] : A53_SCU PGC as LPM power down ack
//[16] : A53_C0 PGC as LPM power up ack
//SLT_CFG0
reg32_write(GPC_IPS_BASE_ADDR + 0xB0,0x00000155) ;
//[6]/[4]/[2]/[0] : A53_C0/A53_C1/ A53_C2/A53_C3 power down in SLOT0
//[8] : A53_SCU power down in SLOT0
//SLT_CFG0_PU
reg32_write(GPC_IPS_BASE_ADDR + 0x200,0x0000001) ;
//[0] : power down in SLOT0
//A53_Cx/SCU are power down in same slot. Special setting is required( see below PGC setting
#A )
//SLT_CFG1_PU
reg32_write(GPC_IPS_BASE_ADDR + 0x204,0x00000002) ;
//[1] : power up in SLOT1
//SLT_CFG2
reg32_write(GPC_IPS_BASE_ADDR + 0xB8,0x00000202) ;
//[9] : A53_SCU power up in SLOT1
//[1] : A53_C0 power up in SLOT1, A53_C1/ A53_C2/ A53_C3 not power up
//A53_Cx/SCU are power up in same slot. Special setting is required( see below PGC setting
#B )
//PGC_CPU_MAPPING
reg32_write(GPC_IPS_BASE_ADDR + 0xEC,0x00000001) ;
//[0] : MIX PGC mapping to A53 LPM
//Special PGC setting #A for C0/C1/C2/C3/SCU power down in same slot(SCU should be always ON
comparing to C0/C1/C2/C3)
reg32_write(GPC_IPS_BASE_ADDR +0x808, (reg32_read(GPC_IPS_BASE_ADDR +0x808) & 0xFFFFC0C0) |
0x0801 ) ;
// set C0.ISO2SW = 8 ; C0.ISO = 1;
reg32_write(GPC_IPS_BASE_ADDR +0x848, (reg32_read(GPC_IPS_BASE_ADDR +0x848) & 0xFFFFC0C0) |
0x0801 ) ;
// set C1.ISO2SW = 8 ; C1.ISO = 1;
reg32_write(GPC_IPS_BASE_ADDR +0x888, (reg32_read(GPC_IPS_BASE_ADDR +0x888) & 0xFFFFC0C0) |
0x0801 ) ;
// set C2.ISO2SW = 8 ; C2.ISO = 1;
reg32_write(GPC_IPS_BASE_ADDR +0x8C8, (reg32_read(GPC_IPS_BASE_ADDR +0x8C8) & 0xFFFFC0C0) |
0x0801 ) ;
// set C3.ISO2SW = 8 ; C3.ISO = 1;
reg32_write(GPC_IPS_BASE_ADDR +0x908, (reg32_read(GPC_IPS_BASE_ADDR +0x908) & 0xFFFFC0C0) |
0x1001 ) ;
// set SCU.ISO2SW = 16 ; SCU.ISO = 1;
//Special PGC setting #B for A53_Cx/SCU power up in same slot(SCU should be always ON
comparing to C0/C1/C2/C3)
```

```
reg32_write(GPC_IPS_BASE_ADDR +0x804, (reg32_read(GPC_IPS_BASE_ADDR +0x804) & 0xFF800040) |
0x11 | (0x20 lt;lt; 7) ) ;
// set C0.SW = 0x11 ; C0.SW2ISO = 0x20 ;
reg32_write(GPC_IPS_BASE_ADDR +0x844, (reg32_read(GPC_IPS_BASE_ADDR +0x844) & 0xFF800040) |
0x11 | (0x20 lt;lt; 7) ) ;
// set C1.SW = 0x11 ; C1.SW2ISO = 0x20 ;
reg32_write(GPC_IPS_BASE_ADDR +0x884, (reg32_read(GPC_IPS_BASE_ADDR +0x884) & 0xFF800040) |
0x11 | (0x20 lt;lt; 7) ) ;
// set C2.SW = 0x11 ; C2.SW2ISO = 0x20 ;
reg32_write(GPC_IPS_BASE_ADDR +0x8C4, (reg32_read(GPC_IPS_BASE_ADDR +0x8C4) & 0xFF800040) |
0x11 | (0x20 lt;lt; 7) ) ;
// set C3.SW = 0x11 ; C3.SW2ISO = 0x20 ;
reg32_write(GPC_IPS_BASE_ADDR +0x904, (reg32_read(GPC_IPS_BASE_ADDR +0x904) & 0xFF800040) |
0x1 | (0x0f lt;lt; 7) ) ;
// set SCU.SW = 0x1 ; SCU.SW2ISO = 0x0f ;
//A53 PGC
reg32_write(GPC_IPS_BASE_ADDR +0x800, reg32_read(GPC_IPS_BASE_ADDR +0x800) | 0x00000001) ;
// enable A53_C0 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x840, reg32_read(GPC_IPS_BASE_ADDR +0x840) | 0x00000001) ;
// enable A53_C1 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x880, reg32_read(GPC_IPS_BASE_ADDR +0x880) | 0x00000001) ;
// enable A53_C2 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x8C0, reg32_read(GPC_IPS_BASE_ADDR +0x8C0) | 0x00000001) ;
// enable A53_C3 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x900, reg32_read(GPC_IPS_BASE_ADDR +0x900) | 0x00000001) ;
// enable A53_SCU PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x910, (0x59 lt;lt; 10) | 0x5B | (0x51 lt;lt; 20) ) ;
// change nL2retn/mempwr/dftram to meet SCU power up timing
// PGC
reg32_write(GPC_IPS_BASE_ADDR +0xA00, reg32_read(GPC_IPS_BASE_ADDR +0xA00) | 0x00000001) ;
// enable MIX PGC power down
```

5.2.9.3 Example Code 3

```
//A53/M4 both enters into low power mode. A53/M4 are in different master domain.
//MIX are mapping to both A53 and M4
//after A53/M4 enters into low power mode, MIX will be also power down.A53/M4 enters into
low power mode any time
//either A53 or M4 exists from low power mode, MIX will be also power up
//IMRx_CORE0_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x30, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x34, 0xFF7FFFFF);
//[23] : GPT1 used as ARM wakeup source
reg32_write(GPC_IPS_BASE_ADDR + 0x38, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x3C, 0xFFFFFFFF);
//IMRx_CORE1_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x40, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x44, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x48, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x4C, 0xFFFFFFFF);
//IMRx_CORE2_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x1C0, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1C4, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1C8, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1CC, 0xFFFFFFFF);
//IMRx_CORE3_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x1D0, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1D4, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1D8, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1DC, 0xFFFFFFFF);
//IMRx_M4
reg32_write(GPC_IPS_BASE_ADDR + 0x50, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x54, 0xFFBFFFFF);
```

```
//[22] : GPT2 used as M4 wakeup source
reg32_write(GPC_IPS_BASE_ADDR + 0x58, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x5C, 0xFFFFFFFF);
//LPCR_A53_BSC
reg32_write(GPC_IPS_BASE_ADDR, 0x0000000A) ;
//[30],[23:22],[29:28] : A53_C0/A53_C1/A53_C2/A53_C3/LPM wakeup from external INT
//[14] : CLOCK OFF during STOP mode
//[3:0] : STOP mode
//LPCR_A53_BSC2
reg32_write(GPC_IPS_BASE_ADDR + 0x108, 0x0000000A) ;
//[3:0] : STOP mode
//LPCR_A53_AD
reg32_write(GPC_IPS_BASE_ADDR + 0x4, 0x0A0A0A1A) ;
//[16] : 1(ALL_OFF mode); 0(L2 retention mode)
//[27]/[25]/[11]/[9]: A53_C0/A53_C1/A53_C2/A53_C3 power up with A53 LPM PUP REQ
//[4] : A53_SCU power down with A53 LPM PDN REQ
//[19]/[17]/[3]/[1] : A53_C0/A53_C1/A53_C2/A53_C3 powr down with A53 LPM PDN REQ
//LPCR_M4
reg32_write(GPC_IPS_BASE_ADDR + 0x8, 0x00003FFE) ;
//[31] : 0(check M4 low power state to enter into DSM)
//[14] : 0(M4 clock OFF during low power mode)
//[3:2] : enable M4 virtual PGC power up/down with M4 LPM
//[1:0] : M4 LPM STOP mode
//SLPCR
reg32_write(GPC_IPS_BASE_ADDR + 0x14,0xe000ffA7) ;
//[31] : enable DSM
//[30] : eneable regulator bypass
//[5:3] : wait 64 ckil clock cycles
//[2] : enable PMIC standby
//[1] : enable OSC power down
//[0] : bypass PMIC ready handshake
//PGC_ACK_SEL_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x24,0x00010004) ;
//[2] : A53_SCU PGC as LPM power down ack
//[16] : A53_C0 PGC as LPM power up ack
//PGC_ACK_SEL_M4
reg32_write(GPC_IPS_BASE_ADDR + 0x28,0x00010001) ;
//[0] : M4 virtual PGC as M4 LPM power down ack
//[16] : M4 virtual PGC as M4 LPM power up ack
//GPC_MISC
reg32_write(GPC_IPS_BASE_ADDR + 0x2C,reg32_read(GPC_IPS_BASE_ADDR + 0x2C) | 0x100) ;
//[8] : not mask M4 power down request to M4 virtual PGC
//SLT_CFG0
reg32_write(GPC_IPS_BASE_ADDR + 0xB0,0x00000155) ;
//[6]/[4]/[2]/[0] : A53_C0/A53_C1/ A53_C2/A53_C3 power down in SLOT0
//[8] : A53_SCU power down in SLOT0
//SLT_CFG0_PU
reg32_write(GPC_IPS_BASE_ADDR + 0x200,0x0001001) ;
//[0] : power down in SLOT0
//[12] : M4 virtual PGC power down in SLOT0
//A53_Cx/SCU are power down in same slot. Special setting is required( see below PGC setting
#A )
//SLT_CFG1_PU
reg32_write(GPC_IPS_BASE_ADDR + 0x204,0x00002002) ;
//[2] : power up in SLOT1
//[13] : M4 virtual PGC power up in SLOT1
//SLT_CFG2
reg32_write(GPC_IPS_BASE_ADDR + 0xB8,0x00000202) ;
//[9] : A53_SCU power up in SLOT1
//[1] : A53_C0 power up in SLOT1, A53_C1/ A53_C2/ A53_C3 not power up
//A53_Cx/SCU are power up in same slot. Special setting is required( see below PGC setting
#B )
//PGC_CPU_MAPPING
reg32_write(GPC_IPS_BASE_ADDR + 0xEC,0x00010001) ;
//[0] : MIX PGC mapping to A53 LPM
//[16] : MIX PGC mapping to M4 LPM
//Special PGC setting #A for C0/C1/SCU power down in same slot(SCU should be always ON
comparing to C0/C1/C2/C3)
reg32_write(GPC_IPS_BASE_ADDR +0x808, (reg32_read(GPC_IPS_BASE_ADDR +0x808) & 0xFFFFC0C0) |
0x0801 ) ;
```

```
// set C0.ISO2SW = 8 ; C0.ISO = 1;
reg32_write(GPC_IPS_BASE_ADDR +0x848, (reg32_read(GPC_IPS_BASE_ADDR +0x848) & 0xFFFFC0C0) |
0x0801 ) ;
// set C1.ISO2SW = 8 ; C1.ISO = 1;
reg32_write(GPC_IPS_BASE_ADDR +0x848, (reg32_read(GPC_IPS_BASE_ADDR +0x848) & 0xFFFFC0C0) |
0x0801 ) ;
// set C2.ISO2SW = 8 ; C2.ISO = 1;
reg32_write(GPC_IPS_BASE_ADDR +0x888, (reg32_read(GPC_IPS_BASE_ADDR +0x888) & 0xFFFFC0C0) |
0x0801 ) ;
// set C3.ISO2SW = 8 ; C3.ISO = 1;
reg32_write(GPC_IPS_BASE_ADDR +0x908, (reg32_read(GPC_IPS_BASE_ADDR +0x908) & 0xFFFFC0C0) |
0x1001 ) ;
// set SCU.ISO2SW = 16 ; SCU.ISO = 1;
//Special PGC setting #B for A53_Cx/SCU power up in same slot(SCU should be always ON
comparing to C0/C1/C2/C3)
reg32_write(GPC_IPS_BASE_ADDR +0x804, (reg32_read(GPC_IPS_BASE_ADDR +0x804) & 0xFF800040) |
0x10 | (0x1f lt:lt: 7) ) ;
// set C0.SW = 0x10 ; C0.SW2ISO = 0x1f ;
reg32_write(GPC_IPS_BASE_ADDR +0x844, (reg32_read(GPC_IPS_BASE_ADDR +0x844) & 0xFF800040) |
0x10 | (0x1f lt:lt: 7) ) ;
// set C1.SW = 0x10 ; C1.SW2ISO = 0x1f ;
reg32_write(GPC_IPS_BASE_ADDR +0x844, (reg32_read(GPC_IPS_BASE_ADDR +0x844) & 0xFF800040) |
0x10 | (0x1f lt:lt: 7) ) ;
// set C2.SW = 0x10 ; C2.SW2ISO = 0x1f ;
reg32_write(GPC_IPS_BASE_ADDR +0x884, (reg32_read(GPC_IPS_BASE_ADDR +0x884) & 0xFF800040) |
0x10 | (0x1f lt:lt: 7) ) ;
// set C3.SW = 0x10 ; C3.SW2ISO = 0x1f ;
reg32_write(GPC_IPS_BASE_ADDR +0x904, (reg32_read(GPC_IPS_BASE_ADDR +0x904) & 0xFF800040) |
0x1 | (0x0f lt:lt: 7) ) ;
// set SCU.SW = 0x1 ; SCU.SW2ISO = 0x0f ;
//A53 PGC
reg32_write(GPC_IPS_BASE_ADDR +0x800, reg32_read(GPC_IPS_BASE_ADDR +0x800) | 0x00000001) ;
// enable A53_C0 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x840, reg32_read(GPC_IPS_BASE_ADDR +0x840) | 0x00000001) ;
// enable A53_C1 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x840, reg32_read(GPC_IPS_BASE_ADDR +0x840) | 0x00000001) ;
// enable A53_C2 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x880, reg32_read(GPC_IPS_BASE_ADDR +0x880) | 0x00000001) ;
// enable A53_C3 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x900, reg32_read(GPC_IPS_BASE_ADDR +0x900) | 0x00000001) ;
// enable A53_SCU PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x910, (0x59 lt:lt: 10) | 0x5B | (0x51 lt:lt: 20) ) ;
// change nL2retn/mempwr/dftram to meet SCU power up timing
// PGC
reg32_write(GPC_IPS_BASE_ADDR +0xA00, reg32_read(GPC_IPS_BASE_ADDR +0xA00) | 0x00000001) ;
// enable MIX PGC power down
```

5.2.9.4 Example Code 4

```
// software power up/down PCIE PHY
//power up PCIE PHY
reg32_write ( GPC_IPS_BASE_ADDR + 0xEC, reg32_read(GPC_IPS_BASE_ADDR + 0x0EC) | 0x8 );
//map PCIE PGC to A53
reg32_write ( GPC_IPS_BASE_ADDR + 0xF8 , reg32_read(GPC_IPS_BASE_ADDR + 0xF8) | 0x2 );
//trigger sw Power up Request
while( read(GPC_IPS_BASE_ADDR + 0xF8) & 0x2 );
//wait software power up request self clear
//power down PCIE PHY
reg32_write ( GPC_IPS_BASE_ADDR + 0xEC, reg32_read(GPC_IPS_BASE_ADDR + 0x0EC) | 0x8 );
//map PCIE PGC to A53
reg32_write ( GPC_IPS_BASE_ADDR + 0xC40, reg32_read(GPC_IPS_BASE_ADDR + 0xC40) | 0x1 );
// enable PCIE PGC power down
reg32_write ( GPC_IPS_BASE_ADDR + 0x104 , reg32_read(GPC_IPS_BASE_ADDR + 0x104) | 0x2 );
```

```
//trigger sw Power Down Request
while( read(GPC_IPS_BASE_ADDR + 0x104) & 0x2 );
//wait software power down request self clear
```

5.2.9.5 Example Code 5

```
// VPU/GPU/DISP/HSIOMIX power up and power down flow:
reg32_write(0x303A00EC,0x0000ffff);        //PGC_CPU_MAPPING
reg32_write(0x303A00F8,0x0000ffff);         //Power up all power domains
for(i=0;ilt;14;i++){                            //Power domain power down enable
    reg32setbit(0x303A0C00+i*0x40,0);
}
reg32_write(0x38330004, 0x7);                 //VPUMIX sft clock enable
reg32_write(0x32e28000,0x0000007f);            //release dispmix sft reset
reg32_write(0x32e28004,0x00001fff);            //dispmix sft clock enable
//GPU_2D power off
reg32clrbit(0x303A01FC,10);                      //power down request to ADB
while((reg32_read(0x303A01FC)) & (0x01lt;lt;28));    //wait ADB ack 0
reg32_write(CCM_CCGR(102),0x00);              //clock off
reg32setbit(0x303A0104,6);                       //PU_PGC_SW_PDN_REQ

//GPU_3D power off
reg32clrbit(0x303A01FC,9);                       //power down request to ADB
while((reg32_read(0x303A01FC)) & (0x01lt;lt;27));    //wait ADB ack 0
reg32_write(CCM_CCGR(79),0x00);               //clock off
reg32setbit(0x303A0104,9);                       //PU_PGC_SW_PDN_REQ

//GPU power off
reg32clrbit(0x303A01FC,11);                      //power down request to ADB
while((reg32_read(0x303A01FC)) & (0x01lt;lt;29));    //wait ADB ack 0
reg32_write(CCM_CCGR(87),0x00);               //clock off
reg32setbit(0x303A0104,7);                       //PU_PGC_SW_PDN_REQ

delay(50);

//GPU power on
reg32_write(CCM_CCGR(87),0x02);                //bus clock on
delay(20);
reg32_write(CCM_CCGR(87),0x00);                //bus clock off
reg32setbit(0x303A00F8,7);                       //PU_PGC_SW_PUP_REQ
delay(20);
reg32_write(CCM_CCGR(87),0x02);                //bus clock on
reg32setbit(0x303A01FC,11);                      //clear power down request to ADB
while(!((reg32_read(0x303A01FC)) & (0x01lt;lt;29))); //wait ADB ack 1

//GPU_2D power on
reg32_write(CCM_CCGR(102),0x02);               //2D clock on
reg32setbit(0x303A00F8,6);                       //PU_PGC_SW_PUP_REQ
reg32setbit(0x303A01FC,10);                      //clear power down request to ADB
while(!((reg32_read(0x303A01FC)) & (0x01lt;lt;28))); //wait ADB ack 1

//GPU_3D power on
//Power up flow:
reg32_write(CCM_CCGR(79),0x02);               //3D clock on
reg32setbit(0x303A00F8,9);                       //PU_PGC_SW_PUP_REQ
reg32setbit(0x303A01FC,9);                       //clear power down request to ADB
while(!((reg32_read(0x303A01FC)) & (0x01lt;lt;27)));   //wait ADB ack 1

//VPU_G1 power off
reg32_write(CCM_CCGR(86),0x00);               //clock off
reg32setbit(0x303A0104,11);                      //PU_PGC_SW_PDN_REQ

//VPU_G2 power off
reg32_write(CCM_CCGR(90),0x00);               //clock off
reg32setbit(0x303A0104,12);                      //PU_PGC_SW_PDN_REQ
```

```
//VPU_H1 power off
reg32_write(CCM_CCGR(89),0x00);                 //clock off
reg32setbit(0x303A0104,13);                       //PU_PGC_SW_PDN_REQ

//VPU power off
reg32clrbit(0x303A01FC,8);                        //power down request to ADB
while((reg32_read(0x303A01FC)) & (0x01lt;lt;26));     //wait ADB ack 0
reg32_write(CCM_CCGR(99),0x00);                 //clock off
reg32setbit(0x303A0104,8);                        //PU_PGC_SW_PDN_REQ

delay(50);

//VPU: power on
reg32_write(CCM_CCGR(99),0x02);                  //bus clock on
delay(20);
reg32_write(CCM_CCGR(99),0x00);                  //bus clock off
reg32setbit(0x303A00F8,8);                         //PU_PGC_SW_PUP_REQ
delay(20);
reg32_write(CCM_CCGR(99),0x02);                 //bus clock on
reg32_write(0x38330004, 0x7);                        //VPUMIX sft clock enable
reg32setbit(0x303A01FC,8);                        //clear power down request to ADB
while(!((reg32_read(0x303A01FC)) & (0x01lt;lt;26))); //wait ADB ack 1

//VPU_G1 power on
reg32_write(CCM_CCGR(86),0x02);                 //clock on
reg32setbit(0x303A00F8,11);                       //PU_PGC_SW_PUP_REQ

//VPU_G2 power on
reg32_write(CCM_CCGR(90),0x02);                 //clock on
reg32setbit(0x303A00F8,12);                       //PU_PGC_SW_PUP_REQ

//VPU_H1 power on
reg32_write(CCM_CCGR(89),0x02);                 //clock on
reg32setbit(0x303A00F8,13);                       //PU_PGC_SW_PUP_REQ

//PCIE power off
reg32_write(CCM_CCGR(37),0x00);                 //clock off
reg32setbit(0x303A0104,1);                        //PU_PGC_SW_PDN_REQ

//USB OTG1 power off
reg32setbit(0x303A0104,2);                        //PU_PGC_SW_PDN_REQ

//USB OTG2 power off
reg32setbit(0x303A0104,3);                        //PU_PGC_SW_PDN_REQ

//HSIO MIX low power
reg32clrbit(0x303A01FC,5);                        //power down request to ADB
while((reg32_read(0x303A01FC)) & (0x01lt;lt;23));     //wait ADB ack 0
reg32clrbit(0x303A01FC,6);                        //power down request to ADB
while((reg32_read(0x303A01FC)) & (0x01lt;lt;24));     //wait ADB ack 0
reg32_write(CCM_CCGR(77),0x00);                   //usb_bus clock off
reg32_write(CCM_CCGR(69),0x00);                   //hsio_bus clock off

delay(50);

//HSIO MIX run
reg32_write(CCM_CCGR(77),0x02);                   //usb_bus clock on
reg32_write(CCM_CCGR(69),0x02);                   //hsio_bus clock on
reg32setbit(0x303A01FC,5);                          //clear power down request to ADB
while(!((reg32_read(0x303A01FC)) & (0x01lt;lt;23))); //wait ADB ack 1
reg32setbit(0x303A01FC,6);                          //clear power down request to ADB
while(!((reg32_read(0x303A01FC)) & (0x01lt;lt;24)));     //wait ADB ack 1

//PCIE power on
reg32_write(CCM_CCGR(37),0x02);                 //clock on
reg32setbit(0x303A00F8,1);                        //PU_PGC_SW_PUP_REQ

//USB OTG1 power on
reg32setbit(0x303A00F8,2);                        //PU_PGC_SW_PUP_REQ
```

```
//USB OTG2 power on
reg32setbit(0x303A00F8,3);                          //PU_PGC_SW_PUP_REQ

//MIPI power off
reg32setbit(0x303A0104,0);                          //PU_PGC_SW_PDN_REQ

//DISP power off
reg32clrbit(0x303A01FC,7);                          //power down request to ADB
while((reg32_read(0x303A01FC)) & (0x01lt;lt;25));     //wait ADB ack 0
reg32_write(CCM_CCGR(93),0x00);                 //clock off
reg32setbit(0x303A0104,10);                         //PU_PGC_SW_PDN_REQ

delay(50);

//DISP power on
reg32_write(CCM_CCGR(93),0x02);                 //clock on
delay(20);
reg32_write(CCM_CCGR(93),0x00);                 //clock on
reg32setbit(0x303A00F8,10);                         //PU_PGC_SW_PUP_REQ
delay(20);
reg32_write(CCM_CCGR(93),0x02);                 //clock on
reg32_write(0x32e28000,0x0000007f);                  //release dispmix sft reset
reg32_write(0x32e28004,0x00001fff);                  //dispmix sft clock enable
reg32setbit(0x303A01FC,7);                          //clear power down request to ADB
while(!((reg32_read(0x303A01FC)) & (0x01lt;lt;25)));   //wait ADB ack 1

//MIPI power on
reg32setbit(0x303A00F8,0);                          //PU_PGC_SW_PUP_REQ

//DDR power off
reg32clrbit(0x303A01FC,2);                          //power down request to ADB

delay(50);

//DDR power on
reg32setbit(0x303A00F8,5);                           //PU_PGC_SW_PUP_REQ
```

5.2.9.6 Example Code 6

```
//ARM enters into ALL_OFF(STOP) mode and enable DSM ,NOC POWER DOWN:
//after "wfi", MIX/C0/C1/C2/C3 power down in SLOT0, SCU power down in SLOT1 when
//after "GPT1_INT" arrived, MIX power up in SLOT2, SCU power up in SLOT3, C0 power up in
SLOT4
//IMRx_CORE0_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x30, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x34, 0xFF7FFFFF); //[23] : GPT1 used as wakeup source
reg32_write(GPC_IPS_BASE_ADDR + 0x38, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x3C, 0xFFFFFFFF);
//IMRx_CORE1_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x40, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x44, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x48, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x4C, 0xFFFFFFFF);
//IMRx_CORE2_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x1C0, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1C4, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1C8, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1CC, 0xFFFFFFFF);
//IMRx_CORE3_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x1D0, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1D4, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1D8, 0xFFFFFFFF);
reg32_write(GPC_IPS_BASE_ADDR + 0x1DC, 0xFFFFFFFF);
```

```
//LPCR_A53_BSC
reg32_write(GPC_IPS_BASE_ADDR, 0x0000000A) ;
//[30],[23:22],[29:28] : A53_CO/A53_C1/A53_C2/A53_C3/LPM wakeup from external INT
//[14] : CLOCK OFF during STOP mode
//[3:0] : STOP mode
//LPCR_A53_BSC2
reg32_write(GPC_IPS_BASE_ADDR + 0x108, 0x0000000A) ;
//[3:0] : STOP mode
//LPCR_A53_AD
reg32_write(GPC_IPS_BASE_ADDR + 0x4, 0x0A0A0A1A) ;
//[16] : 1(ALL_OFF mode); 0(L2 retention mode)
//[27]/[25]/[11]/[9]: A53_C0/A53_C1/A53_C2/A53_C3 power up with A53 LPM PUP REQ
//[4] : A53_SCU power down with A53 LPM PDN REQ
//[19]/[17]/[3]/[1] : A53_C0/A53_C1/A53_C2/A53_C3 powr down with A53 LPM PDN REQ
//LPCR_M4
reg32_write(GPC_IPS_BASE_ADDR + 0x8, 0x80000000) ;
//[31] : DSM ingore to check M4 low power state
//[1:0] : M4 LPM run mode
//SLPCR
reg32_write(GPC_IPS_BASE_ADDR + 0x14,0xe000ffA7) ;
//[31] : enable DSM
//[30] : eneable regulator bypass
//[5:3] : wait 64 ckil clock cycles
//[2] : enable PMIC standby
//[1] : enable OSC power down
//[0] : bypass PMIC ready handshake
//PGC_ACK_SEL_A53
reg32_write(GPC_IPS_BASE_ADDR + 0x24,0x00010008) ;
//[3] : NOC PGC as LPM power down ack
//[16] : A53_C0 PGC as LPM power up ack
//SLT_CFG0
reg32_write(GPC_IPS_BASE_ADDR + 0xB0,0x00000055) ;
//[6]/[4]/[2]/[0] : A53_C0/A53_C1/ A53_C2/A53_C3 power down in SLOT0
//SLT_CFG1
reg32_write(GPC_IPS_BASE_ADDR + 0xB4,0x00000100) ;
//[8] : A53_SCU power down in SLOT1
//SLT_CFG2
reg32_write(GPC_IPS_BASE_ADDR + 0xB8,0x00000400) ;
//[10] :NOC power down in SLOT2
//SLT_CFG3
reg32_write(GPC_IPS_BASE_ADDR + 0xBC,0x00000800) ;
//[11] :NOC power up in SLOT3
//SLT_CFG4
reg32_write(GPC_IPS_BASE_ADDR + 0xC0,0x00000200) ;
//[9] : A53_SCU power up in SLOT4
//SLT_CFG5
reg32_write(GPC_IPS_BASE_ADDR + 0xC4,0x00000002) ;
//[1] : A53_C0 power up in SLOT4, A53_C1/ A53_C2/A53_C3 not power up
//PGC_CPU_MAPPING
reg32_write(GPC_IPS_BASE_ADDR + 0xEC,0x00000002) ;
//[1] : NOC PGC mapping to A53 LPM
//A53 PGC
reg32_write(GPC_IPS_BASE_ADDR +0x800, reg32_read(GPC_IPS_BASE_ADDR +0x800) | 0x00000001) ;
// enable A53_C0 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x840, reg32_read(GPC_IPS_BASE_ADDR +0x840) | 0x00000001) ;
// enable A53_C1 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x880, reg32_read(GPC_IPS_BASE_ADDR +0x880) | 0x00000001) ;
// enable A53_C2 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x8C0, reg32_read(GPC_IPS_BASE_ADDR +0x8C0) | 0x00000001) ;
// enable A53_C3 PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x900, reg32_read(GPC_IPS_BASE_ADDR +0x900) | 0x00000001) ;
// enable A53_SCU PGC power down
reg32_write(GPC_IPS_BASE_ADDR +0x910, (0x59 lt;lt; 10) | 0x5B | (0x51 lt;lt; 20) ) ;
// change nL2retn/mempwr/dftram to meet SCU power up timing
//NOC PGC
reg32_write(GPC_IPS_BASE_ADDR +0xA40, reg32_read(GPC_IPS_BASE_ADDR +0xA40) | 0x00000001) ;
// enable NOC PGC power down
```

5.2.10 GPC Memory Map/Register Definition

Detailed descriptions of each register can be found below.

The total GPC memory map is 4KB

Table 5-10. Memory Regions

Address Range(offset)	Region
0x000 - 0x3FF	GPC configuration register
0x400 - 0x7FF	Reserved
0x800 - 0x9FF	CPU and SCU type PGC register base address
0xA00 - 0xBFF	MIX type PGC register base address
0xC00 - 0xFFF	PU type PGC register base address

Each PGC (CPU type, MIX type, PU type) will occupy 64 Bytes address space, the specific base address of each PGC are listed as below.

- 0x800 ~ 0x83F : PGC for A53 core0
- 0x840 ~ 0x87F: PGC for A53 core1
- 0x880 ~ 0x8BF: PGC for A53 core2
- 0x8C0 ~ 0x8FF: PGC for A53 core3
- 0x900 ~ 0x93F: PGC for A53 SCU
- 0xA00 ~ 0xA3F: Reserved
- 0xA40 ~ 0xA7F: PGC for NOC mix
- 0xC00 ~ 0xC3F: PGC for MIPI DSI PHY
- 0xC40 ~ 0xC7F: PGC for PCIE1 PHY
- 0xC80 ~ 0xCBF: USB_OTG1
- 0xCC0 ~ 0xCFF: USB_OTG2
- 0xD00 ~ 0xD3F: Reserved
- 0xD40 ~ 0xD7F: DDR1
- 0xD80 ~ 0xDBF: GPU_2D
- 0xDC0 ~ 0xDFF: GPUMIX
- 0xE00 ~ 0xE3F: VPUMIX
- 0xE40 ~ 0xE7F: GPU_3D
- 0xE80 ~ 0xEBF: DISPMIX
- 0xEC0 ~ 0xEFF: VPU_G1
- 0xF00 ~ 0xF3F: VPU_G2
- 0xF40 ~ 0xF7F: VPU_H1

For more specific information about PGC register definition, please see the register definition for each PGC.

GPC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303A_0000	Basic Low power control register of A53 platform (GPC_LPCR_A53_BSC)	32	R/W	0000_3FF0h	5.2.10.1/ 609
303A_0004	Advanced Low power control register of A53 platform (GPC_LPCR_A53_AD)	32	R/W	0000_0020h	5.2.10.2/ 612
303A_0008	Low power control register of CPU1 (GPC_LPCR_M4)	32	R/W	0000_3FF0h	5.2.10.3/ 615
303A_0014	System low power control register (GPC_SLPCR)	32	R/W	E000_FF82h	5.2.10.4/ 617
303A_0018	MASTER LPM Handshake (GPC_MST_CPU_MAPPING)	32	R/W	0000_00FFh	5.2.10.5/ 620
303A_0020	Memory low power control register (GPC_MLPCR)	32	R/W	0101_0100h	5.2.10.6/ 621
303A_0024	PGC acknowledge signal selection of A53 platform (GPC_PGC_ACK_SEL_A53)	32	R/W	8000_8000h	5.2.10.7/ 622
303A_0028	PGC acknowledge signal selection of M4 platform (GPC_PGC_ACK_SEL_M4)	32	R/W	8000_8000h	5.2.10.8/ 624
303A_002C	GPC Miscellaneous register (GPC_MISC)	32	R/W	0000_0021h	5.2.10.9/ 625
303A_0030	IRQ masking register 1 of A53 core0 (GPC_IMR1_CORE0_A53)	32	R/W	0000_0000h	5.2.10.10/ 626
303A_0034	IRQ masking register 2 of A53 core0 (GPC_IMR2_CORE0_A53)	32	R/W	0000_0000h	5.2.10.11/ 627
303A_0038	IRQ masking register 3 of A53 core0 (GPC_IMR3_CORE0_A53)	32	R/W	0000_0000h	5.2.10.12/ 627
303A_003C	IRQ masking register 4 of A53 core0 (GPC_IMR4_CORE0_A53)	32	R/W	0000_0000h	5.2.10.13/ 628
303A_0040	IRQ masking register 1 of A53 core1 (GPC_IMR1_CORE1_A53)	32	R/W	0000_0000h	5.2.10.14/ 628
303A_0044	IRQ masking register 2 of A53 core1 (GPC_IMR2_CORE1_A53)	32	R/W	0000_0000h	5.2.10.15/ 628
303A_0048	IRQ masking register 3 of A53 core1 (GPC_IMR3_CORE1_A53)	32	R/W	0000_0000h	5.2.10.16/ 629
303A_004C	IRQ masking register 4 of A53 core1 (GPC_IMR4_CORE1_A53)	32	R/W	0000_0000h	5.2.10.17/ 629
303A_0050	IRQ masking register 1 of M4 (GPC_IMR1_M4)	32	R/W	0000_0000h	5.2.10.18/ 630
303A_0054	IRQ masking register 2 of M4 (GPC_IMR2_M4)	32	R/W	0000_0000h	5.2.10.19/ 630
303A_0058	IRQ masking register 3 of M4 (GPC_IMR3_M4)	32	R/W	0000_0000h	5.2.10.20/ 630

Table continues on the next page...

GPC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303A_005C	IRQ masking register 4 of M4 (GPC_IMR4_M4)	32	R/W	0000_0000h	5.2.10.21/ 631
303A_0070	IRQ status register 1 of A53 (GPC_ISR1_A53)	32	R	0000_0000h	5.2.10.22/ 631
303A_0074	IRQ status register 2 of A53 (GPC_ISR2_A53)	32	R	0000_0000h	5.2.10.23/ 632
303A_0078	IRQ status register 3 of A53 (GPC_ISR3_A53)	32	R	0000_0000h	5.2.10.24/ 632
303A_007C	IRQ status register 4 of A53 (GPC_ISR4_A53)	32	R	0000_0000h	5.2.10.25/ 632
303A_0080	IRQ status register 1 of M4 (GPC_ISR1_M4)	32	R	0000_0000h	5.2.10.26/ 633
303A_0084	IRQ status register 2 of M4 (GPC_ISR2_M4)	32	R	0000_0000h	5.2.10.27/ 633
303A_0088	IRQ status register 3 of M4 (GPC_ISR3_M4)	32	R	0000_0000h	5.2.10.28/ 634
303A_008C	IRQ status register 4 of M4 (GPC_ISR4_M4)	32	R	0000_0000h	5.2.10.29/ 634
303A_00B0	Slot configure register for A53 core (GPC_SLT0_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00B4	Slot configure register for A53 core (GPC_SLT1_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00B8	Slot configure register for A53 core (GPC_SLT2_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00BC	Slot configure register for A53 core (GPC_SLT3_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00C0	Slot configure register for A53 core (GPC_SLT4_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00C4	Slot configure register for A53 core (GPC_SLT5_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00C8	Slot configure register for A53 core (GPC_SLT6_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00CC	Slot configure register for A53 core (GPC_SLT7_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00D0	Slot configure register for A53 core (GPC_SLT8_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00D4	Slot configure register for A53 core (GPC_SLT9_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00D8	Slot configure register for A53 core (GPC_SLT10_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00DC	Slot configure register for A53 core (GPC_SLT11_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00E0	Slot configure register for A53 core (GPC_SLT12_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634

Table continues on the next page...

GPC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303A_00E4	Slot configure register for A53 core (GPC_SLT13_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00E8	Slot configure register for A53 core (GPC_SLT14_CFG)	32	R/W	0000_0000h	5.2.10.30/ 634
303A_00EC	PGC CPU mapping (GPC_PGC_CPU_0_1_MAPPING)	32	R/W	0000_0000h	5.2.10.31/ 637
303A_00F0	CPU PGC software power up trigger (GPC_CPU_PGC_SW_PUP_REQ)	32	R/W	0000_0000h	5.2.10.32/ 639
303A_00F4	MIX PGC software power up trigger (GPC_MIX_PGC_SW_PUP_REQ)	32	R/W	0000_0000h	5.2.10.33/ 640
303A_00F8	PU PGC software up trigger (GPC_PU_PGC_SW_PUP_REQ)	32	R/W	0000_0000h	5.2.10.34/ 641
303A_00FC	CPU PGC software down trigger (GPC_CPU_PGC_SW_PDN_REQ)	32	R/W	0000_0000h	5.2.10.35/ 642
303A_0100	MIX PGC software power down trigger (GPC_MIX_PGC_SW_PDN_REQ)	32	R/W	0000_0000h	5.2.10.36/ 643
303A_0104	PU PGC software down trigger (GPC_PU_PGC_SW_PDN_REQ)	32	R/W	0000_0000h	5.2.10.37/ 644
303A_0108	Basic Low power control register of A53 platform (GPC_LPCR_A53_BSC2)	32	R/W	0000_0000h	5.2.10.38/ 646
303A_0130	CPU PGC software up trigger status1 (GPC_CPU_PGC_PUP_STATUS1)	32	R	0000_0000h	5.2.10.39/ 647
303A_0134	A53 MIX software up trigger status register (GPC_A53_MIX_PGC_PUP_STATUS0)	32	R	0000_0000h	5.2.10.40/ 649
303A_0138	A53 MIX software up trigger status register (GPC_A53_MIX_PGC_PUP_STATUS1)	32	R	0000_0000h	5.2.10.40/ 649
303A_013C	A53 MIX software up trigger status register (GPC_A53_MIX_PGC_PUP_STATUS2)	32	R	0000_0000h	5.2.10.40/ 649
303A_0140	M4 MIX PGC software up trigger status register (GPC_M4_MIX_PGC_PUP_STATUS0)	32	R	0000_0000h	5.2.10.41/ 650
303A_0144	M4 MIX PGC software up trigger status register (GPC_M4_MIX_PGC_PUP_STATUS1)	32	R	0000_0000h	5.2.10.41/ 650
303A_0148	M4 MIX PGC software up trigger status register (GPC_M4_MIX_PGC_PUP_STATUS2)	32	R	0000_0000h	5.2.10.41/ 650
303A_014C	A53 PU software up trigger status register (GPC_A53_PU_PGC_PUP_STATUS0)	32	R	0000_0000h	5.2.10.42/ 652
303A_0150	A53 PU software up trigger status register (GPC_A53_PU_PGC_PUP_STATUS1)	32	R	0000_0000h	5.2.10.42/ 652
303A_0154	A53 PU software up trigger status register (GPC_A53_PU_PGC_PUP_STATUS2)	32	R	0000_0000h	5.2.10.42/ 652
303A_0158	M4 PU PGC software up trigger status register (GPC_M4_PU_PGC_PUP_STATUS0)	32	R	0000_0000h	5.2.10.43/ 655
303A_015C	M4 PU PGC software up trigger status register (GPC_M4_PU_PGC_PUP_STATUS1)	32	R	0000_0000h	5.2.10.43/ 655

Table continues on the next page...

GPC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303A_0160	M4 PU PGC software up trigger status register (GPC_M4_PU_PGC_PUP_STATUS2)	32	R	0000_0000h	5.2.10.43/ 655
303A_0170	CPU PGC software dn trigger status1 (GPC_CPU_PGC_PDN_STATUS1)	32	R	0000_0000h	5.2.10.44/ 657
303A_0174	A53 MIX software down trigger status register (GPC_A53_MIX_PGC_PDN_STATUS0)	32	R	0000_0000h	5.2.10.45/ 659
303A_0178	A53 MIX software down trigger status register (GPC_A53_MIX_PGC_PDN_STATUS1)	32	R	0000_0000h	5.2.10.45/ 659
303A_017C	A53 MIX software down trigger status register (GPC_A53_MIX_PGC_PDN_STATUS2)	32	R	0000_0000h	5.2.10.45/ 659
303A_0180	M4 MIX PGC software power down trigger status register (GPC_M4_MIX_PGC_PDN_STATUS0)	32	R	0000_0000h	5.2.10.46/ 661
303A_0184	M4 MIX PGC software power down trigger status register (GPC_M4_MIX_PGC_PDN_STATUS1)	32	R	0000_0000h	5.2.10.46/ 661
303A_0188	M4 MIX PGC software power down trigger status register (GPC_M4_MIX_PGC_PDN_STATUS2)	32	R	0000_0000h	5.2.10.46/ 661
303A_018C	A53 PU PGC software down trigger status (GPC_A53_PU_PGC_PDN_STATUS0)	32	R	0000_0000h	5.2.10.47/ 662
303A_0190	A53 PU PGC software down trigger status (GPC_A53_PU_PGC_PDN_STATUS1)	32	R	0000_0000h	5.2.10.47/ 662
303A_0194	A53 PU PGC software down trigger status (GPC_A53_PU_PGC_PDN_STATUS2)	32	R	0000_0000h	5.2.10.47/ 662
303A_0198	M4 PU PGC software down trigger status (GPC_M4_PU_PGC_PDN_STATUS0)	32	R	0000_0000h	5.2.10.48/ 665
303A_019C	M4 PU PGC software down trigger status (GPC_M4_PU_PGC_PDN_STATUS1)	32	R	0000_0000h	5.2.10.48/ 665
303A_01A0	M4 PU PGC software down trigger status (GPC_M4_PU_PGC_PDN_STATUS2)	32	R	0000_0000h	5.2.10.48/ 665
303A_01B0	A53 MIX PDN FLG (GPC_A53_MIX_PDN_FLG)	32	R/W	0000_0000h	5.2.10.49/ 668
303A_01B4	A53 PU PDN FLG (GPC_A53_PU_PDN_FLG)	32	R/W	0000_0000h	5.2.10.50/ 669
303A_01B8	M4 MIX PDN FLG (GPC_M4_MIX_PDN_FLG)	32	R/W	0000_0000h	5.2.10.51/ 670
303A_01BC	M4 PU PDN FLG (GPC_M4_PU_PDN_FLG)	32	R/W	0000_0000h	5.2.10.52/ 671
303A_01C0	IRQ masking register 1 of A53 core2 (GPC_IMR1_CORE2_A53)	32	R/W	0000_0000h	5.2.10.53/ 671
303A_01C4	IRQ masking register 2 of A53 core2 (GPC_IMR2_CORE2_A53)	32	R/W	0000_0000h	5.2.10.54/ 672
303A_01C8	IRQ masking register 3 of A53 core2 (GPC_IMR3_CORE2_A53)	32	R/W	0000_0000h	5.2.10.55/ 672
303A_01CC	IRQ masking register 4 of A53 core2 (GPC_IMR4_CORE2_A53)	32	R/W	0000_0000h	5.2.10.56/ 673

Table continues on the next page...

GPC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303A_01D0	IRQ masking register 1 of A53 core3 (GPC_IMR1_CORE3_A53)	32	R/W	0000_0000h	5.2.10.57/ 673
303A_01D4	IRQ masking register 2 of A53 core3 (GPC_IMR2_CORE3_A53)	32	R/W	0000_0000h	5.2.10.58/ 674
303A_01D8	IRQ masking register 3 of A53 core3 (GPC_IMR3_CORE3_A53)	32	R/W	0000_0000h	5.2.10.59/ 674
303A_01DC	IRQ masking register 4 of A53 core3 (GPC_IMR4_CORE3_A53)	32	R/W	0000_0000h	5.2.10.60/ 675
303A_01E0	PGC acknowledge signal selection of A53 platform for PUs (GPC_ACK_SEL_A53_PU)	32	R/W	0000_0000h	5.2.10.61/ 675
303A_01E4	PGC acknowledge signal selection of M4 platform for PUs (GPC_ACK_SEL_M4_PU)	32	R/W	0000_0000h	5.2.10.62/ 678
303A_01E8	Slot configure register for A53 core (GPC_SLT15_CFG)	32	R/W	0000_0000h	5.2.10.63/ 680
303A_01EC	Slot configure register for A53 core (GPC_SLT16_CFG)	32	R/W	0000_0000h	5.2.10.63/ 680
303A_01F0	Slot configure register for A53 core (GPC_SLT17_CFG)	32	R/W	0000_0000h	5.2.10.63/ 680
303A_01F4	Slot configure register for A53 core (GPC_SLT18_CFG)	32	R/W	0000_0000h	5.2.10.63/ 680
303A_01F8	Slot configure register for A53 core (GPC_SLT19_CFG)	32	R/W	0000_0000h	5.2.10.63/ 680
303A_01FC	Power handshake register (GPC_PU_PWRHSK)	32	R/W	0000_FFFFh	5.2.10.64/ 683
303A_0200	Slot configure register for PUs (GPC_SLT0_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0204	Slot configure register for PUs (GPC_SLT1_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0208	Slot configure register for PUs (GPC_SLT2_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_020C	Slot configure register for PUs (GPC_SLT3_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0210	Slot configure register for PUs (GPC_SLT4_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0214	Slot configure register for PUs (GPC_SLT5_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0218	Slot configure register for PUs (GPC_SLT6_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_021C	Slot configure register for PUs (GPC_SLT7_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0220	Slot configure register for PUs (GPC_SLT8_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0224	Slot configure register for PUs (GPC_SLT9_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687

Table continues on the next page...

GPC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303A_0228	Slot configure register for PUs (GPC_SLT10_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_022C	Slot configure register for PUs (GPC_SLT11_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0230	Slot configure register for PUs (GPC_SLT12_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0234	Slot configure register for PUs (GPC_SLT13_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0238	Slot configure register for PUs (GPC_SLT14_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_023C	Slot configure register for PUs (GPC_SLT15_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0240	Slot configure register for PUs (GPC_SLT16_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0244	Slot configure register for PUs (GPC_SLT17_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_0248	Slot configure register for PUs (GPC_SLT18_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687
303A_024C	Slot configure register for PUs (GPC_SLT19_CFG_PU)	32	R/W	0000_0000h	5.2.10.65/ 687

5.2.10.1 Basic Low power control register of A53 platform (GPC_LPCR_A53_BSC)

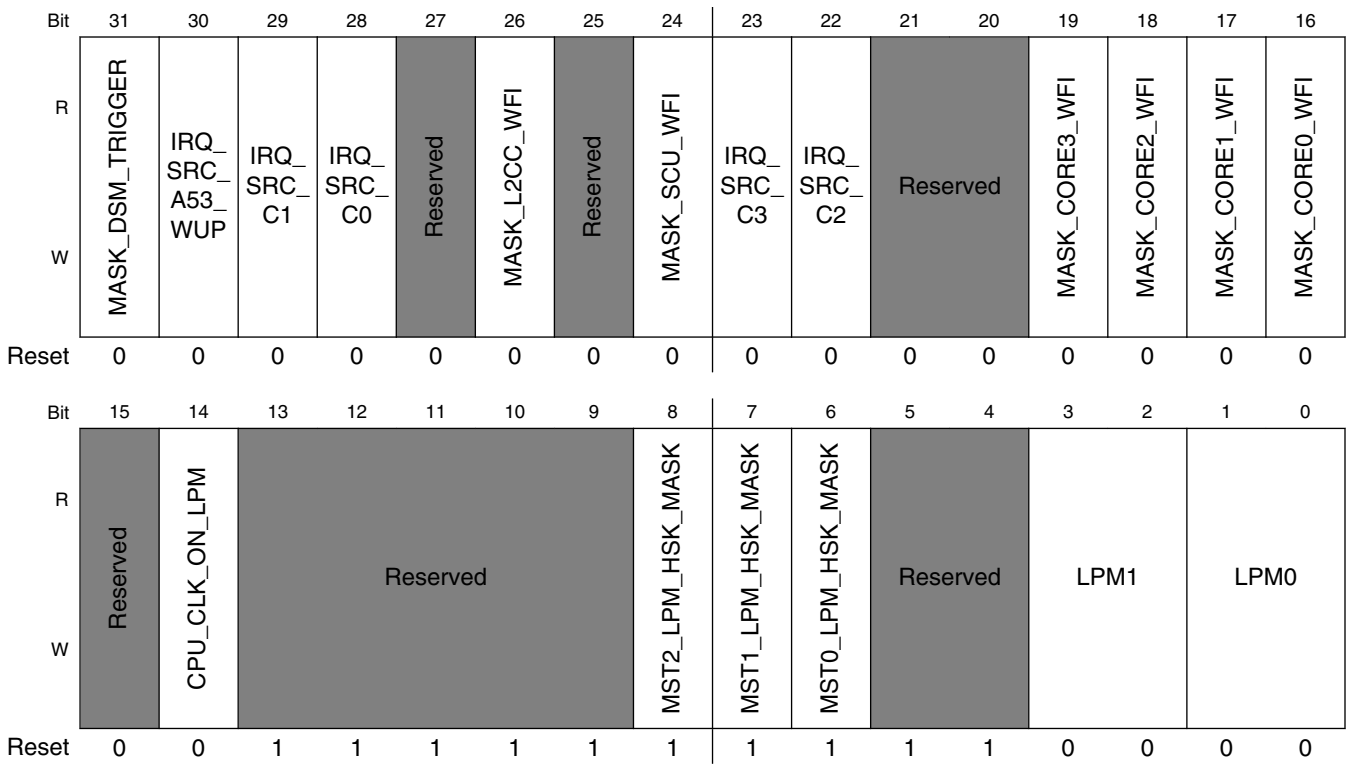
NOTE

LPCR_A53_BSC[CPU_CLK_ON_LPM] should be set 1'b1 when using A53 low power debug feature

NOTE

Always set LPM1/LPM0 with same value

Address: 303A_0000h base + 0h offset = 303A_0000h



GPC_LPCR_A53_BSC field descriptions

Field	Description
31 MASK_DSM_TRIGGER	DSM Trigger Mask 0 DSM trigger of A53 platform will not be masked 1 DSM trigger of A53 platform will be masked
30 IRQ_SRC_A53_WUP	LPCR_A53_BSC[IRQ_SRC_C0], LPCR_A53_BSC[IRQ_SRC_C1], LPCR_A53_BSC[IRQ_SRC_C2], LPCR_A53_BSC[IRQ_SRC_C3], and LPCR_A53_BSC[IRQ_SRC_A53_WUP] work together to decide the wake up source for A53 LPM and core0/core1/core2/core3 power. See "Power up process for A53 platform" for more specific information.

Table continues on the next page...

GPC_LPCR_A53_BSC field descriptions (continued)

Field	Description
	0 LPM wakeup source be “OR” result of LPCR_A53_BSC[IRQ_SRC_C0]/ LPCR_A53_BSC[IRQ_SRC_C1]/LPCR_A53_BSC[IRQ_SRC_C2]/LPCR_A53_BSC[IRQ_SRC_C3] setting 1 LPM wakeup source from external INT[127:0], masked by IMR0
29 IRQ_SRC_C1	LPCR_A53_BSC[IRQ_SRC_C0], LPCR_A53_BSC[IRQ_SRC_C1], LPCR_A53_BSC[IRQ_SRC_C2], LPCR_A53_BSC[IRQ_SRC_C3], and LPCR_A53_BSC[IRQ_SRC_A53_WUP] work together to decide the wake up source for A53 LPM and core0/core1/core2/core3 power. See “Power up process for A53 platform” for more specific information. 0 core1 wakeup source from external INT[127:0], masked by IMR1 refer to “Power up process for A53 platform” for more specific information 1 core1 wakeup source from GIC(nFIQ[1]/nIRQ[1]), SCU should not be power down during low power mode when this bit is set to 1’b1
28 IRQ_SRC_C0	LPCR_A53_BSC[IRQ_SRC_C0], LPCR_A53_BSC[IRQ_SRC_C1], LPCR_A53_BSC[IRQ_SRC_C2], LPCR_A53_BSC[IRQ_SRC_C3], and LPCR_A53_BSC[IRQ_SRC_A53_WUP] work together to decide the wake up source for A53 LPM and core0/core1/core2/core3 power. See “Power up process for A53 platform” for more specific information. 0 core0 wakeup source from external INT[127:0], masked by IMR0 refer to “Power up process for A53 platform” for more specific information 1 core0 wakeup source from GIC(nFIQ[0]/nIRQ[0]), SCU should not be power down during low power mode when this bit is set to 1’b1
27 -	This field is reserved. Reserved
26 MASK_L2CC_ WFI	L2 cache controller Wait For Interrupt Mask Register 0 WFI for L2 cache controller is not masked 1 WFI for L2 cache controller is masked
25 -	This field is reserved. Reserved
24 MASK_SCU_WFI	SCU Wait For Interrupt Mask Register 0 WFI for SCU is not masked 1 WFI for SCU is masked
23 IRQ_SRC_C3	LPCR_A53_BSC[IRQ_SRC_C0], LPCR_A53_BSC[IRQ_SRC_C1], LPCR_A53_BSC[IRQ_SRC_C2], LPCR_A53_BSC[IRQ_SRC_C3], and LPCR_A53_BSC[IRQ_SRC_A53_WUP] work together to decide the wake up source for A53 LPM and core0/core1/core2/core3 power. 0 core3 wakeup source from external INT[127:0], masked by IMR1. See Power Up Process for A53 Platform for more specific information. 1 core3 wakeup source from external GIC(nFIQ[1]/nIRQ[1]), SCU should not be powered down during low power mode when this bit is set to 1'b1.
22 IRQ_SRC_C2	LPCR_A53_BSC[IRQ_SRC_C0], LPCR_A53_BSC[IRQ_SRC_C1], LPCR_A53_BSC[IRQ_SRC_C2], LPCR_A53_BSC[IRQ_SRC_C3], and LPCR_A53_BSC[IRQ_SRC_A53_WUP] work together to decide the wake up source for A53 LPM and core0/core1/core2/core3 power. 0 core2 wakeup source from external INT[127:0], masked by IMR1. See Power Up Process for A53 Platform for more specific information. 1 core2 wakeup source from external GIC(nFIQ[1]/nIRQ[1]), SCU should not be powered down during low power mode when this bit is set to 1'b1.

Table continues on the next page...

GPC_LPCR_A53_BSC field descriptions (continued)

Field	Description
21–20 -	This field is reserved.
19 MASK_CORE3_WFI	CORE3 Wait For Interrupt Mask 0 WFI for CORE3 is not masked 1 WFI for CORE3 is masked
18 MASK_CORE2_WFI	CORE2 Wait For Interrupt Mask 0 WFI for CORE2 is not masked 1 WFI for CORE2 is masked
17 MASK_CORE1_WFI	CORE1 Wait For Interrupt Mask 0 WFI for CORE1 is not masked 1 WFI for CORE1 is masked
16 MASK_CORE0_WFI	CORE0 Wait For Interrupt Mask 0 WFI for CORE0 is not masked 1 WFI for CORE0 is masked
15 -	This field is reserved. Reserved
14 CPU_CLK_ON_LPM	Define if A53 clocks will be disabled on wait/stop mode. 0 A53 clock disabled on wait/stop mode 1 A53 clock enabled on wait/stop mode
13–9 -	This field is reserved. Reserved
8 MST2_LPM_HSK_MASK	MASTER2 LPM handshake mask MASTER2(supermix2noc ADB) will handshake with GPC in LPM, follow this when you want this master power off. This bit should use together with MST_CPU_MAPPING[2] If you want power of supermix2noc ADB, use this setting: LPCR_A53_BSC[8]=0; MST_CPU_MAPPING[2]=1 Otherwise use: LPCR_A53_BSC[8]=1; MST_CPU_MAPPING[2]=0 0 enable MASTER2 LPM handshake, wait ACK from MASTER2 1 disable MASTER2 LPM handshake, mask ACK from MASTER2
7 MST1_LPM_HSK_MASK	MASTER1 LPM handshake mask MASTER1(supermix2noc ADB) will handshake with GPC in LPM, follow this when you want this master power off. This bit should use together with MST_CPU_MAPPING[1] If you want power of supermix2noc ADB, use this setting: LPCR_A53_BSC[7]=0; MST_CPU_MAPPING[1]=1 Otherwise use: LPCR_A53_BSC[7]=1; MST_CPU_MAPPING[1]=0 0 enable MASTER1 LPM handshake, wait ACK from MASTER1 1 disable MASTER1 LPM handshake, mask ACK from MASTER1
6 MST0_LPM_HSK_MASK	MASTER0 LPM handshake mask

Table continues on the next page...

GPC_LPCR_A53_BSC field descriptions (continued)

Field	Description
	MASTER0(SCU) will handshake with GPC in LPM, follow this when you want this master power off. This bit should be used together with MST_CPU_MAPPING[0]. If you want power of SCU, use this setting: LPCR_A53_BSC[6]=0; MST_CPU_MAPPING[0]=1 Otherwise use: LPCR_A53_BSC[6]=1; MST_CPU_MAPPING[0]=0 0 enable MASTER0 LPM handshake, wait ACK from MASTER0 1 disable MASTER0 LPM handshake, mask ACK from MASTER0
5–4 -	This field is reserved. Reserved
3–2 LPM1	CORE1 Setting the low power mode that system will enter on next assertion of dsm_request signal. 00 Remain in RUN mode 01 Transfer to WAIT mode 10 Transfer to STOP mode 11 Reserved
LPM0	CORE0 Setting the low power mode that system will enter on next assertion of dsm_request signal. 00 Remain in RUN mode 01 Transfer to WAIT mode 10 Transfer to STOP mode 11 Reserved

5.2.10.2 Advanced Low power control register of A53 platform (GPC_LPCR_A53_AD)

Address: 303A_0000h base + 4h offset = 303A_0004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L2PGE	Reserved			EN_C3_PUP	EN_C3_IRQ_PUP	EN_C2_PUP	EN_C2_IRQ_PUP	EN_C3_WFI_PDN_DIS	EN_C2_WFI_PDN_DIS	EN_C1_WFI_PDN_DIS	EN_C0_WFI_PDN_DIS	EN_C3_PDN	EN_C3_WFI_PDN	EN_C2_PDN	EN_C2_WFI_PDN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				EN_C1_PUP	EN_C1_IRQ_PUP	EN_C0_PUP	EN_C0_IRQ_PUP	Reserved		EN_L2_WFI_PDN	EN_PLAT_PDN	EN_C1_PDN	EN_C1_WFI_PDN	EN_C0_PDN	EN_C0_WFI_PDN
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

GPC_LPCR_A53_AD field descriptions

Field	Description
31 L2PGE	0 L2 cache RAM will power down with SCU power domain in A53 platform (used for ALL_OFF mode) 1 L2 cache RAM will not power down with SCU power domain in A53 platform (used for ALL_OFF mode)
30–28 -	This field is reserved. Reserved
27 EN_C3_PUP	0 CORE3 will power up with lower power mode request 1 CORE3 will not power up with low power mode request (only used wake up from CPU_OFF)
26 EN_C3_IRQ_PUP	0 CORE3 will power up with IRQ request 1 CORE3 will not power up with IRQ request
25 EN_C2_PUP	0 CORE2 will power up with lower power mode request 1 CORE2 will not power up with low power mode request (only used wake up from CPU_OFF)
24 EN_C2_IRQ_PUP	0 CORE2 will power up with IRQ request 1 CORE2 will not power up with IRQ request
23 EN_C3_WFI_PDN_DIS	0 Disable WFI power down core3 1 Enable WFI power down core3
22 EN_C2_WFI_PDN_DIS	0 Disable WIFI power down core2 1 Enable WIFI power down core2
21 EN_C1_WFI_PDN_DIS	0 Disable WIFI power down core1 1 Enable WIFI power down core1
20 EN_C0_WFI_PDN_DIS	0 Disable WIFI power down core0 1 Enable WIFI power down core0
19 EN_C3_PDN	0 CORE3 will not be power down with low power mode request 1 CORE3 will be power down with low power mode request
18 EN_C3_WFI_PDN	0 CORE3 will not be power down with WFI request 1 CORE3 will be power down with WFI request
17 EN_C2_PDN	0 CORE2 will not be power down with low power mode request 1 CORE2 will be power down with low power mode request
16 EN_C2_WFI_PDN	0 CORE2 will not be power down with WFI request 1 CORE2 will be power down with WFI request
15–12 -	This field is reserved. Reserved
11 EN_C1_PUP	0 CORE1 will not power up with low power mode request (only used wake up from CPU01_OFF mode) 1 CORE1 will power up with low power mode request
10 EN_C1_IRQ_PUP	0 CORE1 will power up with IRQ request 1 CORE1 will not power up with IRQ request
9 EN_C0_PUP	(only used wake up from CPU01_OFF mode)

Table continues on the next page...

GPC_LPCR_A53_AD field descriptions (continued)

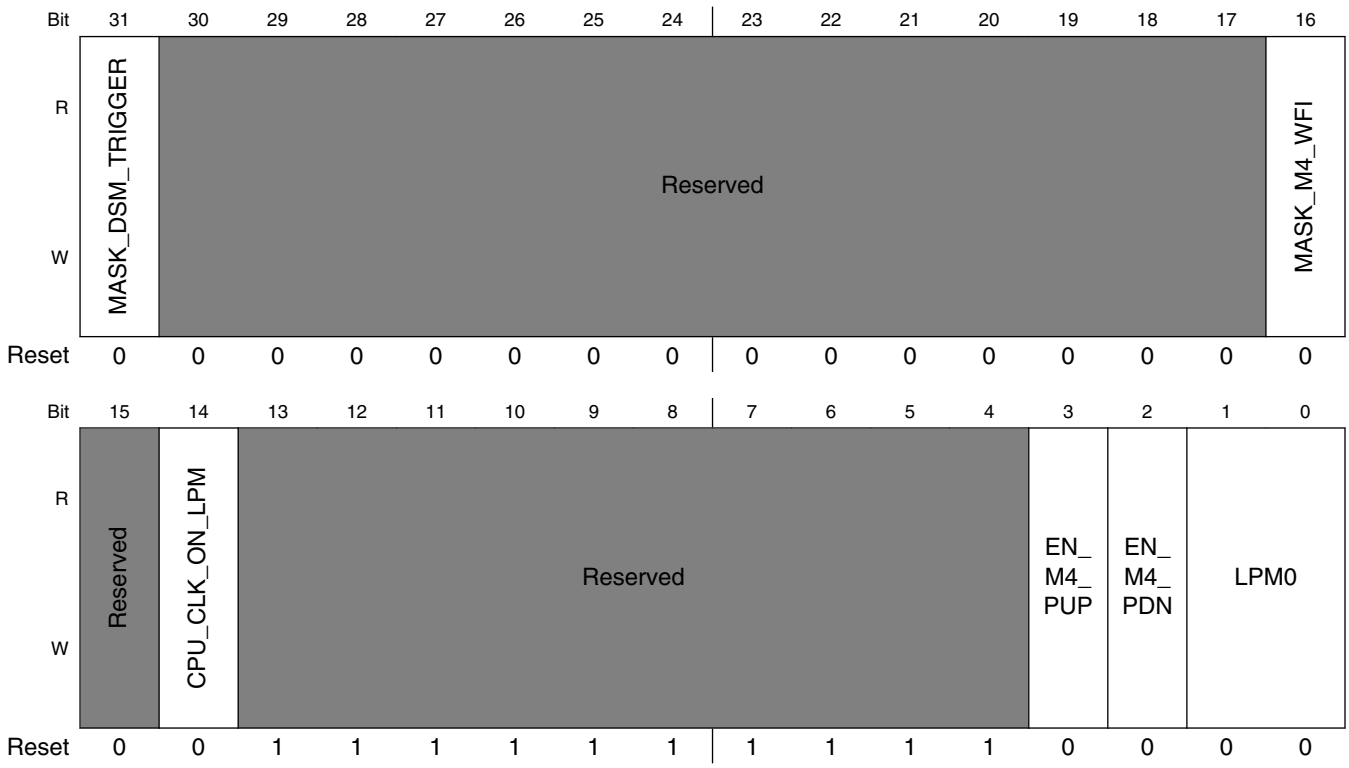
Field	Description
	0 CORE0 will power up with low power mode request 1 CORE0 will not power up with low power mode request
8 EN_C0_IRQ_PUP	0 CORE0 will power up with IRQ request 1 CORE0 will not power up with IRQ request
7–6 -	This field is reserved. Reserved
5 EN_L2_WFI_PDN	**NOTE:** Before reset, L2 WFI is 1 and make GPC generate an error DSM request. This bit is used to mask the L2 WFI before reset. After reset, L2 WFI change to 0, and functions are OK, SW must clear this bit at the beginning of code. 0 SCU and L2 will not be power down with WFI request 1 SCU and L2 will be power down with WFI request (default)
4 EN_PLAT_PDN	0 SCU and L2 cache RAM will not be power down with low power mode request 1 SCU and L2 cache RAM will be power down with low power mode request
3 EN_C1_PDN	0 CORE1 will not be power down with low power mode request 1 CORE1 will be power down with low power mode request
2 EN_C1_WFI_PDN	0 CORE1 will not be power down with WFI request 1 CORE1 will be power down with WFI request
1 EN_C0_PDN	0 CORE0 will not be power down with low power mode request 1 CORE0 will be power down with low power mode request
0 EN_C0_WFI_PDN	0 CORE0 will not be power down with WFI request 1 CORE0 will be power down with WFI request

5.2.10.3 Low power control register of CPU1 (GPC_LPCR_M4)

NOTE

LPCR_M4[CPU_CLK_ON_LPM] should be set 1'b0 if M4 goes to LPM without trigger power down of related domains

Address: 303A_0000h base + 8h offset = 303A_0008h



GPC_LPCR_M4 field descriptions

Field	Description
31 MASK_DSM_TRIGGER	M4 WFI Mask 0 DSM trigger of M4 platform will not be masked 1 DSM trigger of M4 platform will be masked
30–17 -	This field is reserved. Reserved
16 MASK_M4_WFI	M4 WFI Mask 0 WFI for M4 is not masked 1 WFI for M4 is masked
15 -	This field is reserved. Reserved

Table continues on the next page...

GPC_LPCR_M4 field descriptions (continued)

Field	Description
14 CPU_CLK_ON_LPM	Define if M4 clocks will be disabled on wait/stop mode. 0 M4 clock disabled on wait/stop mode. 1 M4 clock enabled on wait/stop mode.
13–4 -	This field is reserved. Reserved
3 EN_M4_PUP	Enable m4 virtual PGC power up with LPM enter
2 EN_M4_PDN	Enable m4 virtual PGC power down with LPM enter
LPM0	Setting the low power mode that system will enter on next assertion of dsm_request signal. 00 Remain in RUN mode 01 Transfer to WAIT mode 10 Transfer to STOP mode 11 Reserved

5.2.10.4 System low power control register (GPC_SLPCR)

NOTE

SLPCR[VSTBY] must be set to 1'b1 if SLPCR[RBC_EN] is set to 1'b1; SLPCR[SBYOS] must be set to 1'b1 if SLPCR[VSTBY] is set to 1'b1.

Address: 303A_0000h base + 14h offset = 303A_0014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EN_DSM	RBC_EN	REG_BYPASS_COUNT						DISABLE_A53_IS_DSM	Reserved			EN_M4_FASTWUP_STOP_MODE	EN_M4_FASTWUP_WAIT_MODE	EN_A53_FASTWUP_STOP_MODE	EN_A53_FASTWUP_WAIT_MODE
Reset	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OSCCNT								COSC_EN	COSC_PWRDOWN	STBY_COUNT			VSTBY	SBYOS	BYPASS_PMIC_READY
Reset	1	1	1	1	1	1	1	1	1	0	0	0	0	0	1	0

GPC_SLPCR field descriptions

Field	Description
31 EN_DSM	DSM enable 0 DSM disabled 1 DSM enabled
30 RBC_EN	Enable for REG_BYPASS_COUNTER. If enabled, REG_BYPASS signal will be asserted after REG_BYPASS_COUNT clocks of CKIL, after standby voltage is requested. If standby voltage is not requested REG_BYPASS won't be asserted, even if counter is enabled. 0 REG_BYPASS_COUNTER disabled 1 REG_BYPASS_COUNTER enabled
29–24 REG_BYPASS_COUNT	Counter for REG_BYPASS signal assertion after standby voltage request by PMIC_STBY_REQ. **NOTE:** When RBC enabled, interrupt will be masked until the counter counts to the value set in GPC_SLPCR[REG_BYPASS_COUNT], this can ignore the unexpected interrupts before CPU enter LPM mode, avoid the process interruption.

Table continues on the next page...

GPC_SLPCR field descriptions (continued)

Field	Description
	000000 no delay 000001 1 CKIL clock period delay 111111 63 CKIL clock period delay
23 DISABLE_A53_IS_DSM	0 Enable A53 isolation signal in DSM 1 Disable A53 isolation signal in DSM
22–20 -	This field is reserved. Reserved
19 EN_M4_FASTWUP_STOP_MODE	Enable M4 fast wake up stop mode, relevant PLLs will not be closed in this mode.
18 EN_M4_FASTWUP_WAIT_MODE	Enable M4 fast wake up wait mode, relevant PLLs will not be closed in this mode.
17 EN_A53_FASTWUP_STOP_MODE	Enable A53 fast wake up stop mode, relevant PLLs will not be closed in this mode.
16 EN_A53_FASTWUP_WAIT_MODE	Enable A53 fast wake up wait mode, relevant PLLs will not be closed in this mode.
15–8 OSCCNT	Oscillator ready counter value. These bits define value of 32KHz counter, that serve as counter for oscillator lock time. This is used for oscillator lock time. Current estimation is ~5ms. This counter will be used in sequence out of DSM and if sbyos bit was defined. GPC will wait the “OSCCNT” number of cycles before it notify CCM to open the relevant PLLs. 00000000 count 1 ckil 11111111 count 256 ckils
7 COSC_EN	On-chip oscillator enable bit - this bit value is reflected on the output cosc_en. The system will start with on-chip oscillator enabled to supply source for the PLLs. Software can change this bit if a transition to the bypass PLL clocks was performed for all the PLLs. In cases that this bit is changed from ‘0’ to ‘1’ then GPC will enable the on-chip oscillator and after counting oscnt ckil clock cycles before it notify CCM to open the relevant PLLs . The cosc_en bit should be changed only when on-chip oscillator is not chosen as the clock source. 0 Disable on-chip oscillator 1 Enable on-chip oscillator
6 COSC_PWRDOWN	In run mode, software can manually control powering down of on chip oscillator, i.e. generating ‘1’ on cosc_pwrdown signal. If software manually powered down the on chip oscillator, then sbyos functionality for on-chip oscillator will be bypassed. The manual closing of on-chip oscillator should be performed only in case the reference oscillator is not the source of all the clocks generation. 0 On-chip oscillator will not be powered down, i.e. cosc_pwrdown = 0 1 On-chip oscillator will be powered down, i.e. cosc_pwrdown = 1

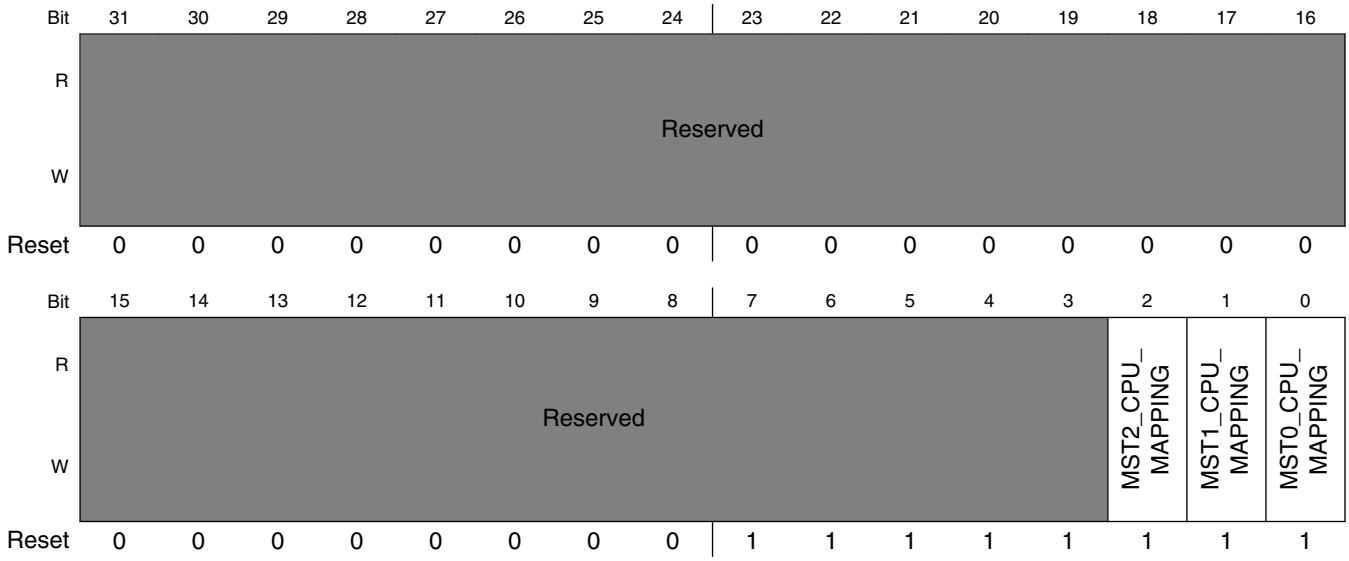
Table continues on the next page...

GPC_SLPCR field descriptions (continued)

Field	Description
5–3 STBY_COUNT	Standby counter definition. These two bits define, in the case of stop exit (if VSTBY bit was set), the amount of time GPC will wait between PMIC_STBY_REQ negation and the check of assertion of PMIC_READY. 000 GPC will wait 4 ckil clock cycles 001 GPC will wait 8 ckil clock cycles 010 GPC will wait 16 ckil clock cycles 011 GPC will wait 32 ckil clock cycles 100 GPC will wait 64 ckil clock cycles 101 GPC will wait 128 ckil clock cycles 110 GPC will wait 256 ckil clock cycles 111 GPC will wait 512 ckil clock cycles
2 VSTBY	Voltage standby request bit. This bit defines if PMIC_STBY_REQ pin, which notifies external power management IC to move from functional voltage to standby voltage, will be asserted in stop mode. 0 Voltage will not be changed to standby voltage after next entrance to stop mode. (PMIC_STBY_REQ will remain negated - ‘0’) 1 Voltage will be changed to standby voltage after next entrance to stop mode.
1 SBYOS	Standby clock oscillator bit. This bit defines if cosc_pwrdown, which power down the on chip oscillator, will be asserted in DSM. 0 On chip oscillator will not be powered down, after next entrance to DSM. 1 On chip oscillator will be powered down, after next entrance to DSM. When returning from DSM, external oscillator will be enabled again, on chip oscillator will return to oscillator mode , and after oscnt count GPC will continue with the exit from DSM process.
0 BYPASS_PMIC_READY	By asserting this bit GPC will bypass waiting for PMIC_READY signal when coming out of DSM. This should be used for PMIC’s that don’t support the PMIC_READY signal. 0 Don’t bypass the PMIC_READY signal - GPC will wait for its assertion during exit of low power mode if standby voltage was enabled 1 Bypass the PMIC_READY signal - GPC will wait for its assertion during exit of low power mode if standby voltage was enabled

5.2.10.5 MASTER LPM Handshake (GPC_MST_CPU_MAPPING)

Address: 303A_0000h base + 18h offset = 303A_0018h

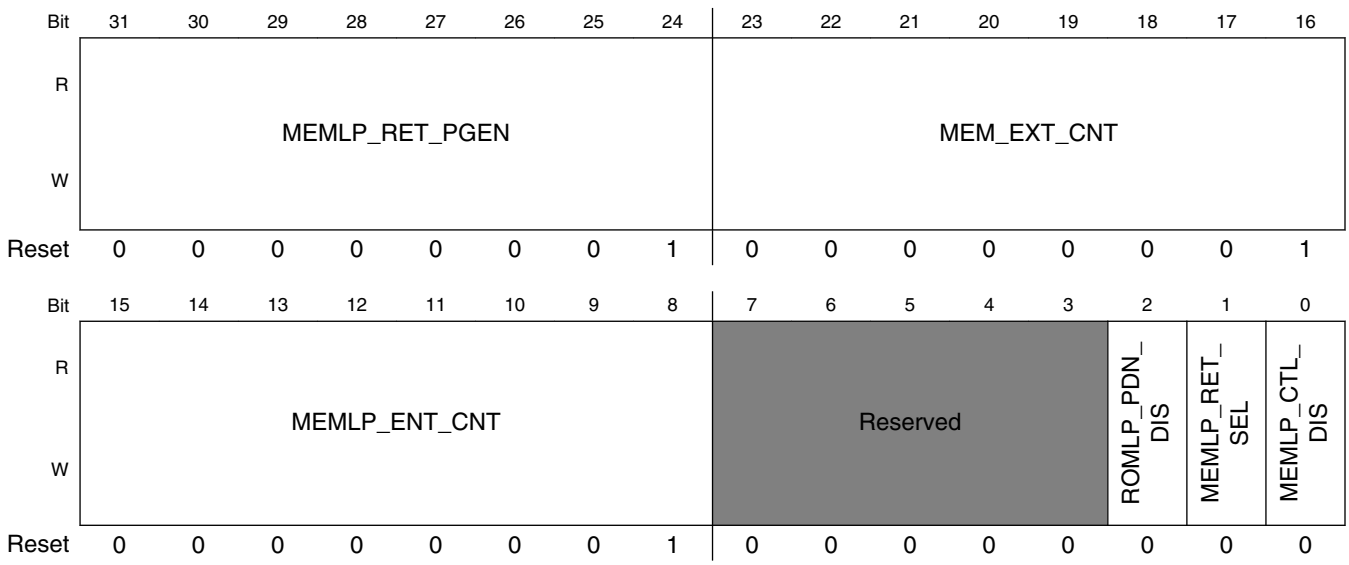


GPC_MST_CPU_MAPPING field descriptions

Field	Description
31–3 -	This field is reserved. Reserved
2 MST2_CPU_MAPPING	MASTER2 CPU Mapping noc2supermix ADB LPM handshake mask. This bit should be used together with LPCR_A53_BSC[8]. If you want power of SCU, use this setting: LPCR_A53_BSC[8]=0; MST_CPU_MAPPING[2]=1 Otherwise, use this: LPCR_A53_BSC[8]=1; MST_CPU_MAPPING[2]=0 0 GPC will not send out power off requirement 1 GPC will send out power off requirement
1 MST1_CPU_MAPPING	MASTER0 CPU Mapping Supermix2noc ADB LPM handshake mask. This bit should be used together with LPCR_A53_BSC[7]. If you want power of SCU, use this setting: LPCR_A53_BSC[7]=0; MST_CPU_MAPPING[1]=1 Otherwise, use this: LPCR_A53_BSC[7]=1; MST_CPU_MAPPING[1]=0 0 GPC will not send out power off requirement 1 GPC will send out power off requirement
0 MST0_CPU_MAPPING	MASTER0 CPU Mapping SCU LPM handshake mask. This bit should be used together with LPCR_A7_BSC[6]. If you want power of SCU, use this setting: LPCR_A7_BSC[6]=0; MST_CPU_MAPPING[0]=1 Otherwise, use this: LPCR_A7_BSC[6]=1; MST_CPU_MAPPING[0]=0 0 GPC will not send out power off requirement 1 GPC will send out power off requirement

5.2.10.6 Memory low power control register (GPC_MLPCR)

Address: 303A_0000h base + 20h offset = 303A_0020h



GPC_MLPCR field descriptions

Field	Description
31–24 MEMLP_RET_PGEN	Delay counter for “retnx” and “pgen”
23–16 MEM_EXT_CNT	Delay counter to start existing from memory low power
15–8 MEMLP_ENT_CNT	Delay counter to make sure all clock off after pll_dis_req is issued by smc
7–3 -	This field is reserved. Reserved
2 ROMLP_PDN_DIS	ROM shut down control 0 Enable ROM shut down control(should also enable RAM low power control); 1 Disable ROM shut down control
1 MEMLP_RET_SEL	Retention select 0 retention mode 2 1 retention mode 1
0 MEMLP_CTL_DIS	RAM low-power control 0 Enable RAM low power control 1 Disable RAM low power control

5.2.10.7 PGC acknowledge signal selection of A53 platform (GPC_PGC_ACK_SEL_A53)

The register can only be accessed by A53 platform

Address: 303A_0000h base + 24h offset = 303A_0024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	A53_PGC_PUP_ACK	A53_C3_PGC_PUP_ACK	A53_C2_PGC_PUP_ACK	Reserved									NOC_PGC_PUP_ACK	A53_PLAT_PGC_PUP_ACK	A53_C1_PGC_PUP_ACK	A53_C0_PGC_PUP_ACK
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	A53_PGC_PDN_ACK	A53_C3_PGC_PDN_ACK	A53_C2_PGC_PDN_ACK	Reserved									NOC_PGC_PDN_ACK	A53_PLAT_PGC_PDN_ACK	A53_C1_PGC_PDN_ACK	A53_C0_PGC_PDN_ACK
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_PGC_ACK_SEL_A53 field descriptions

Field	Description
31 A53_PGC_PUP_ACK	Select power up acknowledge signal of A53 (dummy) PGC as the power up acknowledge for A53 LPM.
30 A53_C3_PGC_PUP_ACK	Select power up acknowledge signal of A53 CORE3 PGC as the power up acknowledge for A53 LPM.
29 A53_C2_PGC_PUP_ACK	Select power up acknowledge signal of A53 CORE2 PGC as the power up acknowledge for A53 LPM.
28–20 -	This field is reserved. Reserved
19 NOC_PGC_PUP_ACK	Select power up acknowledge signal of NOC PGC as the power up acknowledge for A53 LPM.
18 A53_PLAT_PGC_PUP_ACK	Select power up acknowledge signal of A53 PLATFORM PGC as the power up acknowledge for A53 LPM.

Table continues on the next page...

GPC_PGC_ACK_SEL_A53 field descriptions (continued)

Field	Description
17 A53_C1_PGC_PUP_ACK	Select power up acknowledge signal of A53 CORE1 PGC as the power up acknowledge for A53 LPM.
16 A53_C0_PGC_PUP_ACK	Select power up acknowledge signal of A53 CORE0 PGC as the power up acknowledge for A53 LPM.
15 A53_PGC_PDN_ACK	Select power down acknowledge signal of A53 (dummy) PGC as the power down acknowledge for A53 LPM.
14 A53_C3_PGC_PDN_ACK	Select power down acknowledge signal of A53 CORE3 PGC as the power down acknowledge for A53 LPM.
13 A53_C2_PGC_PDN_ACK	Select power down acknowledge signal of A53 CORE2 PGC as the power down acknowledge for A53 LPM.
12–4 -	This field is reserved. Reserved
3 NOC_PGC_PDN_ACK	Select power down acknowledge signal of NOC PGC as the power down acknowledge for A53 LPM.
2 A53_PLAT_PGC_PDN_ACK	Select power down acknowledge signal of A53 PLATFORM PGC as the power down acknowledge for A53 LPM.
1 A53_C1_PGC_PDN_ACK	Select power down acknowledge signal of A53 CORE1 PGC as the power down acknowledge for A53 LPM.
0 A53_C0_PGC_PDN_ACK	Select power down acknowledge signal of A53 CORE0 PGC as the power down acknowledge for A53 LPM.

5.2.10.8 PGC acknowledge signal selection of M4 platform (GPC_PGC_ACK_SEL_M4)

This register can only be accessed by the M4 platform.

NOTE

"dummy" PGC cannot be mapped to time slot control. "virtual" PGC can be mapped to time slot control. When virtual PGC is used, below setting is required - GPC_MISC[M4_PDN_REQ_MASK] should be set to 1'b1 and arrange virtual GPC in same slot with MIX. power/up slot (See example code 3). MIX PGC may possibly power down later than A53 platform power down when virtual PGC is used.

Address: 303A_0000h base + 28h offset = 303A_0028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	M4_DUMMY_PGC_PUP_ACK	Reserved													NOC_PGC_PUP_ACK	M4_VIRTUAL_PGC_PUP_ACK
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	M4_DUMMY_PGC_PDN_ACK	Reserved													NOC_PGC_PDN_ACK	M4_VIRTUAL_PGC_PDN_ACK
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_PGC_ACK_SEL_M4 field descriptions

Field	Description
31 M4_DUMMY_PGC_PUP_ACK	Select power up acknowledge signal of M4 (dummy) PGC as the power up acknowledge for M4 LPM.
30–18 -	This field is reserved. Reserved

Table continues on the next page...

GPC_PGC_ACK_SEL_M4 field descriptions (continued)

Field	Description
17 NOC_PGC_ PUP_ACK	Select power up acknowledge signal of NOC PGC as the power up acknowledge for M4 LPM.
16 M4_VIRTUAL_ PGC_PUP_ACK	Select power up acknowledge signal of M4 virtual PGC as the power up acknowledge for M4 LPM. M4 virtual PGC only acknowledge power up request in the end of current slot time
15 M4_DUMMY_ PGC_PDN_ACK	Select power down acknowledge signal of M4 (dummy) PGC as the power down acknowledge for M4 LPM.
14–2 -	This field is reserved. Reserved
1 NOC_PGC_ PDN_ACK	Select power down acknowledge signal of NOC PGC as the power down acknowledge for M4 LPM.
0 M4_VIRTUAL_ PGC_PDN_ACK	Select power down acknowledge signal of M4 virtual PGC as the power down acknowledge for M4 LPM. M4 virtual PGC only acknowledge power down request in the end of current slot time

5.2.10.9 GPC Miscellaneous register (GPC_MISC)

Address: 303A_0000h base + 2Ch offset = 303A_002Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved						M4_BYPASS_PUP_MASK	A53_BYPASS_PUP_MASK	Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							M4_PDN_REQ_MASK	Reserved		GPC_IRQ_MASK	Reserved			A53_SLEEP_HOLD_REQ_B	M4_SLEEP_HOLD_REQ_B
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1

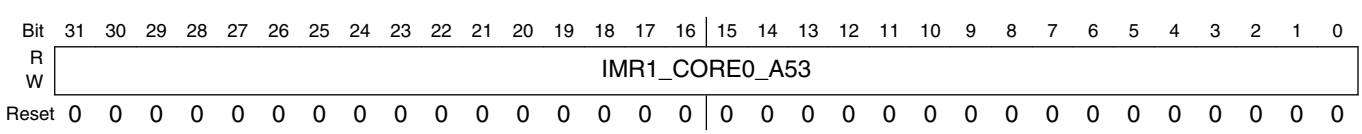
GPC_MISC field descriptions

Field	Description
31–26 -	This field is reserved. Reserved
25 M4_BYPASS_ PUP_MASK	
24 A53_BYPASS_ PUP_MASK	
23–9 -	This field is reserved. Reserved
8 M4_PDN_REQ_ MASK	M4 power-down mask 0 M4 power down request to virtual M4 PGC will be masked. 1 M4 power down request to virtual M4 PGC will not be masked. Set this bit to 1'b1 when M4 virtual PGC is used.
7–6 -	This field is reserved. Reserved
5 GPC_IRQ_MASK	GPC interrupt/event masking 0 Not masked 1 Interrupt / event is masked
4–2 -	This field is reserved. Reserved
1 A53_SLEEP_ HOLD_REQ_B	A53 sleep hold 0 Hold A53 platform in sleep mode. This bit is a software control bit to A53 platform. 1 Don't hold A53 platform in sleep mode.
0 M4_SLEEP_ HOLD_REQ_B	M4 sleep hold 0 Hold M4 platform in sleep mode. This bit is a software control bit to M4 platform. 1 Don't hold M4 platform in sleep mode.

5.2.10.10 IRQ masking register 1 of A53 core0 (GPC_IMR1_CORE0_A53)

The four IMRn_CORE0_A53 (n = 1,2,3,4) registers are used as interrupt mask for A53 core0.

Address: 303A_0000h base + 30h offset = 303A_0030h



GPC_IMR1_CORE0_A53 field descriptions

Field	Description
IMR1_CORE0_A53	A53 core0 IRQ[31:0] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.11 IRQ masking register 2 of A53 core0 (GPC_IMR2_CORE0_A53)

Address: 303A_0000h base + 34h offset = 303A_0034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR2_CORE0_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR2_CORE0_A53 field descriptions

Field	Description
IMR2_CORE0_A53	A53 core0 IRQ[63:32] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.12 IRQ masking register 3 of A53 core0 (GPC_IMR3_CORE0_A53)

Address: 303A_0000h base + 38h offset = 303A_0038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR3_CORE0_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR3_CORE0_A53 field descriptions

Field	Description
IMR3_CORE0_A53	A53 core0 IRQ[95:64] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.13 IRQ masking register 4 of A53 core0 (GPC_IMR4_CORE0_A53)

Address: 303A_0000h base + 3Ch offset = 303A_003Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR4_CORE0_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR4_CORE0_A53 field descriptions

Field	Description
IMR4_CORE0_A53	A53 core0 IRQ[127:96] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.14 IRQ masking register 1 of A53 core1 (GPC_IMR1_CORE1_A53)

The four IMRn_CORE1_A53 (n = 1,2,3,4) registers are used as interrupt mask for A53 core1.

Address: 303A_0000h base + 40h offset = 303A_0040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR1_CORE1_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR1_CORE1_A53 field descriptions

Field	Description
IMR1_CORE1_A53	A53 core1 IRQ[31:0] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.15 IRQ masking register 2 of A53 core1 (GPC_IMR2_CORE1_A53)

Address: 303A_0000h base + 44h offset = 303A_0044h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR2_CORE1_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR2_CORE1_A53 field descriptions

Field	Description
IMR2_CORE1_A53	A53 core1 IRQ[63:32] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.16 IRQ masking register 3 of A53 core1 (GPC_IMR3_CORE1_A53)

Address: 303A_0000h base + 48h offset = 303A_0048h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR3_CORE1_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR3_CORE1_A53 field descriptions

Field	Description
IMR3_CORE1_A53	A53 core1 IRQ[95:64] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.17 IRQ masking register 4 of A53 core1 (GPC_IMR4_CORE1_A53)

Address: 303A_0000h base + 4Ch offset = 303A_004Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR4_CORE1_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR4_CORE1_A53 field descriptions

Field	Description
IMR4_CORE1_A53	A53 core1 IRQ[127:96] masking bits:

GPC_IMR4_CORE1_A53 field descriptions (continued)

Field	Description
	0 IRQ not masked 1 IRQ masked

5.2.10.18 IRQ masking register 1 of M4 (GPC_IMR1_M4)

The four IMRn_M4 (n = 1,2,3,4) registers are used as interrupt mask for M4.

Address: 303A_0000h base + 50h offset = 303A_0050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR1_M4																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR1_M4 field descriptions

Field	Description
IMR1_M4	M4 IRQ[31:0] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.19 IRQ masking register 2 of M4 (GPC_IMR2_M4)

Address: 303A_0000h base + 54h offset = 303A_0054h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR2_M4																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR2_M4 field descriptions

Field	Description
IMR2_M4	M4 IRQ[63:32] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.20 IRQ masking register 3 of M4 (GPC_IMR3_M4)

Address: 303A_0000h base + 58h offset = 303A_0058h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR3_M4																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR3_M4 field descriptions

Field	Description
IMR3_M4	M4 IRQ[95:64] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.21 IRQ masking register 4 of M4 (GPC_IMR4_M4)

Address: 303A_0000h base + 5Ch offset = 303A_005Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR4_M4																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR4_M4 field descriptions

Field	Description
IMR4_M4	M4 IRQ[127:96] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.22 IRQ status register 1 of A53 (GPC_ISR1_A53)

The four ISRn_A53 (n = 1,2,3,4) registers, all of them are read only register

Address: 303A_0000h base + 70h offset = 303A_0070h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR1_A53																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_ISR1_A53 field descriptions

Field	Description
ISR1_A53	A53 IRQ[31:0] status

5.2.10.23 IRQ status register 2 of A53 (GPC_ISR2_A53)

Address: 303A_0000h base + 74h offset = 303A_0074h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR2_A53																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_ISR2_A53 field descriptions

Field	Description
ISR2_A53	A53 IRQ[63:32] status

5.2.10.24 IRQ status register 3 of A53 (GPC_ISR3_A53)

Address: 303A_0000h base + 78h offset = 303A_0078h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR3_A53																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_ISR3_A53 field descriptions

Field	Description
ISR3_A53	A53 IRQ[95:64] status

5.2.10.25 IRQ status register 4 of A53 (GPC_ISR4_A53)

Address: 303A_0000h base + 7Ch offset = 303A_007Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR4_A53																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_ISR4_A53 field descriptions

Field	Description
ISR4_A53	A53 IRQ[127:96] status

5.2.10.26 IRQ status register 1 of M4 (GPC_ISR1_M4)

The four ISRn_M4 (n = 1,2,3,4) registers, all of them are read only register

Address: 303A_0000h base + 80h offset = 303A_0080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR1_M4																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_ISR1_M4 field descriptions

Field	Description
ISR1_M4	M4 IRQ[31:0] status

5.2.10.27 IRQ status register 2 of M4 (GPC_ISR2_M4)

Address: 303A_0000h base + 84h offset = 303A_0084h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR2_M4																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_ISR2_M4 field descriptions

Field	Description
ISR2_M4	M4 IRQ[63:32] status

5.2.10.28 IRQ status register 3 of M4 (GPC_ISR3_M4)

Address: 303A_0000h base + 88h offset = 303A_0088h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR3_M4																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_ISR3_M4 field descriptions

Field	Description
ISR3_M4	M4 IRQ[95:64] status

5.2.10.29 IRQ status register 4 of M4 (GPC_ISR4_M4)

Address: 303A_0000h base + 8Ch offset = 303A_008Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR4_M4																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_ISR4_M4 field descriptions

Field	Description
ISR4_M4	M4 IRQ[127:96] status

5.2.10.30 Slot configure register for A53 core (GPC_SLT*n*_CFG)

There are 20 slots in each SLTn_CFG(n = 0~19) will define the power up or power down behavior of one or more A53 CORE or SCU PGC in that each slot.

In each “SLTn_cfg”, 2 bits (slt_cfg[1:0])are reserved for each PGC:

- 2’b01 (slot controller will power down relevant PGC in corresponding slot if hardware power down request asserted)
- 2’b10 (slot controller will power up relevant PGC in corresponding slot if hardware power up request asserted)
- 2’b00 or 2’b11 (not power down or power up behavior in relevant slot)

The specific bits assignment for each PGC is shown in the table below.

	PGCx	PGCx-1	..	PGC2	PGC1	PGC0
SLT0_CFG	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]
SLT1_CFG	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]
:	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]
SLTn_CFG	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]

Address: 303A_0000h base + B0h offset + (4d × i), where i=0d to 14d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				NOC_PUP_SLOT_CONTROL	NOC_PDN_SLOT_CONTROL	SCU_PUP_SLOT_CONTROL	SCU_PDN_SLOT_CONTROL	CORE3_A53_PUP_SLOT_CONTROL	CORE3_A53_PDN_SLOT_CONTROL	CORE2_A53_PUP_SLOT_CONTROL	CORE2_A53_PDN_SLOT_CONTROL	CORE1_A53_PUP_SLOT_CONTROL	CORE1_A53_PDN_SLOT_CONTROL	CORE0_A53_PUP_SLOT_CONTROL	CORE0_A53_PDN_SLOT_CONTROL
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_SLT*n*_CFG field descriptions

Field	Description
31–12 -	This field is reserved.
11 NOC_PUP_SLOT_CONTROL	NOC Power-up slot control
10 NOC_PDN_SLOT_CONTROL	NOC Power-down slot control
9 SCU_PUP_SLOT_CONTROL	SCU Power-up slot control

Table continues on the next page...

GPC_SLT*n*_CFG field descriptions (continued)

Field	Description
8 SCU_PDN_ SLOT_ CONTROL	SCU Power-down slot control
7 CORE3_A53_ PUP_SLOT_ CONTROL	CORE3 A53 Power-up slot control
6 CORE3_A53_ PDN_SLOT_ CONTROL	CORE3 A53 Power-down slot control
5 CORE2_A53_ PUP_SLOT_ CONTROL	CORE2 A53 Power-up slot control
4 CORE2_A53_ PDN_SLOT_ CONTROL	CORE2 A53 Power-down slot control
3 CORE1_A53_ PUP_SLOT_ CONTROL	CORE1 A53 Power-up slot control
2 CORE1_A53_ PDN_SLOT_ CONTROL	CORE1 A53 Power-down slot control
1 CORE0_A53_ PUP_SLOT_ CONTROL	CORE0 A53 Power-up slot control
0 CORE0_A53_ PDN_SLOT_ CONTROL	CORE0 A53 Power-down slot control

5.2.10.31 PGC CPU mapping (GPC_PGC_CPU_0_1_MAPPING)

Address: 303A_0000h base + ECh offset = 303A_00ECh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	VPU_H1_M4_DOMAIN	VPU_G2_M4_DOMAIN	VPU_G1_M4_DOMAIN	DISPMIX_M4_DOMAIN	GPU_3D_M4_DOMAIN	VPUMIX_M4_DOMAIN	GPUMIX_M4_DOMAIN	GPU_2D_M4_DOMAIN	DDR1_M4_DOMAIN	Reserved	OTG2_M4_DOMAIN	OTG1_M4_DOMAIN	PCIE_M4_DOMAIN	MIPI_M4_DOMAIN	NOC_M4_DOMAIN	MF_M4_DOMAIN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VPU_H1_A53_DOMAIN	VPU_G2_A53_DOMAIN	VPU_G1_A53_DOMAIN	DISPMIX_A53_DOMAIN	GPU_3D_A53_DOMAIN	VPUMIX_A53_DOMAIN	GPUMIX_A53_DOMAIN	GPU_2D_A53_DOMAIN	DDR1_A53_DOMAIN	Reserved	OTG2_A53_DOMAIN	OTG1_A53_DOMAIN	PCIE_A53_DOMAIN	MIPI_A53_DOMAIN	NOC_A53_DOMAIN	MF_A53_DOMAIN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_PGC_CPU_0_1_MAPPING field descriptions

Field	Description
31 VPU_H1_M4_DOMAIN	VPU_H1_M4_DOMAIN
30 VPU_G2_M4_DOMAIN	VPU_G2_M4_DOMAIN
29 VPU_G1_M4_DOMAIN	VPU_G1_M4_DOMAIN
28 DISPMIX_M4_DOMAIN	DISPMIX_M4_DOMAIN
27 GPU_3D_M4_DOMAIN	GPU_3D_M4_DOMAIN

Table continues on the next page...

GPC_PGC_CPU_0_1_MAPPING field descriptions (continued)

Field	Description
26 VPUMIX_M4_DOMAIN	VPUMIX_M4_DOMAIN
25 GPUMIX_M4_DOMAIN	GPUMIX_M4_DOMAIN
24 GPU_2D_M4_DOMAIN	GPU_2D_M4_DOMAIN
23 DDR1_M4_DOMAIN	DDR1_M4_DOMAIN
22 -	This field is reserved.
21 OTG2_M4_DOMAIN	OTG2_M4_DOMAIN
20 OTG1_M4_DOMAIN	OTG1_M4_DOMAIN
19 PCIE_M4_DOMAIN	PCIE_M4_DOMAIN
18 MIPI_M4_DOMAIN	MIPI_M4_DOMAIN
17 NOC_M4_DOMAIN	NOC_M4_DOMAIN
16 MF_M4_DOMAIN	MF_M4_DOMAIN
15 VPU_H1_A53_DOMAIN	VPU_H1_A53_DOMAIN
14 VPU_G2_A53_DOMAIN	VPU_G2_A53_DOMAIN
13 VPU_G1_A53_DOMAIN	VPU_G1_A53_DOMAIN
12 DISPMIX_A53_DOMAIN	DISP_MIXA53_DOMAIN
11 GPU_3D_A53_DOMAIN	GPU_3D_A53_DOMAIN

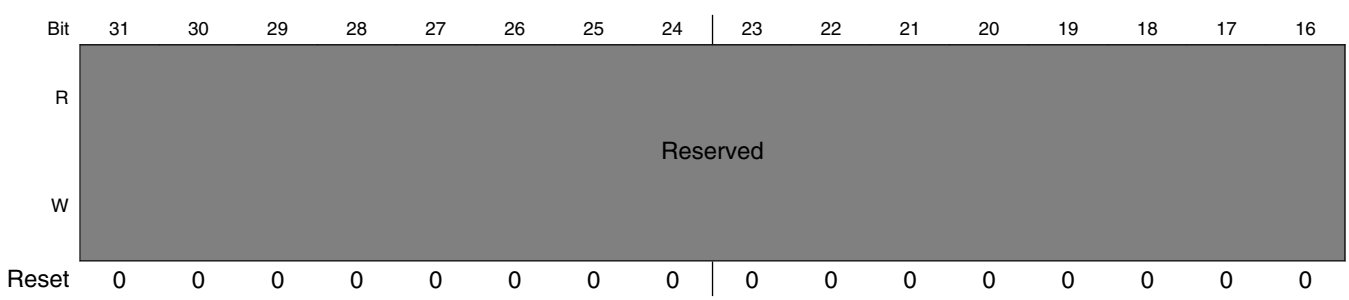
Table continues on the next page...

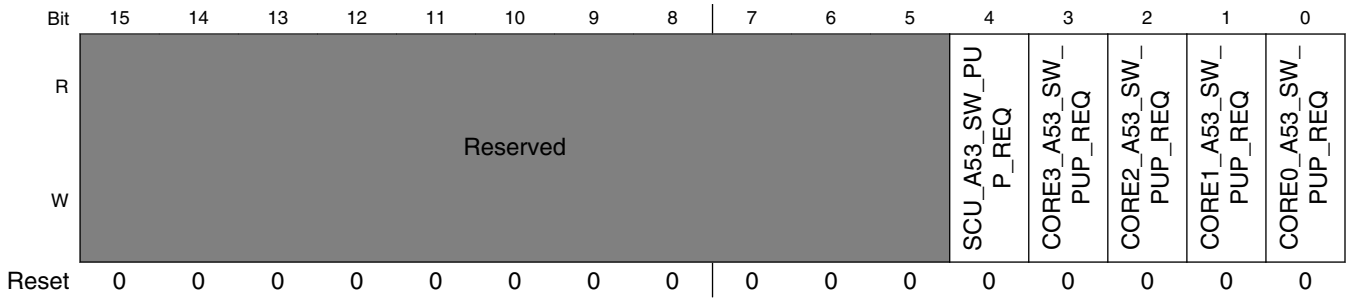
GPC_PGC_CPU_0_1_MAPPING field descriptions (continued)

Field	Description
10 VPUMIX_A53_DOMAIN	VPUMIX_A53_DOMAIN
9 GPUMIX_A53_DOMAIN	GPUMIX_A53_DOMAIN
8 GPU_2D_A53_DOMAIN	GPU_2D_A53_DOMAIN
7 DDR1_A53_DOMAIN	DDR1_A53_DOMAIN
6 -	This field is reserved.
5 OTG2_A53_DOMAIN	OTG2_A53_DOMAIN
4 OTG1_A53_DOMAIN	OTG1_A53_DOMAIN
3 PCIE_A53_DOMAIN	PCIE_A53_DOMAIN
2 MIPI_A53_DOMAIN	MIPI A53 DOMAIN
1 NOC_A53_DOMAIN	NOC_A53_DOMAIN
0 MF_A53_DOMAIN	MF_A53_DOMAIN

5.2.10.32 CPU PGC software power up trigger (GPC_CPU_PGC_SW_PUP_REQ)

Address: 303A_0000h base + F0h offset = 303A_00F0h



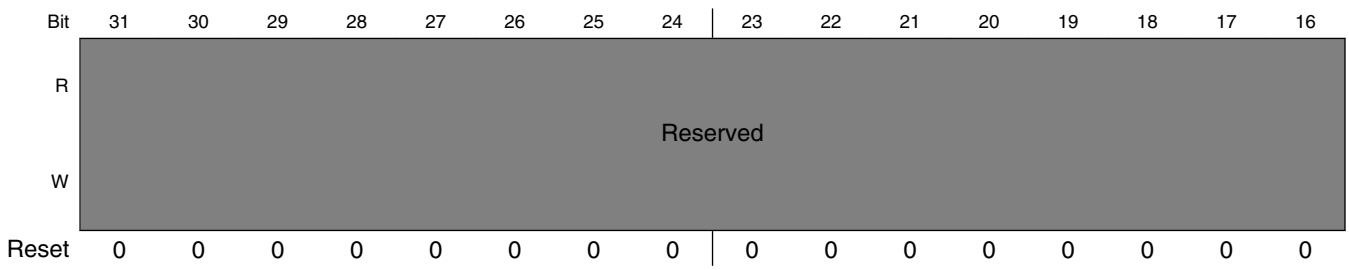


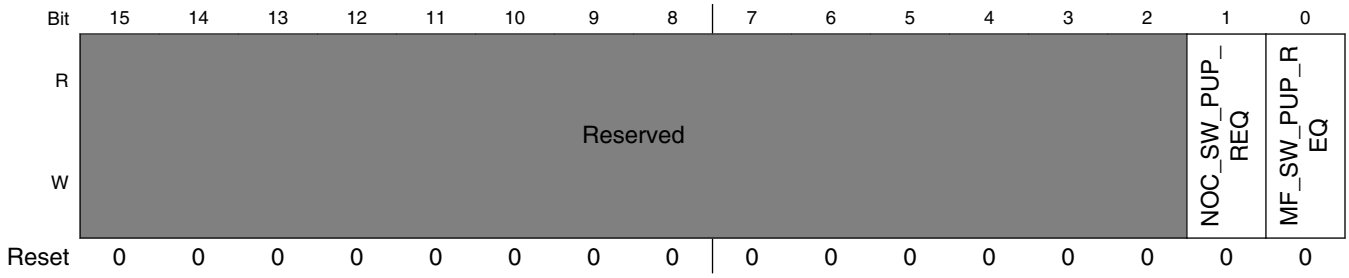
GPC_CPU_PGC_SW_PUP_REQ field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SCU_A53_SW_PUP_REQ	Software power up trigger for SCU A53 PGC
3 CORE3_A53_SW_PUP_REQ	Software power up trigger for Core3 A53 PGC
2 CORE2_A53_SW_PUP_REQ	Software power up trigger for Core2 A53
1 CORE1_A53_SW_PUP_REQ	Software power up trigger for Core1 A53 PGC
0 CORE0_A53_SW_PUP_REQ	Software power up trigger for Core0 A53 PGC

5.2.10.33 MIX PGC software power up trigger (GPC_MIX_PGC_SW_PUP_REQ)

Address: 303A_0000h base + F4h offset = 303A_00F4h



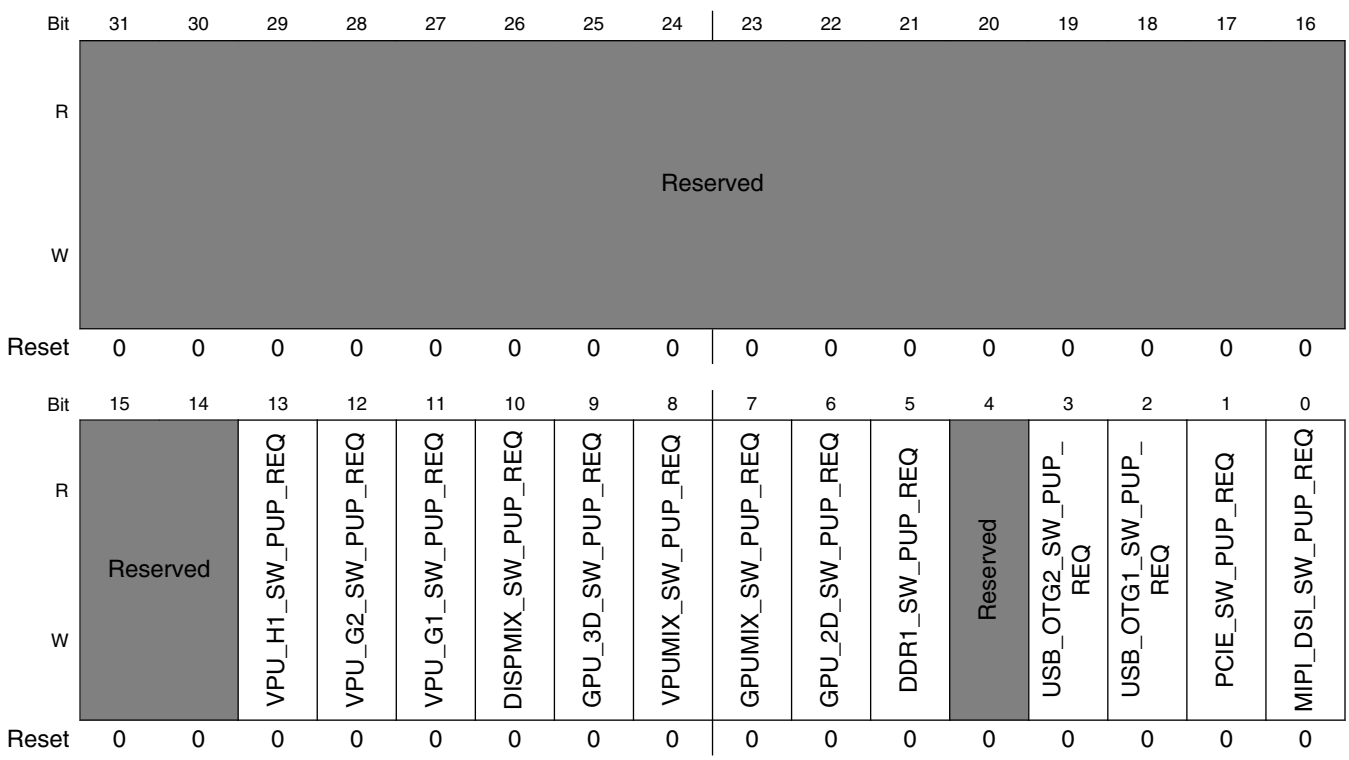


GPC_MIX_PGC_SW_PUP_REQ field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
1 NOC_SW_PUP_REQ	Software power up trigger for NOC PGC
0 MF_SW_PUP_REQ	Software power up trigger for MIX PGC

5.2.10.34 PU PGC software up trigger (GPC_PU_PGC_SW_PUP_REQ)

Address: 303A_0000h base + F8h offset = 303A_00F8h

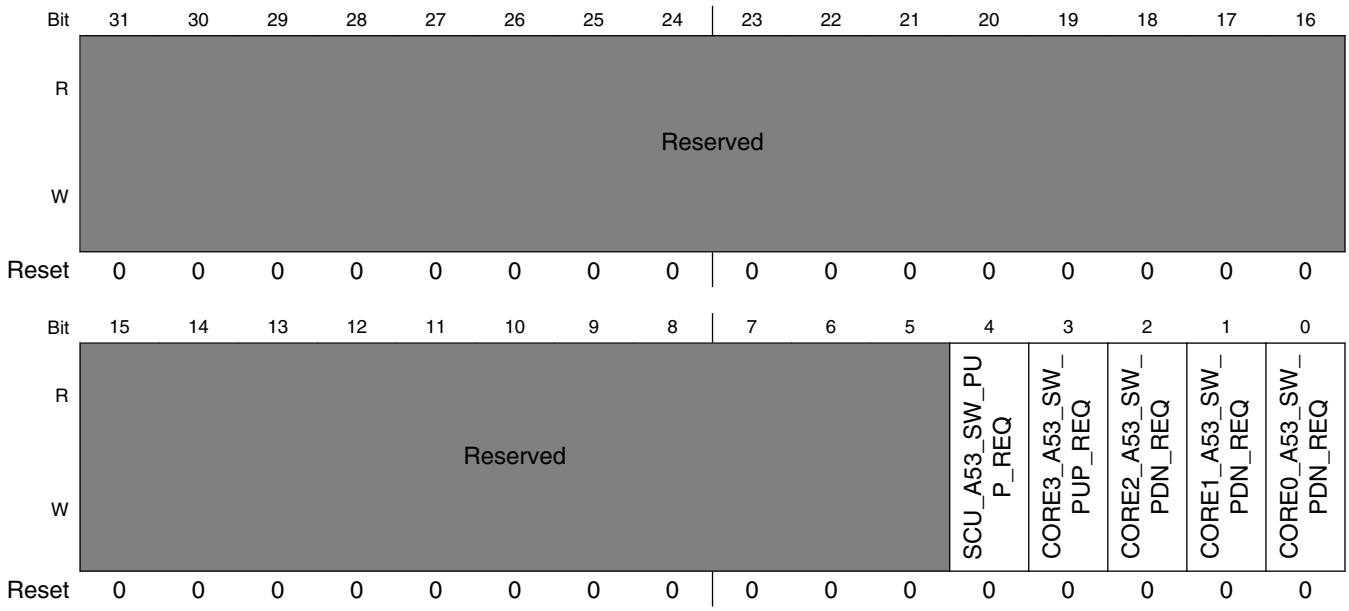


GPC_PU_PGC_SW_PUP_REQ field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13 VPU_H1_SW_PUP_REQ	Software power up trigger for VPU_H1
12 VPU_G2_SW_PUP_REQ	Software power up trigger for MIPI CSI2
11 VPU_G1_SW_PUP_REQ	Software power up trigger for VPU_G1
10 DISPMIX_SW_PUP_REQ	Software power up trigger for DISPMIX
9 GPU_3D_SW_PUP_REQ	Software power up trigger for GPU_3D
8 VPUMIX_SW_PUP_REQ	Software power up trigger for VPUMIX
7 GPUMIX_SW_PUP_REQ	Software power up trigger for GPUMIX
6 GPU_2D_SW_PUP_REQ	Software power up trigger for GPU_2D
5 DDR1_SW_PUP_REQ	Software power up trigger for DDR1
4 -	This field is reserved.
3 USB_OTG2_SW_PUP_REQ	Software power up trigger for USB_OTG2
2 USB_OTG1_SW_PUP_REQ	Software power up trigger for USB_OTG1
1 PCIE_SW_PUP_REQ	Software power up trigger for PCIE
0 MIPI_DSI_SW_PUP_REQ	Software power up trigger for MIPI_DSI

5.2.10.35 CPU PGC software down trigger (GPC_CPU_PGC_SW_PDN_REQ)

Address: 303A_0000h base + FCh offset = 303A_00FCh

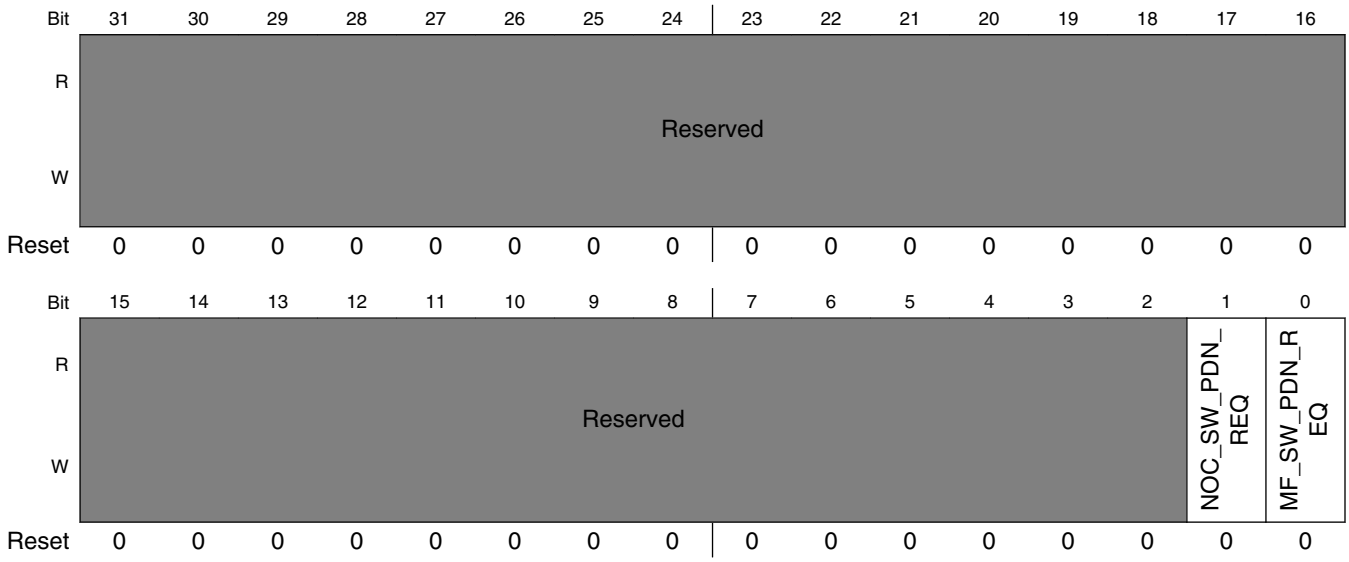


GPC_CPU_PGC_SW_PDN_REQ field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SCU_A53_SW_PUP_REQ	Software power up trigger for SCU A53 PGC
3 CORE3_A53_SW_PUP_REQ	Software power up trigger for Core3 A53 PGC
2 CORE2_A53_SW_PDN_REQ	Software power down trigger for Core2 A53
1 CORE1_A53_SW_PDN_REQ	Software power down trigger for Core1 A53 PGC
0 CORE0_A53_SW_PDN_REQ	Software power down trigger for Core0 A53 PGC

5.2.10.36 MIX PGC software power down trigger (GPC_MIX_PGC_SW_PDN_REQ)

Address: 303A_0000h base + 100h offset = 303A_0100h

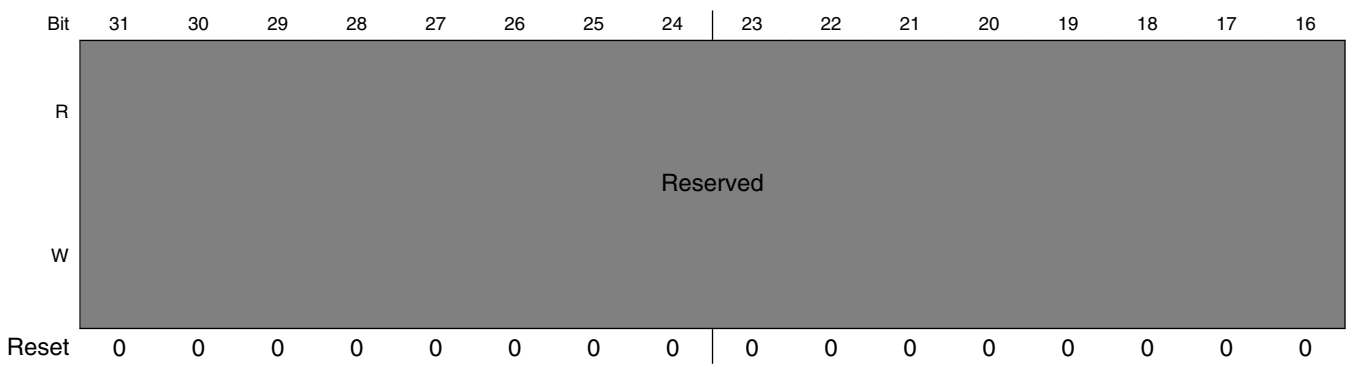


GPC_MIX_PGC_SW_PDN_REQ field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
1 NOC_SW_PDN_REQ	Software power down trigger for NOC PGC
0 MF_SW_PDN_REQ	Software power down trigger for MIX PGC

5.2.10.37 PU PGC software down trigger (GPC_PU_PGC_SW_PDN_REQ)

Address: 303A_0000h base + 104h offset = 303A_0104h



Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		VPU_H1_SW_PDN_REQ	VPU_G2_SW_PDN_REQ	VPU_G1_SW_PDN_REQ	DISPMIX_SW_PDN_REQ	GPU_3D_SW_PDN_REQ	VPUMIX_SW_PDN_REQ	GPUMIX_SW_PDN_REQ	GPU_2D_SW_PDN_REQ	DDR1_SW_PDN_REQ	Reserved	USB_OTG2_SW_PDN_REQ	USB_OTG1_SW_PDN_REQ	PCIE_SW_PDN_REQ	MIPI_DSI_SW_PDN_REQ
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_PU_PGC_SW_PDN_REQ field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13 VPU_H1_SW_PDN_REQ	Software power down trigger for VPU_H1
12 VPU_G2_SW_PDN_REQ	Software power down trigger for VPU_G2
11 VPU_G1_SW_PDN_REQ	Software power down trigger for VPU_G1
10 DISPMIX_SW_PDN_REQ	Software power down trigger for DISPMIX
9 GPU_3D_SW_PDN_REQ	Software power down trigger for GPU_3D
8 VPUMIX_SW_PDN_REQ	Software power down trigger for VPUMIX
7 GPUMIX_SW_PDN_REQ	Software power down trigger for GPUMIX
6 GPU_2D_SW_PDN_REQ	Software power down trigger for GPU_2D
5 DDR1_SW_PDN_REQ	Software power down trigger for DDR1
4 -	This field is reserved.
3 USB_OTG2_SW_PDN_REQ	Software power down trigger for USB_OTG2

Table continues on the next page...

GPC_PU_PGC_SW_PDN_REQ field descriptions (continued)

Field	Description
2 USB_OTG1_SW_PDN_REQ	Software power down trigger for USB_OTG1
1 PCIE_SW_PDN_REQ	Software power down trigger for PCIE
0 MIPI_DSI_SW_PDN_REQ	Software power down trigger for MIPI_DSI

5.2.10.38 Basic Low power control register of A53 platform (GPC_LPCR_A53_BSC2)

Address: 303A_0000h base + 108h offset = 303A_0108h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												LPM3		LPM2	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_LPCR_A53_BSC2 field descriptions

Field	Description
31–4 -	This field is reserved. Reserved
3–2 LPM3	CORE3 Setting the low power mode that system will enter on next assertion of dsm_request signal. 00 Remain in RUN mode 01 Transfer to WAIT mode 10 Transfer to STOP mode 11 Reserved
LPM2	CORE2 Setting the low power mode that system will enter on next assertion of dsm_request signal. 00 Remain in RUN mode 01 Transfer to WAIT mode 10 Transfer to STOP mode 11 Reserved

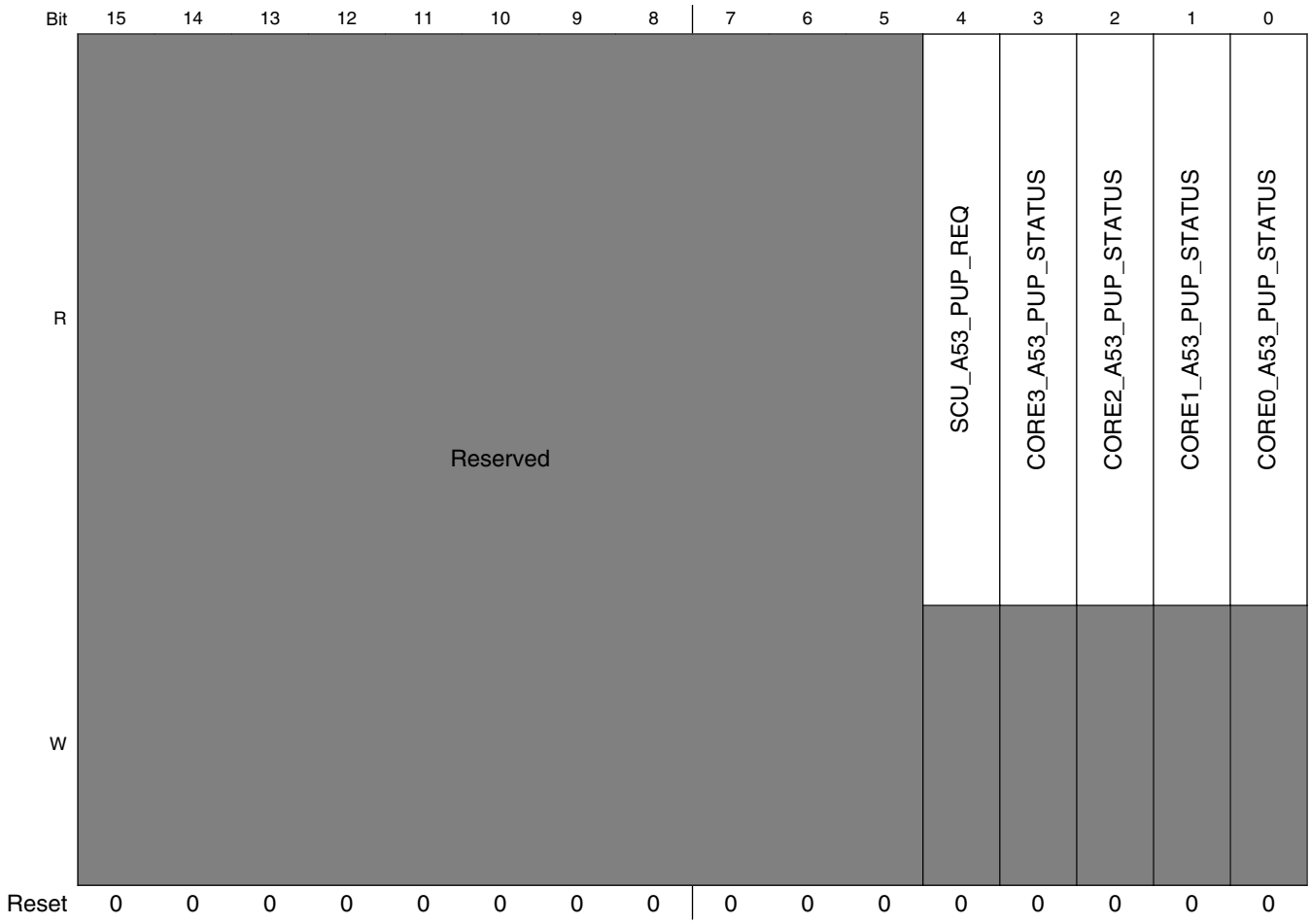
5.2.10.39 CPU PGC software up trigger status1 (GPC_CPU_PGC_PUP_STATUS1)

CPU_PGC_PUP_STATUS1 is a read only register, represents the results for power up software trigger for CPU type PGCs.

The field description is show in table below, the value of “1’b1” represent the software power up trigger failed because the relevant PGC is in a power down process. The relevant bit will be cleared after a success operation of power up software trigger for CPU type PGCs.

Address: 303A_0000h base + 130h offset = 303A_0130h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



GPC_CPU_PGC_PUP_STATUS1 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SCU_A53_PUP_REQ	
3 CORE3_A53_PUP_STATUS	
2 CORE2_A53_PUP_STATUS	
1 CORE1_A53_PUP_STATUS	
0 CORE0_A53_PUP_STATUS	

5.2.10.40 A53 MIX software up trigger status register (GPC_A53_MIX_PGC_PUP_STATUS*n*)

A53_MIX_PGC_PUP_STATUSn (n = 0,1,2) are a read only register, represents the results for power up software trigger from A53 platform to MIX type PGCs.

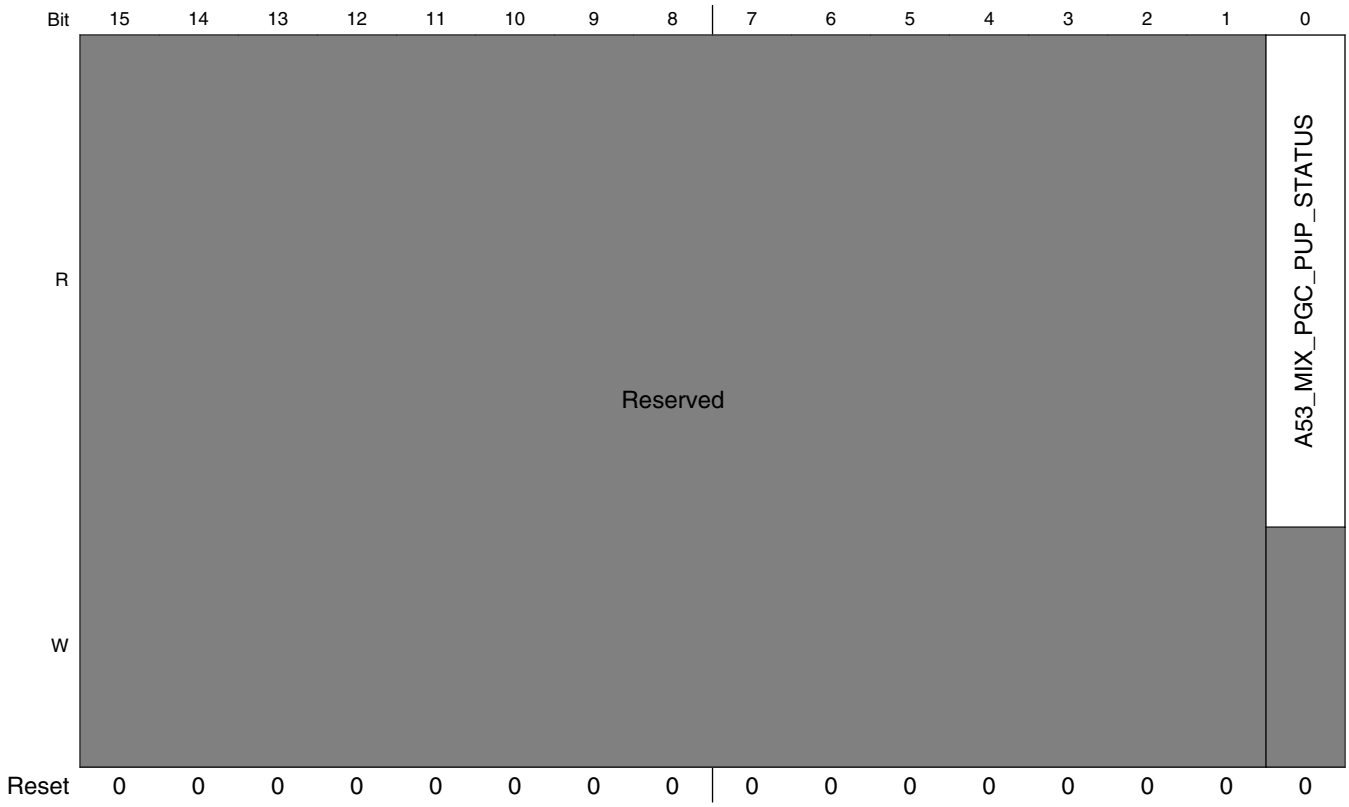
A53_MIX_PGC_PUP_STATUS0: value of “1’b1” represent the software power up trigger failed because domain control condition. The relevant bit will be cleared after a success operation of power up software trigger for MIX type PGCs.

A53_MIX_PGC_PUP_STATUS1: value of “1’b1” represent the software power up trigger failed because the relevant PGC is in a power up process. The relevant bit will be cleared after a success operation of power up software trigger for MIX type PGCs.

A53_MIX_PGC_PUP_STATUS2: value of “1’b1” represent the software power up trigger failed because time slot control is busy. The relevant bit will be cleared after a success operation of power up software trigger for MIX type PGCs.

Address: 303A_0000h base + 134h offset + (4d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



GPC_A53_MIX_PGC_PUP_STATUS*n* field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 A53_MIX_PGC_PUP_STATUS	

5.2.10.41 M4 MIX PGC software up trigger status register (GPC_M4_MIX_PGC_PUP_STATUS*n*)

M4_MIX_PGC_PUP_STATUSn (n = 0,1,2) are a read only register, represents the results for power up software trigger from M4 platform to MIX type PGCs.

M4_MIX_PGC_PUP_STATUS0: value of “1’b1” represent the software power up trigger failed because domain control condition. The relevant bit will be cleared after a success operation of power up software trigger for MIX type PGCs.

M4_MIX_PGC_PUP_STATUS1: value of “1’b1” represent the software power up trigger failed because the relevant PGC is in a power up process. The relevant bit will be cleared after a success operation of power up software trigger for MIX type PGCs.

M4_MIX_PGC_PUP_STATUS2: value of “1’b1” represent the software power up trigger failed because time slot control is busy. The relevant bit will be cleared after a success operation of power up software trigger for MIX type PGCs.

Address: 303A_0000h base + 140h offset + (4d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															M4_MIX_PGC_PUP_STATUS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_M4_MIX_PGC_PUP_STATUS*n* field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 M4_MIX_PGC_ PUP_STATUS	

5.2.10.42 A53 PU software up trigger status register (GPC_A53_PU_PGC_PUP_STATUS*n*)

A53_PU_PGC_PUP_STATUSn (n = 0,1,2) are a read only register, represents the results for power up software trigger from A53 platform to PU type PGCs.

A53_PU_PGC_PUP_STATUS0: value of “1’b1” represent the software power up trigger failed because domain control condition. The relevant bit will be cleared after a success operation of power up software trigger for PU type PGCs.

A53_PU_PGC_PUP_STATUS1: value of “1’b1” represent the software power up trigger failed because the relevant PGC is in a power up process. The relevant bit will be cleared after a success operation of power up software trigger for PU type PGCs.

A53_PU_PGC_PUP_STATUS2: value of “1’b1” represent the software power up trigger failed because time slot control is busy. The relevant bit will be cleared after a success operation of power up software trigger for PU type PGCs.

Address: 303A_0000h base + 14Ch offset + (4d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		A53_VPU_H1_PUP_STATUS	A53_VPU_G2_PUP_STATUS	A53_VPU_G1_PUP_STATUS	A53_DISPMIX_PUP_STATUS	A53_GPU_3D_PUP_STATUS	A53_VPUMIX_PUP_STATUS	A53_GPUMIX_PUP_STATUS	A53_GPU_2D_PUP_STATUS	A53_DDR1_PUP_STATUS	Reserved	A53_OTG2_PUP_STATUS	A53_OTG1_PUP_STATUS	A53_PCIE_PUP_STATUS	A53_MIPI_PUP_STATUS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_A53_PU_PGC_PUP_STATUS*n* field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13 A53_VPU_H1_PUP_STATUS	
12 A53_VPU_G2_PUP_STATUS	
11 A53_VPU_G1_PUP_STATUS	
10 A53_DISPMIX_PUP_STATUS	
9 A53_GPU_3D_PUP_STATUS	
8 A53_VPUMIX_PUP_STATUS	
7 A53_GPUMIX_PUP_STATUS	
6 A53_GPU_2D_PUP_STATUS	
5 A53_DDR1_PUP_STATUS	
4 -	This field is reserved.
3 A53_OTG2_PUP_STATUS	
2 A53_OTG1_PUP_STATUS	
1 A53_PCIE_PUP_STATUS	
0 A53_MIPI_PUP_STATUS	

5.2.10.43 M4 PU PGC software up trigger status register (GPC_M4_PU_PGC_PUP_STATUS*n*)

M4_PU_PGC_PUP_STATUSn (n = 0,1,2) are a read only register, represents the results for power up software trigger from M4 platform to PU type PGCs.

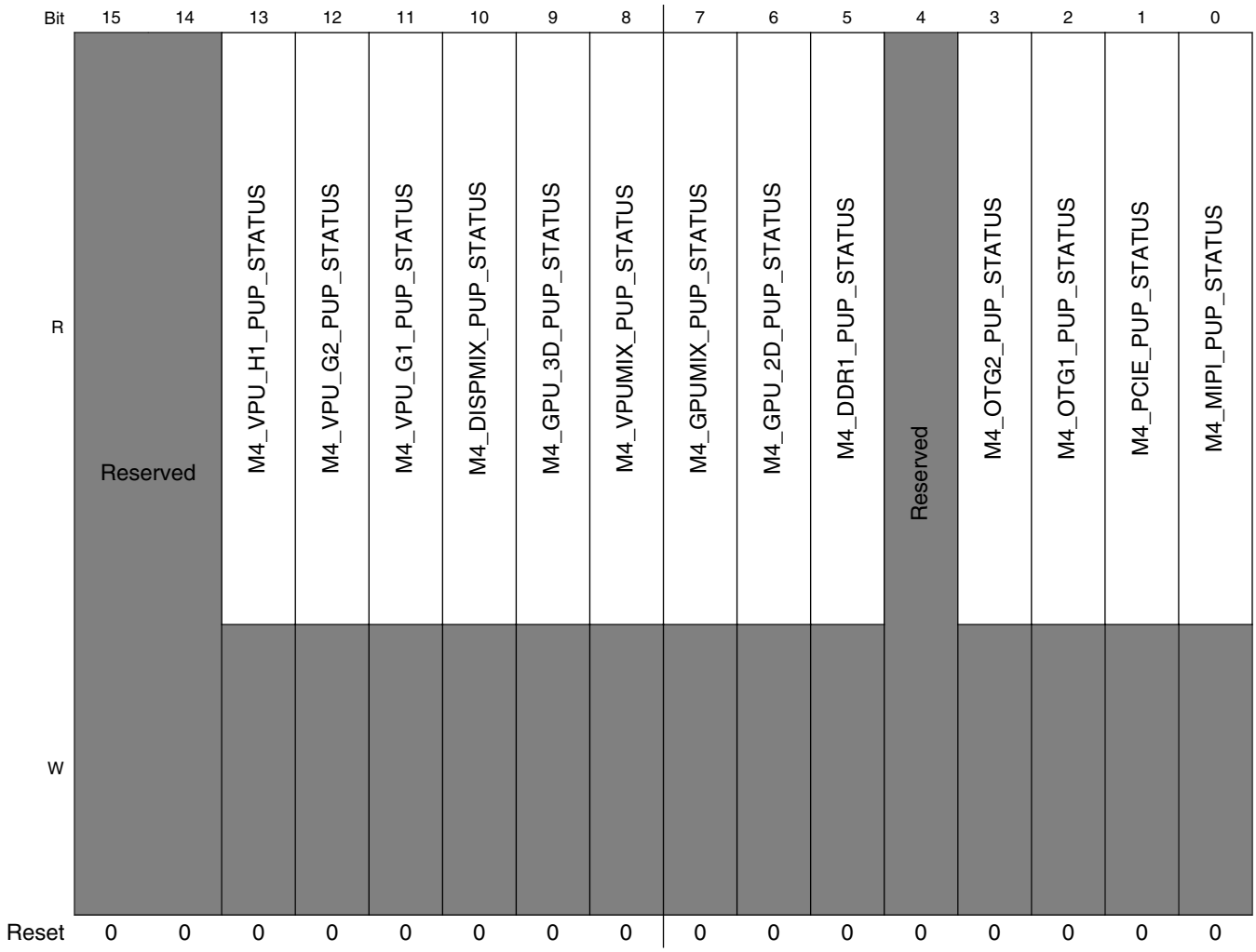
M4_PU_PGC_PUP_STATUS0: value of “1’b1” represent the software power up trigger failed because domain control condition. The relevant bit will be cleared after a success operation of power up software trigger for PU type PGCs.

M4_PU_PGC_PUP_STATUS1: value of “1’b1” represent the software power up trigger failed because the relevant PGC is in a power up process. The relevant bit will be cleared after a success operation of power up software trigger for PU type PGCs.

M4_PU_PGC_PUP_STATUS2: value of “1’b1” represent the software power up trigger failed because time slot control is busy. The relevant bit will be cleared after a success operation of power up software trigger for PU type PGCs.

Address: 303A_0000h base + 158h offset + (4d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



GPC_M4_PU_PGC_PUP_STATUS*n* field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13 M4_VPU_H1_PUP_STATUS	
12 M4_VPU_G2_PUP_STATUS	
11 M4_VPU_G1_PUP_STATUS	
10 M4_DISPMIX_PUP_STATUS	
9 M4_GPU_3D_PUP_STATUS	

Table continues on the next page...

GPC_M4_PU_PGC_PUP_STATUS*n* field descriptions (continued)

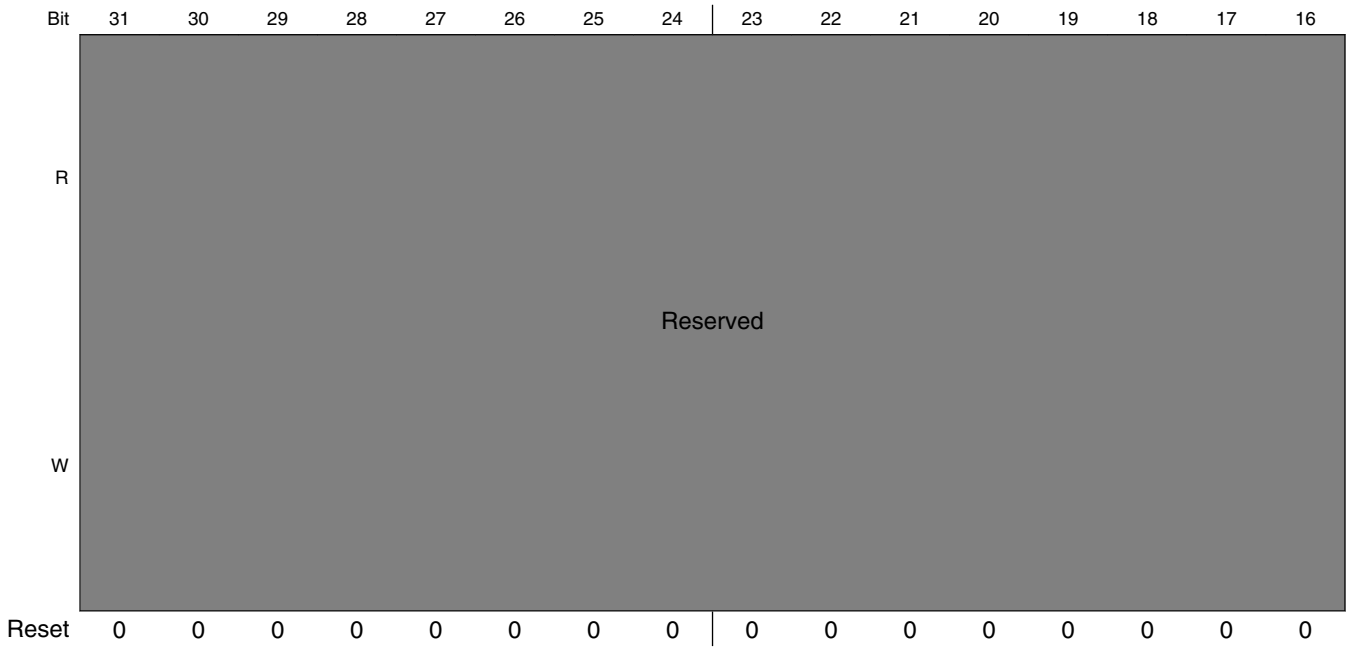
Field	Description
8 M4_VPUMIX_PUP_STATUS	
7 M4_GPUMIX_PUP_STATUS	
6 M4_GPU_2D_PUP_STATUS	
5 M4_DDR1_PUP_STATUS	
4 -	This field is reserved.
3 M4_OTG2_PUP_STATUS	
2 M4_OTG1_PUP_STATUS	
1 M4_PCIE_PUP_STATUS	
0 M4_MIPI_PUP_STATUS	

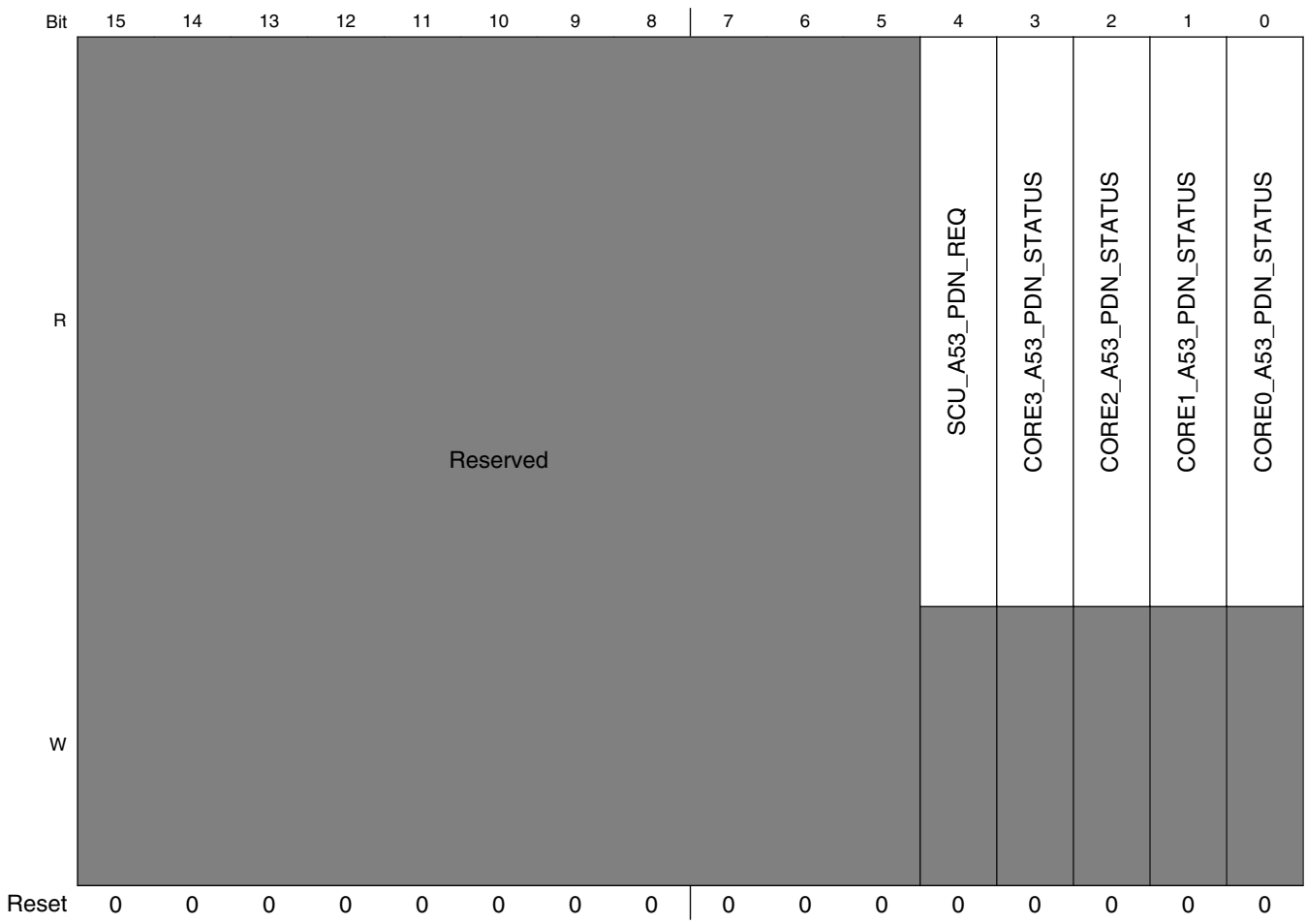
5.2.10.44 CPU PGC software dn trigger status1 (GPC_CPU_PGC_PDN_STATUS1)

CPU_PGC_PDN_STATUS1 is a read only register, represents the results for power DN software trigger for CPU type PGCs.

The field description is show in table below, the value of “1’b1” represent the software power DN trigger failed because the relevant PGC is in a power down process. The relevant bit will be cleared after a success operation of power DN software trigger for CPU type PGCs.

Address: 303A_0000h base + 170h offset = 303A_0170h





GPC_CPU_PGC_PDN_STATUS1 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SCU_A53_PDN_REQ	
3 CORE3_A53_PDN_STATUS	
2 CORE2_A53_PDN_STATUS	
1 CORE1_A53_PDN_STATUS	
0 CORE0_A53_PDN_STATUS	

5.2.10.45 A53 MIX software down trigger status register (GPC_A53_MIX_PGC_PDN_STATUS*n*)

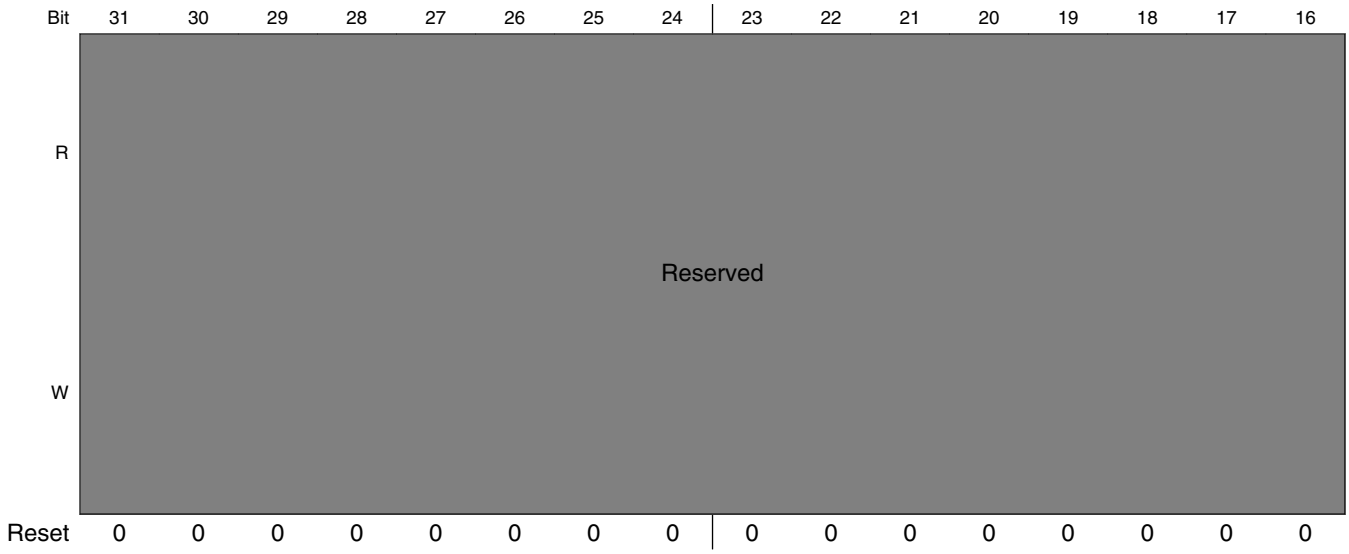
A53_MIX_PGC_PDN_STATUSn (n = 0,1,2) are a read only register, represents the results for power down software trigger from A53 platform to MIX type PGCs.

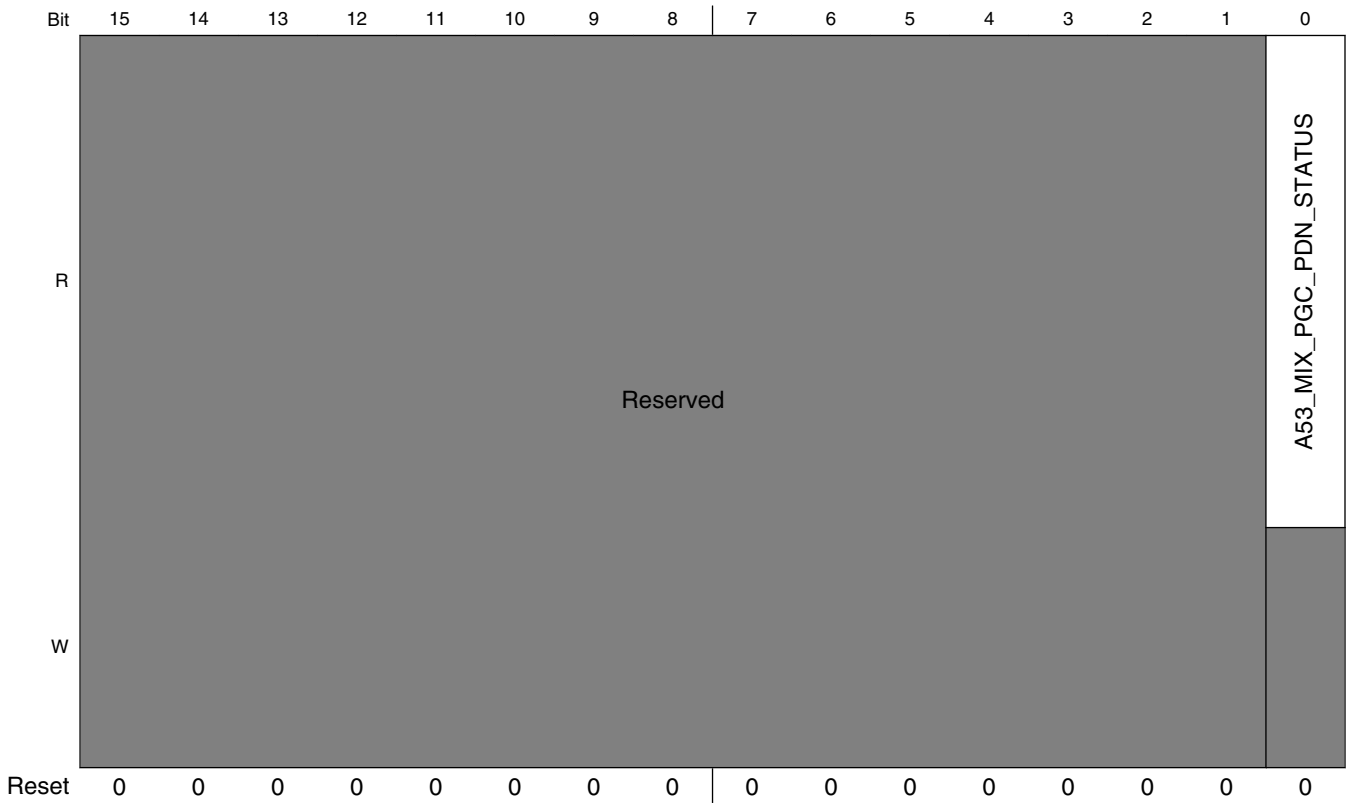
A53_MIX_PGC_PDN_STATUS0: value of “1’b1” represent the software power up trigger failed because domain control condition. The relevant bit will be cleared after a success operation of power down software trigger for MIX type PGCs.

A53_MIX_PGC_PDN_STATUS1: value of “1’b1” represent the software power up trigger failed because the relevant PGC is in a power up process. The relevant bit will be cleared after a success operation of power down software trigger for MIX type PGCs.

A53_MIX_PGC_PDN_STATUS2: value of “1’b1” represent the software power up trigger failed because time slot control is busy. The relevant bit will be cleared after a success operation of power down software trigger for MIX type PGCs.

Address: 303A_0000h base + 174h offset + (4d × i), where i=0d to 2d





GPC_A53_MIX_PGC_PDN_STATUS*n* field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 A53_MIX_PGC_PDN_STATUS	

5.2.10.46 M4 MIX PGC software power down trigger status register (GPC_M4_MIX_PGC_PDN_STATUS*n*)

M4_MIX_PGC_PDN_STATUSn (n = 0,1,2) are a read only register, represents the results for power down software trigger from M4 platform to MIX type PGCs.

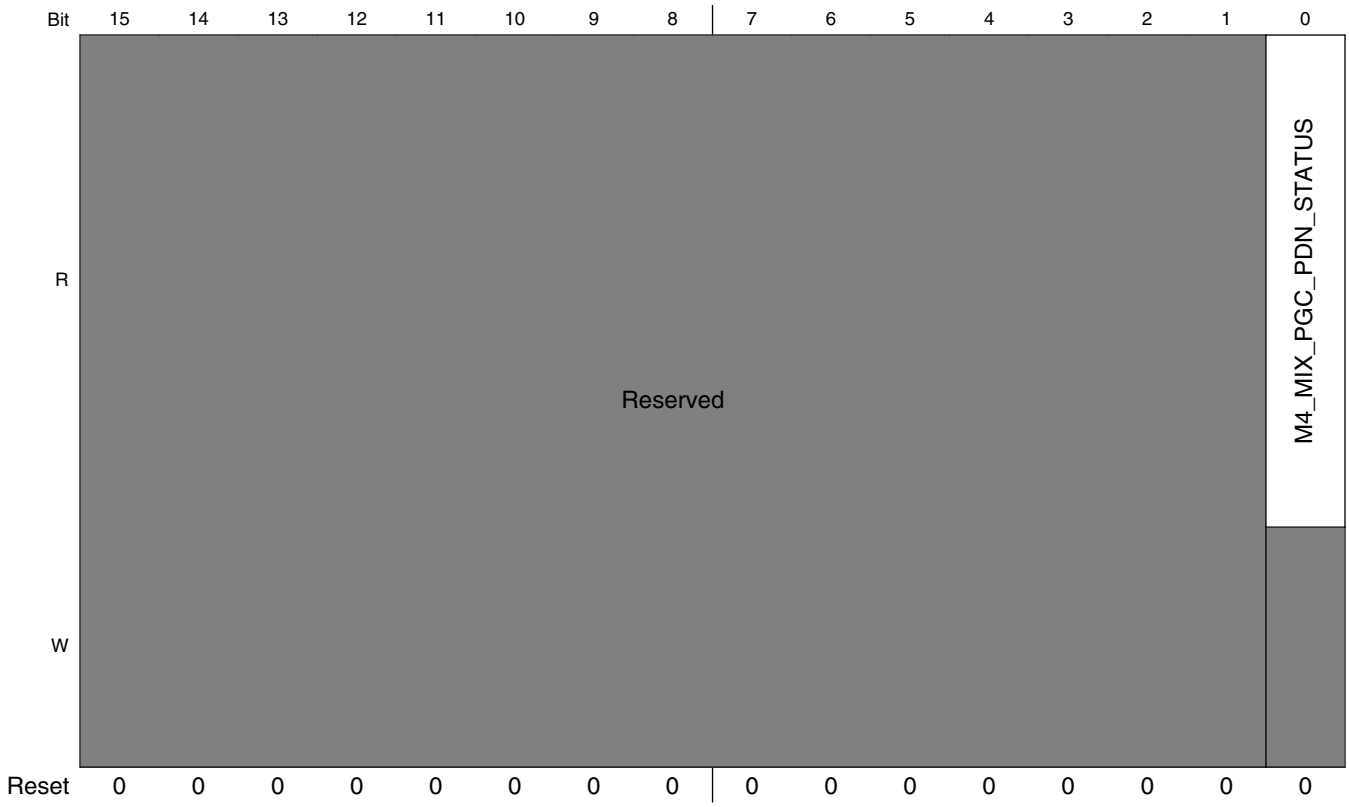
M4_MIX_PGC_PDN_STATUS0: value of “1’b1” represent the software power up trigger failed because domain control condition. The relevant bit will be cleared after a success operation of power down software trigger for MIX type PGCs.

M4_MIX_PGC_PDN_STATUS1: value of “1’b1” represent the software power up trigger failed because the relevant PGC is in a power up process. The relevant bit will be cleared after a success operation of power down software trigger for MIX type PGCs.

M4_MIX_PGC_PDN_STATUS2: value of “1’b1” represent the software power up trigger failed because time slot control is busy. The relevant bit will be cleared after a success operation of power down software trigger for MIX type PGCs.

Address: 303A_0000h base + 180h offset + (4d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



GPC_M4_MIX_PGC_PDN_STATUS*n* field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 M4_MIX_PGC_ PDN_STATUS	

5.2.10.47 A53 PU PGC software down trigger status (GPC_A53_PU_PGC_PDN_STATUS*n*)

A53_PU_PGC_PDN_STATUSn (n = 0,1,2) are a read only register, represents the results for power DN software trigger from A53 platform to PU type PGCs.

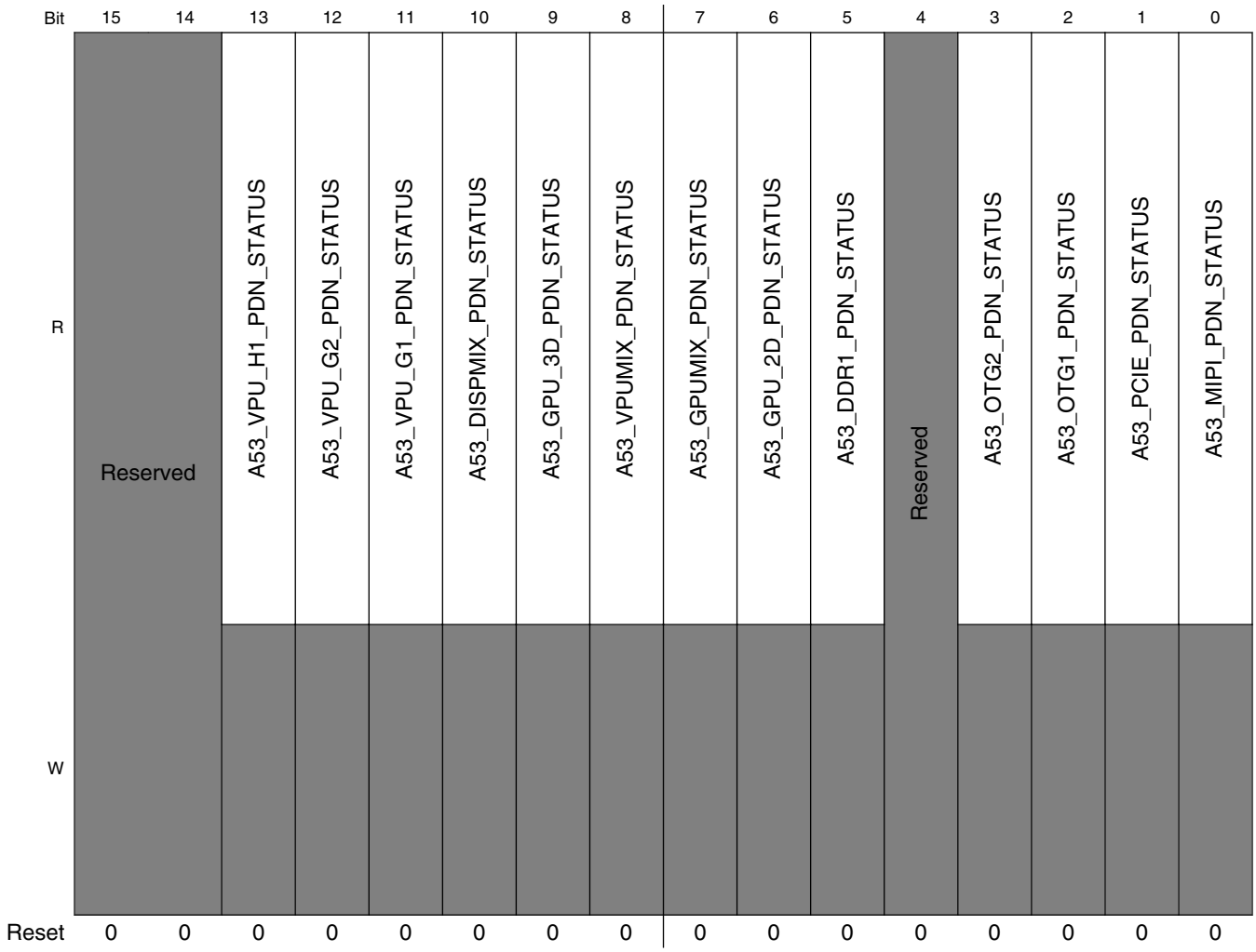
A53_PU_PGC_PDN_STATUS0: value of “1’b1” represent the software power DN trigger failed because domain control condition. The relevant bit will be cleared after a success operation of power DN software trigger for PU type PGCs.

A53_PU_PGC_PDN_STATUS1: value of “1’b1” represent the software power DN trigger failed because the relevant PGC is in a power DN process. The relevant bit will be cleared after a success operation of power DN software trigger for PU type PGCs.

A53_PU_PGC_PDN_STATUS2: value of “1’b1” represent the software power DN trigger failed because time slot control is busy. The relevant bit will be cleared after a success operation of power DN software trigger for PU type PGCs.

Address: 303A_0000h base + 18Ch offset + (4d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



GPC_A53_PU_PGC_PDN_STATUS*n* field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13 A53_VPU_H1_PDN_STATUS	
12 A53_VPU_G2_PDN_STATUS	
11 A53_VPU_G1_PDN_STATUS	
10 A53_DISPMIX_PDN_STATUS	
9 A53_GPU_3D_PDN_STATUS	

Table continues on the next page...

GPC_A53_PU_PGC_PDN_STATUS*n* field descriptions (continued)

Field	Description
8 A53_VPUMIX_PDN_STATUS	
7 A53_GPUMIX_PDN_STATUS	
6 A53_GPU_2D_PDN_STATUS	
5 A53_DDR1_PDN_STATUS	
4 -	This field is reserved.
3 A53_OTG2_PDN_STATUS	
2 A53_OTG1_PDN_STATUS	
1 A53_PCIE_PDN_STATUS	
0 A53_MIPI_PDN_STATUS	

5.2.10.48 M4 PU PGC software down trigger status (GPC_M4_PU_PGC_PDN_STATUS*n*)

M4_PU_PGC_PDN_STATUSn (n = 0,1,2) are a read only register, represents the results for power DN software trigger from M4 platform to PU type PGCs.

M4_PU_PGC_PDN_STATUS0: value of “1’b1” represent the software power DN trigger failed because domain control condition. The relevant bit will be cleared after a success operation of power DN software trigger for PU type PGCs.

M4_PU_PGC_PDN_STATUS1: value of “1’b1” represent the software power DN trigger failed because the relevant PGC is in a power DN process. The relevant bit will be cleared after a success operation of power DN software trigger for PU type PGCs.

M4_PU_PGC_PDN_STATUS2: value of “1’b1” represent the software power DN trigger failed because time slot control is busy. The relevant bit will be cleared after a success operation of power DN software trigger for PU type PGCs.

Address: 303A_0000h base + 198h offset + (4d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		M4_VPU_H1_PDN_STATUS	M4_VPU_G2_PDN_STATUS	M4_VPU_G1_PDN_STATUS	M4_DISPMIX_PDN_STATUS	M4_GPU_3D_PDN_STATUS	M4_VPUMIX_PDN_STATUS	M4_GPUMIX_PDN_STATUS	M4_GPU_2D_PDN_STATUS	M4_DDR1_PDN_STATUS	Reserved	M4_OTG2_PDN_STATUS	M4_OTG1_PDN_STATUS	M4_PCIE_PDN_STATUS	M4_MIPI_PDN_STATUS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

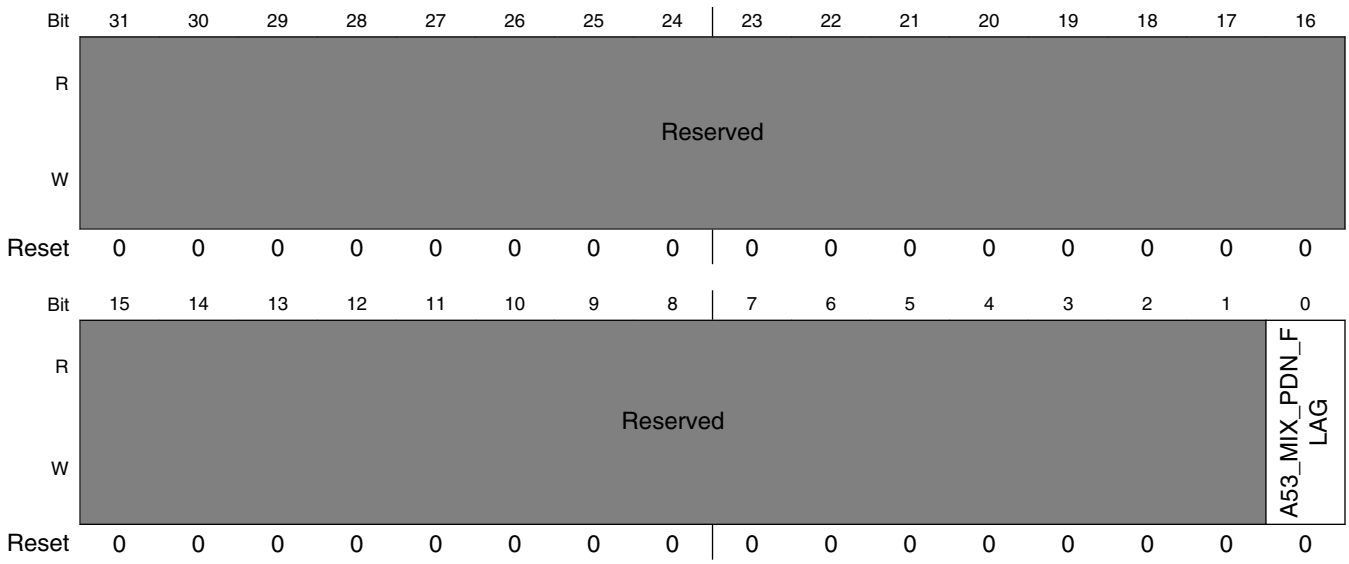
GPC_M4_PU_PGC_PDN_STATUS*n* field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13 M4_VPU_H1_PDN_STATUS	
12 M4_VPU_G2_PDN_STATUS	
11 M4_VPU_G1_PDN_STATUS	
10 M4_DISPMIX_PDN_STATUS	
9 M4_GPU_3D_PDN_STATUS	
8 M4_VPUMIX_PDN_STATUS	
7 M4_GPUMIX_PDN_STATUS	
6 M4_GPU_2D_PDN_STATUS	
5 M4_DDR1_PDN_STATUS	
4 -	This field is reserved.
3 M4_OTG2_PDN_STATUS	
2 M4_OTG1_PDN_STATUS	
1 M4_PCIE_PDN_STATUS	
0 M4_MIPI_PDN_STATUS	

5.2.10.49 A53 MIX PDN FLG (GPC_A53_MIX_PDN_FLG)

This is flag bit relevant domain control, represents A53 CPU platform wants to power down MIX PGC. The register can only be accessed by A53 platform.

Address: 303A_0000h base + 1B0h offset = 303A_01B0h



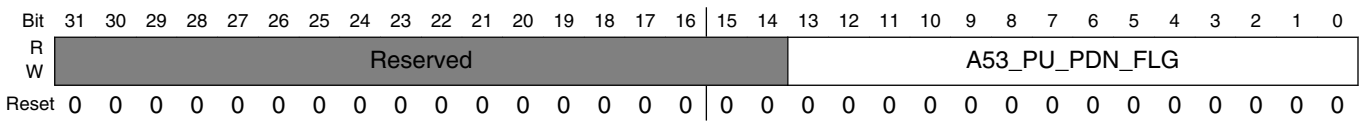
GPC_A53_MIX_PDN_FLG field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 A53_MIX_PDN_FLAG	A53 MIX power-down flag

5.2.10.50 A53 PU PDN FLG (GPC_A53_PU_PDN_FLG)

The register field is show in the table below. The 1'b1 represents A53 CPU platform wants to power down certain PU PGC. The register is a read only register. The register bits will be set when corresponding A53 software power down trigger happens and will be clear when corresponding A53 software power up trigger happens.

Address: 303A_0000h base + 1B4h offset = 303A_01B4h



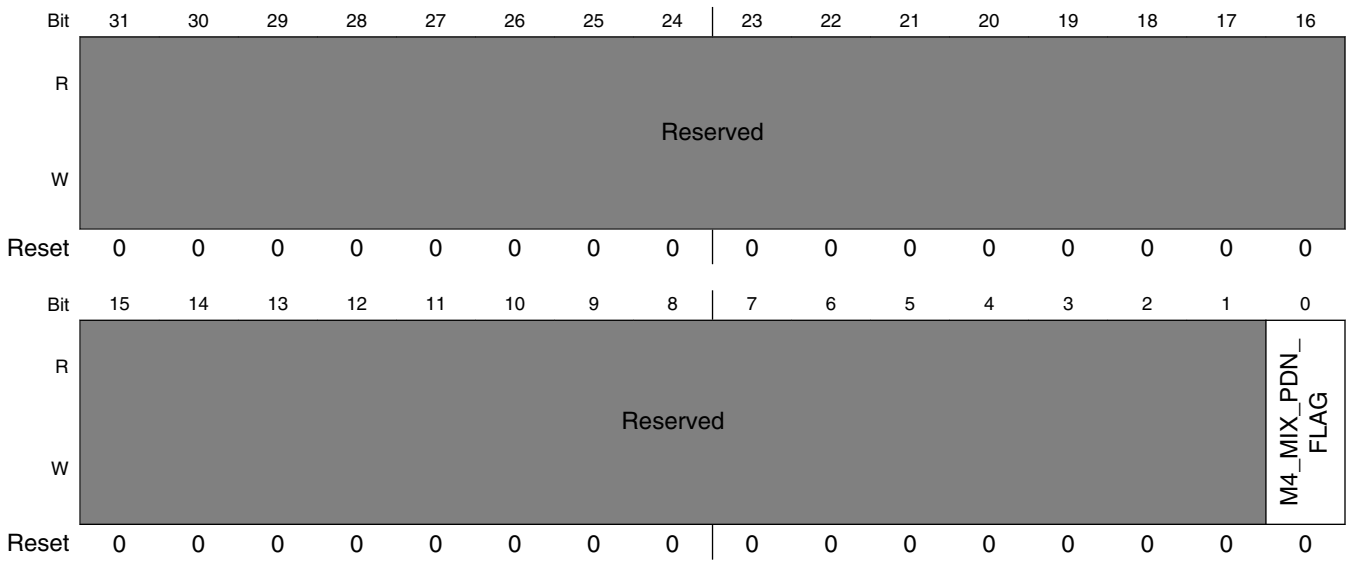
GPC_A53_PU_PDN_FLG field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
A53_PU_PDN_FLG	A53 PGC power-down flag

5.2.10.51 M4 MIX PDN FLG (GPC_M4_MIX_PDN_FLG)

This is flag bit relevant domain control, represents M4 CPU platform wants to power down MIX PGC. The register can only be accessed by M4 platform.

Address: 303A_0000h base + 1B8h offset = 303A_01B8h



GPC_M4_MIX_PDN_FLG field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 M4_MIX_PDN_FLAG	M4_MIX power-down flag

5.2.10.52 M4 PU PDN FLG (GPC_M4_PU_PDN_FLG)

The register field is show in the table below. The 1'b1 represents M4 CPU platform wants to power down certain PU PGC. The register is a read only register. The register bits will be set when corresponding M4 software power down trigger happens and will be clear when corresponding M4 software power up trigger happens.

Address: 303A_0000h base + 1BCh offset = 303A_01BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																M4_PU_PDN_FLG																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_M4_PU_PDN_FLG field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
M4_PU_PDN_FLG	M4 power-down flag

5.2.10.53 IRQ masking register 1 of A53 core2 (GPC_IMR1_CORE2_A53)

The four IMRn_CORE2_A53 (n = 1,2,3,4) registers are used as interrupt mask for A53 core2.

Address: 303A_0000h base + 1C0h offset = 303A_01C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR1_CORE2_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR1_CORE2_A53 field descriptions

Field	Description
IMR1_CORE2_A53	A53 core2 IRQ[31:0] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.54 IRQ masking register 2 of A53 core2 (GPC_IMR2_CORE2_A53)

Address: 303A_0000h base + 1C4h offset = 303A_01C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR2_CORE2_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR2_CORE2_A53 field descriptions

Field	Description
IMR2_CORE2_A53	A53 core2 IRQ[63:32] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.55 IRQ masking register 3 of A53 core2 (GPC_IMR3_CORE2_A53)

Address: 303A_0000h base + 1C8h offset = 303A_01C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR3_CORE2_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR3_CORE2_A53 field descriptions

Field	Description
IMR3_CORE2_A53	A53 core2 IRQ[95:64] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.56 IRQ masking register 4 of A53 core2 (GPC_IMR4_CORE2_A53)

Address: 303A_0000h base + 1CCh offset = 303A_01CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR4_CORE2_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR4_CORE2_A53 field descriptions

Field	Description
IMR4_CORE2_A53	A53 core2 IRQ[127:96] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.57 IRQ masking register 1 of A53 core3 (GPC_IMR1_CORE3_A53)

The four IMRn_CORE2_A53 (n = 1,2,3,4) registers are used as interrupt mask for A53 core3.

Address: 303A_0000h base + 1D0h offset = 303A_01D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR1_CORE3_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR1_CORE3_A53 field descriptions

Field	Description
IMR1_CORE3_A53	A53 core3 IRQ[31:0] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.58 IRQ masking register 2 of A53 core3 (GPC_IMR2_CORE3_A53)

Address: 303A_0000h base + 1D4h offset = 303A_01D4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR2_CORE3_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR2_CORE3_A53 field descriptions

Field	Description
IMR2_CORE3_A53	A53 core3 IRQ[63:32] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.59 IRQ masking register 3 of A53 core3 (GPC_IMR3_CORE3_A53)

Address: 303A_0000h base + 1D8h offset = 303A_01D8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR3_CORE3_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR3_CORE3_A53 field descriptions

Field	Description
IMR3_CORE3_A53	A53 core3 IRQ[95:64] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.60 IRQ masking register 4 of A53 core3 (GPC_IMR4_CORE3_A53)

Address: 303A_0000h base + 1DCh offset = 303A_01DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR4_CORE3_A53																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_IMR4_CORE3_A53 field descriptions

Field	Description
IMR4_CORE3_A53	A53 core3 IRQ[127:96] masking bits: 0 IRQ not masked 1 IRQ masked

5.2.10.61 PGC acknowledge signal selection of A53 platform for PUs (GPC_ACK_SEL_A53_PU)

Address: 303A_0000h base + 1E0h offset = 303A_01E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	VPU_H1_PGC_PUP_ACK	VPU_G2_PGC_PUP_ACK	VPU_G1_PGC_PUP_ACK	DISPMIX_PGC_PUP_ACK	GPU_3D_PGC_PUP_ACK	VPUMIX_PGC_PUP_ACK	GPUMIX_PGC_PUP_ACK	GPU_2D_PGC_PUP_ACK	DDR1_PGC_PUP_ACK	Reserved	USB_OTG2_PGC_PUP_ACK	USB_OTG1_PGC_PUP_ACK	PCIE_PGC_PUP_ACK	MIPI_PGC_PUP_ACK	Reserved	MF_PGC_PUP_ACK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VPUMIX_H1_PGC_PDN_ACK	VPU_G2_PGC_PDN_ACK	VPU_G1_PGC_PDN_ACK	DISPMIX_PGC_PDN_ACK	GPU_3D_PGC_PDN_ACK	VPUMIX_PGC_PDN_ACK	GPUMIX_PGC_PDN_ACK	GPU_2D_PGC_PDN_ACK	DDR1_PGC_PDN_ACK	Reserved	USB_OTG2_PGC_PDN_ACK	USB_OTG1_PGC_PDN_ACK	PCIE_PGC_PDN_ACK	MIPI_PGC_PDN_ACK	Reserved	MF_PGC_PDN_ACK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_ACK_SEL_A53_PU field descriptions

Field	Description
31 VPU_H1_PGC_PUP_ACK	Select power down acknowledge signal of VPU_H1 PGC as the power up acknowledge for A53 LPM.
30 VPU_G2_PGC_PUP_ACK	Select power down acknowledge signal of VPU_G2 PGC as the power up acknowledge for A53 LPM.
29 VPU_G1_PGC_PUP_ACK	Select power down acknowledge signal of VPU_G1 PGC as the power up acknowledge for A53 LPM.
28 DISPMIX_PGC_PUP_ACK	Select power down acknowledge signal of DISPMIX PGC as the power up acknowledge for A53 LPM.
27 GPU_3D_PGC_PUP_ACK	Select power down acknowledge signal of GPU_3D PGC as the power up acknowledge for A53 LPM.
26 VPUMIX_PGC_PUP_ACK	Select power down acknowledge signal of VPUMIX PGC as the power up acknowledge for A53 LPM.
25 GPUMIX_PGC_PUP_ACK	Select power down acknowledge signal of GPUMIX PGC as the power up acknowledge for A53 LPM.
24 GPU_2D_PGC_PUP_ACK	Select power down acknowledge signal of GPU_2D PGC as the power up acknowledge for A53 LPM.
23 DDR1_PGC_PUP_ACK	Select power down acknowledge signal of DDR1 PGC as the power up acknowledge for A53 LPM.
22 -	This field is reserved.
21 USB_OTG2_PGC_PUP_ACK	Select power down acknowledge signal of USB_OTG2 PGC as the power up acknowledge for A53 LPM.
20 USB_OTG1_PGC_PUP_ACK	Select power down acknowledge signal of USB_OTG1 PGC as the power up acknowledge for A53 LPM.

Table continues on the next page...

GPC_ACK_SEL_A53_PU field descriptions (continued)

Field	Description
19 PCIE_PGC_PUP_ACK	Select power down acknowledge signal of PCIE PGC as the power up acknowledge for A53 LPM.
18 MIPI_PGC_PUP_ACK	Select power down acknowledge signal of MIPI PGC as the power up acknowledge for A53 LPM.
17 -	This field is reserved.
16 MF_PGC_PUP_ACK	Select power down acknowledge signal of MIX PGC as the power up acknowledge for A53 LPM.
15 VPUMIX_H1_PGC_PDN_ACK	Select power down acknowledge signal of VPUMIX_H1 PGC as the power down acknowledge for A53 LPM.
14 VPU_G2_PGC_PDN_ACK	Select power down acknowledge signal of VPU_G2 PGC as the power down acknowledge for A53 LPM.
13 VPU_G1_PGC_PDN_ACK	Select power down acknowledge signal of VPU_G1 PGC as the power down acknowledge for A53 LPM.
12 DISPMIX_PGC_PDN_ACK	Select power down acknowledge signal of DISPMIX PGC as the power down acknowledge for A53 LPM.
11 GPU_3D_PGC_PDN_ACK	Select power down acknowledge signal of GPU_3D PGC as the power down acknowledge for A53 LPM.
10 VPUMIX_PGC_PDN_ACK	Select power down acknowledge signal of VPUMIX PGC as the power down acknowledge for A53 LPM.
9 GPUMIX_PGC_PDN_ACK	Select power down acknowledge signal of GPUMIX PGC as the power down acknowledge for A53 LPM.
8 GPU_2D_PGC_PDN_ACK	Select power down acknowledge signal of GPU_2D PGC as the power down acknowledge for A53 LPM.
7 DDR1_PGC_PDN_ACK	Select power down acknowledge signal of DDR1 PGC as the power down acknowledge for A53 LPM.
6 -	This field is reserved. Reserved
5 USB_OTG2_PGC_PDN_ACK	Select power down acknowledge signal of USB_OTG2 PGC as the power down acknowledge for A53 LPM.
4 USB_OTG1_PGC_PDN_ACK	Select power down acknowledge signal of USB_OTG1 PGC as the power down acknowledge for A53 LPM.
3 PCIE_PGC_PDN_ACK	Select power down acknowledge signal of PCIE PGC as the power down acknowledge for A53 LPM.

Table continues on the next page...

GPC_ACK_SEL_A53_PU field descriptions (continued)

Field	Description
2 MIPI_PGC_ PDN_ACK	Select power down acknowledge signal of MIPI PGC as the power down acknowledge for A53 LPM.
1 -	This field is reserved. Reserved
0 MF_PGC_PDN_ ACK	Select power down acknowledge signal of MIX PGC as the power down acknowledge for A53 LPM.

5.2.10.62 PGC acknowledge signal selection of M4 platform for PUs (GPC_ACK_SEL_M4_PU)

Address: 303A_0000h base + 1E4h offset = 303A_01E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	VPU_H1_PGC_PUP_ACK	VPU_G2_PGC_PUP_ACK	VPU_G1_PGC_PUP_ACK	DISPMIX_PGC_PUP_ACK	GPU_3D_PGC_PUP_ACK	VPUMIX_PGC_PUP_ACK	GPUMIX_PGC_PUP_ACK	GPU_2D_PGC_PUP_ACK	DDR1_PGC_PUP_ACK	Reserved	USB_OTG2_PGC_PUP_ACK	USB_OTG1_PGC_PUP_ACK	PCIE_PGC_PUP_ACK	MIPI_PGC_PUP_ACK	Reserved	MF_PGC_PUP_ACK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VPU_H1_PGC_PDN_ACK	VPU_G2_PGC_PDN_ACK	VPU_G1_PGC_PDN_ACK	DISPMIX_PGC_PDN_ACK	GPU_3D_PGC_PDN_ACK	VPUMIX_PGC_PDN_ACK	GPUMIX_PGC_PDN_ACK	GPU_2D_PGC_PDN_ACK	DDR1_PGC_PDN_ACK	Reserved	USB_OTG2_PGC_PDN_ACK	USB_OTG1_PGC_PDN_ACK	PCIE_PGC_PDN_ACK	MIPI_PGC_PDN_ACK	Reserved	MF_PGC_PDN_ACK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_ACK_SEL_M4_PU field descriptions

Field	Description
31 VPU_H1_PGC_P UP_ACK	Select power down acknowledge signal of VPU_H1 PGC as the power up acknowledge for M4 LPM.

Table continues on the next page...

GPC_ACK_SEL_M4_PU field descriptions (continued)

Field	Description
30 VPU_G2_PGC_PUP_ACK	Select power down acknowledge signal of VPU_G2 PGC as the power up acknowledge for M4 LPM.
29 VPU_G1_PGC_PUP_ACK	Select power down acknowledge signal of VPU_G1 PGC as the power up acknowledge for M4 LPM.
28 DISPMIX_PGC_PUP_ACK	Select power down acknowledge signal of DISPMIX PGC as the power up acknowledge for M4 LPM.
27 GPU_3D_PGC_PUP_ACK	Select power down acknowledge signal of GPU_3D PGC as the power up acknowledge for M4 LPM.
26 VPUMIX_PGC_PUP_ACK	Select power down acknowledge signal of VPUMIX PGC as the power up acknowledge for M4 LPM.
25 GPUMIX_PGC_PUP_ACK	Select power down acknowledge signal of GPUMIX PGC as the power up acknowledge for M4 LPM.
24 GPU_2D_PGC_PUP_ACK	Select power down acknowledge signal of GPU_2D PGC as the power up acknowledge for M4 LPM.
23 DDR1_PGC_PUP_ACK	Select power down acknowledge signal of DDR1 PGC as the power up acknowledge for M4 LPM.
22 -	This field is reserved.
21 USB_OTG2_PGC_PUP_ACK	Select power down acknowledge signal of USB_OTG2 PGC as the power up acknowledge for M4 LPM.
20 USB_OTG1_PGC_PUP_ACK	Select power down acknowledge signal of USB_OTG1 PGC as the power up acknowledge for M4 LPM.
19 PCIE_PGC_PUP_ACK	Select power down acknowledge signal of PCIE PGC as the power up acknowledge for M4 LPM.
18 MIPI_PGC_PUP_ACK	Select power down acknowledge signal of MIPI PGC as the power up acknowledge for M4 LPM.
17 -	This field is reserved.
16 MF_PGC_PUP_ACK	Select power down acknowledge signal of MIX PGC as the power up acknowledge for M4 LPM.
15 VPU_H1_PGC_PDN_ACK	Select power down acknowledge signal of VPU_H1 PGC as the power down acknowledge for M4 LPM.
14 VPU_G2_PGC_PDN_ACK	Select power down acknowledge signal of VPU_G2 PGC as the power down acknowledge for M4 LPM.

Table continues on the next page...

GPC_ACK_SEL_M4_PU field descriptions (continued)

Field	Description
13 VPU_G1_PGC_PDN_ACK	Select power down acknowledge signal of VPU_G1 PGC as the power down acknowledge for M4 LPM.
12 DISPMIX_PGC_PDN_ACK	Select power down acknowledge signal of DISPMIX PGC as the power down acknowledge for M4 LPM.
11 GPU_3D_PGC_PDN_ACK	Select power down acknowledge signal of GPU_3D PGC as the power down acknowledge for M4 LPM.
10 VPUMIX_PGC_PDN_ACK	Select power down acknowledge signal of VPUMIX PGC as the power down acknowledge for M4 LPM.
9 GPUMIX_PGC_PDN_ACK	Select power down acknowledge signal of GPUMIX PGC as the power down acknowledge for M4 LPM.
8 GPU_2D_PGC_PDN_ACK	Select power down acknowledge signal of GPU_2D PGC as the power down acknowledge for M4 LPM.
7 DDR1_PGC_PDN_ACK	Select power down acknowledge signal of DDR1 PGC as the power down acknowledge for M4 LPM.
6 -	This field is reserved.
5 USB_OTG2_PGC_PDN_ACK	Select power down acknowledge signal of USB_OTG2 PGC as the power down acknowledge for M4 LPM.
4 USB_OTG1_PGC_PDN_ACK	Select power down acknowledge signal of USB_OTG1 PGC as the power down acknowledge for M4 LPM.
3 PCIE_PGC_PDN_ACK	Select power down acknowledge signal of PCIE PGC as the power down acknowledge for M4 LPM.
2 MIPI_PGC_PDN_ACK	Select power down acknowledge signal of MIPI PGC as the power down acknowledge for M4 LPM.
1 -	This field is reserved.
0 MF_PGC_PDN_ACK	Select power down acknowledge signal of MIX PGC as the power down acknowledge for M4 LPM.

5.2.10.63 Slot configure register for A53 core (GPC_SLT*n*_CFG)

There are 20 slots in each SLTn_CFG(n = 0~19) will define the power up or power down behavior of one or more A53 CORE or SCU PGC in that each slot.

In each “SLTn_cfg”, 2 bits (slt_cfg[1:0])are reserved for each PGC:

- 2’b01 (slot controller will power down relevant PGC in corresponding slot if hardware power down request asserted)
- 2’b10 (slot controller will power up relevant PGC in corresponding slot if hardware power up request asserted)
- 2’b00 or 2’b11 (not power down or power up behavior in relevant slot)

The specific bits assignment for each PGC is shown in the table below.

	PGCx	PGCx-1	..	PGC2	PGC1	PGC0
SLT0_CFG	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]
SLT1_CFG	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]
:	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]
SLTn_CFG	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]	slt_cfg[1:0]

Address: 303A_0000h base + 1E8h offset + (4d × i), where i=0d to 4d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				NOC_PUP_SLOT_CONTROL	NOC_PDN_SLOT_CONTROL	SCU_PUP_SLOT_CONTROL	SCU_PDN_SLOT_CONTROL	CORE3_A53_PUP_SLOT_CONTROL	CORE3_A53_PDN_SLOT_CONTROL	CORE2_A53_PUP_SLOT_CONTROL	CORE2_A53_PDN_SLOT_CONTROL	CORE1_A53_PUP_SLOT_CONTROL	CORE1_A53_PDN_SLOT_CONTROL	CORE0_A53_PUP_SLOT_CONTROL	CORE0_A53_PDN_SLOT_CONTROL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_SLT*n*_CFG field descriptions

Field	Description
31–12 -	This field is reserved.
11 NOC_PUP_SLOT_CONTROL	NOC Power-up slot control

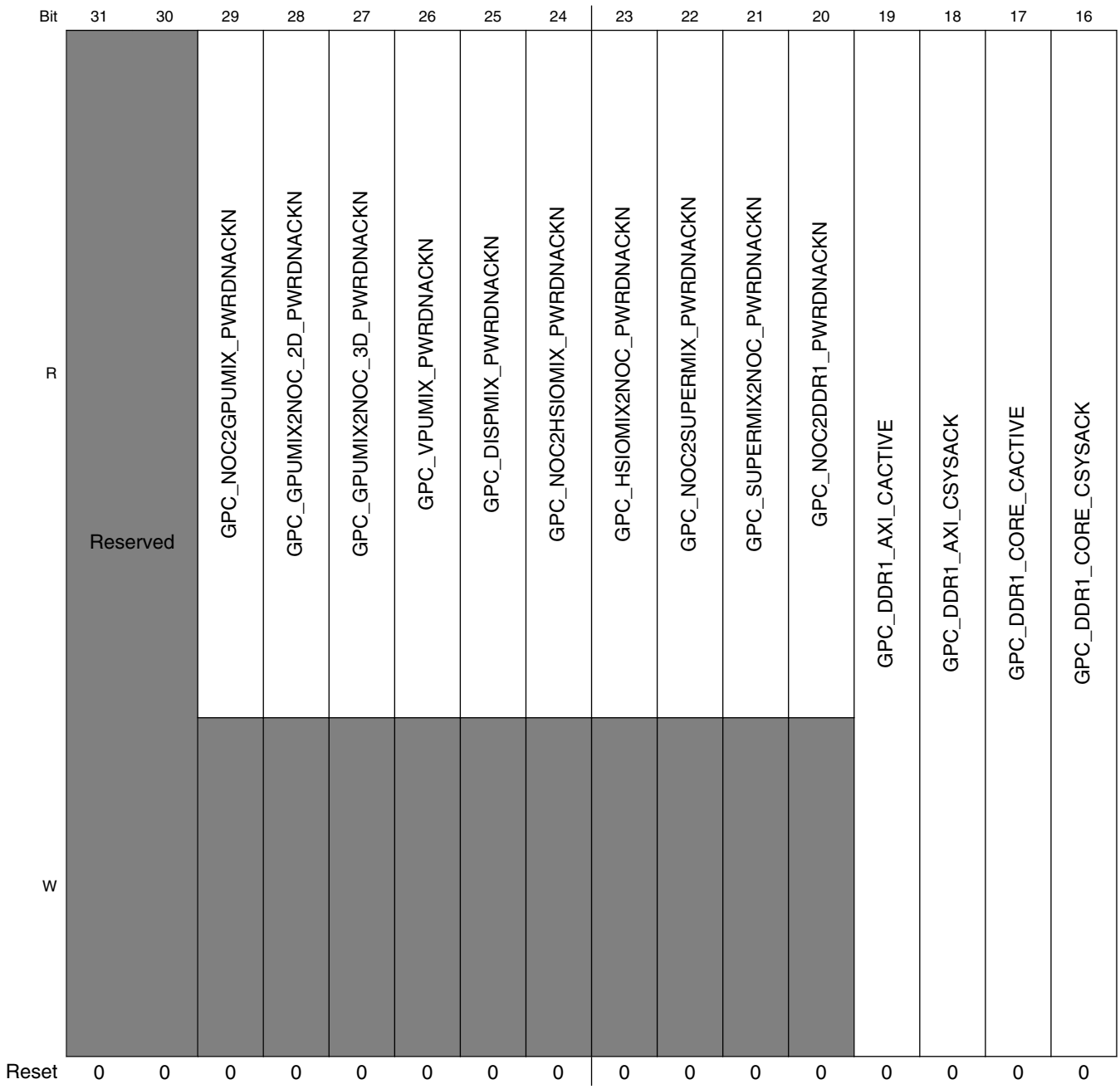
Table continues on the next page...

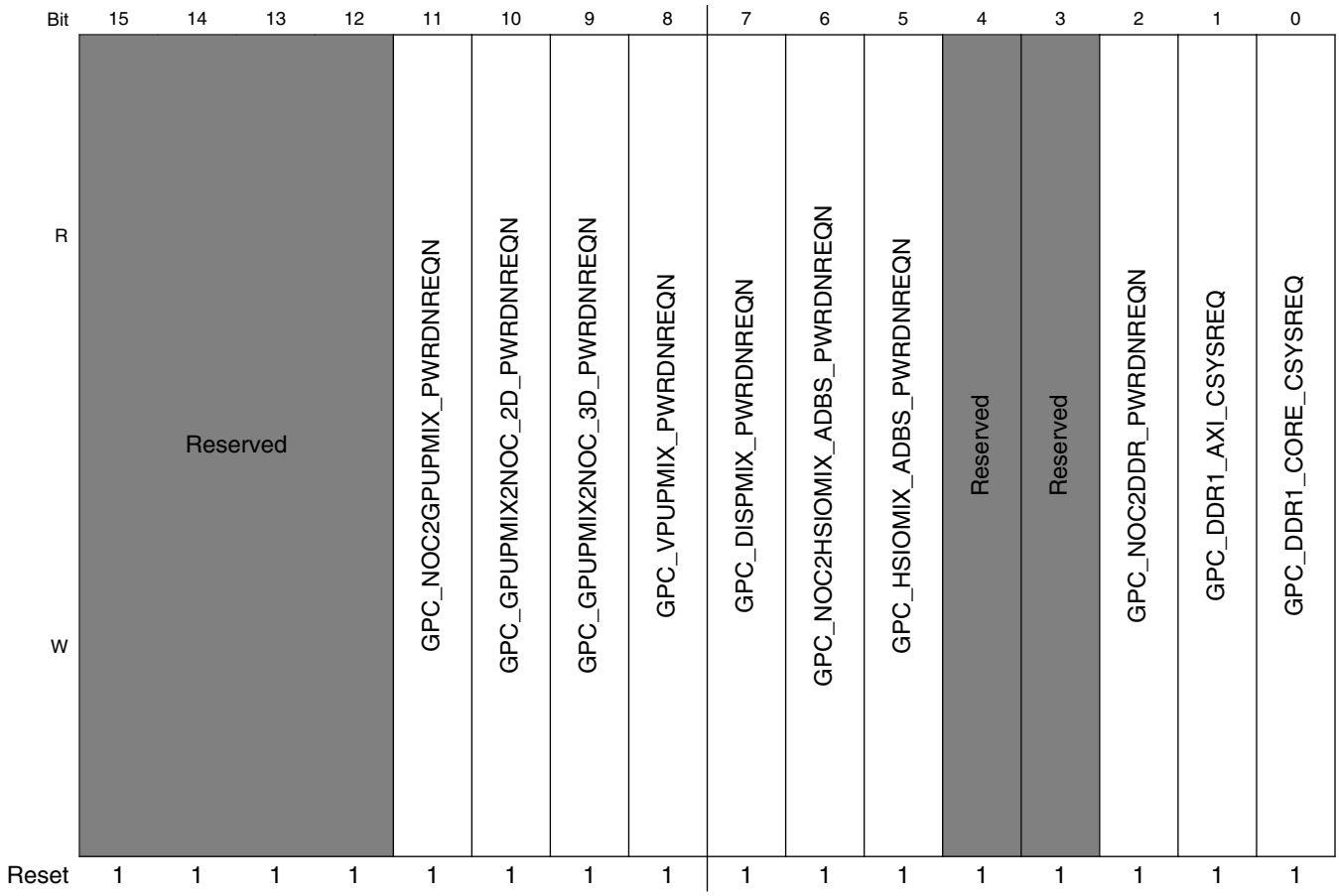
GPC_SLT*n*_CFG field descriptions (continued)

Field	Description
10 NOC_PDN_ SLOT_ CONTROL	NOC Power-down slot control
9 SCU_PUP_ SLOT_ CONTROL	SCU Power-up slot control
8 SCU_PDN_ SLOT_ CONTROL	SCU Power-down slot control
7 CORE3_A53_ PUP_SLOT_ CONTROL	CORE3 A53 Power-up slot control
6 CORE3_A53_ PDN_SLOT_ CONTROL	CORE3 A53 Power-down slot control
5 CORE2_A53_ PUP_SLOT_ CONTROL	CORE2 A53 Power-up slot control
4 CORE2_A53_ PDN_SLOT_ CONTROL	CORE2 A53 Power-down slot control
3 CORE1_A53_ PUP_SLOT_ CONTROL	CORE1 A53 Power-up slot control
2 CORE1_A53_ PDN_SLOT_ CONTROL	CORE1 A53 Power-down slot control
1 CORE0_A53_ PUP_SLOT_ CONTROL	CORE0 A53 Power-up slot control
0 CORE0_A53_ PDN_SLOT_ CONTROL	CORE0 A53 Power-down slot control

5.2.10.64 Power handshake register (GPC_PU_PWRHSK)

Address: 303A_0000h base + 1FCh offset = 303A_01FCh





GPC_PU_PWRHSK field descriptions

Field	Description
31–30 -	This field is reserved.
29 GPC_ NOC2GPUMIX_ PWRDNACKN	NOC2GPUMIX ADB400 power down ack. Active 0
28 GPC_ GPUMIX2NOC_ 2D_PWRDNACKN	GPUMIX2NOC(2D) ADB400 power down ack. Active 0
27 GPC_ GPUMIX2NOC_ 3D_PWRDNACKN	GPUMIX2NOC(3D) ADB400 power down ack. Active 0
26 GPC_VPUMIX_ PWRDNACKN	VPU ADB400 power down ack. Active 0
25 GPC_DISPMIX_ PWRDNACKN	DISPMIX ADB400 power down ack. Active 0

Table continues on the next page...

GPC_PU_PWRHSK field descriptions (continued)

Field	Description
24 GPC_ NOC2HSIOMIX_ PWRDNACKN	NOC2HSIOMIX ADB400 power down ack. Active 0
23 GPC_ HSIOMIX2NOC_ PWRDNACKN	HSIOMIX2NOC ADB400 power down ack.Active 0
22 GPC_ NOC2SUPERMIX_ PWRDNACKN	NOC2SUPERMIX ADB400 power down ack. Active 0
21 GPC_ SUPERMIX2NOC_ PWRDNACKN	SUPERMIX2NOC ADB400 power down ack. Active 0
20 GPC_NOC2DDR1_ PWRDNACKN	NOC2DDR ADB400 power down ack. Active 0
19 GPC_DDR1_AXI_ CACTIVE	DDR1 AXI Clock Active
18 GPC_DDR1_AXI_ CSYSACK	DDR1 AXI Low-Power Request ack
17 GPC_DDR1_ CORE_CACTIVE	DDR1 controller Hardware Low-Power Clock active
16 GPC_DDR1_ CORE_CSYSACK	DDR1 controller Hardware Low_Power ack
15–12 -	This field is reserved.
11 GPC_ NOC2GPUPMIX_ PWRDNREQN	NOC2GPUMIX ADB400 power down request. Active 0
10 GPC_ GPUPMIX2NOC_ 2D_PWRDNREQN	GPUMIX2NOC 2D ADB400 power down request. Active 0
9 GPC_ GPUPMIX2NOC_ 3D_PWRDNREQN	GPUMIX2NOC 3D ADB400 power down request. Active 0
8 GPC_VPUPMIX_ PWRDNREQN	VPUPMIX ADB400 power down request. Active 0

Table continues on the next page...

GPC_PU_PWRHSK field descriptions (continued)

Field	Description
7 GPC_DISPMIX_ PWRDNREQN	DISPMIX ADB400 power down request. Active 0
6 GPC_ NOC2HSIOMIX_ ADBS_ PWRDNREQN	NOC2HSIOMIX ADB400 power down request. Active 0
5 GPC_HSIOMIX_ ADBS_ PWRDNREQN	HSIOMIX2NOC ADB400 power down request. Active 0
4 -	This field is reserved.
3 -	This field is reserved.
2 GPC_NOC2DDR_ PWRDNREQN	NOC2DDR ADB400 power down request. Active 0
1 GPC_DDR1_AXI_ CSYSREQ	DDR1 AXI Low-Power Request
0 GPC_DDR1_ CORE_CSYSREQ	DDR1 controller Hardware Low-Power Request

5.2.10.65 Slot configure register for PUs (GPC_SLT*n*_CFG_PU)

Slot configure register for PUs (M4 dummy)

Address: 303A_0000h base + 200h offset + (4d × i), where i=0d to 19d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	VPU_H1_PUP_SLOT_CONTROL	VPU_H1_PDN_SLOT_CONTROL	VPU_G2_PUP_SLOT_CONTROL	VPU_G2_PDN_SLOT_CONTROL	VPU_G1_PUP_SLOT_CONTROL	VPU_G1_PDN_SLOT_CONTROL	DISPMIX_PUP_SLOT_CONTROL	DISPMIX_PDN_SLOT_CONTROL	GPU_3D_PUP_SLOT_CONTROL	GPU_3D_PDN_SLOT_CONTROL	VPUMIX_PUP_SLOT_CONTROL	VPUMIX_PDN_SLOT_CONTROL	GPUMIX_PUP_SLOT_CONTROL	GPUMIX_PDN_SLOT_CONTROL	GPU_2D_PUP_SLOT_CONTROL	GPU_2D_PDN_SLOT_CONTROL
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DDR1_PUP_SLOT_CONTROL	DDR1_PDN_SLOT_CONTROL	M4_PUP_SLOT_CONTROL	M4_PDN_SLOT_CONTROL	Reserved		OTG2_PUP_SLOT_CONTROL	OTG2_PDN_SLOT_CONTROL	OTG1_PUP_SLOT_CONTROL	OTG1_PDN_SLOT_CONTROL	PCIE_PUP_SLOT_CONTROL	PCIE_PDN_SLOT_CONTROL	MIPI_PUP_SLOT_CONTROL	MIPI_PDN_SLOT_CONTROL	MF_PUP_SLOT_CONTROL	MF_PDN_SLOT_CONTROL
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPC_SLT*n*_CFG_PU field descriptions

Field	Description
31 VPU_H1_PUP_SLOT_CONTROL	VPU_H1 Power-up slot control
30 VPU_H1_PDN_SLOT_CONTROL	VPU_H1 Power-down slot control
29 VPU_G2_PUP_SLOT_CONTROL	VPU_G2 Power-up slot control
28 VPU_G2_PDN_SLOT_CONTROL	VPU_G2 Power-down slot control
27 VPU_G1_PUP_SLOT_CONTROL	VPU_G1 Power-up slot control
26 VPU_G1_PDN_SLOT_CONTROL	VPU_G1 Power-down slot control

Table continues on the next page...

GPC_SLT*n*_CFG_PU field descriptions (continued)

Field	Description
25 DISPMIX_PUP_SLOT_CONTROL	DISPMIX Power-up slot control
24 DISPMIX_PDN_SLOT_CONTROL	DISPMIX Power-down slot control
23 GPU_3D_PUP_SLOT_CONTROL	GPU_3D Power-up slot control
22 GPU_3D_PDN_SLOT_CONTROL	GPU_3D Power-down slot control
21 VPUMIX_PUP_SLOT_CONTROL	VPUMIX Power-up slot control
20 VPUMIX_PDN_SLOT_CONTROL	VPUMIX Power-down slot control
19 GPUMIX_PUP_SLOT_CONTROL	GPUMIX Power-up slot control
18 GPUMIX_PDN_SLOT_CONTROL	GPUMIX Power-down slot control
17 GPU_2D_PUP_SLOT_CONTROL	GPU_2D Power-up slot control
16 GPU_2D_PDN_SLOT_CONTROL	GPU_2D Power-down slot control
15 DDR1_PUP_SLOT_CONTROL	DDR1 Power-up slot control
14 DDR1_PDN_SLOT_CONTROL	DDR1 Power-down slot control
13 M4_PUP_SLOT_CONTROL	M4 Power-up slot control
12 M4_PDN_SLOT_CONTROL	M4 Power-down slot control
11–10 -	This field is reserved.

Table continues on the next page...

GPC_SLT*n*_CFG_PU field descriptions (continued)

Field	Description
9 OTG2_PUP_ SLOT_ CONTROL	OTG2 Power-up slot control
8 OTG2_PDN_ SLOT_ CONTROL	OTG2 Power-down slot control
7 OTG1_PUP_ SLOT_ CONTROL	OTG1 Power-up slot control
6 OTG1_PDN_ SLOT_ CONTROL	OTG1 Power-down slot control
5 PCIE_PUP_ SLOT_ CONTROL	PCIE Power-up slot control
4 PCIE_PDN_ SLOT_ CONTROL	SCU Power-down slot control
3 MIPI_PUP_ SLOT_ CONTROL	MIPI Power-up slot control
2 MIPI_PDN_ SLOT_ CONTROL	MIPI Power-down slot control
1 MF_PUP_SLOT_ CONTROL	MF Power-up slot control
0 MF_PDN_SLOT_ CONTROL	MF Power-down slot control

5.2.11 GPC PGC Memory Map/Register Definition

There are 14 PGC inside GPCv2, with 4 different types: CPU/SCU/MIX/PU. PCIE/ MIPI/USB OTGx PGC belongs to PU type PGC. Each type PGC has 4 different control words PGC_CTRL,PGC_PUPSCR,PGC_PDNSCR and PGC_SR. Different types PGC may have different field definition in these four registers. There is another extra control word PGC_AUXSW for SCU type PGC.

The total GPC memory map is 4KB

Table 5-11. Memory Regions

Address Range(offset)	Region
0x000 - 0x3FF	GPC configuration register
0x400 - 0x7FF	Reserved
0x800 - 0x9FF	CPU and SCU type PGC register base address
0xA00 - 0xBFF	MIX type PGC register base address
0xC00 - 0xFFF	PU type PGC register base address

Each PGC (CPU type, MIX type, PU type) will occupy 64 Bytes address space, the specific base address of each PGC are listed as below.

- 0x800 ~ 0x83F: PGC for A53 core0
- 0x840 ~ 0x87F: PGC for A53 core1
- 0x880 ~ 0x8BF: PGC for A53 core2
- 0x8C0 ~ 0x8FF: PGC for A53 core3
- 0x900 ~ 0x93F: PGC for A53 SCU
- 0xA40 ~ 0xA7F: PGC for NOC mix
- 0xC00 ~ 0xC3F: PGC for MIPI PHY (PU0)
- 0xC40 ~ 0xC7F: PGC for PCIE1 PHY (PU1)
- 0xC80 ~ 0xCBF: USB_OTG1 (PU2)
- 0xCC0 ~ 0xCFF: USB_OTG2 (PU3)
- 0xD00 ~ 0xD3F: Reserved (PU4)
- 0xD40 ~ 0xD7F: DDR1 (PU5)
- 0xD80 ~ 0xDBF: GPU_2D (PU6)
- 0xDC0 ~ 0xDFF: GPUMIX (PU7)
- 0xE00 ~ 0xE3F: VPUMIX (PU8)
- 0xE40 ~ 0xE7F: GPU_3D (PU9)
- 0xE80 ~ 0xEBF: DISPMIX (PU10)
- 0xEC0 ~ 0xEFF: VPU_G1 (PU11)
- 0xF00 ~ 0xF3F: VPU_G2 (PU12)
- 0xF40 ~ 0xF7F: VPU_H1 (PU13)

GPC_PGC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303A_0800	GPC PGC Control Register (GPC_PGC_A53CORE0_CTRL)	32	R/W	0604_0202h	5.2.11.1/ 695
303A_0804	GPC PGC Up Sequence Control Register (GPC_PGC_A53CORE0_PUPSCR)	32	R/W	0009_97C1h	5.2.11.2/ 696
303A_0808	GPC PGC Down Sequence Control Register (GPC_PGC_A53CORE0_PDNSCR)	32	R/W	2100_0801h	5.2.11.3/ 697
303A_080C	GPC PGC Status Register (GPC_PGC_A53CORE0_SR)	32	R/W	0000_1000h	5.2.11.4/ 698
303A_0840	GPC PGC Control Register (GPC_PGC_A53CORE1_CTRL)	32	R/W	0604_0202h	5.2.11.1/ 695
303A_0844	GPC PGC Up Sequence Control Register (GPC_PGC_A53CORE1_PUPSCR)	32	R/W	0009_97C1h	5.2.11.2/ 696
303A_0848	GPC PGC Down Sequence Control Register (GPC_PGC_A53CORE1_PDNSCR)	32	R/W	2100_0801h	5.2.11.3/ 697
303A_084C	GPC PGC Status Register (GPC_PGC_A53CORE1_SR)	32	R/W	0000_1000h	5.2.11.4/ 698
303A_0880	GPC PGC Control Register (GPC_PGC_A53CORE2_CTRL)	32	R/W	0604_0202h	5.2.11.1/ 695
303A_0884	GPC PGC Up Sequence Control Register (GPC_PGC_A53CORE2_PUPSCR)	32	R/W	0009_97C1h	5.2.11.2/ 696
303A_0888	GPC PGC Down Sequence Control Register (GPC_PGC_A53CORE2_PDNSCR)	32	R/W	2100_0801h	5.2.11.3/ 697
303A_088C	GPC PGC Status Register (GPC_PGC_A53CORE2_SR)	32	R/W	0000_1000h	5.2.11.4/ 698
303A_08C0	GPC PGC Control Register (GPC_PGC_A53CORE3_CTRL)	32	R/W	0604_0202h	5.2.11.1/ 695
303A_08C4	GPC PGC Up Sequence Control Register (GPC_PGC_A53CORE3_PUPSCR)	32	R/W	0009_97C1h	5.2.11.2/ 696
303A_08C8	GPC PGC Down Sequence Control Register (GPC_PGC_A53CORE3_PDNSCR)	32	R/W	2100_0801h	5.2.11.3/ 697
303A_08CC	GPC PGC Status Register (GPC_PGC_A53CORE3_SR)	32	R/W	0000_1000h	5.2.11.4/ 698
303A_0900	GPC PGC Control Register (GPC_PGC_A53SCU_CTRL)	32	R/W	0604_0202h	5.2.11.1/ 695
303A_0904	GPC PGC Up Sequence Control Register (GPC_PGC_A53SCU_PUPSCR)	32	R/W	0009_97C1h	5.2.11.2/ 696
303A_0908	GPC PGC Down Sequence Control Register (GPC_PGC_A53SCU_PDNSCR)	32	R/W	2100_0801h	5.2.11.3/ 697
303A_090C	GPC PGC Status Register (GPC_PGC_A53SCU_SR)	32	R/W	0000_1000h	5.2.11.4/ 698
303A_0910	GPC PGC Auxiliary Power Switch Control Register (GPC_PGC_A53SCU_AUXSW)	32	R/W	0000_0131h	5.2.11.5/ 700
303A_0A00	GPC PGC Control Register (GPC_PGC_MF_MIX_CTRL)	32	R/W	0604_0202h	5.2.11.6/ 701

Table continues on the next page...

GPC_PGC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303A_0A04	GPC PGC Up Sequence Control Register (GPC_PGC_MF_MIX_PUPSCR)	32	R/W	0009_97C1h	5.2.11.7/ 703
303A_0A08	GPC PGC Down Sequence Control Register (GPC_PGC_MF_MIX_PDNSCR)	32	R/W	2100_0801h	5.2.11.8/ 704
303A_0A0C	GPC PGC Status Register (GPC_PGC_MF_MIX_SR)	32	R/W	0000_1000h	5.2.11.9/ 705
303A_0A40	GPC PGC Control Register (GPC_PGC_NOC_MIX_CTRL)	32	R/W	0604_0202h	5.2.11.6/ 701
303A_0A44	GPC PGC Up Sequence Control Register (GPC_PGC_NOC_MIX_PUPSCR)	32	R/W	0009_97C1h	5.2.11.7/ 703
303A_0A48	GPC PGC Down Sequence Control Register (GPC_PGC_NOC_MIX_PDNSCR)	32	R/W	2100_0801h	5.2.11.8/ 704
303A_0A4C	GPC PGC Status Register (GPC_PGC_NOC_MIX_SR)	32	R/W	0000_1000h	5.2.11.9/ 705
303A_0C00	GPC PGC Control Register (GPC_PGC_PU0_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0C04	GPC PGC Up Sequence Control Register (GPC_PGC_PU0_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0C08	GPC PGC Down Sequence Control Register (GPC_PGC_PU0_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0C0C	GPC PGC Status Register (GPC_PGC_PU0_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0C40	GPC PGC Control Register (GPC_PGC_PU1_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0C44	GPC PGC Up Sequence Control Register (GPC_PGC_PU1_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0C48	GPC PGC Down Sequence Control Register (GPC_PGC_PU1_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0C4C	GPC PGC Status Register (GPC_PGC_PU1_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0C80	GPC PGC Control Register (GPC_PGC_PU2_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0C84	GPC PGC Up Sequence Control Register (GPC_PGC_PU2_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0C88	GPC PGC Down Sequence Control Register (GPC_PGC_PU2_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0C8C	GPC PGC Status Register (GPC_PGC_PU2_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0CC0	GPC PGC Control Register (GPC_PGC_PU3_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0CC4	GPC PGC Up Sequence Control Register (GPC_PGC_PU3_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0CC8	GPC PGC Down Sequence Control Register (GPC_PGC_PU3_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710

Table continues on the next page...

GPC_PGC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303A_0CCC	GPC PGC Status Register (GPC_PGC_PU3_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0D00	GPC PGC Control Register (GPC_PGC_PU4_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0D04	GPC PGC Up Sequence Control Register (GPC_PGC_PU4_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0D08	GPC PGC Down Sequence Control Register (GPC_PGC_PU4_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0D0C	GPC PGC Status Register (GPC_PGC_PU4_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0D40	GPC PGC Control Register (GPC_PGC_PU5_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0D44	GPC PGC Up Sequence Control Register (GPC_PGC_PU5_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0D48	GPC PGC Down Sequence Control Register (GPC_PGC_PU5_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0D4C	GPC PGC Status Register (GPC_PGC_PU5_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0D80	GPC PGC Control Register (GPC_PGC_PU6_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0D84	GPC PGC Up Sequence Control Register (GPC_PGC_PU6_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0D88	GPC PGC Down Sequence Control Register (GPC_PGC_PU6_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0D8C	GPC PGC Status Register (GPC_PGC_PU6_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0DC0	GPC PGC Control Register (GPC_PGC_PU7_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0DC4	GPC PGC Up Sequence Control Register (GPC_PGC_PU7_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0DC8	GPC PGC Down Sequence Control Register (GPC_PGC_PU7_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0DCC	GPC PGC Status Register (GPC_PGC_PU7_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0E00	GPC PGC Control Register (GPC_PGC_PU8_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0E04	GPC PGC Up Sequence Control Register (GPC_PGC_PU8_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0E08	GPC PGC Down Sequence Control Register (GPC_PGC_PU8_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0E0C	GPC PGC Status Register (GPC_PGC_PU8_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0E40	GPC PGC Control Register (GPC_PGC_PU9_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708

Table continues on the next page...

GPC_PGC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
303A_0E44	GPC PGC Up Sequence Control Register (GPC_PGC_PU9_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0E48	GPC PGC Down Sequence Control Register (GPC_PGC_PU9_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0E4C	GPC PGC Status Register (GPC_PGC_PU9_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0E80	GPC PGC Control Register (GPC_PGC_PU10_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0E84	GPC PGC Up Sequence Control Register (GPC_PGC_PU10_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0E88	GPC PGC Down Sequence Control Register (GPC_PGC_PU10_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0E8C	GPC PGC Status Register (GPC_PGC_PU10_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0EC0	GPC PGC Control Register (GPC_PGC_PU11_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0EC4	GPC PGC Up Sequence Control Register (GPC_PGC_PU11_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0EC8	GPC PGC Down Sequence Control Register (GPC_PGC_PU11_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0ECC	GPC PGC Status Register (GPC_PGC_PU11_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0F00	GPC PGC Control Register (GPC_PGC_PU12_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0F04	GPC PGC Up Sequence Control Register (GPC_PGC_PU12_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0F08	GPC PGC Down Sequence Control Register (GPC_PGC_PU12_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0F0C	GPC PGC Status Register (GPC_PGC_PU12_SR)	32	R/W	0000_1000h	5.2.11.13/ 711
303A_0F40	GPC PGC Control Register (GPC_PGC_PU13_CTRL)	32	R/W	0604_0202h	5.2.11.10/ 708
303A_0F44	GPC PGC Up Sequence Control Register (GPC_PGC_PU13_PUPSCR)	32	R/W	0009_97C1h	5.2.11.11/ 709
303A_0F48	GPC PGC Down Sequence Control Register (GPC_PGC_PU13_PDNSCR)	32	R/W	2100_0801h	5.2.11.12/ 710
303A_0F4C	GPC PGC Status Register (GPC_PGC_PU13_SR)	32	R/W	0000_1000h	5.2.11.13/ 711

5.2.11.1 GPC PGC Control Register (GPC_PGC_*n*CTRL)

GPC PGC Control Register

Address: 303A_0000h base + 800h offset + (64d × i), where i=0d to 4d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		MEMPWR_TCD1_TDR_TRM						Reserved		L2RETN_TCD1_TDR					
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		DFTRAM_TCD1						Reserved	L2RSTDIS						PCR
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

GPC_PGC_*n*CTRL field descriptions

Field	Description
31–30 -	This field is reserved. Reserved
29–24 MEMPWR_ TCD1_TDR_ TRM	After scu pdn_req, count this value to assert A53 mempwr to 1'b1 **NOTE:** Can't be programmed to zero (This register control only for SCU Type PGC)
23–22 -	This field is reserved. Reserved
21–16 L2RETN_TCD1_ TDR	After scu pdn_req, count this value to assert A53 l2retn to 1'b0 **NOTE:** Can't be programmed to zero (This register control only for SCU Type PGC)
15–14 -	This field is reserved. Reserved
13–8 DFTRAM_TCD1	After scu pdn_req, count this value to assert A53 dftram to 1'b1

Table continues on the next page...

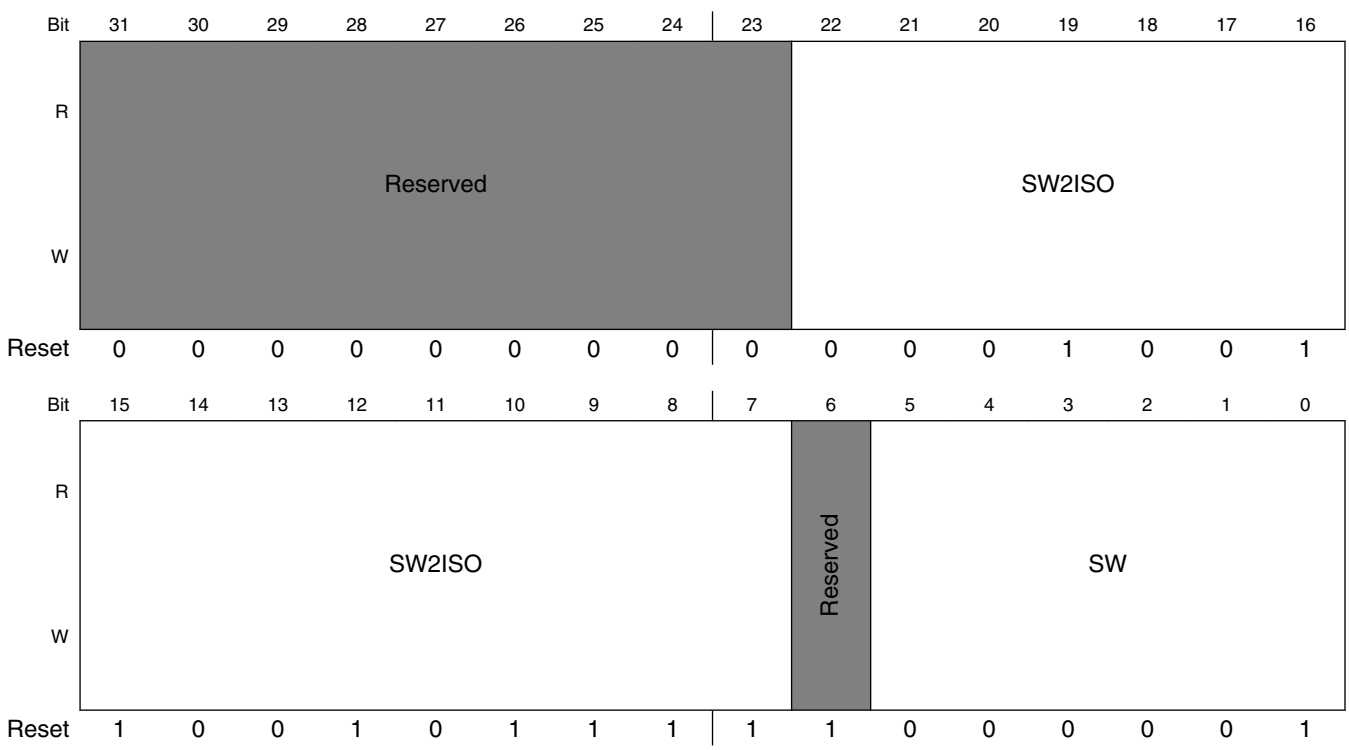
GPC_PGC_*n*CTRL field descriptions (continued)

Field	Description
	NOTE: Can't be programmed to zero (This register control only for SCU Type PGC)
7 -	This field is reserved. Reserved
6–1 L2RSTDIS	After scu pdn_req, count this value to assert A53 l2rstdis to 1'b1, it will be clear automatically once any of A53 core0/core1/core2/core3 is wakeup **NOTE:** Can't be programmed to zero (This register control only for SCU Type PGC)
0 PCR	Power Control **NOTE:** PCR must not change from power-down request (pdn_req) assertion until the target subsystem is completely powered up. 0 Do not switch off power even if pdn_req is asserted. 1 Switch off power when pdn_req is asserted.

5.2.11.2 GPC PGC Up Sequence Control Register (GPC_PGC_*n*PUPSCR)

GPC PGC Up Sequence Control Register

Address: 303A_0000h base + 804h offset + (64d × i), where i=0d to 4d



GPC_PGC_*n*PUPSCR field descriptions

Field	Description
31–23 -	This field is reserved. Reserved
22–7 SW2ISO	After asserting switch_b, the PGC waits a number of clocks equal to the value of SW2ISO before negating isolation.
6 -	This field is reserved. Reserved
SW	After a power-up request (pup_req assertion), the PGC waits a number of clocks equal to the value of SW before asserting switch_b **NOTE:** SW must not be programmed to zero.

5.2.11.3 GPC PGC Down Sequence Control Register (GPC_PGC_*n*PDNSCR)

GPC PGC Down Sequence Control Register

Address: 303A_0000h base + 808h offset + (64d × i), where i=0d to 4d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		ISO2SW						Reserved		ISO					
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1

GPC_PGC_*n*PDNSCR field descriptions

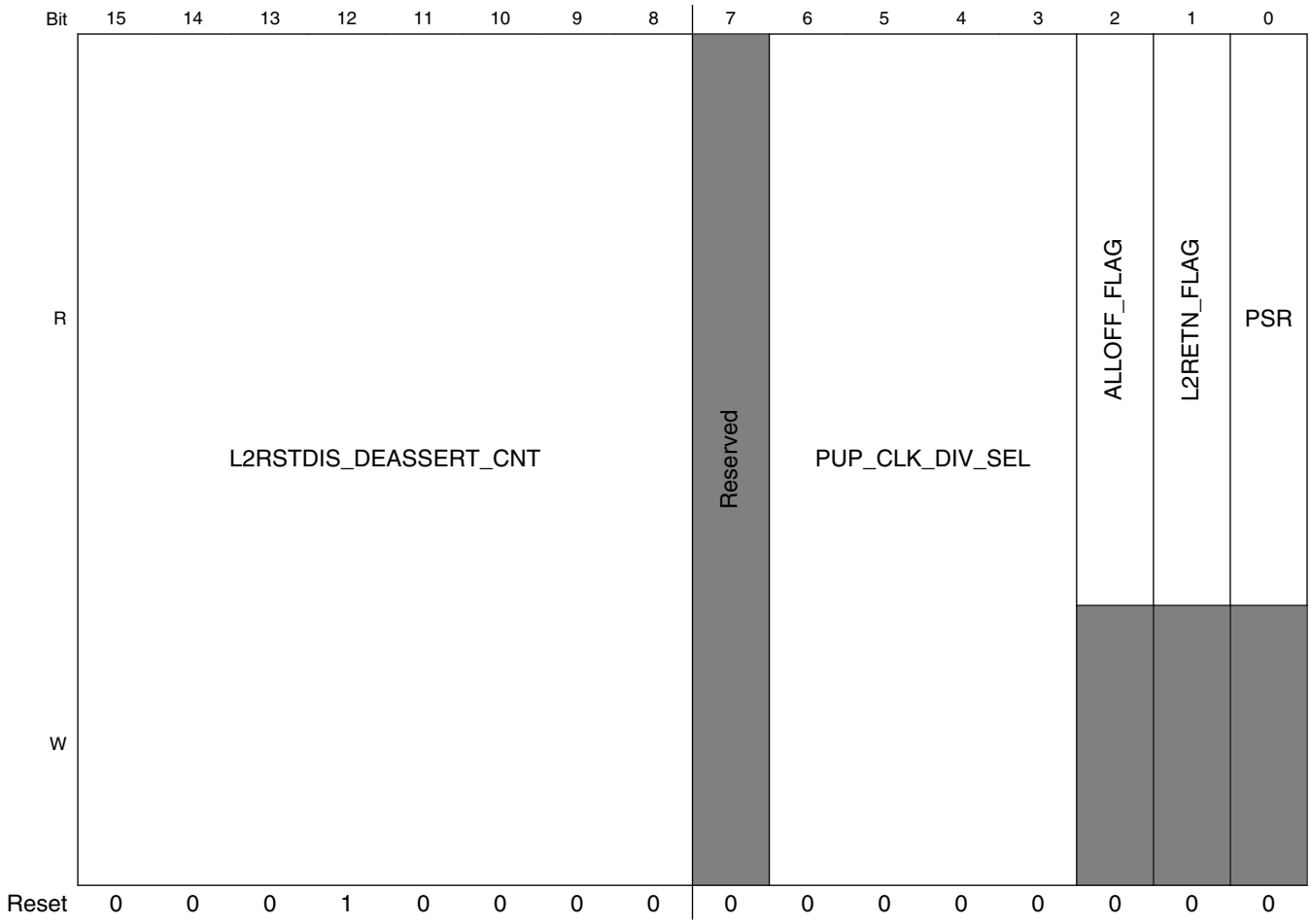
Field	Description
31–14 -	This field is reserved. Reserved
13–8 ISO2SW	After asserting isolation(by pdn_req assertion), the PGC waits a number of clocks equal to the value of ISO2SW before negating switch_b **NOTE:** ISO2SW must not be programmed to zero.
7–6 -	This field is reserved. Reserved
ISO	After a power-down request (pdn_req assertion), the PGC waits a number of clocks equal to the value of ISO before asserting isolation **NOTE:** ISO must not be programmed to zero.

5.2.11.4 GPC PGC Status Register (GPC_PGC_*n*SR)

GPC PGC Status Register

Address: 303A_0000h base + 80Ch offset + (64d × i), where i=0d to 4d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														L2RSTDIS_DEASSERT_CNT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



GPC_PGC_*n*SR field descriptions

Field	Description
31–18 -	This field is reserved. Reserved
17–8 L2RSTDIS_ DEASSERT_ CNT	Count this value to de-assert L2RSTDISABLE to LOW after CPU0 or CPU1 power up **NOTE:** This value can't be programmed to zero (This register control only for SCU Type PGC)
7 -	This field is reserved. Reserved
6–3 PUP_CLK_DIV_ SEL	Clock divider select for the clock of power up counter(count_clk is 32KHz for CPU/SCU type PGC, ipg_clk(66MHz) for MIX/PU Type PGC) 0000 1 0001 1/2 count_clk 0010 1/4 count_clk 0011 1/8 count_clk 0100 1/16 count_clk 0101 1/32 count_clk 0110 1/64 count_clk 0111 1/128 count_clk

Table continues on the next page...

GPC_PGC_*n*SR field descriptions (continued)

Field	Description
	1000 1/256 count_clk 1001 1/512 count_clk 1010 1/1024 count_clk 1011 1/2056 count_clk 1100 1/4096 count_clk 1101 1/8192 count_clk 1110 1/16384 count_clk 1111 1/32768 count_clk
2 ALLOFF_FLAG	All-off flag. **NOTE:** Software should write “1” to clear this flag after A53 is wakeup from ALL_OFF mode, otherwise, it will always keep to 1 (This register control only for SCU Type PGC) 0 A53 is not wakeup from ALL_OFF mode. 1 A53 is wakeup from ALL_OFF mode.
1 L2RETN_FLAG	L2 Retention Flag **NOTE:** Software should write “1” to clear this flag after A53 is wakeup from L2 retention mode, otherwise it will always keep to 1 (This register control only for SCU Type PGC) 0 A53 is not wakeup from L2 retention mode. 1 A53 is wakeup from L2 retention mode.
0 PSR	Power status. When in functional (or software-controlled debug) mode, PGC hardware sets PSR as soon as any of the power control output changes its state to one. Write one to clear this bit. Software should clear this bit after power up; otherwise, PSR continues to reflect the power status of the initial power down. 0 The target subsystem was not powered down for the previous power-down request. 1 The target subsystem was powered down for the previous power-down request.

5.2.11.5 GPC PGC Auxiliary Power Switch Control Register (GPC_PGC_A53SCU_AUXSW)

GPC PGC Auxiliary Power Switch Control Register.

Address: 303A_0000h base + 910h offset = 303A_0910h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		MEMPWR_TRC1_TMC										L2RETN_RTC1_TMC_TMR			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L2RETN_RTC1_TMC_TMR						DFTRAM_TRC1_TMC_TMR_TCD2									
Reset	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	1

GPC_PGC_A53SCU_AUXSW field descriptions

Field	Description
31–30 -	This field is reserved. Reserved
29–20 MEMPWR_ TRC1_TMC	After scu starts pup reset, count this value to assert a53 l2 memory switch to 1’b0.
19–10 L2RETN_RTC1_ TMC_TMR	After scu starts pup reset, count this value to assert a53 l2 memory retention to 1’b1.
DFTRAM_TRC1_ TMC_TMR_ TCD2	After scu starts pup reset, count this value to assert a53 l2 memory dftram to 1’b0.

5.2.11.6 GPC PGC Control Register (GPC_PGC_*n*MIX_CTRL)

GPC PGC Control Register

Address: 303A_0000h base + A00h offset + (64d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		MEMPWR_TCD1_TDR_TRM						Reserved		L2RETN_TCD1_TDR					
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		DFTRAM_TCD1						Reserved	L2RSTDIS						MIX_ PCR
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

GPC_PGC_*n*MIX_CTRL field descriptions

Field	Description
31–30 -	This field is reserved. Reserved
29–24 MEMPWR_ TCD1_TDR_ TRM	After scu pdn_req, count this value to assert A53 mempwr to 1'b1 **NOTE:** Can't be programmed to zero (This register control only for SCU Type PGC)
23–22 -	This field is reserved. Reserved
21–16 L2RETN_TCD1_ TDR	After scu pdn_req, count this value to assert A53 l2retn to 1'b0 **NOTE:** Can't be programmed to zero (This register control only for SCU Type PGC)
15–14 -	This field is reserved. Reserved
13–8 DFTRAM_TCD1	After scu pdn_req, count this value to assert A53 dftram to 1'b1 **NOTE:** Can't be programmed to zero (This register control only for SCU Type PGC)
7 -	This field is reserved. Reserved
6–1 L2RSTDIS	After scu pdn_req, count this value to assert A53 l2rstdis to 1'b1, it will be clear automatically once any of A53 core0/core1/core2/core3 is wakeup **NOTE:** Can't be programmed to zero (This register control only for SCU Type PGC)
0 MIX_PCR	Power Control **NOTE:** PCR must not change from power-down request (pdn_req) assertion until the target subsystem is completely powered up. 0 Do not switch off power even if pdn_req is asserted. 1 Switch off power when pdn_req is asserted.

5.2.11.7 GPC PGC Up Sequence Control Register (GPC_PGC_*n*MIX_PUPSCR)

GPC PGC Up Sequence Control Register

Address: 303A_0000h base + A04h offset + (64d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved									SW2ISO						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW2ISO									PUP_WAIT_SCALL_OUT	Reserved					
Reset	1	0	0	1	0	1	1	1	1	1	0	0	0	0	0	1

GPC_PGC_*n*MIX_PUPSCR field descriptions

Field	Description
31–23 -	This field is reserved. Reserved
22–7 SW2ISO	After asserting switch_b, the PGC waits a number of clocks equal to the value of SW2ISO before negating isolation.
6 PUP_WAIT_SCALL_OUT	After SCALL asserting to 1'b0, wait handshake signal SCALL_OUT to return to 1'b0 (This register control only for MIX Type PGC)
-	This field is reserved. Reserved

5.2.11.8 GPC PGC Down Sequence Control Register (GPC_PGC_*n*MIX_PDNSCR)

GPC PGC Down Sequence Control Register

Address: 303A_0000h base + A08h offset + (64d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		ISO2SW						Reserved		ISO					
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1

GPC_PGC_*n*MIX_PDNSCR field descriptions

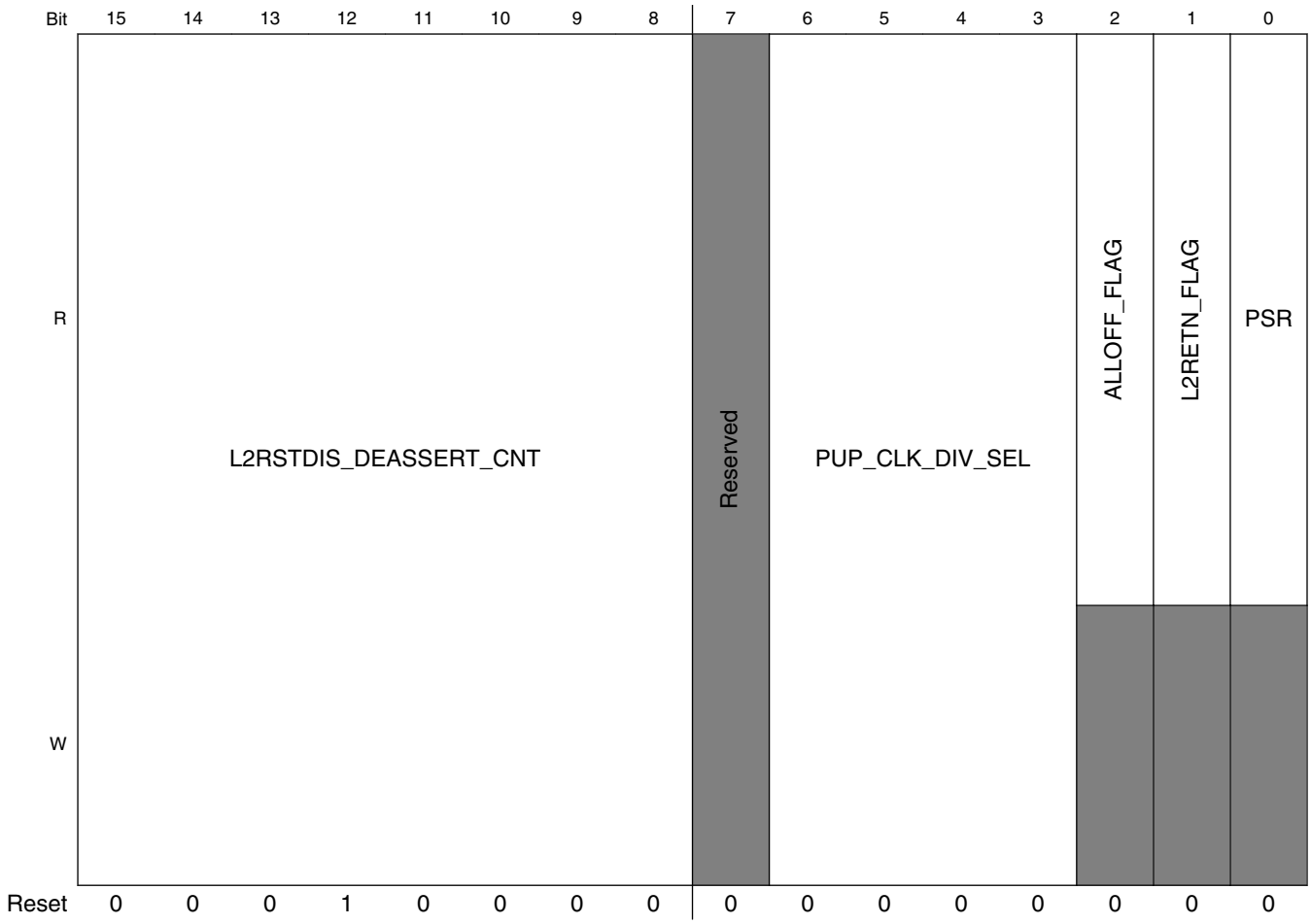
Field	Description
31–14 -	This field is reserved. Reserved
13–8 ISO2SW	After asserting isolation(by pdn_req assertion), the PGC waits a number of clocks equal to the value of ISO2SW before negating switch_b **NOTE:** ISO2SW must not be programmed to zero.
7–6 -	This field is reserved. Reserved
ISO	After a power-down request (pdn_req assertion), the PGC waits a number of clocks equal to the value of ISO before asserting isolation **NOTE:** ISO must not be programmed to zero.

5.2.11.9 GPC PGC Status Register (GPC_PGC_*n*MIX_SR)

GPC PGC Status Register

Address: 303A_0000h base + A0Ch offset + (64d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														L2RSTDIS_DEASSERT_CNT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



GPC_PGC_*n*MIX_SR field descriptions

Field	Description
31–18 -	This field is reserved. Reserved
17–8 L2RSTDIS_ DEASSERT_ CNT	Count this value to de-assert L2RSTDISABLE to LOW after CPU0 or CPU1 power up **NOTE:** This value can't be programmed to zero (This register control only for SCU Type PGC)
7 -	This field is reserved. Reserved
6–3 PUP_CLK_DIV_ SEL	Clock divider select for the clock of power up counter(count_clk is 32KHz for CPU/SCU type PGC, ipg_clk(66MHz) for MIX/PU Type PGC) 0000 1 0001 1/2 count_clk 0010 1/4 count_clk 0011 1/8 count_clk 0100 1/16 count_clk 0101 1/32 count_clk 0110 1/64 count_clk 0111 1/128 count_clk

Table continues on the next page...

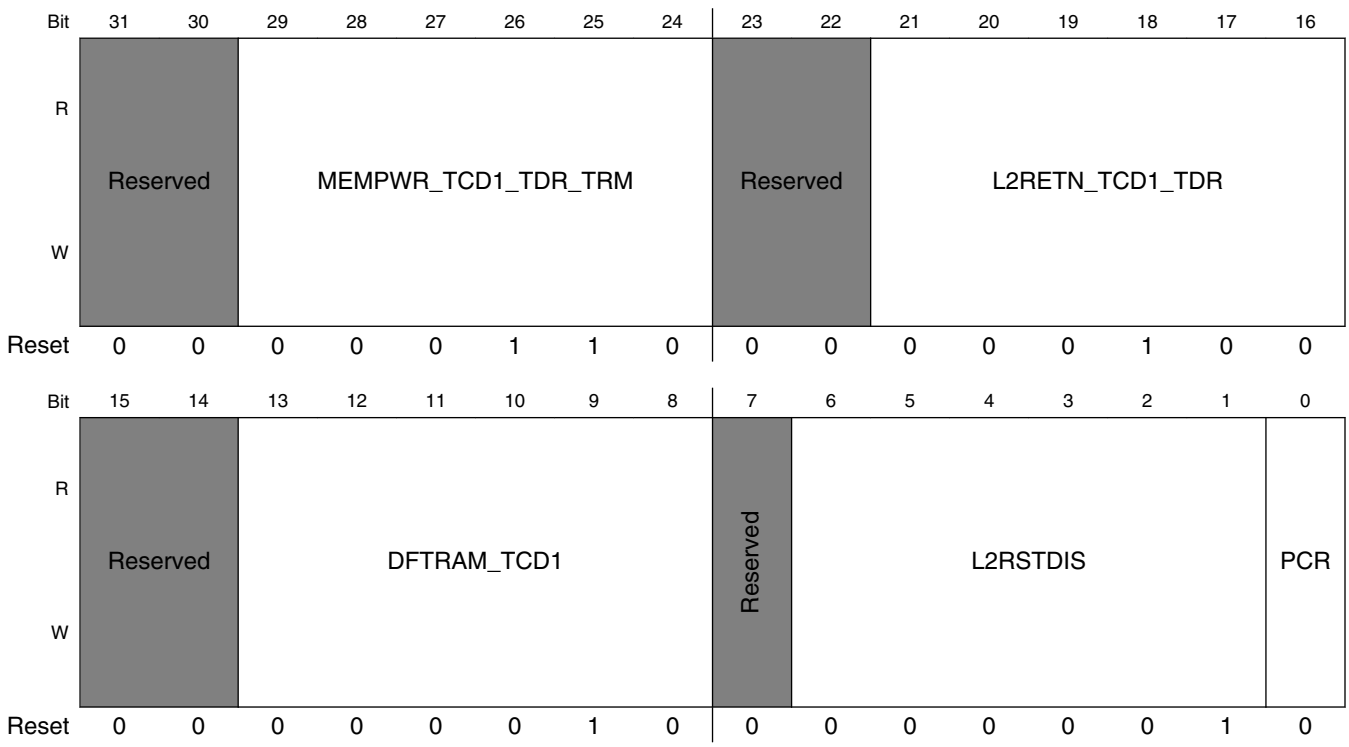
GPC_PGC_*n*MIX_SR field descriptions (continued)

Field	Description
	1000 1/256 count_clk 1001 1/512 count_clk 1010 1/1024 count_clk 1011 1/2056 count_clk 1100 1/4096 count_clk 1101 1/8192 count_clk 1110 1/16384 count_clk 1111 1/32768 count_clk
2 ALLOFF_FLAG	All-off flag. **NOTE:** Software should write “1” to clear this flag after A53 is wakeup from ALL_OFF mode, otherwise, it will always keep to 1 (This register control only for SCU Type PGC) 0 A53 is not wakeup from ALL_OFF mode. 1 A53 is wakeup from ALL_OFF mode.
1 L2RETN_FLAG	L2 Retention Flag **NOTE:** Software should write “1” to clear this flag after A53 is wakeup from L2 retention mode, otherwise it will always keep to 1 (This register control only for SCU Type PGC) 0 A53 is not wakeup from L2 retention mode. 1 A53 is wakeup from L2 retention mode.
0 PSR	Power status. When in functional (or software-controlled debug) mode, PGC hardware sets PSR as soon as any of the power control output changes its state to one. Write one to clear this bit. Software should clear this bit after power up; otherwise, PSR continues to reflect the power status of the initial power down. 0 The target subsystem was not powered down for the previous power-down request. 1 The target subsystem was powered down for the previous power-down request.

5.2.11.10 GPC PGC Control Register (GPC_PGC_*n*CTRL)

GPC PGC Control Register

Address: 303A_0000h base + C00h offset + (64d × i), where i=0d to 13d



GPC_PGC_*n*CTRL field descriptions

Field	Description
31–30 -	This field is reserved. Reserved
29–24 MEMPWR_ TCD1_TDR_ TRM	After scu pdn_req, count this value to assert A53 mempwr to 1'b1 **NOTE:** Can't be programmed to zero (This register control only for SCU Type PGC)
23–22 -	This field is reserved. Reserved
21–16 L2RETN_TCD1_ TDR	After scu pdn_req, count this value to assert A53 l2retn to 1'b0 **NOTE:** Can't be programmed to zero (This register control only for SCU Type PGC)
15–14 -	This field is reserved. Reserved
13–8 DFTRAM_TCD1	After scu pdn_req, count this value to assert A53 dftram to 1'b1

Table continues on the next page...

GPC_PGC_*n*CTRL field descriptions (continued)

Field	Description
	NOTE: Can't be programmed to zero (This register control only for SCU Type PGC)
7 -	This field is reserved. Reserved
6–1 L2RSTDIS	After scu pdn_req, count this value to assert A53 l2rstdis to 1'b1, it will be clear automatically once any of A53 core0/core1/core2/core3 is wakeup **NOTE:** Can't be programmed to zero (This register control only for SCU Type PGC)
0 PCR	Power Control **NOTE:** PCR must not change from power-down request (pdn_req) assertion until the target subsystem is completely powered up. 0 Do not switch off power even if pdn_req is asserted. 1 Switch off power when pdn_req is asserted.

5.2.11.11 GPC PGC Up Sequence Control Register (GPC_PGC_*n*PUPSCR)

GPC PGC Up Sequence Control Register

Address: 303A_0000h base + C04h offset + (64d × i), where i=0d to 13d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved									SW2ISO						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW2ISO								Reserved	SW						
Reset	1	0	0	1	0	1	1	1	1	1	0	0	0	0	0	1

GPC_PGC_*n*PUPSCR field descriptions

Field	Description
31–23 -	This field is reserved. Reserved
22–7 SW2ISO	After asserting switch_b, the PGC waits a number of clocks equal to the value of SW2ISO before negating isolation.
6 -	This field is reserved. Reserved
SW	After a power-up request (pup_req assertion), the PGC waits a number of clocks equal to the value of SW before asserting switch_b **NOTE:** SW must not be programmed to zero.

5.2.11.12 GPC PGC Down Sequence Control Register (GPC_PGC_*n*PDNSCR)

GPC PGC Down Sequence Control Register

Address: 303A_0000h base + C08h offset + (64d × i), where i=0d to 13d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		ISO2SW						Reserved		ISO					
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1

GPC_PGC_*n*PDNSCR field descriptions

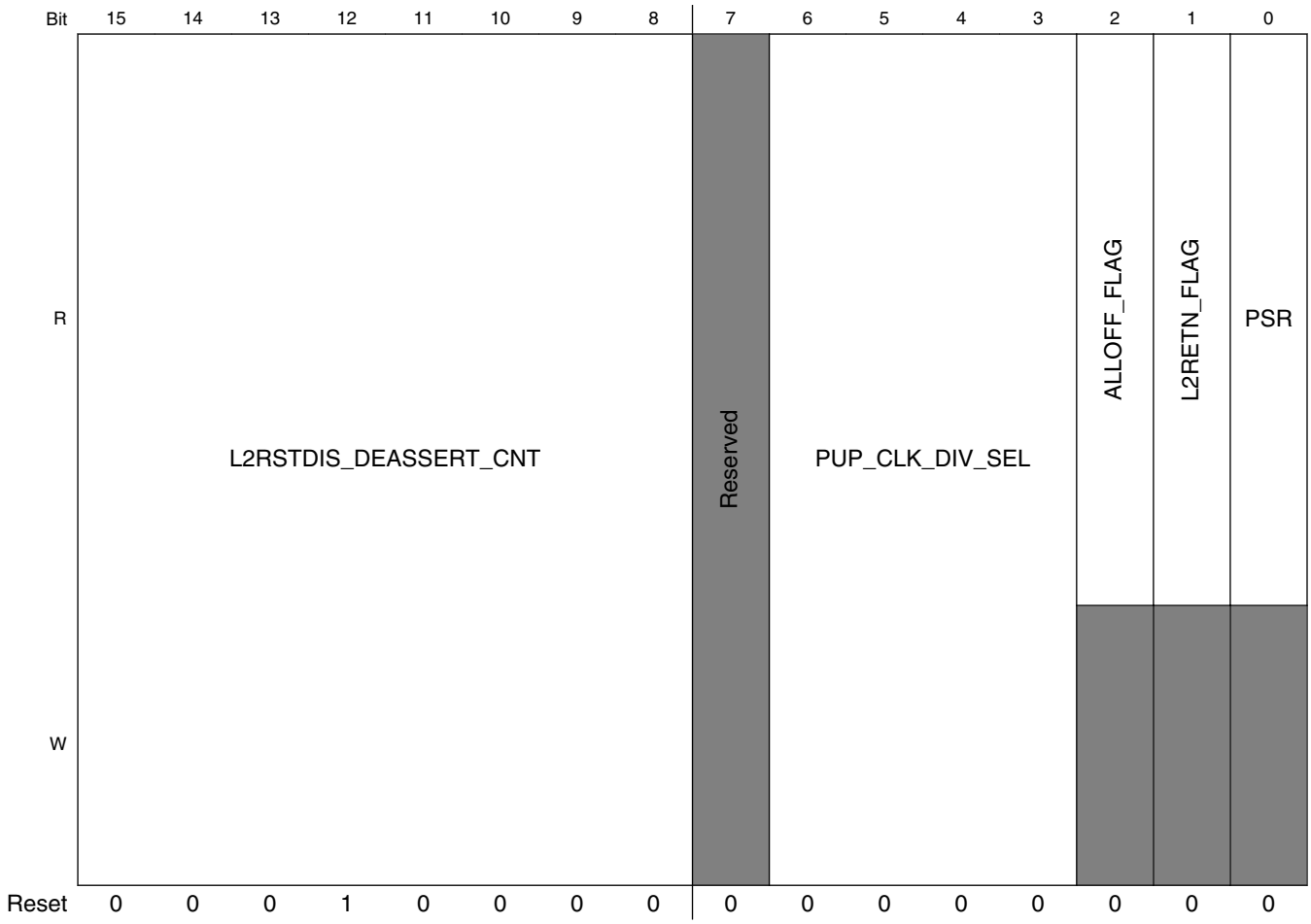
Field	Description
31–14 -	This field is reserved. Reserved
13–8 ISO2SW	After asserting isolation(by pdn_req assertion), the PGC waits a number of clocks equal to the value of ISO2SW before negating switch_b **NOTE:** ISO2SW must not be programmed to zero.
7–6 -	This field is reserved. Reserved
ISO	After a power-down request (pdn_req assertion), the PGC waits a number of clocks equal to the value of ISO before asserting isolation **NOTE:** ISO must not be programmed to zero.

5.2.11.13 GPC PGC Status Register (GPC_PGC_*n*SR)

GPC PGC Status Register

Address: 303A_0000h base + C0Ch offset + (64d × i), where i=0d to 13d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														L2RSTDIS_DEASSERT_CNT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



GPC_PGC_*n*SR field descriptions

Field	Description
31–18 -	This field is reserved. Reserved
17–8 L2RSTDIS_ DEASSERT_ CNT	Count this value to de-assert L2RSTDISABLE to LOW after CPU0 or CPU1 power up **NOTE:** This value can't be programmed to zero (This register control only for SCU Type PGC)
7 -	This field is reserved. Reserved
6–3 PUP_CLK_DIV_ SEL	Clock divider select for the clock of power up counter(count_clk is 32KHz for CPU/SCU type PGC, ipg_clk(66MHz) for MIX/PU Type PGC) 0000 1 0001 1/2 count_clk 0010 1/4 count_clk 0011 1/8 count_clk 0100 1/16 count_clk 0101 1/32 count_clk 0110 1/64 count_clk 0111 1/128 count_clk

Table continues on the next page...

GPC_PGC_*n*SR field descriptions (continued)

Field	Description
	1000 1/256 count_clk 1001 1/512 count_clk 1010 1/1024 count_clk 1011 1/2056 count_clk 1100 1/4096 count_clk 1101 1/8192 count_clk 1110 1/16384 count_clk 1111 1/32768 count_clk
2 ALLOFF_FLAG	All-off flag. **NOTE:** Software should write “1” to clear this flag after A53 is wakeup from ALL_OFF mode, otherwise, it will always keep to 1 (This register control only for SCU Type PGC) 0 A53 is not wakeup from ALL_OFF mode. 1 A53 is wakeup from ALL_OFF mode.
1 L2RETN_FLAG	L2 Retention Flag **NOTE:** Software should write “1” to clear this flag after A53 is wakeup from L2 retention mode, otherwise it will always keep to 1 (This register control only for SCU Type PGC) 0 A53 is not wakeup from L2 retention mode. 1 A53 is wakeup from L2 retention mode.
0 PSR	Power status. When in functional (or software-controlled debug) mode, PGC hardware sets PSR as soon as any of the power control output changes its state to one. Write one to clear this bit. Software should clear this bit after power up; otherwise, PSR continues to reflect the power status of the initial power down. 0 The target subsystem was not powered down for the previous power-down request. 1 The target subsystem was powered down for the previous power-down request.

5.3 Crystal Oscillator (XTALOSC)

5.3.1 Overview

The chip has two XTAL modules, 24MHz XTAL module and 32KHz XTAL module. The 24MHz XTAL module is instantiated from the XTAL IP, which includes:

- 24MHz crystal oscillator to generate reference clock
- Digital control logics for the XTAL

The 32KHz XTAL module uses a different IP and it is used as the clock source for the RTC, located in the SNVS.

The 24MHz oscillator will be used as the primary clock source for the PLLs to generate the clock for CPU, BUS, and high-speed interfaces. For all PLLs, the 24MHz clock from the oscillator can be used as the PLL reference clock directly.

The OSC IP used by the 24MHz XTAL module has three modes, Internal clock generation mode, External clock receive mode and Retention mode. The figure below shows the OSC IP integration diagram:

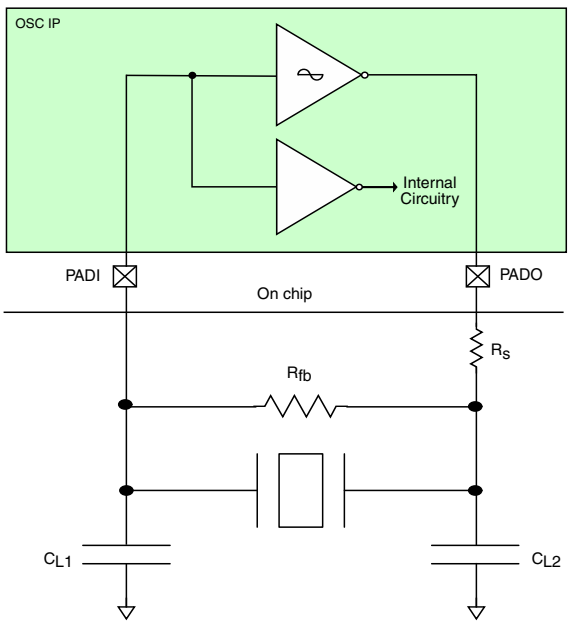


Figure 5-17. OSC IP Integration Diagram

During internal clock generation mode, a suitable quartz crystal is connected between PADI and PADO to generate the clock signal at the CK pin.

During external clock generation mode, the cell acts like a buffer, reflecting the PADI signal at CK.

The RTO (retention enable) signal retains the previous state of all the core input control signals. Logic at the RTO signal enables the retention operation.

Each XTAL module supports the following modes through register configuration:

- Normal oscillator mode - In normal mode, the XTAL IP generates stable square wave based on the crystal oscillator input.
- Bypass mode - In bypass mode, an external clock can be input through the XTAL pad.

5.3.2 XTALOSC Memory Map/Register Definition

XTALOSC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3027_0000	OSC Normal Clock Generation Control Register0 (XTALOSC_SYS_OSCNML_CTL0)	32	R/W	0000_0000h	5.3.2.1/ 715
3027_0004	OSC Normal Clock Generation Control Register1 (XTALOSC_SYS_OSCNML_CTL1)	32	R/W	0000_0000h	5.3.2.2/ 716

5.3.2.1 OSC Normal Clock Generation Control Register0 (XTALOSC_SYS_OSCNML_CTL0)

OSC Normal Clock Generation Control Register0

Address: 3027_0000h base + 0h offset = 3027_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EN	Reserved														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved											RTO	Reserved	SP	SF1	SF0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

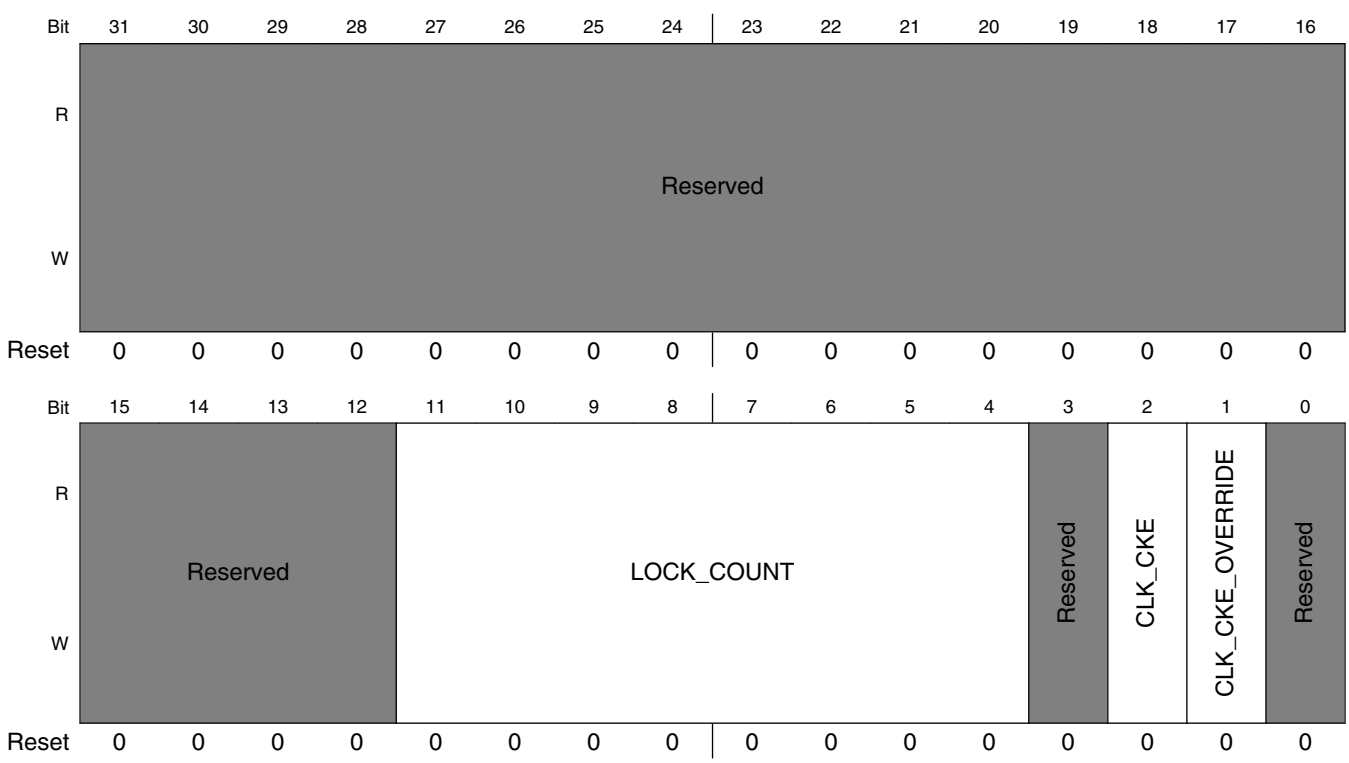
XTALOSC_SYS_OSCNML_CTL0 field descriptions

Field	Description
31 EN	Enable Oscillator
30–5 -	This field is reserved. Reserved
4 RTO	Retention Enable
3 -	This field is reserved. Reserved
2 SP	Select Power
1 SF1	Select Frequency1
0 SF0	Select Frequency0

5.3.2.2 OSC Normal Clock Generation Control Register1 (XTALOSC_SYS_OSCNML_CTL1)

OSC Normal Clock Generation Control Register1

Address: 3027_0000h base + 4h offset = 3027_0004h



XTALOSC_SYS_OSCNML_CTL1 field descriptions

Field	Description
31–12 -	This field is reserved. Reserved
11–4 LOCK_COUNT	Lock Signal Gen Counter
3 -	This field is reserved. Reserved
2 CLK_CKE	Oscillator Clock Gating Enable
1 CLK_CKE_ OVERRIDE	Oscillator Clock Gating Enable Override
0 -	This field is reserved. Reserved

5.4 Thermal Monitoring Unit (TMU)

5.4.1 Overview

The chip uses a temperature sensor to monitor the die temperature. The temperature sensor has a digitizer and local probe, and has the resolution of 1°C.

5.4.2 Features

- Junction temperature sensor function
- 1°C of sensing resolution
- 8-bit output codes with software calibration

5.4.3 Block Diagram

The TMU consists of a reference generation block, a positive TC voltage generator and a digitizer to convert the sensed voltage to 8-bit output codes.

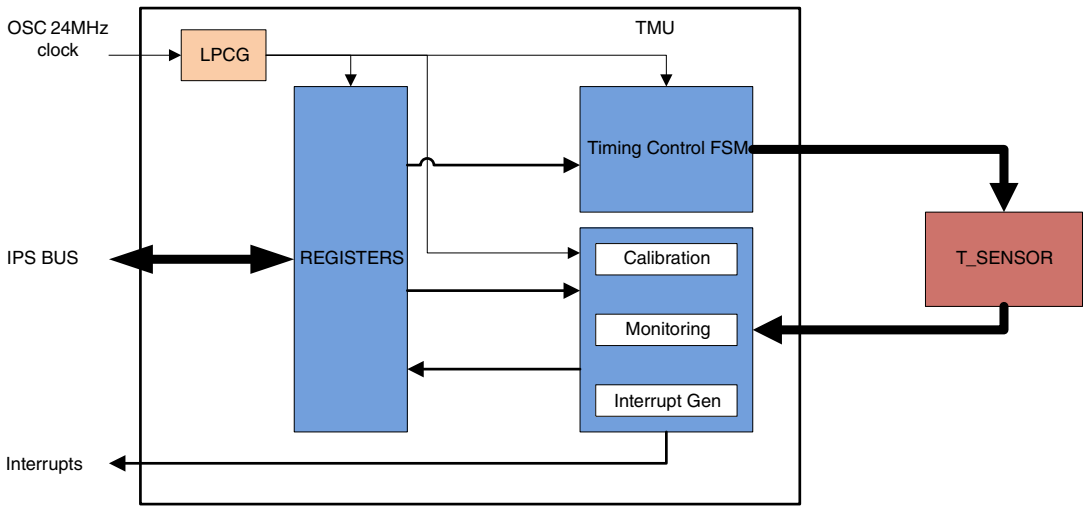


Figure 5-18. Block Diagram

5.4.4 Timing control FSM

The finite-state machine below describes the 8-bit clock input flow.

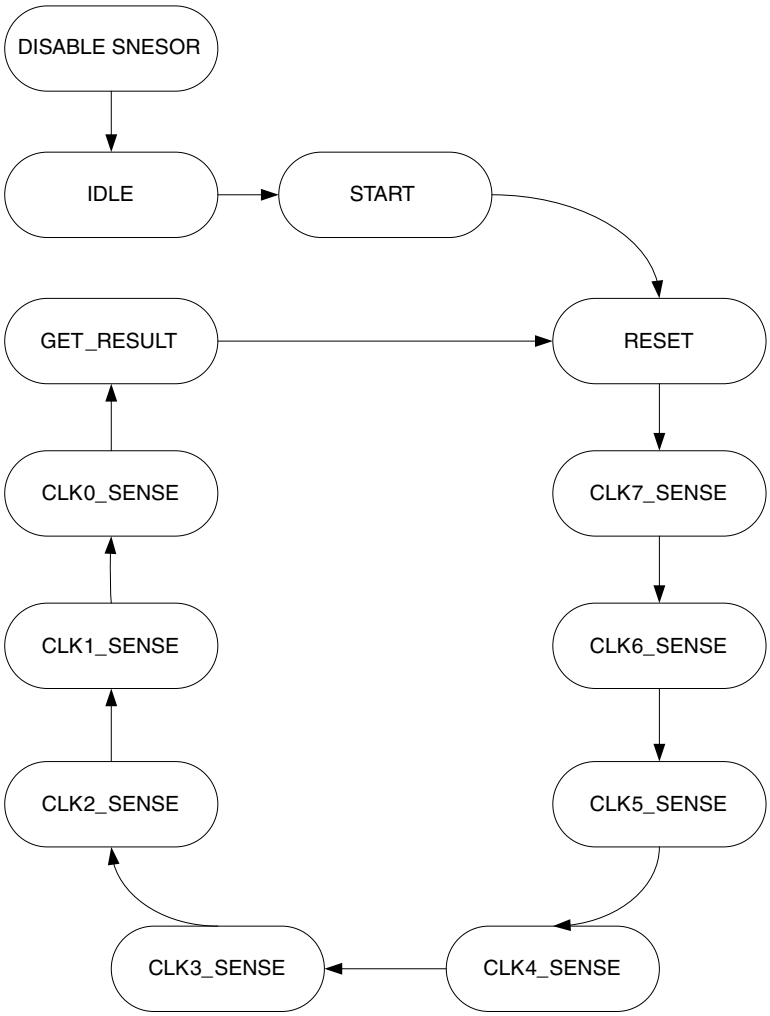


Figure 5-19. Timing Control FSM Diagram

5.4.5 Temperature Sensor Error Correction Method

The following figure shows an example of the difference between a reference curve and a real measured curve. The reference curve is based on ideal output codes vs. temperature, and the real measured curve is a waveform with process and mismatch error terms to the reference curve. There is DC offset between the real measured curve and the reference curve. In order to correct the error term, the temperature sensor uses software calibration method.

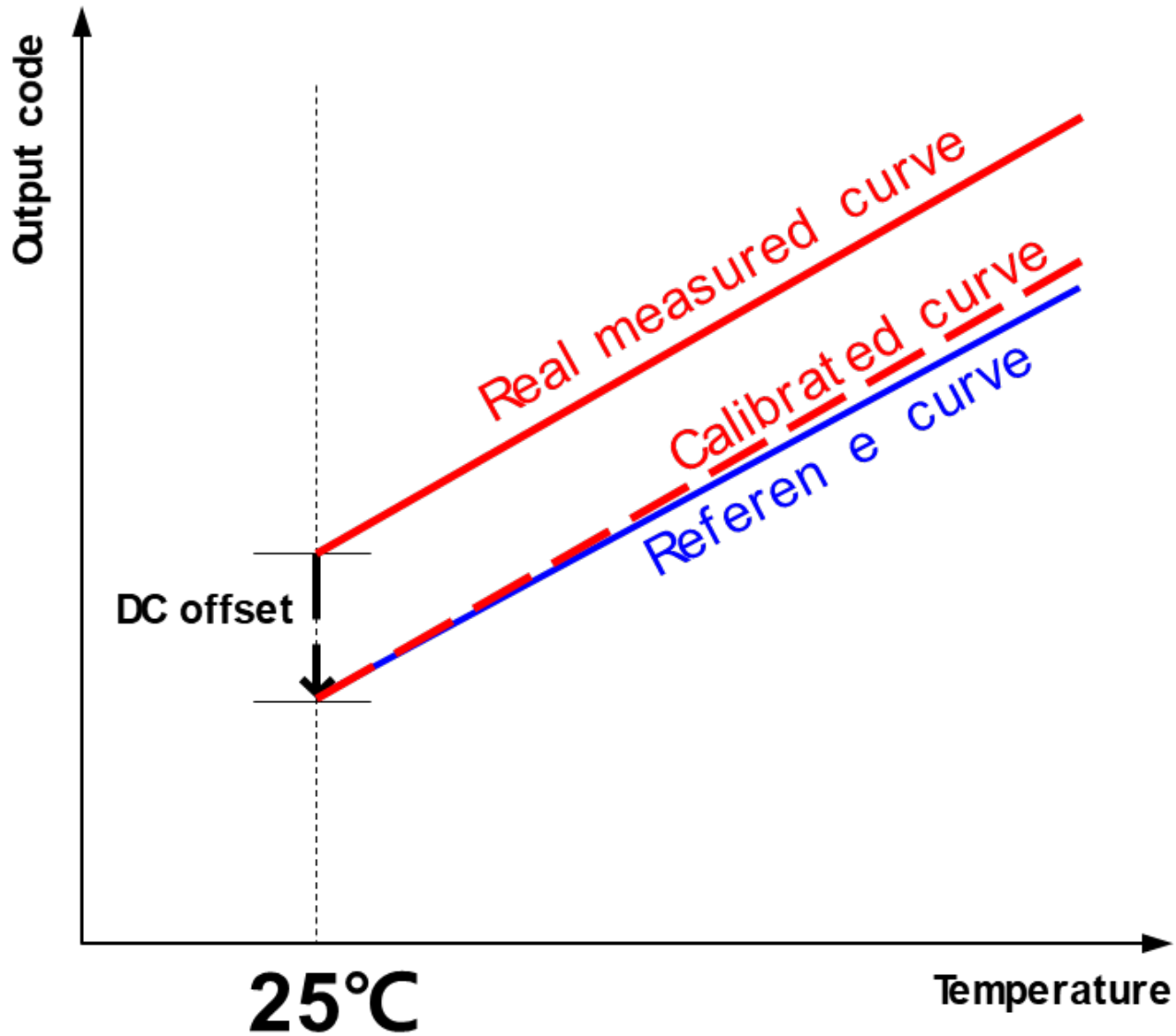


Figure 5-20. Reference Curve vs. Real Measured Curve

The software calibration method works as follows.

First, the temperature sensor senses die temperature at 25°C, and then 8-bit output codes are stored in the 8-bit OTP cells. With 8-bit data stored in OTP cells, the logic block (from the outside of the temperature sensor) is able to compensate DC offset.

Below equation shows the 1-point calibration method at 25°C.

$$T_{calib} = T_{sense} - (T_{E1} - 25)$$

Next equation shows the 1-point calibration method at 85°C.

$$T_{calib} = T_{sense} - (T_{E2} - 85)$$

where Tsense is the real measured output code including the DC offset, Tcalib is the calibrated output code of Tsense, TE1 (TE2) is the stored code in the first 8-bit OTP cell at 25°C (85°C) , and T25 (T85) is the ideal output code at 25°C (85°C). In other words, the DC offset is the difference between TE1 and T25. When the above equation is programmed in the glue logic block, the calibrated data is simply obtained by applying measured Tsense value.

Next equation shows the 2-point calibration method.

$$T_{calib} = (T_{sense} - T_{E1}) \times \frac{85-25}{T_{E2}-T_{E1}} + T_{E1} - (T_{E1} - 25)$$

where TE1 and TE2 are the temperature values stored in the 8-bit OTP cells from the 25°C and 85°C measurement, respectively. Units of all terms are degree (°C).

According to the above equation, (TE1 – 25) represents DC offset term and

$$(T_{sense} - T_{E1}) \times \frac{85 - 25}{T_{E2} - T_{E1}} + T_{E1}$$

represents slope compensation term. DC offset is simply difference between 25°C and TE1. Slope compensation term is reciprocal of temperature slope, (85 – 25)/(TE2 – TE1) times (Tsense – TE1). To obtain calibrated data, TE1 is added to slope compensation term for the slope error correction and DC offset tem is finally subtracted. If the above equation is programmed in the software, the calibrated data is simply obtained by applying measured Tsense value.

By using software slope calibration, it is possible to control a temperature slope of Tcalib. The software calibration method is below.

$$T_{calib\_SW} = \alpha \times (T_{sense} - T_{E1}) + 25$$

$$\alpha = \frac{(85 - 25)}{(Tcalib - T_{E1})}$$

5.4.6 Temperature Result Process

When the new sensor result is received, hardware will do 1 point calibration (software can do 2 points calibration). The calculated results and temperature value will then be transferred to registers, based on the calibration result and sensor-temperature mapping relationship. The hardware monitor checks the result effectiveness every time it gets the new sensor result. Hardware can also generate a normal interrupt or critical interrupt to the system when one of the three sensors reports a temperature value higher than the threshold value programmed in the registers

5.4.7 TMU Memory Map/Register Definition

This section provides a detailed description of all accessible TMU memory and registers. The table below lists the TMU registers. Note that the full register address is comprised of the programmable CCSRBAR together with the offset listed.

NOTE

The EN bit field of the TMU Enable Register (TMU_TER[EN]) must always be enabled for the part to operate correctly.

TMU memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3026_0000	TMU Enable Register (TMU_TER)	32	R/W	0000_0001h	5.4.7.1/ 723
3026_0004	TMU Status register (TMU_TSR)	32	R	0000_0000h	5.4.7.2/ 724
3026_0008	TMU Interrupt Enable register (TMU_TIER)	32	R/W	0000_0000h	5.4.7.3/ 725
3026_000C	TMU Interrupt Detect register (TMU_TIDR)	32	w1c	0000_0000h	5.4.7.4/ 726
3026_0010	TMU Monitor High Temperature Immediate Threshold register (TMU_TMHTITR)	32	R/W	0000_0000h	5.4.7.5/ 727
3026_0014	TMU Monitor High Temperature Average threshold register (TMU_TMHTATR)	32	R/W	0000_0000h	5.4.7.6/ 728
3026_0018	TMU Monitor High Temperature Average Critical Threshold register (TMU_TMHTACTR)	32	R/W	0000_0000h	5.4.7.7/ 729
3026_001C	TMU Sensor Calibration register (TMU_TSCR)	32	R	0000_0000h	5.4.7.8/ 729

Table continues on the next page...

TMU memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3026_0020	TMU Report Immediate Temperature Site register n (TMU_TRITSR)	32	R	0000_0000h	5.4.7.9/ 730
3026_0024	TMU Report Average Temperature Site register n (TMU_TRATSR)	32	R	0000_0000h	5.4.7.10/ 731

5.4.7.1 TMU Enable Register (TMU_TER)

Address: 3026_0000h base + 0h offset = 3026_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EN	Reserved														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														ALPF	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

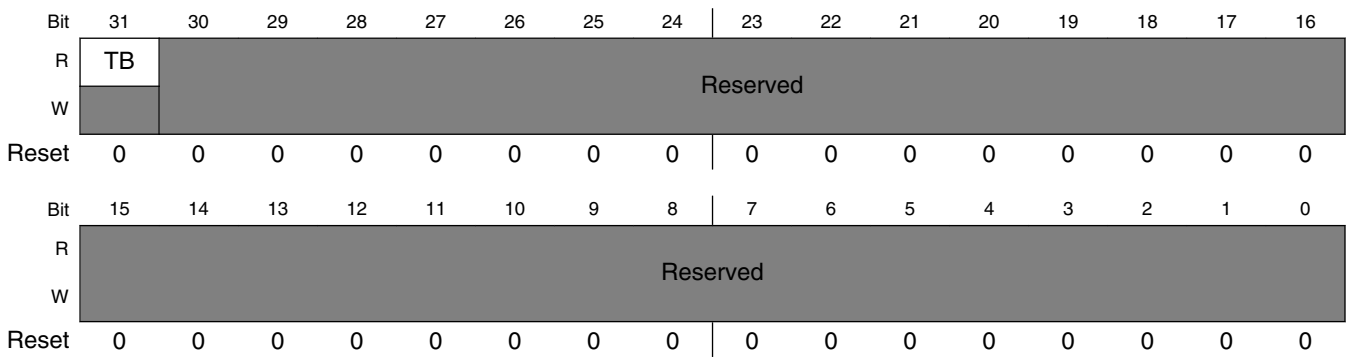
TMU_TER field descriptions

Field	Description
31 EN	Enable monitoring the temperature sensor **NOTE:** The POR value is disabled, but is enabled in boot. To ensure proper operation, this field must be enabled. 0 No monitoring 1 Enable monitoring
30–2 -	This field is reserved. Reserved
ALPF	Average low pass filter setting. The average temperature is calculated as: ALPF x Current_Temp + (1 - ALPF) x Average_Temp. If no previous (average) temperature is valid, current temperature is used. For proper operation, this field should only change when monitoring is disabled. 00 1.0 01 0.5 10 0.25 11 0.125

5.4.7.2 TMU Status register (TMU_TSR)

The TMU status register reports the monitoring status during operation.

Address: 3026_0000h base + 4h offset = 3026_0004h



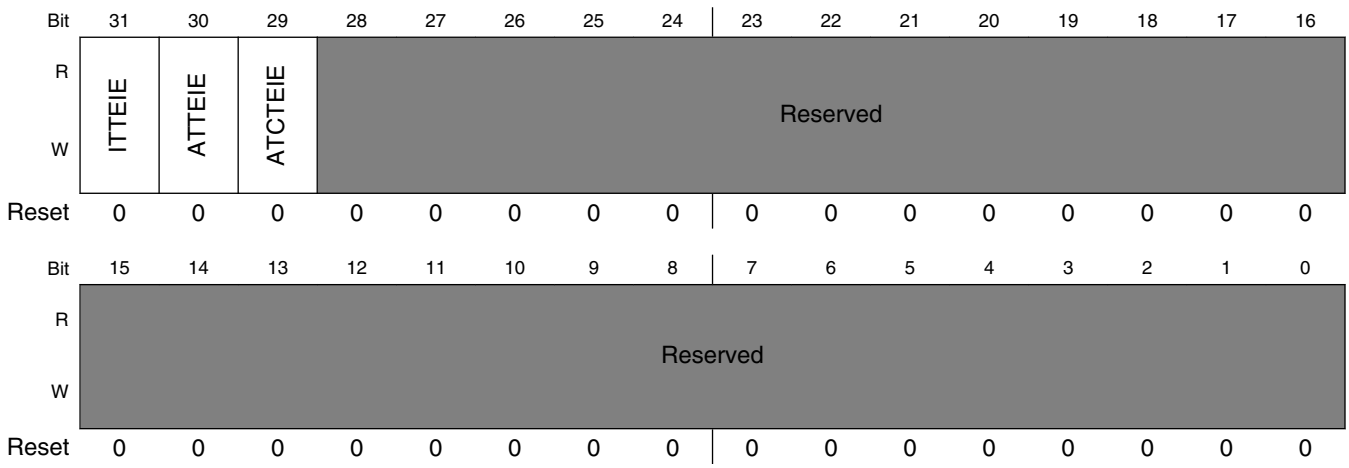
TMU_TSR field descriptions

Field	Description
31 TB	TMU busy. 0 TMU idle. 1 TMU busy. In monitoring mode this indicates a temperature measurement is pending. In calibration mode, sensor result has not yet been determined based on last given ambient temperature.
-	This field is reserved. Reserved

5.4.7.3 TMU Interrupt Enable register (TMU_TIER)

The TMU interrupt enable register determines if a detected status condition should cause a system interrupt. A system interrupt occurs if a bit in this register is set and the corresponding bit in the interrupt detect register is also set. To clear the interrupt, write a 1 to the interrupt detect register.

Address: 3026_0000h base + 8h offset = 3026_0008h



TMU_TIER field descriptions

Field	Description
31 ITTEIE	Immediate temperature threshold exceeded interrupt enable. 0 Disabled. 1 Interrupt enabled. Generate an interrupt if TIDR[ITTE] is set. Write 1 to this bit will clear bit TIDR[ITTE].
30 ATTEIE	Average temperature threshold exceeded interrupt enable. 0 Disabled. 1 Interrupt enabled. Generate an interrupt if TIDR[ATTE] is set. Write 1 to this bit will clear bit TIDR[ATTE].
29 ATCTEIE	Average temperature critical threshold exceeded interrupt enable. 0 Disabled. 1 Interrupt enabled. Generate an interrupt if TIDR[ATCTE] is set. Write 1 to this bit will clear bit TIDR[ATCTE].
-	This field is reserved. Reserved

5.4.7.4 TMU Interrupt Detect register (TMU_TIDR)

The TMU interrupt detect register indicates if an status condition was detected that could generate an interrupt. Write 1 to clear the detected condition and the interrupt, if enabled.

Address: 3026_0000h base + Ch offset = 3026_000Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	ITTE	ATTE	ATCTE	Reserved											
W		w1c	w1c	w1c												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TMU_TIDR field descriptions

Field	Description
31 -	This field is reserved. Reserved
30 ITTE	Immediate temperature threshold exceeded. Write 1 to clear. 0 No threshold exceeded. 1 Immediate temperature threshold, as defined by TMHTITR, has been exceeded. This includes an out-of-range measured temperature above 125°C.
29 ATTE	Average temperature threshold exceeded. Write 1 to clear. 0 No threshold exceeded. 1 Average temperature threshold, as defined by TMHTATR, has been exceeded.
28 ATCTE	Average temperature critical threshold exceeded. Write 1 to clear. 0 No threshold exceeded. 1 Average temperature critical threshold, as defined by TMHTACTR, has been exceeded.
-	This field is reserved. Reserved

5.4.7.5 TMU Monitor High Temperature Immediate Threshold register (TMU_TMHTITR)

This TMU monitor register determines the high current temperature threshold for generating the TIDR[ITTE] event.

Address: 3026_0000h base + 10h offset = 3026_0010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EN	Reserved														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								TEMP							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TMU_TMHTITR field descriptions

Field	Description
31 EN	Enable threshold. 0 Disabled. 1 Threshold enabled.
30–8 -	This field is reserved. Reserved

Table continues on the next page...

TMU_TMHTITR field descriptions (continued)

Field	Description
TEMP	High temperature immediate threshold value. Determines the current upper temperature threshold when EN=1. 10-125 °C Sensor range 0-9 and 126-255 °C Reserved

5.4.7.6 TMU Monitor High Temperature Average threshold register (TMU_TMHTATR)

This TMU monitor register determines the high average temperature threshold for generating the TIDR[ATTE] event. The low-pass filter setting, TMR[ALPF], determines the function for calculating average temperature.

Address: 3026_0000h base + 14h offset = 3026_0014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EN	Reserved														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								TEMP							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TMU_TMHTATR field descriptions

Field	Description
31 EN	Enable threshold. 0 Disabled. 1 Threshold enabled.
30–8 -	This field is reserved. Reserved
TEMP	High temperature average threshold value. Determines the average upper temperature threshold when EN=1. 10-125 °C Sensor range 0-9 and 126-255 °C Reserved

5.4.7.7 TMU Monitor High Temperature Average Critical Threshold register (TMU_TMHTACTR)

This TMU monitor register determines the high average critical temperature threshold for generating the TIDR[ATCTE] event. The low-pass filter setting, TMR[ALPF], determines the function for calculating average temperature.

Address: 3026_0000h base + 18h offset = 3026_0018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EN	Reserved														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								TEMP							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TMU_TMHTACTR field descriptions

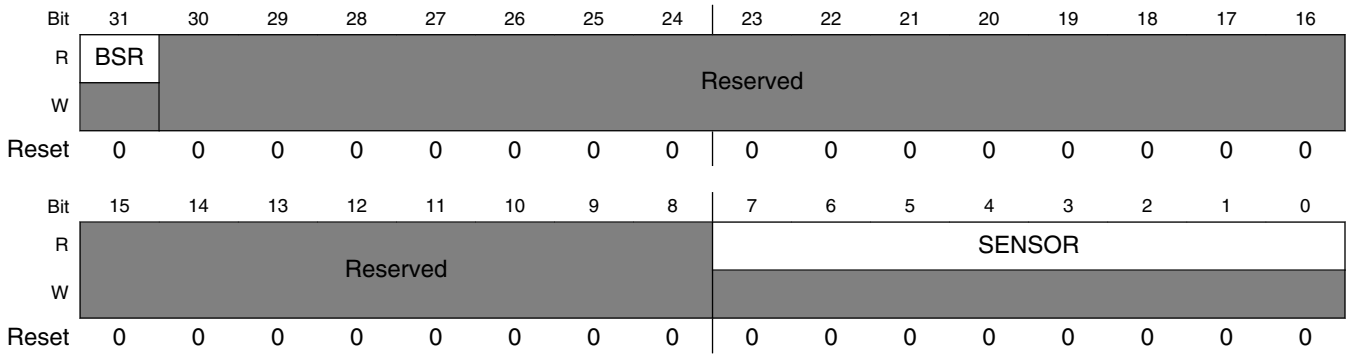
Field	Description
31 EN	Enable threshold. 0 Disabled. 1 Threshold enabled.
30–8 -	This field is reserved. Reserved
TEMP	High temperature average critical threshold value. Determines the average upper critical temperature threshold when EN=1. 10-125 °C Sensor range 0-9 and 126-255 °C Reserved

5.4.7.8 TMU Sensor Calibration register (TMU_TSCR)

The TMU sensor calibration register, in conjunction with the temperature calibration register, is used to build the internal sensor translation table used during monitoring.

During the calibration phase, TMR[CE]=1, reading this register will return the sensor reading. After writing the temperature calibration register, user should poll the TSR[TB] bit to determine when the sensor reading is complete and then read the register. If BSR=1 the sensor reading could not be established due to temperature outside sensor range.

Address: 3026_0000h base + 1Ch offset = 3026_001Ch



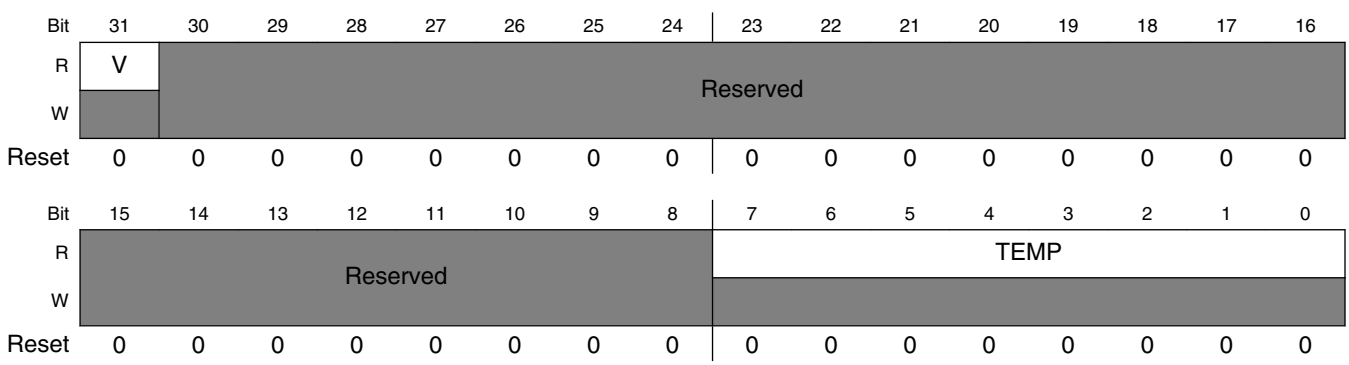
TMU_TSCR field descriptions

Field	Description
31 BSR	Beyond temperature sensor range of 10-125 °C. The sensor reading is invalid. Read-only.
30–8 -	This field is reserved. Reserved
SENSOR	Sensor reading. Read only.

5.4.7.9 TMU Report Immediate Temperature Site register n (TMU_TRITSR)

This TMU report register returns the last measured temperature at site. The site must be part of the list of enabled monitored sites as defined by TMR[MSITE].

Address: 3026_0000h base + 20h offset = 3026_0020h



TMU_TRITSR field descriptions

Field	Description
31 V	Valid measured temperature.

Table continues on the next page...

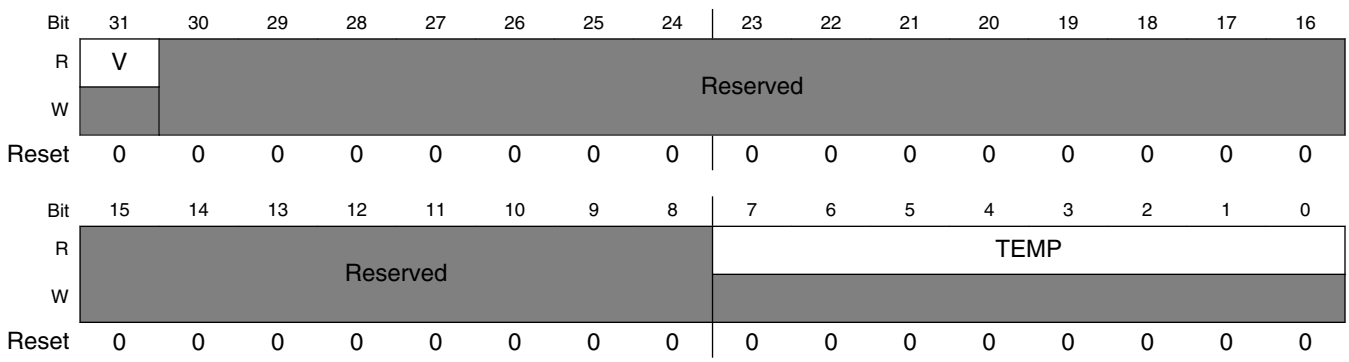
TMU_TRITSR field descriptions (continued)

Field	Description
	0 Not valid. Temperature out of sensor range or first measurement still pending. 1 Valid.
30–8 -	This field is reserved. Reserved
TEMP	Last temperature reading at site when V=1.

5.4.7.10 TMU Report Average Temperature Site register n (TMU_TRATSR)

This TMU report register returns the average measured temperature at site.

Address: 3026_0000h base + 24h offset = 3026_0024h



TMU_TRATSR field descriptions

Field	Description
31 V	Valid measured temperature. 0 Not valid. Temperature out of sensor range or first measurement still pending. 1 Valid.
30–8 -	This field is reserved. Reserved
TEMP	Average temperature reading at site0 when V=1.

Chapter 6
SNVS, Reset, Fuse, and Boot

6.1 System Boot

6.1.1 Overview

The boot process begins at the Power-On Reset (POR) where the hardware reset logic forces the Arm core to begin the execution starting from the on-chip boot ROM.

The boot ROM code uses the state of the internal register BOOT_MODE[1:0] as well as the state of various eFUSEs and/or GPIO settings to determine the boot flow behavior of the device.

The main features of the ROM include:

- Support for booting from various boot devices
- Serial downloader support (USB OTG)
- Device Configuration Data (DCD) and plugin
- Wake-up from the low-power modes

The boot ROM supports these boot devices:

- Serial NOR Flash via FlexSPI
- NAND flash
- SD/MMC
- Serial (SPI) NOR

The boot ROM uses the state of the BOOT_MODE and eFUSEs to determine the boot device. For development purposes, the eFUSEs used to determine the boot device may be overridden using the GPIO pin inputs.

The boot ROM code also allows to download the programs to be run on the device. The example is a provisioning program that can make further use of the serial connection to provide a boot device with a new image. Typically, the provisioning program is

downloaded to the internal RAM and allows to program the boot devices, such as the SD/MMC flash. The ROM serial downloader uses a high-speed USB in a non-stream mode connection.

The Device Configuration Data (DCD) feature allows the boot ROM code to obtain the SOC configuration data from an external program image residing on the boot device. As an example, the DCD can be used to program the DDR controller for optimal settings, improving the boot performance. The DCD is restricted to the memory areas and peripheral addresses that are considered essential for the boot purposes (see Write data command).

A key feature of the boot ROM is the ability to perform a secure boot, also known as a High-Assurance Boot (HAB). This is supported by the HAB security library which is a subcomponent of the ROM code. The HAB uses a combination of hardware and software together with the Public Key Infrastructure (PKI) protocol to protect the system from executing unauthorized programs. Before the HAB allows the user image to execute, the image must be signed. The signing process is done during the image build process by the private key holder and the signatures are then included as a part of the final program image. If configured to do so, the ROM verifies the signatures using the public keys included in the program image. A secure boot with HAB can be performed on all boot devices supported on the chip in addition to the serial downloader. The HAB library in the boot ROM also provides the API functions, allowing the additional boot chain components (bootloaders) to extend the secure boot chain. The out-of-fab setting for the SEC_CONFIG is the open configuration, in which the ROM/HAB performs the image authentication, but all authentication errors are ignored and the image is still allowed to execute.

6.1.2 Boot modes

During reset, the chip checks the power gating controller status register.

During boot, the core's behavior is defined by the boot mode pin settings, as described in Boot mode pin settings. When waking up from the low-power boot mode, the core skips the clock settings. The boot ROM checks that the PERSISTENT_ENTRY0 (see Persistent bits) is a pointer to a valid address space (OCRAM, DDR, or EIM). If the PERSISTENT_ENTRY0 is a pointer to a valid range, it starts the execution using the entry point from the PERSISTENT_ENTRY0 register. If the PERSISTENT_ENTRY0 is a pointer to an invalid range, the core performs the system reset.

6.1.2.1 Boot mode pin settings

The device has four boot modes (one is reserved for NXP use). The boot mode is selected based on the binary value stored in the internal BOOT_MODE register.

The BOOT_MODE is initialized by sampling the BOOT_MODE0 and BOOT_MODE1 inputs on the rising edge of the POR_B. After these inputs are sampled, their subsequent state does not affect the contents of the BOOT_MODE internal register. The state of the internal BOOT_MODE register may be read from the BMOD[1:0] field of the SRC Boot Mode Register (SRC_SBMR2). The available boot modes are: Boot From Fuses, serial boot via USB, and Internal Boot. See this table for settings:

Table 6-1. Boot MODE pin settings

BOOT_MODE[1:0]	Boot Type
00	Boot From Fuses
01	Serial Downloader
10	Internal Boot
11	Reserved

6.1.2.2 High-level boot sequence

The figure found here show the high-level boot ROM code flow.

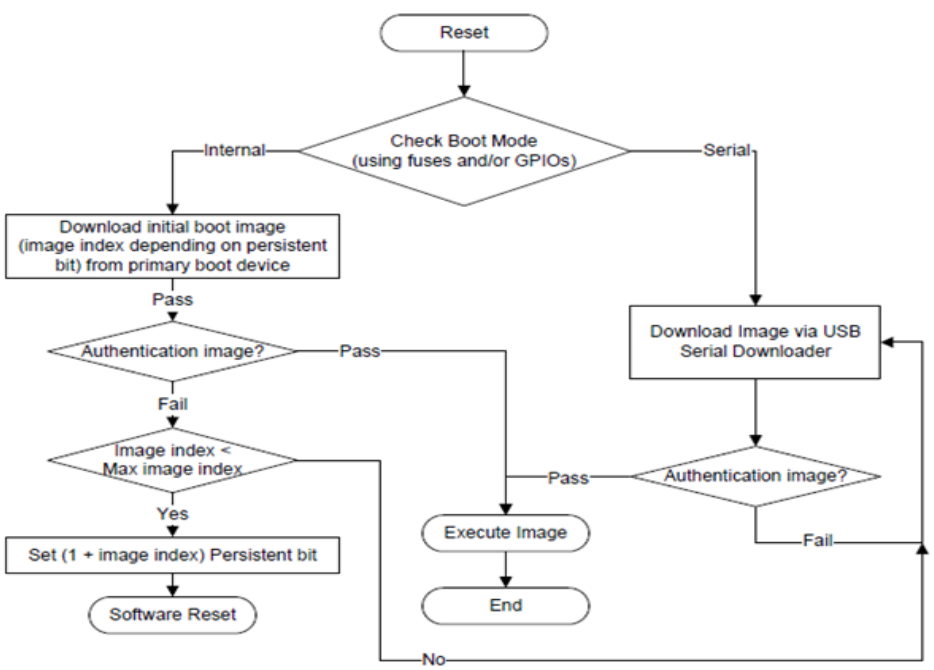


Figure 6-1. Boot flow

6.1.2.3 Boot From Fuses mode (BOOT_MODE[1:0] = 00b)

A value of 00b in the BOOT_MODE[1:0] register selects the Boot From Fuses mode.

This mode is similar to the Internal Boot mode described in Internal Boot mode (BOOT_MODE[1:0] = 0b10) with one difference. In this mode, the GPIO boot override pins are ignored. The boot ROM code uses the boot eFUSE settings only. This mode also supports a secure boot using HAB.

If set to Boot From Fuses, the boot flow is controlled by the BT_FUSE_SEL eFUSE value. If BT_FUSE_SEL = 0, indicating that the boot device (for example, flash, SD/MMC) was not programmed yet, the boot flow jumps directly to the Serial Downloader. If BT_FUSE_SEL = 1, the normal boot flow is followed, where the ROM attempts to boot from the selected boot device.

The first time a board is used, the default eFUSEs may be configured incorrectly for the hardware on the platform. In such case, the Boot ROM code may try to boot from a device that does not exist. This may cause an electrical/logic violation on some pads. Using the Boot From Fuses mode addresses this problem.

Setting the BT_FUSE_SEL=0 forces the ROM code to jump directly to the Serial Downloader. This allows a bootloader to be downloaded which can then provision the boot device with a program image and blow the BT_FUSE_SEL and the other boot configuration eFUSEs. After the reset, the boot ROM code determines that the BT_FUSE_SEL is blown (BT_FUSE_SEL = 1) and the ROM code performs an internal boot according to the new eFUSE settings. This allows the user to set BOOT_MODE[1:0]=00b on a production device and burn the fuses on the same device (by forcing the entry to the Serial Downloader), without changing the value of the BOOT_MODE[1:0] or the pullups/pulldowns on the BOOT_MODE pins.

6.1.2.4 Serial Downloader

The Serial Downloader provides a means to download a program image to the chip over the USB serial connection.

In this mode, the ROM programs the WDOG1 for a time-out specified by the fuse WDOG Time-out Select (See fusemap for details) if the WDOG_ENABLE eFuse is 1 and continuously polls for the USB connection. If no activity is found on the USB OTG1 and the watchdog timer expires, the Arm core is reset.

NOTE

After the downloaded image is loaded, it is responsible for managing the watchdog resets properly.

This figure shows the USB boot flow:

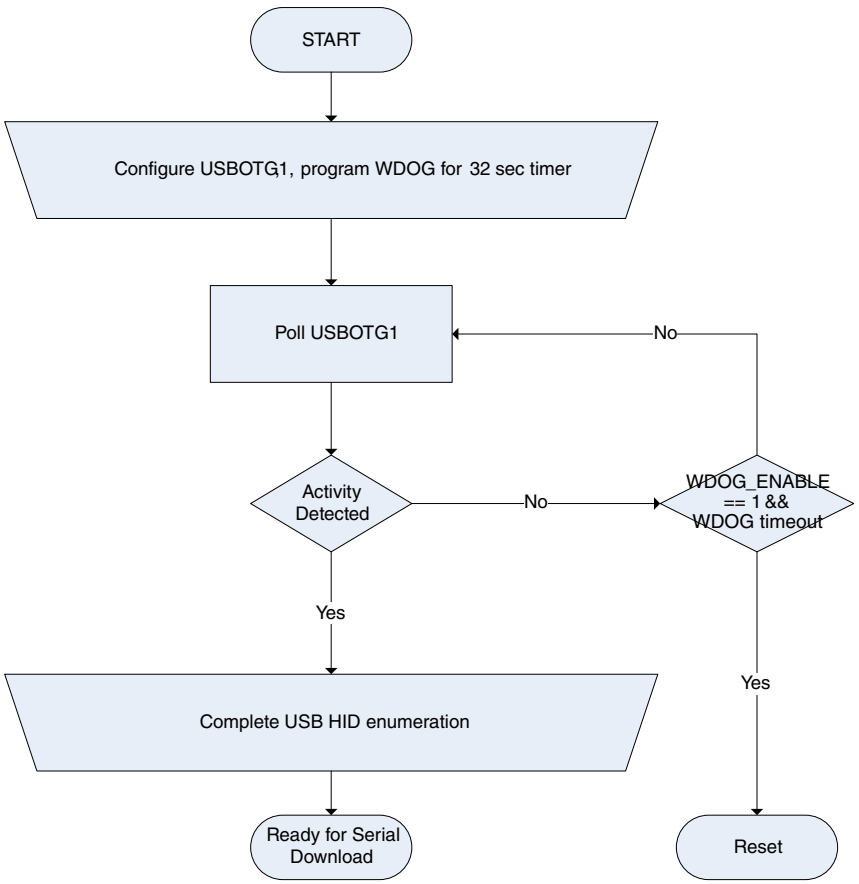


Figure 6-2. Serial Downloader boot flow

NOTE

Before going into USB serial mode, Boot ROM detect SD/MMC card on USDHC2 port. If a card is inserted, ROM will try to boot from it. This is the so-called Manufacture SD/MMC boot. This feature can be disabled by blowing fuse “Disable SD/MMC Manufacture Mode”. See SD/MMC manufacture mode for details.

6.1.2.5 Internal Boot mode (BOOT_MODE[1:0] = 0b10)

A value of 0b10 in the BOOT_MODE[1:0] register selects the Internal Boot mode. In this mode, the processor continues to execute the boot code from the internal boot ROM.

The boot code performs the hardware initialization, loads the program image from the chosen boot device, performs the image validation using the HAB library (see Boot security settings), and then jumps to an address derived from the program image. If an error occurs during the internal boot, the boot code jumps to the Serial Downloader (see Serial Downloader). A secure boot using the HAB is possible in all the three boot modes.

When set to the Internal Boot, the boot flow may be controlled by a combination of eFUSE settings with an option of overriding the fuse settings using the General Purpose I/O (GPIO) pins. The GPIO Boot Select FUSE (BT_FUSE_SEL) determines whether the ROM uses the GPIO pins for a selected number of configuration parameters or eFUSEs in this mode.

- If BT_FUSE_SEL = 1, all boot options are controlled by the eFUSEs described in Table 6-2.
- If BT_FUSE_SEL = 0, the specific boot configuration parameters may be set using the GPIO pins rather than eFUSEs. The fuses that can be overridden when in this mode are indicated in the GPIO column of Table 6-2. Table 6-3 provides the details of the GPIO pins.

The use of the GPIO overrides is intended for development since these pads are used for other purposes in the deployed products. NXP recommends controlling the boot configuration by the eFUSEs in the deployed products and reserving the use of the GPIO mode for the development and testing purposes only.

6.1.2.6 Boot security settings

The internal boot modes use one of three security configurations.

- Closed: This level is intended for use with shipping-secure products. All HAB functions are executed and the security hardware is initialized (the Security Controller or SNVS enters the Secure state), the DCD is processed if present, and the program image is authenticated by the HAB before its execution. All detected errors are logged, and the boot flow is aborted with the control being passed to the serial downloader. At this level, the execution does not leave the internal ROM unless the target executable image is authenticated.
- Open: This level is intended for use in non-secure products or during the development phases of a secure product. All HAB functions are executed as for a closed device. The security hardware is initialized (except for the SNVS which is left

in the Non-Secure state), the DCD is processed if present, and the program image is authenticated by the HAB before its execution. All detected errors are logged, but have no influence on the boot flow which continues as if the errors did not occur. This configuration is useful for a secure product development because the program image runs even if the authentication data is missing or incorrect, and the error log can be examined to determine the cause of the authentication failure.

- Field Return: This level is intended for the parts returned from the shipped products.

6.1.3 Device configuration

This section describes the external inputs that control the behavior of the Boot ROM code.

This includes the boot device selection (SD, MMC, and so on), boot device configuration (SD bus width, speed, and so on), and other. In general, the source for this configuration comes from the eFUSEs embedded inside the chip. However, certain configuration parameters can be sourced from the GPIO pins, allowing further flexibility during the development process.

6.1.3.1 Boot eFUSE descriptions

This table is a comprehensive list of the configuration parameters that the ROM uses.

Table 6-2. Boot eFUSE descriptions

Fuse	Configuration	Definition	GPIO[1]	Shipped value	Settings[2]
BT_FUSE_SEL	OEM	In the Internal Boot mode BOOT_MODE[1:0] = 10, the BT_FUSE_SEL fuse determines whether the boot settings indicated by a Yes in the GPIO column are controlled by the GPIO pins or the eFUSE settings in the On-Chip OTP Controller (OCOTP). In the Boot From Fuse mode BOOT_MODE[1:0] = 00, the BT_FUSE_SEL fuse indicates whether the bit configuration eFuses are programmed.	NA	0	If BOOT_MODE[1:0] = 0b10: • 0—The bits of the SBMR are overridden by the GPIO pins. • 1—The specific bits of the SBMR are controlled by the eFUSE settings. If BOOT_MODE[1:0] = 0b00 • 0—The BOOT configuration eFuses are not programmed yet. The boot flow jumps to the serial downloader. • 1—The BOOT configuration eFuses are programmed. The regular boot flow is performed.

Table continues on the next page...

Table 6-2. Boot eFUSE descriptions (continued)

Fuse	Configuration	Definition	GPIO[1]	Shipped value	Settings[2]
SEC_CONFIG[1:0]	SEC_CONFIG[0] - NXP SEC_CONFIG[1] - OEM	Security Configuration, as defined in Boot security settings	NA	01	00—Reserved 01—Open (allows any program image, even if the authentication fails) 1x—Closed (The program image executes only if authenticated)
FIELD_RETURN	OEM	Configure device for field return testing. Fuse burning is protected by CSF command, with proper parameter passed. Non-lockable.			0—Device is in functional/secure mode. 1—Device is open for "field-return" testing.
SRK_HASH[255:0]	OEM	256-bit hash value of the super root key (SRK_HASH)	NA	0	Settings vary—used by HAB
BT_MMU_DISABLE (BOOT_CFG[16])	OEM	The MMU/L1 D Cache disable bit used by the boot ROM for fast HAB processing.	No	0	0—MMU/L1 D Cache is enabled by the ROM during the boot. 1—MMU/L1 D Cache is disabled by the ROM during the boot.
L1 I-Cache DISABLE	OEM	L1 I Cache disable bit used by the boot during the entire execution.	No	0	0—L1 I Cache is enabled by the ROM during the boot. 1—L1 I Cache is disabled by the ROM during the boot.
LPB_BOOT	OEM	USB Low-Power Boot	No	0	0x—LPB Disable 10—Divide by 2 11—Divide by 4
BT_LPB_POLARITY	OEM	USB Low-Power Boot GPIO polarity	No	0	0—Low on the GPIO pad indicates the low-power condition. 1—High on the GPIO pad indicates the low-power condition.
WDOG_ENABLE	OEM	Watchdog reset counter enable	No	0	0—The watchdog reset counter is disabled during the serial downloader. 1—The watchdog reset counter is enabled during the serial downloader.
Recovery Boot Enable	OEM	SPI recovery boot enable	No	0	0—Disabled 1—Enabled
MMC_DLL_DLY[6:0]	OEM	uSDHC Delay Line settings	No	0	uSDHC Delay Line settings
SRK_REVOKE[2:0]	OEM	SRK revocation mask	No	0	SRK revocation mask
DISABLE_SDMMC_MFG	OEM	Disable the SDMMC manufacture mode	Yes	0	0—enable the SD/MMC MFG mode 1—disable the SD/MMC MFG mode

Table continues on the next page...

Table 6-2. Boot eFUSE descriptions (continued)

Fuse	Configuration	Definition	GPIO[1]	Shipped value	Settings[2]
USDHC_PAD_SETTINGS NAND_PAD_SETTINGS	OEM	Override values for the SD/MMC and NAND boot modes	No	0	Override the following IO PAD settings: [1:0] Driver Strength [2] Slew Rate [3] Hysteresis > [4] Pull/Keeper select [6:5] Pull up/down config .
OVERRIDE_HYS_SDMMC_PADS	OEM	Overrides the HYS bit for the SD pads	No	0	Override the IO PAD setting HYS to 1 for the SD pads.
eMMC_4.4_RESET_TO_PRE-IDLE_STATE	OEM	ROM resets the boot device in the pre-idle state using the eMMC 4.4 feature, CMD0 with the argument value 0xf0f0f0f0.	No	0	Applicable for booting from the eMMC 4.4 spec or greater version devices. The fuse must not be blown for the eMMC 4.3 or lesser spec version devices.

1. This setting can be overridden by the GPIO settings when the BT_FUSE_SEL fuse is intact. See GPIO Boot Overrides for the corresponding GPIO pin.
2. 0 = intact fuse and 1= blown fuse

6.1.3.2 GPIO boot overrides

This table provides a list of the GPIO boot overrides:

Table 6-3. GPIO override contact assignments

Package pin	Direction on reset	eFuse
BOOT_MODE1	Input	Boot mode selection
BOOT_MODE0	Input	
SAI1_RXD0	Input	BOOT_CFG[0]
SAI1_RXD1	Input	BOOT_CFG[1]
SAI1_RXD2	Input	BOOT_CFG[2]
SAI1_RXD3	Input	BOOT_CFG[3]
SAI1_RXD4	Input	BOOT_CFG[4]
SAI1_RXD5	Input	BOOT_CFG[5]
SAI1_RXD6	Input	BOOT_CFG[6]
SAI1_RXD7	Input	BOOT_CFG[7]
SAI1_TXD0	Input	BOOT_CFG[8]

Table continues on the next page...

Table 6-3. GPIO override contact assignments (continued)

Package pin	Direction on reset	eFuse
SAI1_TXD1	Input	BOOT_CFG[9]
SAI1_TXD2	Input	BOOT_CFG[10]
SAI1_TXD3	Input	BOOT_CFG[11]
SAI1_TXD4	Input	BOOT_CFG[12]
SAI1_TXD5	Input	BOOT_CFG[13]
SAI1_TXD6	Input	BOOT_CFG[14]
SAI1_TXD7	Input	BOOT_CFG[15]

The input pins provided are sampled at boot, and can be used to override the corresponding eFUSE values, depending on the setting of the BT_FUSE_SEL fuse.

6.1.3.3 Device Configuration Data (DCD)

The DCD is the configuration information contained in the program image (external to the ROM) that the ROM interprets to configure various on-chip peripherals. See Device Configuration Data (DCD) for more details on DCD.

6.1.4 Device initialization

This section describes the details of the ROM and provides the initialization details.

This includes details on:

- The ROM memory map
- The RAM memory map
- On-chip blocks that the ROM must use or change the POR register default values
- Clock initialization
- Enabling the cache
- Exception handling and interrupt handling

6.1.4.1 Internal ROM/RAM memory map

These figures show the iROM memory map:

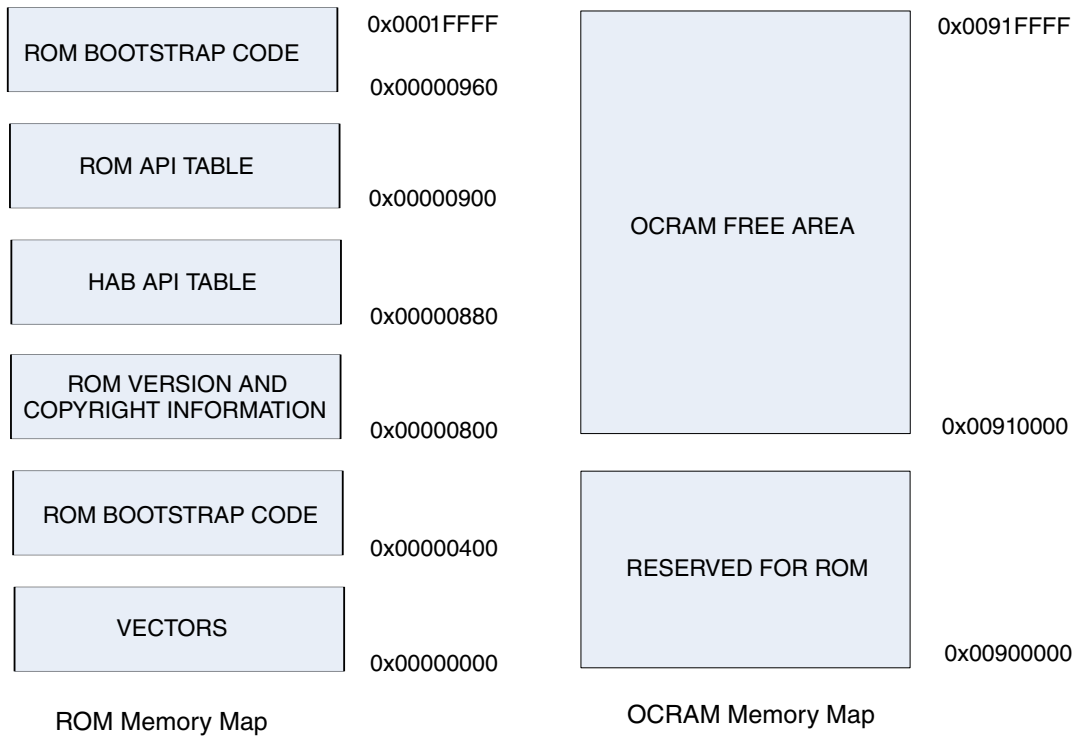


Figure 6-3. Internal ROM and RAM memory map

NOTE

The entire OCRAM region can be used freely after the boot.

6.1.4.2 Boot block activation

The boot ROM affects a number of different hardware blocks which are activated and play a vital role in the boot flow.

The ROM configures and uses the following blocks (listed in an alphabetical order) during the boot process. Note that the blocks actually used depend on the boot mode and the boot device selection:

- APBH—the DMA engine to drive the GPMI module
- BCH—62-bit error correction hardware engine with the AXI bus master and a private connection to the GPMI
- CCM—Clock Control Module
- ECSPI—Enhanced Configurable Serial Peripheral Interface
- FlexSPI—Flexible SPI Interface which supports serial NOR devices
- GPMI—NAND controller pin interface
- OCOTP_CTRL—On-Chip OTP Controller; the OCOTP contains the eFUSEs

- IOMUXC—I/O Multiplexer Control which allows the GPIO use to override the eFUSE boot settings;
- IOMUXC GPR—I/O Multiplexer Control General-Purpose Registers
- CAAM—Cryptographic Acceleration and Assurance Module
- SNVS—Secure Non-Volatile Storage
- SRC—System Reset Controller
- USB—used for the serial download of a boot device provisioning program
- USDHC—Ultra-Secure Digital Host Controller
- WDOG-1—Watchdog timer

6.1.4.3 Clocks at boot time

The table below show the various clocks and their sources used by the ROM.

After the reset, each Arm core has access to all peripherals. The ROM code disables the clocks listed in the following table, except for the boot devices listed in the second column.

Table 6-4. PLL setting by ROM

PLL name	Frequency	Comment
ARM_PLL	800 MHz	
SYS_PLL1	800 MHz	
SYS_PLL2	1000 MHz	
SYS_PLL3	1000 MHz	

NOTE

All other PLLs are in the default status.

Table 6-5. Clock root setting by ROM

Clock Name	Frequency (MHz)	Source	Enable
ARM_A53_ROOT	800	arm_pll_clk	Yes
ARM_M4_CLK_ROOT	200	system_pll2_200m_clk	
AHB_CLK_ROOT	133	system_pll1_133m_clk	Yes
MAIN_AXI_CLK_ROOT	333	system_pll2_333m_clk	Yes
VPU_A53_CLK_ROOT	800	arm_pll_clk	No
DRAM_ALT_CLK_ROOT	800	system_pll1_800m_clk	yes
NAND_CLK_ROOT	500	system_pll2_500m_clk	Enabled by driver
NAND_USDHC_BUS_CLK_ROOT	266	system_pll1_266m_clk	Enabled by driver
USB_BUS_CLK_ROOT		system_pll2_500m_clk	Enabled by driver

Table continues on the next page...

Table 6-5. Clock root setting by ROM (continued)

Clock Name	Frequency (MHz)	Source	Enable
NOC_CLK_ROOT	400	system_pll1_800m_clk	Yes
USDHC1_CLK_ROOT	200	system_pll1_400m_clk	Enabled by driver
USDHC2_CLK_ROOT	200	system_pll1_400m_clk	Enabled by driver
USB_PHY_REF_CLK_ROOT		system_pll1_100m_clk	Enabled by driver
ECSPI1_CLK_ROOT	50	system_pll2_200m_clk	No
ECSPI2_CLK_ROOT	50	system_pll2_200m_clk	No
ECSPI3_CLK_ROOT	50	system_pll2_200m_clk	No
WRCLK_CLK_ROOT		system_pll1_40m_clk	No

NOTE

All other clock roots are in the default status.

Table 6-6. NAND_CLK_ROOT setting

NAND data rate	NAND_CLK_ROOT source	Frequency
Async/Legacy NAND	system_pll1_400m_clk	25 MHz
Sync 40M	system_pll1_400m_clk	40 MHz
Toggle/Sync 66M	system_pll1_400m_clk	66 MHz
Toggle 80M	system_pll1_400m_clk	80 MHz
Sync 100M	system_pll1_400m_clk	100 MHz
Toggle/Sync 133M	system_pll1_400m_clk	133 MHz
Sync 160M	system_pll1_400m_clk	133 MHz
Toggle/Sync 200M	system_pll1_400m_clk	200 MHz

NOTE

The NAND_CLK_ROOT source depends on the NAND data rate.

The ROM code disables the clocks listed in the following table, except for the boot devices listed in the "Enabled for boot device" column below.

Table 6-7. CCGR setting by ROM

CCGR Register	LPCG Enable	Enabled for boot device
CCM_CCGR0	Dvfs	
CCM_CCGR1	Anamix	
CCM_CCGR2	Cpu	
CCM_CCGR3	Csu	Security related
CCM_CCGR4	debug	
CCM_CCGR5	Dram1	

Table continues on the next page...

Table 6-7. CCGR setting by ROM (continued)

CCGR Register	LPCG Enable	Enabled for boot device
CCM_CCGR6	reserved	
CCM_CCGR7	Ecspi1	
CCM_CCGR8	Ecspi2	
CCM_CCGR9	Ecspi3	
CCM_CCGR10	Enet1	
CCM_CCGR11	Gpio1	Never gate GPIO clock. uSDHC and test mode use GPIO.
CCM_CCGR12	Gpio2	
CCM_CCGR13	Gpio3	
CCM_CCGR14	Gpio4	
CCM_CCGR15	Gpio5	
CCM_CCGR16	Gpt1	Used by ROM as tick. Keep no changed so it is 25MHz.
CCM_CCGR17	Gpt2	Can be used in DCD. Keep no changed so it is 25MHz.
CCM_CCGR18	Gpt3	
CCM_CCGR19	Gpt4	
CCM_CCGR20	Gpt5	
CCM_CCGR21	Gpt6	
CCM_CCGR22	Hs	
CCM_CCGR23	I2c1	No I2C to be enabled
CCM_CCGR24	I2c2	
CCM_CCGR25	I2c3	
CCM_CCGR26	I2c4	
CCM_CCGR27	Iomux	
CCM_CCGR28	Iomux1	
CCM_CCGR29	Iomux2	
CCM_CCGR30	Iomux3	
CCM_CCGR31	Iomux4	
CCM_CCGR32	SNVSMIX_IPG_CLK	
CCM_CCGR33	Mu	
CCM_CCGR34	Ocotp	
CCM_CCGR35	Ocram	
CCM_CCGR36	Ocram_s	
CCM_CCGR37	Pcie	
CCM_CCGR38	Perfmon1	
CCM_CCGR39	Perfmon2	
CCM_CCGR40	Pwm1	
CCM_CCGR41	Pwm2	
CCM_CCGR42	Pwm3	

Table continues on the next page...

Table 6-7. CCGR setting by ROM (continued)

CCGR Register	LPCG Enable	Enabled for boot device
CCM_CCGR43	Pwm4	
CCM_CCGR44	Qos	
CCM_CCGR45	Dispmix	
CCM_CCGR46	Ethernet	
CCM_CCGR47	Qspi	
CCM_CCGR48	Rawnand	Will be gated on if boot from NAND is issued.
CCM_CCGR49	Rdc	Never used by ROM. Gate it off.
CCM_CCGR50	Rom	
CCM_CCGR51	Sai1	
CCM_CCGR52	Sai2	
CCM_CCGR53	Sai3	
CCM_CCGR54	Sai4	
CCM_CCGR55	Sai5	
CCM_CCGR56	Sai6	
CCM_CCGR57	Sctr	System counter. Do not gate off.
CCM_CCGR58	Sdma1	Not used by ROM. Gate it off.
CCM_CCGR59	Sdma2	
CCM_CCGR60	Sec_debug	
CCM_CCGR61	Sema1	Not used by ROM. Gate it off.
CCM_CCGR62	Sema2	
CCM_CCGR63	Sim_display	Bus clock. Do not gate off.
CCM_CCGR64	Sim_enet	Bus clock. Do not gate off.
CCM_CCGR65	Sim_m	Bus clock. Do not gate off.
CCM_CCGR66	Sim_main	Bus clock. Do not gate off.
CCM_CCGR67	Sim_s	Bus clock. Do not gate off.
CCM_CCGR68	Sim_wakeup	Bus clock. Do not gate off.
CCM_CCGR69	Sim_usb	Bus clock. Do not gate off.
CCM_CCGR70	Sim_vpu	Bus clock. Do not gate off.
CCM_CCGR71	Snvs	Secure Non-Volatile Storage
CCM_CCGR72	Trace	
CCM_CCGR73	Uart1	
CCM_CCGR74	Uart2	
CCM_CCGR75	Uart3	
CCM_CCGR76	Uart4	
CCM_CCGR77	Usb_ctrl1	Used by ROM USB driver, to enable USB clock
CCM_CCGR78	Reserved	
CCM_CCGR79	GPU3D	
CCM_CCGR80	Reserved	

Table continues on the next page...

Table 6-7. CCGR setting by ROM (continued)

CCGR Register	LPCG Enable	Enabled for boot device
CCM_CCGR81	Usdhc1	Used by USDHC driver
CCM_CCGR82	Usdhc2	Used by USDHC driver
CCM_CCGR83	Wdog1	WDOG1 used by ROM.
CCM_CCGR84	Wdog2	
CCM_CCGR85	Wdog3	
CCM_CCGR86	VPUG1	
CCM_CCGR87	Gpu	
CCM_CCGR88	Reserved	
CCM_CCGR89	VPUH1	
CCM_CCGR90	VPUG2	
CCM_CCGR91	PDM	
CCM_CCGR92	Gic	Leave on for Software.
CCM_CCGR93	Display	
CCM_CCGR94	USDHC3	
CCM_CCGR95	SDMA3	
CCM_CCGR96	Xtal	
CCM_CCGR97	Pll	
CCM_CCGR98	Tsensor	
CCM_CCGR99	Vpu_dec	
CCM_CCGR100	Reserved	
CCM_CCGR101	Reserved	
CCM_CCGR102	GPU2D	

6.1.4.4 Enabling MMU and caches

The boot ROM includes a feature that enables the Memory Management Unit (MMU) and the caches to improve the boot speed.

The L1 instruction cache is enabled at the start of the image download. The L1 data cache, L2 cache, and MMU are enabled during the image authentication. When the HAB authentication completes, the ROM disables the L1 data cache, L2 cache, and MMU.

The L1 Instruction cache, L1 data cache, L2 cache, and MMU is controlled by eFuse. By default, these features are enabled.

Enabling the MMU when booting non-securely with SEC_CONFIG=Open and setting the CSF pointer in the Image Vector Table to NULL has no impact on the boot performance. With this configuration, it is recommended to blow the BT_MMU_DISABLE fuse.

6.1.4.5 Exception handling

The exception vectors located at the start of the ROM are used to map all the Arm exceptions (except the reset exception) to a duplicate exception vector table in the internal RAM.

During the boot phase of CPU0, the RAM vectors point to the serial downloader in the ROM.

After the boot, the program image can overwrite the vectors as required. The code shown below is used to map the ROM exception vector table to the duplicate exception vector table in the RAM.

Mapping ROM Exception Vector Table

```
;; Define linker area for ROM exception vector table
AREA IROM_VECTORS, CODE, READONLY
LDR     PC, Reset_Addr
LDR     PC, Undefined_Addr
LDR     PC, SWI_Addr
LDR     PC, Prefetch_Addr
LDR     PC, Abort_Addr
NOP                         ; Reserved vector
LDR     PC, IRQ_Addr
LDR     PC, FIQ_Addr

;; Define exception vector table
Reset_Addr      DCD     start_address
Undefined_Addr  DCD     iRAM_Undefined_Handler
SWI_Addr        DCD     iRAM_SWI_Handler
Prefetch_Addr   DCD     iRAM_Prefetch_Handler
Abort_Addr      DCD     iRAM_Abort_Handler
                DCD     0               ; Reserved vector
IRQ_Addr        DCD     iRAM_IRQ_Handler
FIQ_Addr        DCD     iRAM_FIQ_Handler

start_address DCD start ;reset handler vector
```

6.1.4.6 Interrupt handling during boot

No special interrupt-handling routines are required during the boot process. The interrupts are disabled during the boot ROM execution and may be enabled in a later boot stage.

6.1.4.7 Persistent bits

Some modes of the boot ROM require the registers that keep their values after a warm reset. The SRC General-Purpose registers are used for this purpose.

See this table for persistent bits list and description:

Table 6-8. Persistent bits

Bit name	Bit location	Description
PERSIST_SECONDARY_BOOT	SRC_GPR10[30]	This bit identifies which image must be used—primary and secondary. Used only for the boot modes that support redundant boot.
PERSIST_BLOCK_REWRITE	SRC_GPR10[29]	This bit is used as a warning. It identifies that there are errors in the NAND blocks that hold the application image. See NAND flash for more details.
PERSISTENT_ENTRY0[31:0]	SRC_GPR1[31:0]	Holds the entry function for the CPU0 to wake up from the low-power mode.
PERSISTENT_ARG0[31:0]	SRC_GPR2[31:0]	Holds the argument of entry function for the CPU0 to wake up from the low-power mode.
PERSISTENT_ENTRY1[31:0]	SRC_GPR3[31:0]	Holds the entry function for the CPU1 to wake up from the low-power mode.
PERSISTENT_ARG1[31:0]	SRC_GPR4[31:0]	Holds the argument of the entry function for the CPU1 to wake up from the low-power mode.

6.1.5 Boot devices (internal boot)

The chip supports these boot flash devices:

- Serial NOR flash via FlexSPI interface
- Raw NAND (MLC and SLC), and Toggle-mode NAND flash through GPMI-2 interface, located at CS0. Page sizes of 2 KB, 4 KB, and 8 KB. The bus widths of 8-bit with 2 through 62-bit BCH hardware ECC (Error Correction) are supported.
- SD/MMC/eSD/SDXC/eMMC4.4 via USDHC interface, supporting high capacity cards.
- EEPROM boot via SPI (serial flash).
- EEPROM boot via SPI (serial flash)

The selection of the external boot device type is controlled by the BOOT_CFG eFUSEs. See this table for more details:

Table 6-9. Boot device selection

BOOT_CFG[14:12]	Boot device
001	SD/eSD
010	MMC/eMMC
011	NAND
100	Serial NOR boot via FlexSPI
110	Serial (SPI) NOR

6.1.5.1 Serial NOR Flash Boot via FlexSPI

6.1.5.1.1 Serial NOR eFUSE Configuration

Table 6-10. Fuse definition for Serial NOR over FlexSPI

Fuse Config	Config	Definitions	GPIO	Shipped Value	Settings
BOOT_CFG[3:0]	OEM	xSPI FLASH Dummy Cycle	Yes	0	0 – Dummy cycles is auto-probed Others – Actual dummy cycles for Read command
BOOT_CFG[5:4]	OEM	xSPI FLASH Auto Probe Type	Yes	0	0 – QuadSPI NOR 1 – MXIC Octal 2 – Micron Octal 3 – Adesto Octal
BOOT_CFG[7:6]	OEM	Hold time before read from device	Yes	0	0 – 500us 1 – 1ms 2 – 3ms 3 – 10ms
BOOT_CFG[10:8]	OEM	Flash Type	Yes	0	000b–Device supports 3B read by default 001b–Device supports 4B read by default 010b–HyperFlash 1V8 011b–HyperFlash 3V3 100b–MXIC Octal DDR
BOOT_CFG[11]	OEM	xSPI FLASH Auto Probe	Yes	0	0 – Disabled 1 – Enabled
BOOT_CFG[14:12]	OEM	Boot device selection	Yes	0	100 – Serial NOR device is selected as boot device.
0x480[2:0] (BOOT_CFG_PARAMETER)	OEM	xSPI FLASH Frequency	No	0	0 - 100 MHz 1 - 133 MHz 2 - 166 MHz 3 - 200 MHz 4 - 80 MHz 5 - 20 MHz Others – Reserved

NOTE

If the xSPI FLASH Auto Probe feature is enabled, the following is the logic how this feature works with other fuse combinations:

- Flash Type - If Flash type is 0, the "xSPI FLASH Auto Probe Type" takes effect for the Flash type selection. If Flash Type is greater than 1, the "Flash Type" Fuse is used for Flash type selection, ROM will issue specific command to probe the presence of Serial NOR FLASH.
- xSPI FLASH Frequency - This field is used for specifying the Flash working frequency.

6.1.5.1.2 FlexSPI Serial NOR Flash Boot Operation

The Boot ROM attempts to boot from Serial NOR flash if the BOOT_CFG [7:6] fuses are programmed to 0b’00 as shown in the Serial NOR eFUSE Configuration table, then the ROM will initialize FlexSPI1 interface. FlexSPI interface initialization is a two-step process.

The ROM expects the 512-byte FlexSPI NOR configuration parameters as explained in next section to be present at offset 0 in Serial NOR flash. The ROM reads these configuration parameters using the read command specified by BOOT_CFG [5:3] with Serial clock operating at 30 MHz.

In the second step, ROM configures FlexSPI1 interface with the parameters provided in configuration block read from Serial NOR flash and starts the boot procedure. Refer to Table 25-14 for details regarding FlexSPI configuration parameters and to the FlexSPI NOR boot flow chart for detailed boot flow chart of FlexSPI NOR.

Both booting an XIP and non XIP image are supported from Serial NOR Flash. For XIP boot, the image has to be built for FlexSPI address space and for non XIP the image can be built to execute from Internal RAM.

6.1.5.1.3 FlexSPI NOR boot flow chart

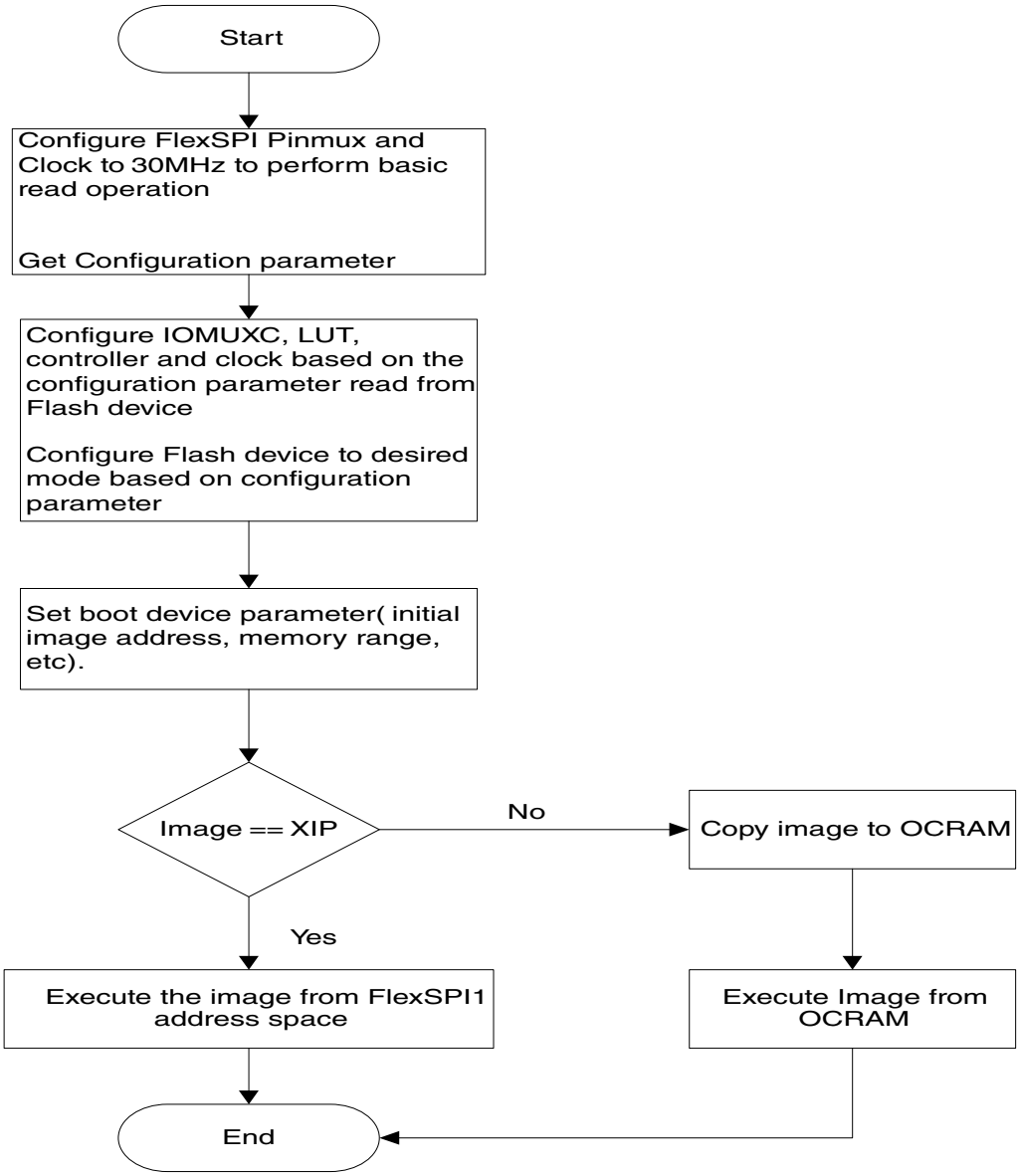


Figure 6-4. FlexSPI NOR boot flow

6.1.5.2 Serial NOR configuration based on FlexSPI interface

The ROM SW supports Serial NOR based on FlexSPI module, using a 448-bytes common FlexSPI configuration block and several specified parameters for Serial NOR respectively. See below sections for more details.

6.1.5.2.1 FlexSPI Configuration Block

FlexSPI Configuration block consists of parameters regarding specific Flash devices including read command sequence, quad mode enablement sequence (optional), etc.

Table 6-11. FlexSPI Configuration block

Name	Offset	Size(bytes)	Description
Tag	0x000	4	0x42464346, ascii:" FCFB"
Version	0x004	4	0x56010000 [07:00] bugfix = 0 [15:08] minor = 0 [23:16] major = 1 [31:24] ascii 'V'
-	0x008	4	Reserved
readSampleClkSrc	0x00C	1	0 – internal loopback 1 – loopback from DQS pad 3 – Flash provided DQS
dataHoldTime	0x00D	1	Serial Flash CS Hold Time Recommend default value is 0x03
dataSetupTime	0x00E	1	Serial Flash CS setup time Recommended default value is 0x03
columnAdressWidth	0x00F	1	3 – For HyperFlash 12/13 – For Serial NAND, see datasheet to find correct value 0 – Other devices
deviceModeCfgEnable	0x010	1	Device Mode Configuration Enable feature 0 – Disabled 1 – Enabled
-	0x011	3	Reserved
deviceModeSeq	0x014	4	Sequence parameter for device mode configuration
deviceModeArg	0x018	4	Device Mode argument, effective only when deviceModeCfgEnable = 1
configCmdEnable	0x01C	1	Config Command Enable feature 0 – Disabled 1 – Enabled
-	0x01D	3	Reserved
configCmdSeqs	0x020	16	Sequences for Config Command, allow 4 separate

Table continues on the next page...

Table 6-11. FlexSPI Configuration block (continued)

Name	Offset	Size(bytes)	Description
			configuration command sequences.
cfgCmdArgs	0x030	16	Arguments for each separate configuration command sequence.
controllerMiscOption	0x040	4	Bit0 – differential clock enable Bit1 – CK2 enable, must set to 0 in this silicon Bit2 – ParallelModeEnable, must set to 0 for this silicon Bit3 – wordAddressableEnable Bit4 – Safe Configuration Frequency enable set to 1 for the devices that support DDR Read instructions Bit5 – Pad Setting Override Enable Bit6 – DDR Mode Enable, set to 1 for device supports DDR read command
deviceType	0x044	1	1 – Serial NOR 2 – Serial NAND
sflashPadType	0x045	1	1 – Single pad 2 – Dual pads 4 – Quad pads 8 – Octal pads
serialClkFreq	0x046	1	Chip specific value, for this silicon 1 – 30 MHz 2 – 50 MHz 3 – 60 MHz 4 – 75 MHz 5 – 80 MHz 6 – 100 MHz 7 – 133 MHz 8 – 166 MHz Other value: 30 MHz
lutCustomSeqEnable	0x047	1	0 – Use pre-defined LUT sequence index and number 1 - Use LUT sequence parameters provided in this block

Table continues on the next page...

Table 6-11. FlexSPI Configuration block (continued)

Name	Offset	Size(bytes)	Description
-	0x048	8	Reserved
sflashA1Size	0x050	4	For SPI NOR, need to fill with actual size For SPI NAND, need to fill with actual size * 2
sflashA2Size	0x054	4	The same as above
sflashB1Size	0x058	4	The same as above
sflashB2Size	0x05C	4	The same as above
csPadSettingOverride	0x060	4	Set to 0 if it is not supported
sclkPadSettingOverride	0x064	4	Set to 0 if it is not supported
dataPadSettingOverride	0x068	4	Set to 0 if it is not supported
dqsPadSettingOverride	0x06C	4	Set to 0 if it is not supported
timeoutInMs	0x070	0	Maximum wait time during read busy status 0 – Disabled timeout checking feature Other value – Timeout if the wait time exceeds this value.
commandInterval	0x074	4	Unit: ns Currently, it is used for SPI NAND only at high frequency
dataValidTime	0x078	4	Time from clock edge to data valid edge, unit ns. This field is used when the FlexSPI Root clock is less than 100 MHz and the read sample clock source is device provided DQS signal without CK2 support. [31:16] data valid time for DLLB in terms of 0.1 ns [15:0] data valid time for DLLA in terms of 0.1 ns
busyOffset	0x07C	2	busy bit offset, valid range : 0-31
busyBitPolarity	0x07E	2	0 – busy bit is 1 if device is busy 1 – busy bit is 0 if device is busy
lookupTable	0x080	256	Lookup table
lutCustomSeq	0x180	48	Customized LUT sequence, see below table for details.
	0x1B0	16	Reserved for future use

Note:

1. To customize the LUT sequence for some specific device, users need to enable “lutCustomSeqEnable” and fill in corresponding “lutCustomSeq” field specified by command index below.
2. For Serial (SPI) NOR, the pre-defined LUT index is as follows:

Table 6-12. LUT sequence definition for Serial NOR

Command Index	Name	Index in lookup table	Description
0	Read	0	Read command Sequence
1	ReadStatus	1	Read Status command
2	WriteEnable	3	Write Enable command sequence
3	EraseSector	5	Erase Sector Command
4	PageProgram	9	Page Program Command
5	ChipErase	11	Full Chip Erase
6	Dummy	15	Dummy Command as needed
	Reserved	2,4,6,7,8,10,12,13,14	All reserved indexes can be freely used for other purpose

6.1.5.2.2 Serial NOR configuration block (512 bytes)

Table 6-13. Serial NOR configuration block

Name	Offset	Size (Bytes)	Description
memCfg	0	448	The common memory configuration block, see FlexSPI configuration block for more details
pageSize	0x1C0	4	Page size in terms of bytes, not used by ROM
sectorSize	0x1C4	4	Sector size in terms of bytes, not used by ROM
ipCmdSerialClkFreq	0x1C8	4	Chip specific value, not used by ROM For Ultra 0 – No change, keep current serial clock unchanged 1 – 30 MHz 2 – 50 MHz 3 – 60 MHz 4 – 75 MHz

Table continues on the next page...

Table 6-13. Serial NOR configuration block (continued)

Name	Offset	Size (Bytes)	Description
			5 – 80 MHz 6 – 100 MHz 7 – 133 MHz 8 – 166 MHz
Reserved	0x1CC	52	Reserved for future use

6.1.5.3 NAND flash

The boot ROM supports a number of MLC/SLC NAND flash devices from different vendors and LBA NAND flash devices. The Error Correction and Control (ECC) subblock (BCH) is used to detect the errors.

6.1.5.3.1 NAND eFUSE configuration

The boot ROM determines the configuration of the external NAND flash by parameters, either provided by the eFUSE, or sampled on the GPIO pins during boot. See Table 6-14 for parameters details.

NOTE

BOOT_CFGx sampled on the GPIO pins depends on the BT_FUSE_SEL setting. See Boot Fusemap for details.

Table 6-14. NAND boot eFUSE descriptions

Fuse	Config	Definition	GPIO[1]	Shipped value	Settings
BOOT_CFG[15:12]	OEM	Boot device selection	Yes	0	0011—boot from the NAND interface
BOOT_CFG[7]	OEM	BT_TOGGLEMODE	Yes	0	0—raw NAND 1—toggle mode NAND
BOOT_CFG[11:10]	OEM	Pages in block	Yes	0	00—128 01—64 10—32 11—256
BOOT_CFG[9:8]	OEM	Row address cycles	Yes	00	00—3 01—2 10—4 11—5

Table continues on the next page...

Table 6-14. NAND boot eFUSE descriptions (continued)

Fuse	Config	Definition	GPIO[1]	Shipped value	Settings
BOOT_CFG[4:1]	OEM	Toggle mode 33 MHz preamble delay, read latency	Yes	000	0000—16 GPMICLK cycles 0001—1 GPMICLK cycles 0010—2 GPMICLK cycles 0011—3 GPMICLK cycles 0100—4 GPMICLK cycles 0101—5 GPMICLK cycles 0110—6 GPMICLK cycles 0111—7 GPMICLK cycles 1000—8 GPMICLK cycles 1001—9 GPMICLK cycles 1010—10 GPMICLK cycles 1011—11 GPMICLK cycles 1100—12 GPMICLK cycles 1101—13 GPMICLK cycles 1110—14 GPMICLK cycles 1111—15 GPMICLK cycles
BOOT_CFG[6:5]	OEM	Boot search count	Yes	00	00—2 01—2 10—4 11—8
0x4B0[7]	OEM	Override pad settings	No	0	Override the NAND pad settings 0—use the default values 1—use the PAD_SETTINGS value
0x4A0[31:24]	OEM	PAD_SETTINGS[7:0]	No	0	NAND pad settings value
0x4B0[15:12]	OEM	READ_RETRY_SEQ_ID[3:0]	No	0000	0000—don't use the ROM embedded read-retry sequence 0001—use Micron 20 nm read-retry sequence 0010—use Toshiba A19nm read-retry sequence 0011—use Toshiba 19nm read-retry sequence 0100—use SanDisk 19nm read-retry sequence 0101—use SanDisk 1ynm read-retry sequence 0110 to 1111—reserved

1. The setting can be overridden by the GPIO settings when the BT_FUSE_SEL fuse is intact. See Table 1 for the corresponding GPIO pin.

6.1.5.3.2 NAND flash boot flow and Boot Control Blocks (BCB)

There are two BCB data structures:

- FCB
- DBBT

As a part of the NAND media initialization, the ROM driver uses safe NAND timings to search for the Firmware Configuration Block (FCB) that contains the optimum NAND timings, the page address of the Discovered Bad Block Table (DBBT) Search Area, and the start page address of the primary and secondary firmware.

The hardware ECC level to use is embedded inside the FCB block. The FCB data structure is also protected using the ECC. The driver reads raw 2112 bytes of the first sector and runs through the software ECC engine that determines whether the FCB data is valid or not.

If the FCB is found, the optimum NAND timings are loaded for further reads. If the ECC fails, or the fingerprints do not match, the Block Search state machine increments the page number to the Search Stride number of pages to read for the next BCB until the SearchCount pages have been read.

If the search fails to find a valid FCB, the NAND driver responds with an error and the boot ROM enters the serial download mode.

The FCB contains the page address of the DBBT Search Area, and the page address for primary and secondary boot images. The DBBT is searched in the DBBT Search Area, just like the FCB is searched. After the FCB is read, the DBBT is loaded, and the primary or secondary boot image is loaded using the starting page address from the FCB.

This figure shows the state diagram of the FCB search:

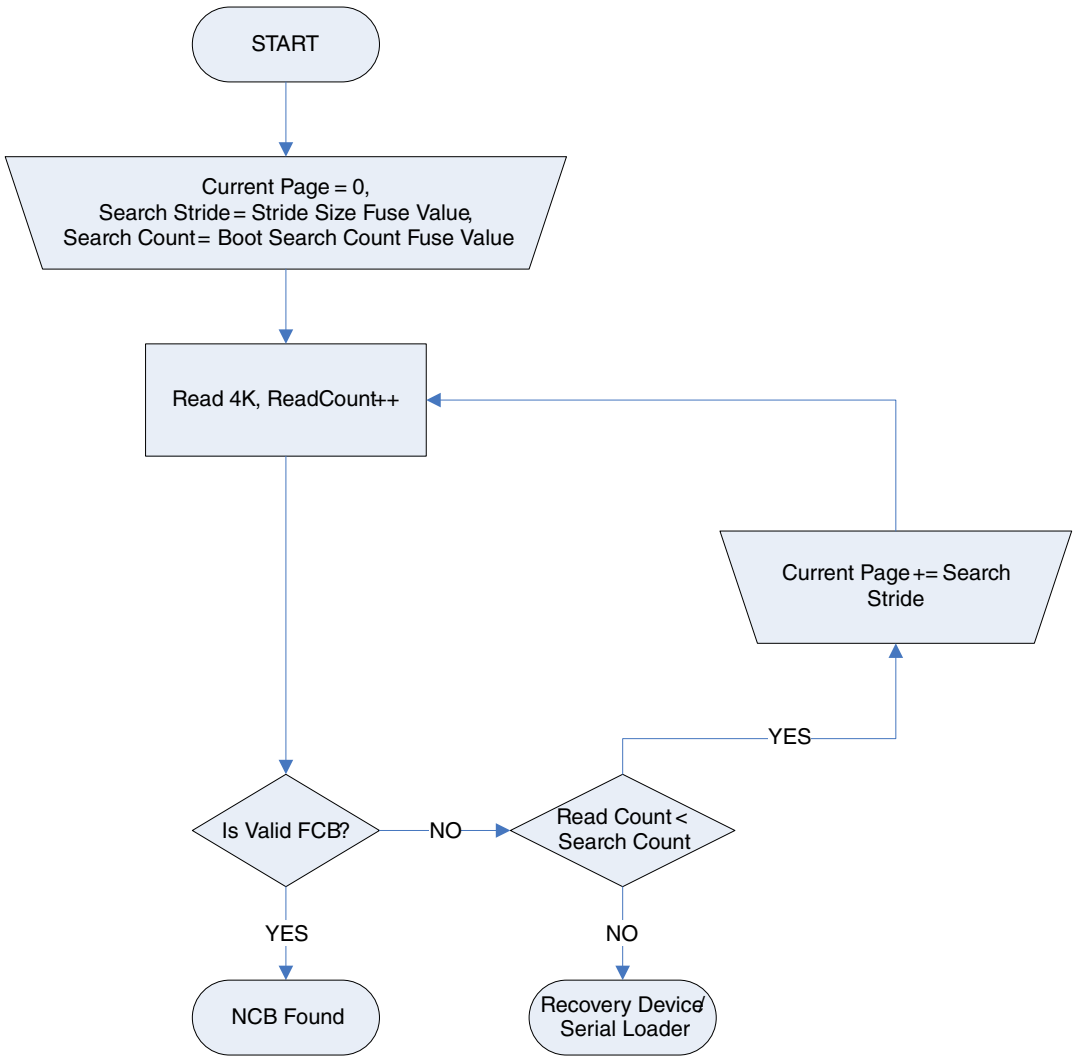


Figure 6-5. FCB search flow

When the FCB is found, the boot ROM searches for the Discovered Bad Blocks Table (DBBT). If the DBBT Search Area is 0 in the FCB, the ROM assumes that there are no bad blocks on the NAND device boot area. See this figure for the DBBT search flow:

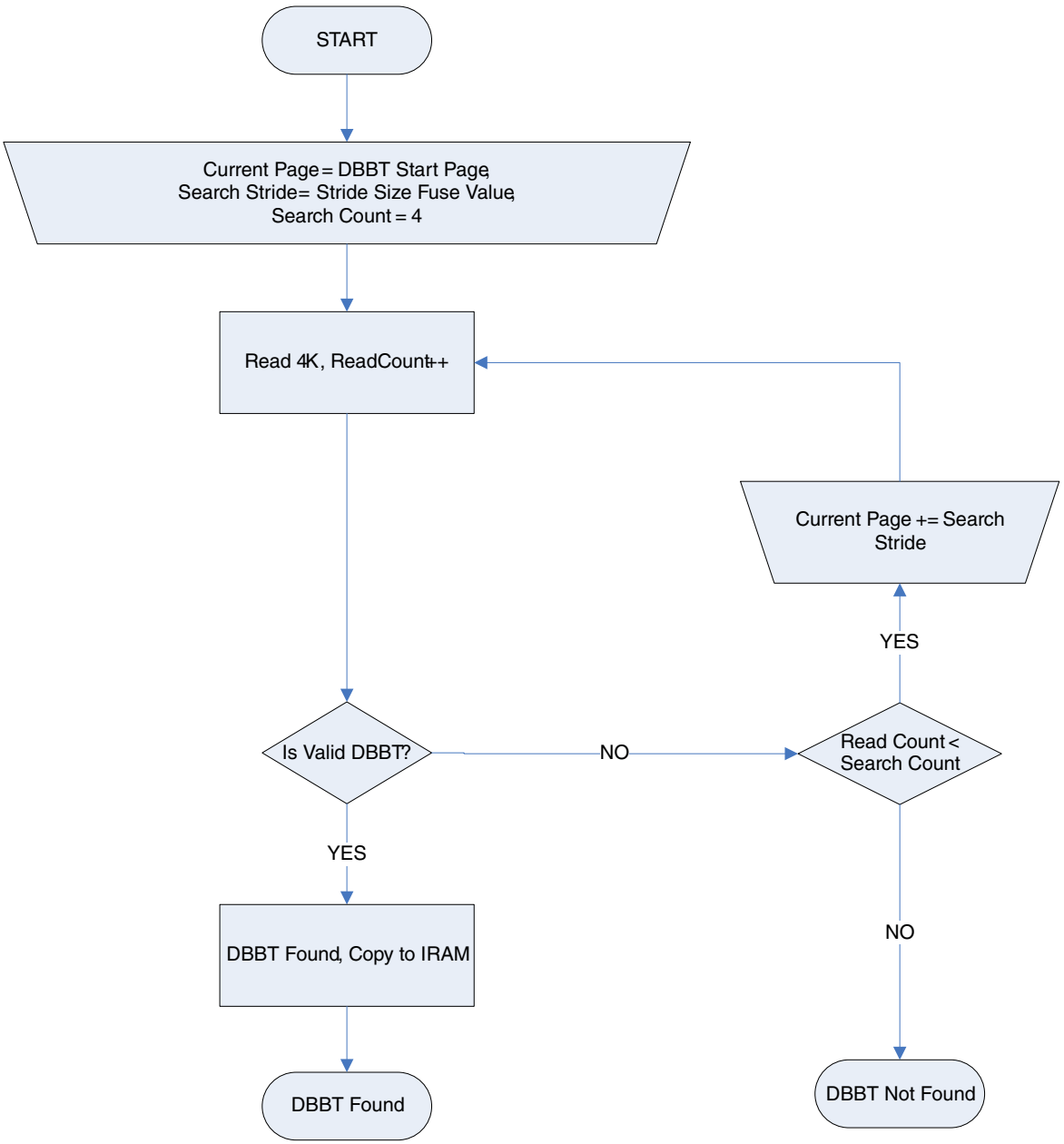


Figure 6-6. DBBT search flow

The BCB search and load function also monitors the ECC correction threshold and sets the PERSIST_BLOCK_REWRITE persistent bit if the threshold exceeds the maximum ECC correction ability.

If there is a page with a number of errors higher than ECC can correct during the primary image read, the boot ROM turns on the PERSIST_SECONDARY_BOOT bit and performs the software reset (After the software reset, the secondary image is used).

If there is a page with number of errors higher than ECC can correct during secondary image read, the boot ROM goes to the serial loader.

6.1.5.3.3 Firmware configuration block

The FCB is the first sector in the first good block. The FCB must be present at each search stride of the search area.

The search area contains copies of the FCB at each stride distance, so, in case the first NAND block becomes corrupted, the ROM finds its copy in the next NAND block. The search area must span over at least two NAND blocks. The location information for the DBBT search area, FW1, and FW2 are all specified in the FCB. This table shows the flash control block structure:

Table 6-15. Flash control block structure

Name	Start byte	Size in bytes	Description
Reserved	0	4	Reserved for Fingerprint #1(Checksum)
FingerPrint	4	4	32-bit word with a value of 0x20424346, in ascii "FCB"
Version	8	4	32-bit version number; this version of FCB is 0x00000001
m_NANDTiming	12	8	8 B of data for eight NAND timing parameters from the NAND datasheet. The eight parameters are: m_NandTiming[0]=data_setup, m_NandTiming[1]=data_hold, m_NandTiming[2]=address_setup, m_NandTiming[3]=dsample_time, m_NandTiming[4]=nand_timing_state, m_NandTiming[5]=REA, m_NandTiming[6]=RLOH, m_NandTiming[7]=RHOH. The ROM only uses the first four parameters, but the FCB provides space for other four parameters to be used by the bootloader or other applications.
PageDataSize	20	4	The number of bytes of data in a page. Typically, this is 2048 bytes for 2112 bytes page size or 4096 bytes for 4314/4224 bytes page size or 8192 for 8568 bytes page size.
TotalPageSize	24	4	The total number of bytes in a page. Typically, 2112 for 2-KB page or 4224 or 4314 for 4-KB page or 8568 for 8-KB page.
SectorsPerBlock	28	4	The number of pages per block. Typically 64 or 128 or depending on the NAND device type.

Table continues on the next page...

Table 6-15. Flash control block structure (continued)

Name	Start byte	Size in bytes	Description
NumberOfNANDs	32	4	Not used by ROM
TotalInternalDie	36	4	Not used by ROM
CellType	40	4	Not used by ROM
EccBlockNEccType	44	4	Value from 0 to is used to set the BCH Error Corrrection level 0, 2, 4, .. or 62 for Block BN of ECC page, used in configuring the BCH62 page layout registers.
EccBlock0Size	48	4	Size of block B0 used in configuring the BCH62 page-layout registers.
EccBlockNSize	52	4	Size of block BN used in configuring the BCH62 page-layout registers.
EccBlock0EccType	56	4	Value from 0 to used to set the BCH Error Corrrection level 0, 2, 4, .. or 62 for Block BN of ECC page, used in configuring the BCH62 page layout registers.
MetadataBytes	60	4	Size of metadata bytes used in configuring the BCH62 page-layout registers.
NumEccBlocksPerPage	64	4	Number of the ECC blocks BN not including B0. This value is used in configuring the BCH62 page-layout registers.
EccBlockNEccLevelSDK	68	4	Not used by ROM
EccBlock0SizeSDK	72	4	Not used by ROM
EccBlockNSizeSDK	76	4	Not used by ROM
EccBlock0EccLevelSDK	80	4	Not used by ROM
NumEccBlocksPerPageSDK	84	4	Not used by ROM
MetadataBytesSDK	88	4	Not used by ROM
EraseThreshold	92	4	Not used by ROM
Firmware1_startingPage	104	4	Page number address where the first copy of bootable firmware is located.
Firmware2_startingPage	108	4	Page number address where the second copy of bootable firmware is located.
PagesInFirmware1	112	4	Size of the first copy of firmware in pages.
PagesInFirmware2	116	4	Size of the second copy of firmware in pages.
DBBTSearchAreaStartAddress	120	4	Page address for the bad block table search area.
BadBlockMarkerByte	124	4	This is an input offset in the BCH page for the ROM to swap with the first byte of metadata after reading a page using the BCH62. The ROM supports the restoration of manufacturer-marked bad block markers in the page and this offset is the bad block marker offset location.
BadBlockMarkerStartBit	128	4	This is an input bit offset in the BadBlockMarkerByte for the ROM to use when swapping eight bits with the first byte of metadata.

Table continues on the next page...

Table 6-15. Flash control block structure (continued)

Name	Start byte	Size in bytes	Description
BBMarkerPhysicalOffset	132	4	This is the offset where the manufacturer leaves the bad block marker on a page.
BCHType	136	4	0 for BCH20 and 1 for BCH62. The chip is backwards compatible to BCH20 and this field tells the ROM to use the BCH20 or BCH62 block.
TMTiming2_ReadLatency	140	4	Toggle mode NAND timing parameter read latency, the ROM uses this value to configure the timing2 register of the GPMI.
TMTiming2_PreambleDelay	144	4	Toggle mode NAND timing parameter Preamble Delay. The ROM uses this value to configure the timing2 register of the GPMI.
TMTiming2_CEDelay	148	4	Toggle mode NAND timing parameter CE Delay. The ROM uses this value to configure the timing2 register of the GPMI.
TMTiming2_PostambleDelay	152	4	Toggle mode NAND timing parameter Postamble Delay. The ROM uses this value to configure the timing2 register of the GPMI.
TMTiming2_CmdAddPause	156	4	Toggle mode NAND timing parameter Cmd Add Pause. The ROM uses this value to configure the timing2 register of the GPMI.
TMTiming2_DataPause	160	4	Toggle mode NAND timing parameter Data Pause. The ROM uses this value to configure the timing2 register of the GPMI.
TMSpeed	164	4	This is the toggle mode speed for the ROM to configure the gpmi clock. 0 for 33 MHz, 1 for 40 MHz, and 2 for 66 MHz.
TMTiming1_BusyTimeout	168	4	Toggle mode NAND timing parameter Busy Timeout. The ROM uses this value to configure the timing1 register of the GPMI.
DISBBM	172	4	If 0, the ROM swaps the BadBlockMarkerByte with metadata[0] after reading a page using the BCH62. If the value is 1, the ROM does not swap.
BBMark_spare_offset	176	4	The offset in the metadata place which stores the data in the bad block marker place.
Onfi_sync_enable	180	4	Enable the Onfi nand sync mode support.
Onfi_sync_speed	184	4	Speed for the Onfi nand sync mode: 0 - 24 MHz, 1 - 33 MHz, 2 - 40 MHz, 3 - 50 MHz, 4 - 66 MHz, 5 - 80 MHz, 6 - 100 MHz, 7 - 133 MHz, 8 - 160 MHz, 9 - 200 MHz
Onfi_syncNANDData	188	28	The parameters for the Onfi nand sync mode timing. They are read latency, ce_delay, preamble_delay, postamble_delay, cmdadd_pause, data_pause, and busy_timeout.
DISBB_Search	216	4	Disable the bad block search function when reading the firmware, only using DBBT.

The FCB data structure is protected using a 62-bit ECC. The layout of the FCB page is illustrated in this figure:

Figure 6-7. Layout of the FCB page

The detailed parameters of the FCB pages are listed in this table:

Table 6-16. Parameters setting for FCB page

Parameter	Value
TotalPageSize	2048+64=2112
MetadataBytes	32
EccBlock0Size	128
EccBlock0EccType	31
BCHType	0
EccBlockNSize	128
EccBlockNEccType	31
NumEccBlocksPerPage	7

To reduce the disturbances caused by a neighboring cell in the FCB page in the NAND chip, a randomizer is enabled when reading the FCB page. BCH ECC has a Randomizer module that is interfaced through the GPMI APBHDMA chain. The Randomizer can generate random data based on BCH ECC encoded/decoded data. It can be employed to reduce the disturbances caused by a neighboring cell in the NAND chip, thus reducing bit errors. The randomizer is used to reduce the bit errors in the FCB. Ensure that the randomizer is enabled when burning the FCB pages in the NAND flash. To control the randomizer for the pages (except for FCB), a new field called Randomizer_Enable is added into the FCB structure. If the Randomizer_Enable field is set to 0, the randomizer is disabled. Reading the pages (except for FCB) being set to a non-zero value enables the randomizer. For detailed randomizer information, see Randomizer.

6.1.5.3.4 Discovered Bad Block Table (DBBT)

See this table for the DBBT format:

Table 6-17. DBBT structure

Name	Start byte	Size in bytes	Description
reserved	0	4	-
FingerPrint	4	4	32-bit word with a value of 0x44424254,in ascii "DBBT"
Version	8	4	32-bit version number; this version of DBBT is 0x00000001
reserved	12	4	-
DBBT_NUM_OF_PAGES	16	4	Size of the DBBT in pages
reserved	20	4*PageSize-20	-
reserved	4*PageSize	4	-
Number of Entries	4*PageSize + 4	4	Number of bad blocks
Bad Block Number	4*PageSize + 8	4	First bad block number
Bad Block Number	4*PageSize + 12	4	Second bad block number
-	-	-	Next bad block number
-	-	-	-
Last bad block number	-	-	Last bad block number

6.1.5.3.5 Bad block handling in ROM

During the firmware boot, at the block boundary, the Bad Block table is searched for a match to the next block.

If no match is found, the next block can be loaded. If a match is found, the block must be skipped and the next block checked.

If the Bad Block table start page is null, check the manufactory made Bad Block marker. The location of the Bad Block maker is at the first three or last three pages in every block of the NAND flash. The NAND manufacturers normally use one byte in the spare area of certain pages within a block to mark that a block is bad or not. A value of 0xFF means good block, non-FF means bad block.

To preserve the BI (bad block information), the flash updater or gang programmer applications must swap the Bad Block Information (BI) data to byte 0 of the metadata area for every page before programming the NAND flash. When the ROM loads the firmware, it copies back the value at metadata[0] to the BI offset in the page data. This figure shows how the factory bad block marker is preserved:

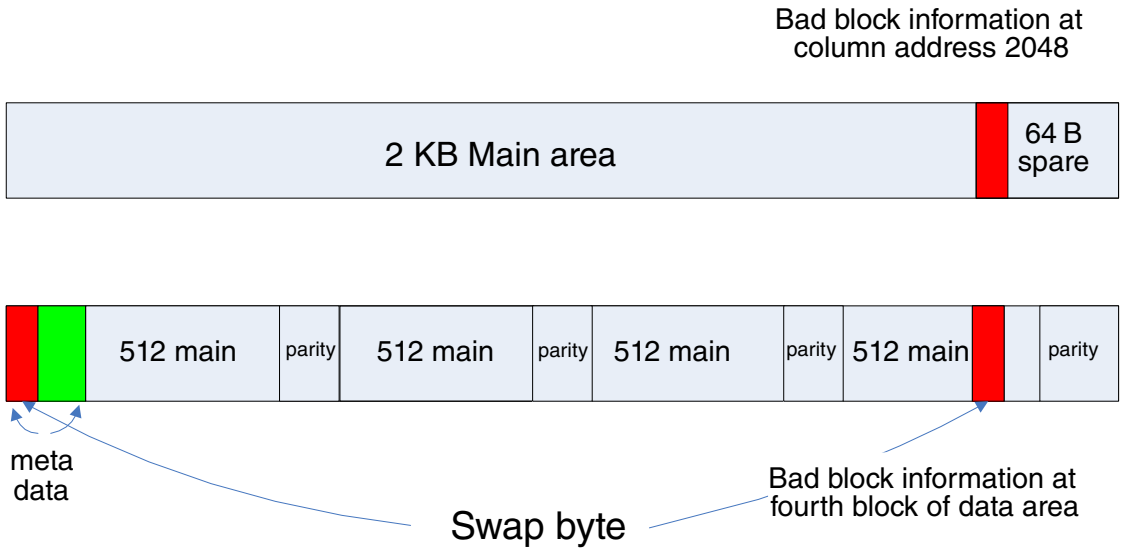


Figure 6-8. Factory bad block marker preservation

In the FCB structure, there are two elements (m_u32BadBlockMarkerByte and m_u32BadBlockMarkerStartBit) to indicate the byte and bit place in the page data that the manufacturer marked the bad block marker.

6.1.5.3.6 Toggle mode DDR NAND boot

If the BT_TOGGLEMODE efuse is blown, the ROM does the following to boot from the Samsung's toggle mode DDR NAND.

6.1.5.3.6.1 GPMI and BCH clocks configuration

The ROM sets the clock source and the dividers in the CCM registers.

If the BOOT_CFG is set (toggle mode), the GPMI/BCH CLK source is PLL2PFD4, and running at 66 MHz, otherwise the GPMI/ BCH CLK souce is PLL3, running at 24 MHz. The ROM sets the default values to timing0, timing1, and timing2 gpmi registers for 24 MHz clock speed. It uses the BOOT_CFG fuse to configure the GPMI timing2 register parameters preamble delay and read latency. The default value for these parameters is 2 when the fuses are not blown.

The default timing parameter values used by the ROM for the toggle-mode device are:

- Timing0.ADDRESS_SETUP = 5
- Timing0.DATA_SETUP = 10
- Timing0.DATA_HOLD = 10
- Timing1.DEVICE_BUSY_TIMEOUT = 0 x 500

- Timing2.READ_LATENCY = BOOT_CFG if blown, otherwise 2
- Timing2.CE_DELAY = 2
- Timing2.PREAMBLE_DELAY = BOOT_CFG if blown, otherwise 2
- Timing2.POSTAMBLE_DELAY = 3
- Timing2.CMDADD_PAUSE = 4
- Timing2.DATA_PAUSE = 6

The default timing parameters can be overriden by the TMTiming2_ReadLatency, TMTiming2_PreambleDelay, TMTiming2_CEDelay, TMTiming2_PostambleDelay, TMTiming2_CmdAddPause, and TMTiming2_DataPause parameters of the FCB.

6.1.5.3.6.2 Setup DMA for DDR transfers

In the DMA descriptors, the GPMI is configured to read the page data at a double data rate, the word length is set to 16, and the transfer count to a half of the page size.

6.1.5.3.6.3 Reconfigure timing and speed using values in FCB

After reading the FCB page with the GPMI set to default timings and a speed of 33 MHz, the ROM reconfigures the CCM dividers to run the gpmi/bch clks to a desired speed specified in the FCB for the rest of the boot process. The GPMI timing registers are also reconfigured to the values specified in the FCB.

The GPMI speed can be configured using the FCB parameter TMSpeed:

- 0—25 MHz
- 1—33 MHz
- 2—40 MHz
- 3—50 MHz
- 4—66 MHz
- 5—80 MHz
- 6—100 MHz
- 7—133 MHz
- 8—133 MHz
- 9—200 MHz

The GPMI timing0 register fields data_setup, data_hold, and address_setup are set to the values specified for the data_setup and data_hold and address_setup in the FCB member m_NANDTiming.

The GPMI timing1.DEVICE_BUSY_TIMEOUT is set to the value specified in the FCB member TMTiming1_BusyTimeout.

The GPMI timing2 register values are set using the FCB members TMTiming2.READ_LATENCY, CE_DELAY, PREAMBLE_DELAY, POSTAMBLE_DELAY, CMDADD_PAUSE, and DATA_PAUSE.

6.1.5.3.7 Typical NAND page organization

6.1.5.3.7.1 BCH ECC page organization

The first data block is called block 0 and the rest of the blocks are called block N. A separate ECC level scan is used for block 0 and block N.

The metadata bytes must be located at the beginning of a page, starting at byte 0, followed by the data block 0, the ECC bytes for data block 0, the block 1 and its ECC bytes, and so on, up until the N data blocks. The ECC level for the block 0 can be different from the ECC level for the rest of the blocks.

For the NAND boot with page-size restrictions and the data block size restricted to 512 B, only few combinations of the ECC for block 0 and block N are possible.

This figure shows the valid layout for 2112-byte sized page.

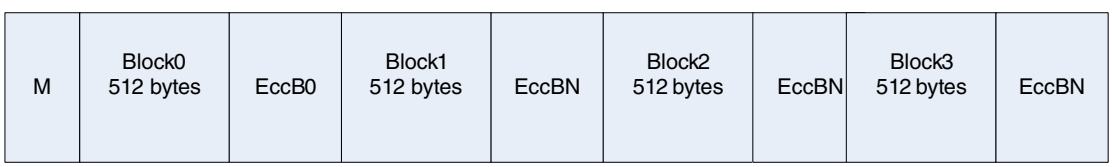


Figure 6-9. Valid layout for 2112-byte sized page

The example below is for 13 bits of parity (GF13). The number of ECC bits required for a data block is calculated using the (ECC_Correction_Level * 13) bits.

In the above layout, the ECC size for EccB0 and EccBN must be selected to not exceed a total page size of 2112 bytes. The EccB0 and EccBN can be one of the 2, 4, 6, 8, 10, 12, 14, 16, 18, and 20 bits on the ECC correction level. The total bytes are:

[M + (data_block_size x 4) + ([EccB0 + (EccBN x 3)] x 13) / 8] <= 2112;

M = metadata bytes and data_block_size is 512.

There are four data blocks of 512 bytes each in a page of 2-KB page sized NAND. The values of EccB0 and EccBN must be such that the above calculation does not result in a value greater than 2112 bytes.

M	Block0 512 bytes	EccB0	Block1 512 bytes	EccBN	Block2 512 bytes	EccBN	Block3 512 bytes	EccBN
	Block4 512 bytes	EccBN	Block5 512 bytes	EccBN	Block6 512 bytes	EccBN	Block7 512 bytes	EccBN

Figure 6-10. Valid layout for 4-KB sized page

Different NAND manufacturers have different sizes for a 4-KB page; 4314 bytes is typical.

[M + (data_block_size x 8) + ([EccB0 + (EccBN x 7)] x 13) / 8] <= 4314;

M= metadata bytes and data_block_size is 512.

There are eight data blocks of 512 bytes each in a page of a 4-KB page sized NAND. The values of the EccB0 and EccBN must be such that the above calculation does not result in a value greater than the size of a page in a 4-KB page NAND.

6.1.5.3.7.2 Metadata

The number of bytes used for the metadata is specified in the FCB. The metadata for the BCH encoded pages is placed at the beginning of a page. The ROM only cares about the first byte of metadata to swap it with a bad block marker byte in the page data after each page read; it is important to have at least one byte for the metadata bytes field in the FCB data structure.

6.1.5.3.8 IOMUX configuration for NAND

The following table shows the RawNAND IOMUX pin configuration.

Table 6-18. NAND IOMUX pin configuration

Signal	Pad name
NAND_CLE	SD3_CLK.alt1
NAND_ALE	SD3_CMD.alt1
NAND_WP_B	SAI1_MCLK.alt1
NAND_RE_B	SD3_STROBE.alt1
NAND_WE_B	SD3_RESET_B.alt1
NAND_READY_B	SAI1_TXD.alt1
NAND_DQS	SAI1_TXFS.alt1

Table continues on the next page...

Table 6-18. NAND IOMUX pin configuration (continued)

NAND_CE0_B	SAI1_TXC.alt1
NAND_DATA00	SD3_DATA0.alt1
NAND_DATA01	SD3_DATA1.alt1
NAND_DATA02	SD3_DATA2.alt2
NAND_DATA03	SD3_DATA3.alt3
NAND_DATA04	SD3_DATA4.alt4
NAND_DATA05	SD3_DATA5.alt5
NAND_DATA06	SD3_DATA6.alt6
NAND_DATA07	SD3_DATA7.alt7

6.1.5.4 Expansion device

The ROM supports booting from the MMC/eMMC and SD/eSD compliant devices.

6.1.5.4.1 Expansion device eFUSE configuration

The SD/MMC/eSD/eMMC/SDXC boot can be performed using either the USDHC ports, based on the setting of the BOOT_CFG[11:10] (Port Select) fuse or it is associated to the GPIO input value at the boot.

All USDHC ports support the fast boot. See this table for details:

Table 6-19. USDHC boot eFUSE descriptions

Fuse	Config	Definition	GPIO[1]	Shipped value	Settings
BOOT_CFG[7]	OEM	Fast boot support	Yes	000	0 - Normal boot 1 - Fast boot
BOOT_CFG[6:4]	OEM	Bus width	Yes	000	0 - SD/eSD/SDXC 1 - MMC/eMMC
BOOT_CFG[3:1]	OEM	SD/MMC speed mode/ USDHC1 IO voltage selection	Yes	000	MMC speed selection 00 - Normal 01 - High else - Reserved SD speed selection speed 000 - Normal/SDR12 001 - High/SDR25 010 - SDR50 011 - SDR104

Table continues on the next page...

Table 6-19. USDHC boot eFUSE descriptions (continued)

Fuse	Config	Definition	GPIO[1]	Shipped value	Settings
					else - Reserved USDHC1 IO VOLTAGE SELECTION (only for MMC/eMMC boot) 0 - 3.3 V 1 - 1.8 V
MMC Speed Mode (CFG[3:2])	Yes	00	MMC speed selection 00 - Normal 01 - High else - Reserved		
USDHC1 IO Voltage Selection (CFG[1])	Yes	0	USDHC1 IO VOLTAGE SELECTION (only for MMC/eMMC boot) 0 - 3.3 V 1 - 1.8 V		
BOOT_CFG[0]	OEM	USDHC2 IO VOLTAGE	Yes	0	USDHC2 IO VOLTAGE SELECTION (only for MMC/eMMC boot) 0 - 3.3 V 1 - 1.8 V
BOOT_CFG[15:12]	OEM	Boot device selection	Yes	0000	0001 - Boot from SD/eSD 0010 - Boot from MMC/eMMC
BOOT_CFG[11:10]	OEM	USDHC port selection	Yes	00	00 - USDHC-1 01 - USDHC-2 10 - USDHC-3 else - reserved
BOOT_CFG[9]	OEM	SD power cycle enable/ eMMC reset enable	Yes	0	SD power cycle/eMMC reset 0 - Disabled 1 - Enabled
BOOT_CFG[8]	OEM	USDHC loopback clock selection	Yes	0	USDHC loopback clock source selection 0 - Through SD pad 1 - Direct
0x490[14:8]	OEM	SD/MMC DLL DLY config	No	0	Delay target for USDHC DLL, it is applied to the slave mode target delay or overrides the mode target delay, depending on the DLL override fuse bit value.

Table continues on the next page...

Table 6-19. USDHC boot eFUSE descriptions (continued)

Fuse	Config	Definition	GPIO[1]	Shipped value	Settings
0x490[15]	OEM	USDHC DLL override enabled	No	0	0 - No override 1 - Override
0x490[16]	OEM	USDHC DLL enabled	No	0	0 - Disable the DLL for SD/eMMC 1 - Enable the DLL for SD/eMMC
0x490[17]	OEM	USDHC override pad settings selection			0 - Use default pad settings 1 - Override the USDHC pad settings by using the PAD_SETTINGS value
0x490[18]	OEM	USDHC_IOMUX_SION_BIT_ENABLE	No	0	0 - Disable 1 - Enable
0x490[19]	OEM	ENABLE_EMMC_5K_PULLUP	No	0	0 - 47 K pullup 1 - 5 K pullup
0x490[20]	OEM	USDHC_PAD_PULL_DOWN	No	0	0 - No action 1 - Pull down
0x490[21]	OEM	Issue pre-idle command enabled (for eMMC4.4)	No	0	0 - Enable 1 - Disable
0x490[23]	OEM	Disable SDMMC manufacture mode	No	0	0 - Enable 1 - Disable
0X490[31:24]	OEM	USDHC pad setting override	No	0	Override pad settings default if 0X490[17] is set
0x4A0[0]	OEM	Fast boot acknowledge enable	No	0	0 - Boot Ack disabled 1 - Boot Ack enabled
0x4A0[1]	OEM	USDHC3 IO voltage selection	No	0	0 - 3.3 V 1 - 1.8 V
0x4A0[2]	OEM	uSDHC power-off polarity selection	No	0	0 - Low 1 - High
0x4A0[3]	OEM	uSDHC power cycle delay selection	No	0	0 - 5 ms 1 - 2.5 ms
0x4A0[5:4]	OEM	uSDHC power cycle interval	No	0	00 - 20 ms 01 - 10 ms 10 - 5 ms 11 - 2.5 ms

1. The setting can be overridden by the GPIO settings when the BT_FUSE_SEL fuse is intact. See GPIO boot overrides for the corresponding GPIO pin.

The boot code supports these standards:

- MMCv4.4 or less
- eMMCv5.0 or less

- SDv2.0 or less
- eSDv2.10 rev-0.9, with or without FAST_BOOT
- SDXCv3.0

The MMC/SD/eSD/SDXC/eMMC can be connected to any of the USDHC blocks and can be booted by copying 4 KB of data from the MMC/SD/eSD/eMMC device to the internal RAM. After checking the Image Vector Table header value (0xD1) from program image, the ROM code performs a DCD check. After a successful DCD extraction, the ROM code extracts from the Boot Data Structure the destination pointer and length of image to be copied to the RAM device from where the code execution occurs.

The maximum image size to load into the SD/MMC boot is 32 MB. This is due to a limited number of uSDHC ADMA Buffer Descriptors allocated by the ROM.

NOTE

The initial 4 KB of the program image must contain the IVT, DCD, and the Boot Data structures.

Table 6-20. SD/MMC frequencies

	SD	MMC	MMC (DDR mode)
Identification (KHz)	347.22		
Normal-speed mode (MHz)	25	20	25
High-speed mode (MHz)	50	40	50
UHSI SDR50 (MHz)	100		
UHSI SDR104 (MHz)	200		

NOTE

The boot ROM code reads the application image length and the application destination pointer from the image.

6.1.5.4.2 MMC and eMMC boot

This table provides the MMC and eMMC boot details.

Table 6-21. MMC and eMMC boot details

Normal boot mode	During the initialization (normal boot mode), the MMC frequency is set to 347.22 KHz. When the MMC card enters the identification portion of the initialization, the voltage validation is performed, and the ROM boot code checks the high-voltage settings and the card capacity. The ROM boot code supports both the high-capacity and low-capacity MMC/ eMMC cards. After the initialization phase is complete, the ROM boot code switches to a higher frequency (20 MHz in

Table continues on the next page...

Table 6-21. MMC and eMMC boot details (continued)

	the normal boot mode or 40 MHz in the high-speed mode). The eMMC is also interfaced via the USDHC and follows the same flow as the MMC. The boot partition can be selected for an MMC4.x card after the card initialization is complete. The ROM code reads the BOOT_PARTITION_ENABLE field in the Ext_CSD[179] to get the boot partition to be set. If there is no boot partition mentioned in the BOOT_PARTITION_ENABLE field or the user partition was mentioned, the ROM boots from the user partition.
eMMC4.3 or eMMC4.4 device supporting special boot mode	If using an eMMC4.3 or eMMC4.4 device that supports the special boot mode, it can be initiated by pulling the CMD line low. If the BOOT ACK is enabled, the eMMC4.3/eMMC4.4 device sends the BOOT ACK via the DATA lines and the ROM can read the BOOT ACK [S010E] to identify the eMMC4.3/eMMC4.4 device. If the BOOT ACK is enabled, the ROM waits 50 ms to get the BOOT ACK and if the BOOT ACK is received by the ROM. If BOOT ACK is disabled ROM waits 1 second for data. If the BOOT ACK or data was received, the eMMC4.3/eMMC4.4 is booted in the "boot mode", otherwise the eMMC4.3/eMMC4.4 boots as a normal MMC card from the selected boot partition. This boot mode can be selected by the BOOT_CFG (fast boot) fuse. The BOOT ACK is selected by the .
eMMC4.4 device	If using the eMMC4.4 device, the Double Data Rate (DDR) mode can be used. This mode can be selected by the BOOT_CFG2[7:5] (bus width) fuse.

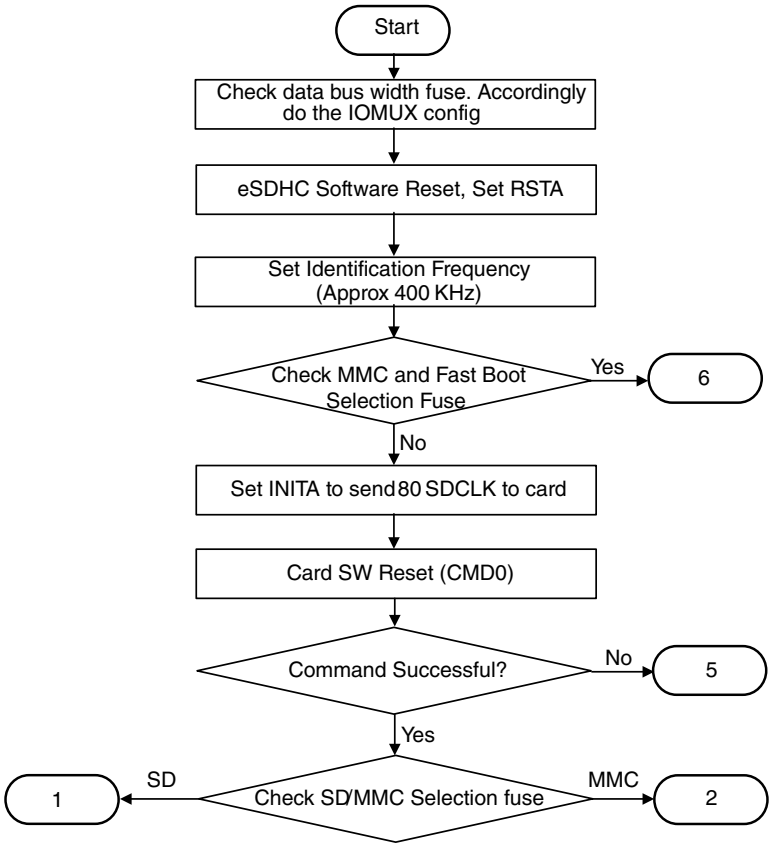


Figure 6-11. Expansion device boot flow (1 of 6)

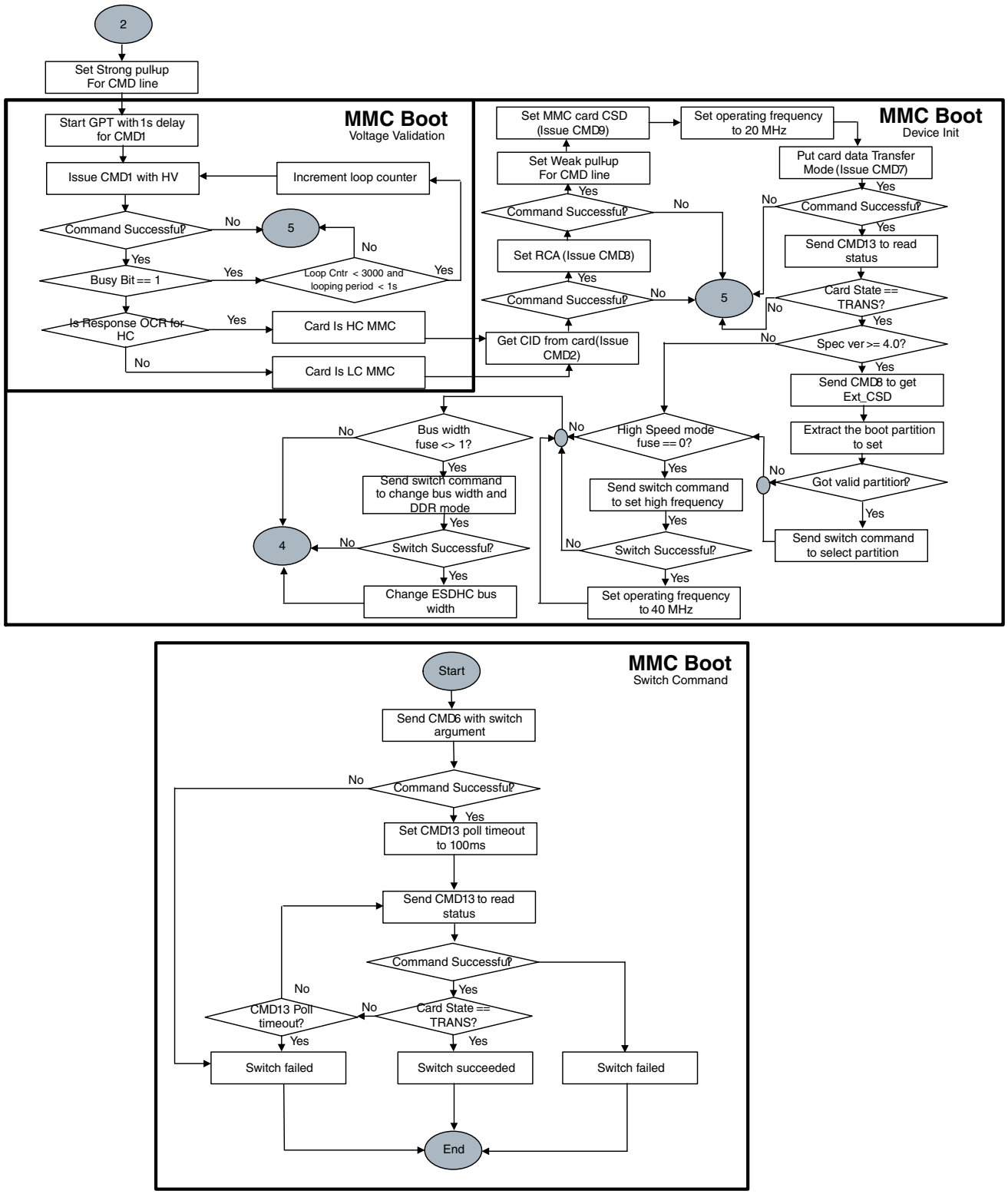


Figure 6-12. Expansion device (MMC) boot flow (2 of 6)

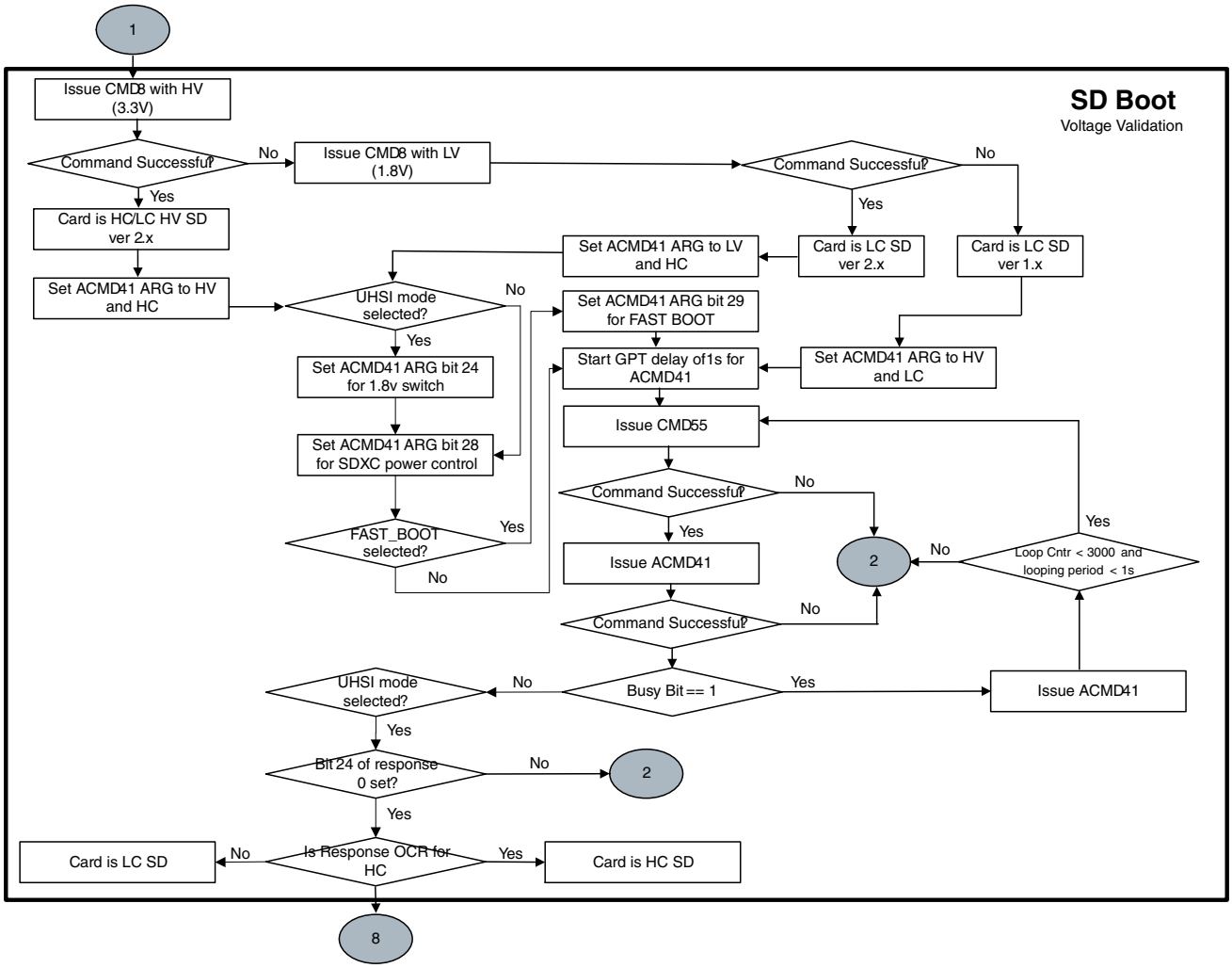


Figure 6-13. Expansion device (SD/eSD/SDXC) boot flow (3 of 6) part 1

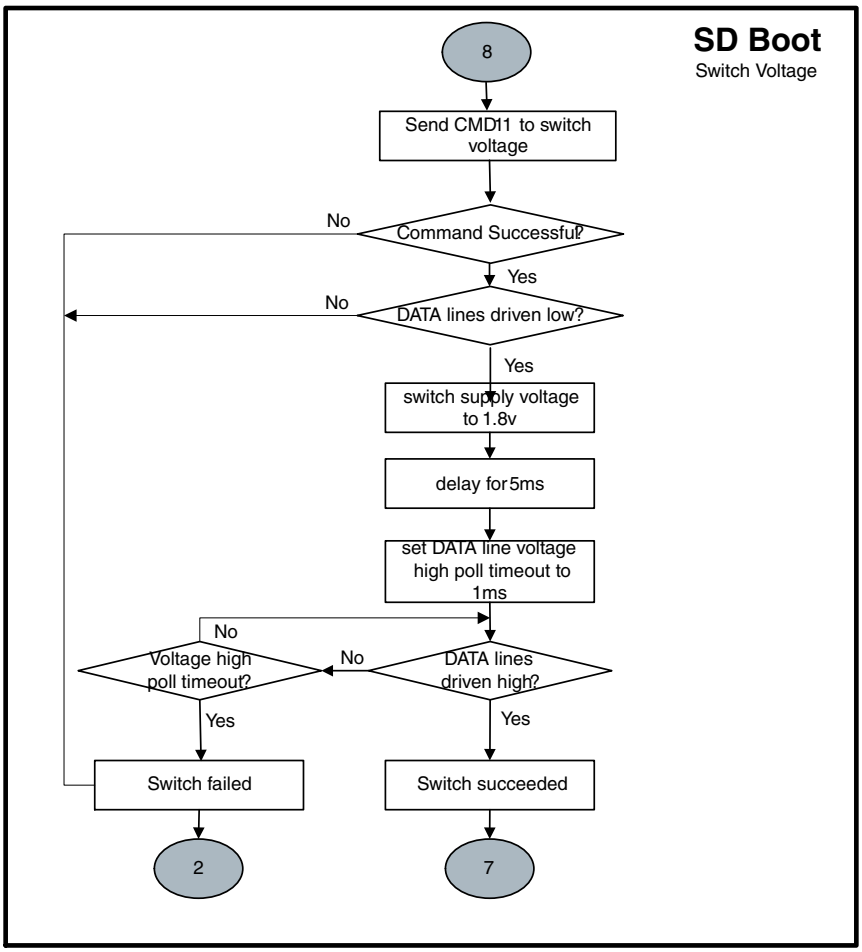


Figure 6-14. Expansion device (SD/eSD/SDXC) boot flow (3 of 6) part 2

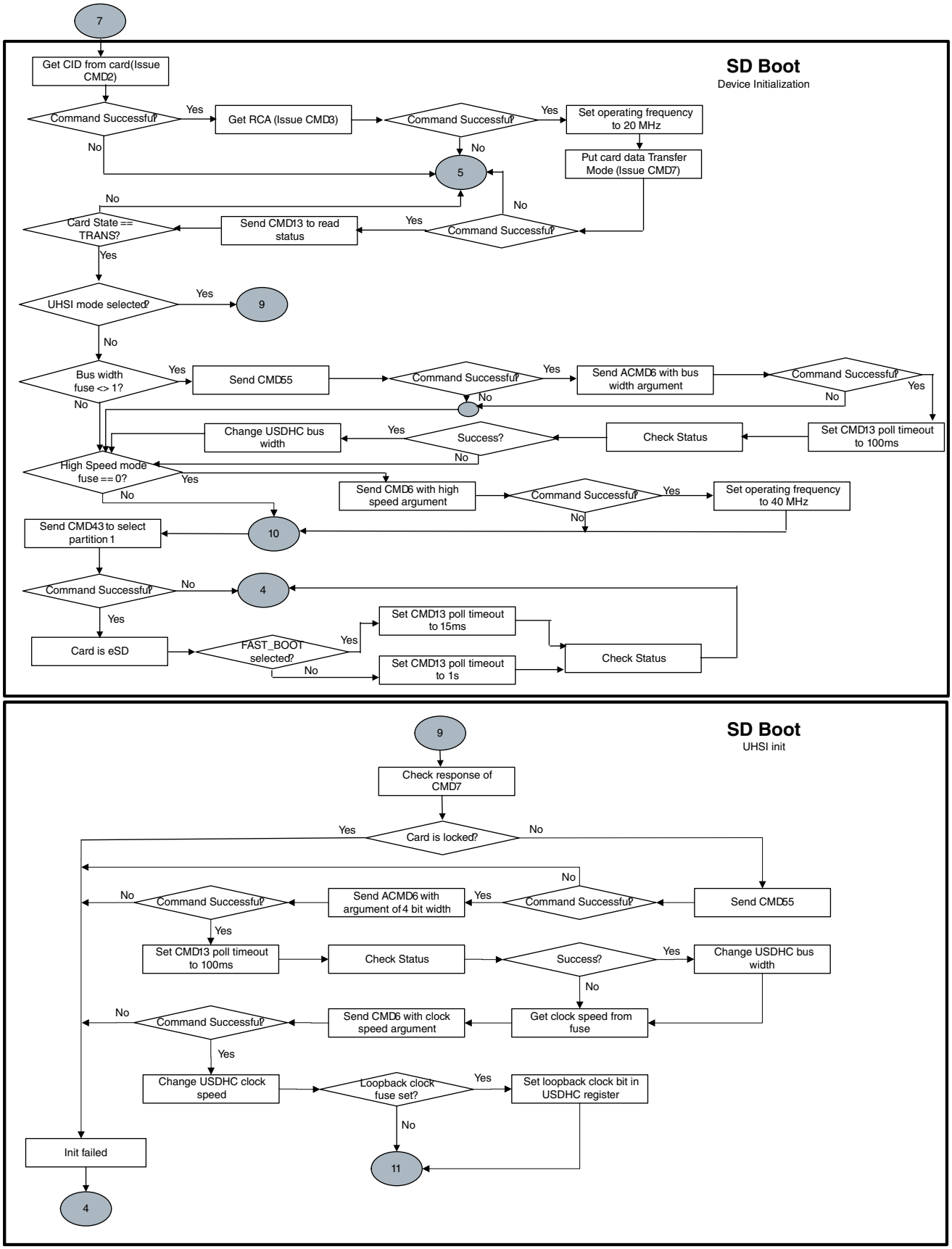


Figure 6-15. Expansion device (MMCSD/eSD/SDXC) boot flow (4 of 6)

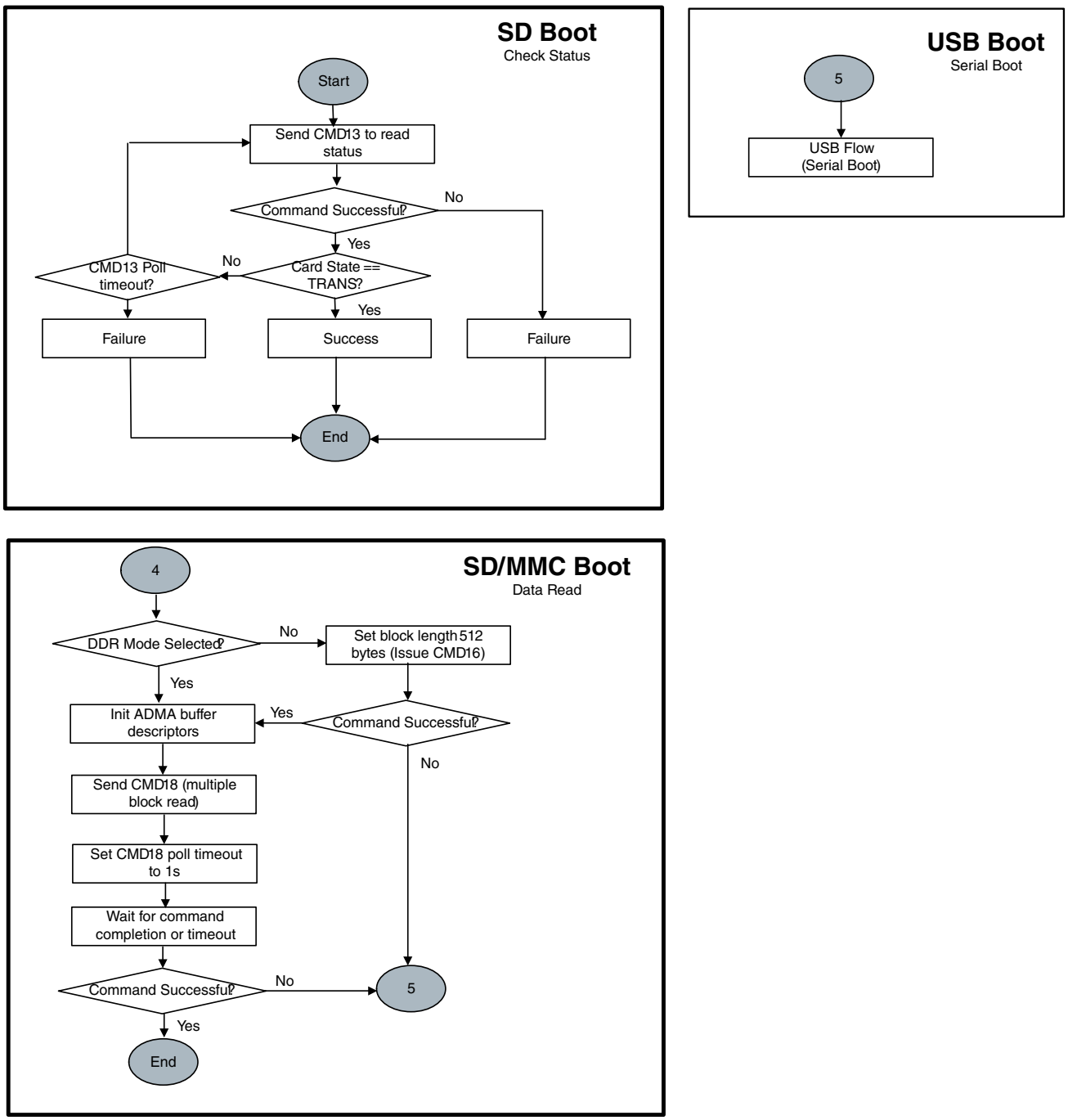


Figure 6-16. Expansion device (SD/eSD) boot flow (5 of 6)

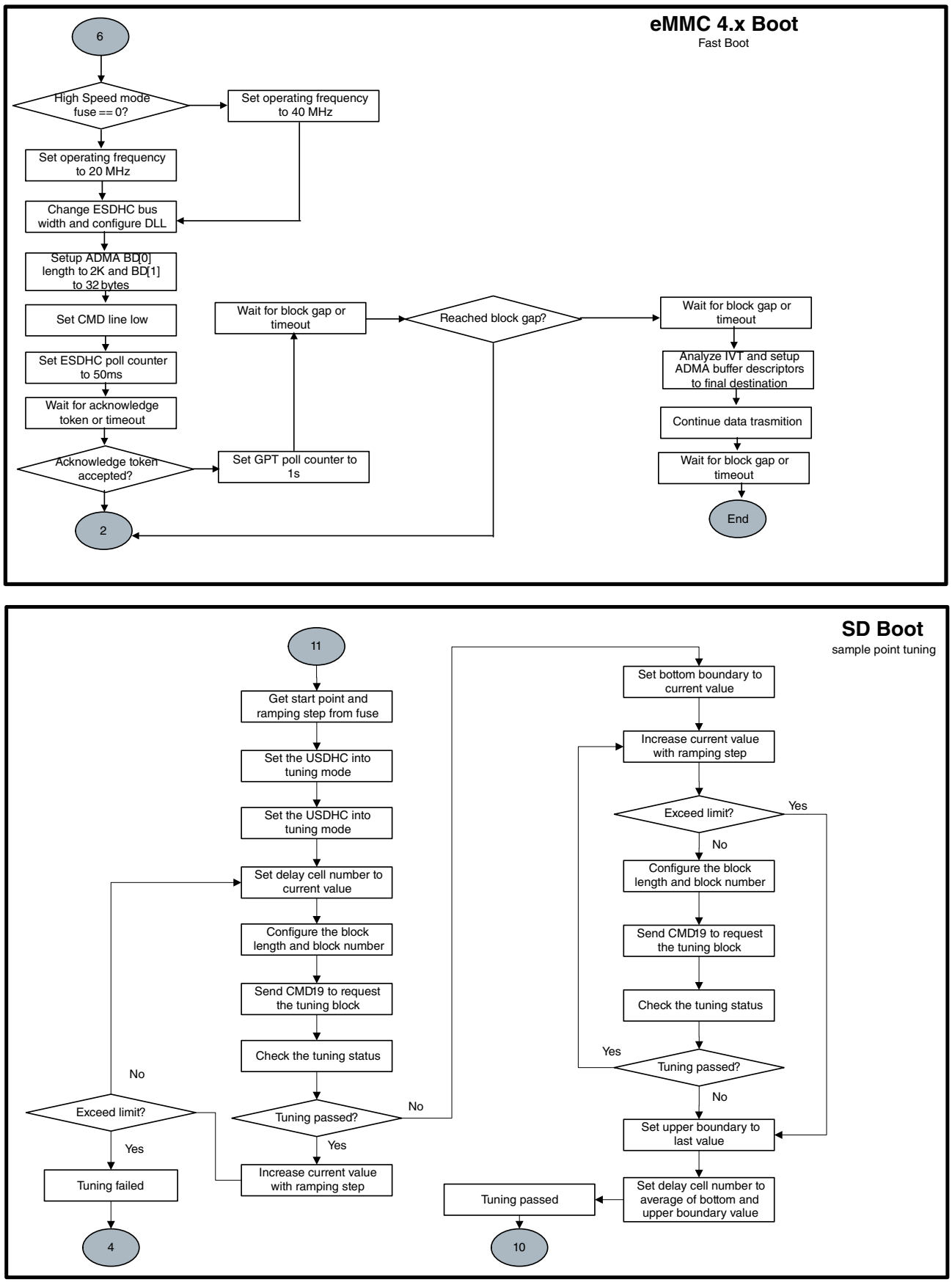


Figure 6-17. Expansion device boot flow (6 of 6)

6.1.5.4.3 SD, eSD, and SDXC

After the normal boot mode initialization begins, the SD/eSD/SDXC frequency is set to 347.22 kHz. During the identification phase, the SD/eSD/SDXC card voltage validation is performed. During the voltage validation, the boot code first checks with the high-voltage settings; if that fails, it checks with the low-voltage settings.

The capacity of the card is also checked. The boot code supports the high-capacity and low-capacity SD/eSD/SDXC cards after the voltage validation card initialization is done.

During the card initialization, the ROM boot code attempts to set the boot partition for all SD, eSD, and SDXC devices. If this fails, the boot code assumes that the card is a normal SD or SDXC card. If it does not fail, the boot code assumes it is an eSD card. After the initialization phase is over, the boot code switches to a higher frequency (25 MHz in the normal-speed mode or 50 MHz in the high-speed mode). The ROM also supports the FAST_BOOT mode booting from the eSD card. This mode can be selected by the BOOT_CFG[4] (Fast Boot) fuse described in .

For the UHSI cards, the clock speed fuses can be set to SDR50 or SDR104 on ports. This enables the voltage switch process to set the signaling voltage to 1.8 V during the voltage validation. The bus width is fixed at a 4-bit width and a sampling point tuning process is needed to calibrate the number of the delay cells. If the SD Loopback Clock eFuse is set, the feedback clock comes directly from the loopback SD clock, instead of the card clock (by default). The SD clock speed can be selected by the BOOT_CFG[3:2], and the SD Loopback Clock is selected by the BOOT_CFG[0].

The UHSI calibration start value (MMC_DLL_DLY[6:0]) and the step value can be set to optimize the sample point tuning process.

If the SD Power Cycle Enable eFuse is 1, the ROM sets the SD_RST pad low, waits for 5 ms, and then sets the SD_RST pad high. If the SD_RST pad is connected to the SD power supply enable logic on board, it enables the power cycle of the SD card. This may be crucial in case the SD logic is in the 1.8 V states and must be reset to the 3.3 V states.

6.1.5.4.4 IOMUX configuration for SD/MMC

Table 6-22. SD/MMC IOMUX pin configuration

Signal	USDHC1	USDHC2	USDHC3
CLK	SD1_CLK.alt0	SD2_CLK.alt0	SD3_CLK.alt0
CMD	SD1_CMD.alt0	SD2_CMD.alt0	SD3_CMD.alt0
DATA0	SD1_DATA0.alt0	SD2_DATA0.alt0	SD3_DATA0.alt0
DATA1	SD1_DATA1.alt0	SD2_DATA1.alt0	SD3_DATA1.alt0
DATA2	SD1_DATA2.alt0	SD2_DATA2.alt0	SD3_DATA2.alt0

Table continues on the next page...

Table 6-22. SD/MMC IOMUX pin configuration (continued)

Signal	USDHC1	USDHC2	USDHC3
DATA3	SD1_DATA3.alt0	SD2_DATA3.alt0	SD3_DATA3.alt0
DATA4	ECSPI2_SCLK.alt2	ECSPI1_SCLK.alt2	SD3_DATA4.alt0
DATA5	ECSPI2_MOSI.alt2	ECSPI1_MOSI.alt2	SD3_DATA5.alt0
DATA6	ECSPI2_MISO.alt2	ECSPI1_MISO.alt2	SD3_DATA6.alt0
DATA7	ECSPI2_SS0.alt2	ECSPI1_SS0.alt2	SD3_DATA7.alt0
VSELECT	GPIO1_IO08.alt1	GPIO1_IO12.alt1	GPIO1_IO13.alt1
RESET_B	SD1_RESET_B.alt5	SD2_RESET_B.alt5	SD3_RESET_B.alt5
CD_B	SD1_CD_B.alt0	-	-

6.1.5.4.5 Redundant boot support for expansion device

The ROM supports the redundant boot for an expansion device. The primary or secondary image is selected, depending on the PERSIST_SECONDARY_BOOT setting. (see Table 6-8).

If the PERSIST_SECONDARY_BOOT is 0, the boot ROM uses address 0x8400 for the primary image.

If the PERSIST_SECONDARY_BOOT is 1, the boot ROM reads the secondary image table from address 0x8200 on the boot media and uses the address specified in the table.

Table 6-23. Secondary image table format

Reserved (chipNum)
Reserved (driveType)
tag
firstSectorNumber
Reserved (sectorCount)

Where:

- The tag is used as an indication of the valid secondary image table. It must be 0x00112233.
- The firstSectorNumber is the first 512-byte sector number of the secondary image.

For the secondary image support, the primary image must reserve the space for the secondary image table. See this figure for the typical structures layout on an expansion device.

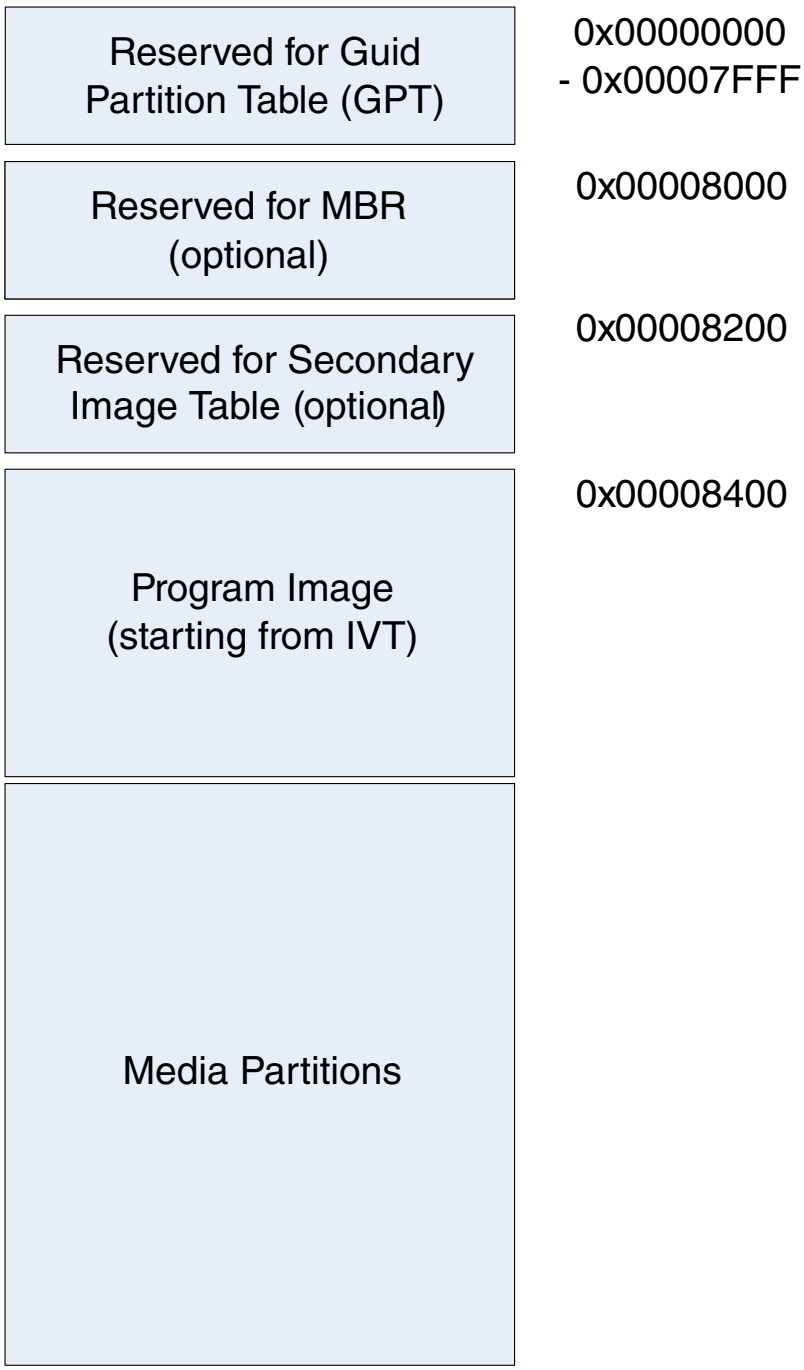


Figure 6-18. Expansion device structures layout

For the Closed mode, if there are failures during primary image authentication, the boot ROM turns on the PERSIST_SECONDARY_BOOT bit (see Table 6-8) and performs the software reset. (After the software reset, the secondary image is used.)

6.1.5.5 Serial NOR through SPI

The chip supports booting from serial memory devices, such as EEPROM and serial flash, using the SPI.

These ports are available for serial boot: eCSPI (eCSPI1, eCSPI2, eCSPI3) interfaces.

6.1.5.5.1 Serial(SPI) NOR eFUSE configuration

The boot ROM code determines the type of device using the following parameters, either provided by the eFUSE settings or sampled on the I/O pins, during boot.

See this table for details:

Table 6-24. Serial(SPI) NOR boot eFUSE descriptions

Fuse	Config	Definition	GPIO[1]	Shipped value	Settings
BOOT_CFG[14:12]	OEM	Boot device selection	Yes	0000	110 - Boot from the serial(SPI) NOR
BOOT_CFG[11:9]	OEM	ECSPI port selection	Yes	000	000 - eCSPI1 001 - eCSPI2 010 - eCSPI3
BOOT_CFG[8]	OEM	SPI addressing	Yes	0	0 - 3 B (24-bit) 1 - 2 B (16-bit)
0x480[25]	OEM	Recovery boot enable	No	0	0 – Disabled 1 – Enabled
BOOT_CFG[7:6]	OEM	CS selection (SPI only)	Yes	00	00 – CS#0
0x480[31:29]	OEM	Recovery port selection	No	000	000 - eCSPI1 001 - eCSPI2 010 - eCSPI3
0x480[28]	OEM	Recovery SPI addressing	No	0	0 - 3 B (24-bit) 1 - 2 B (16-bit)
0x480[27:26]	OEM	Recovery CS selection (SPI only)	No	00	00 - CS#0

1. The setting can be overridden by the GPIO settings when the BT_FUSE_SEL fuse is intact. See Table 1 for the corresponding GPIO pin.

The ECPSI-1/ECPSI-2/ECPSI-3 block can be used as a boot device using the ECSPI interface for the serial(SPI) NOR boot. The SPI interface is configured to operate at 12.5 MHz for 3-byte addressing devices and at 3.125 MHz for 2-byte addressing devices.

The boot ROM copies 4 KB of data from the serial ROM device to the internal RAM. After checking the Image Vector Table header value (0xD1) from the program image, the ROM code performs a DCD check. After a successful DCD extraction, the ROM code extracts the destination pointer and length of image from the Boot Data Structure to be copied to the RAM device from where the code execution occurs.

NOTE

The Initial 4 KB of program image must contain the IVT, DCD, and the Boot Data Structures.

6.1.6 Program image

This section describes the data structures that are required to be included in the user's program image. The program image consists of:

- Image vector table—a list of pointers located at a fixed address that the ROM examines to determine where the other components of the program image are located.
- Boot data—a table that indicates the program image location, program image size in bytes, and the plugin flag.
- Device configuration data—IC configuration data.
- User code and data.

6.1.6.1 Image Vector Table and Boot Data

The Image Vector Table (IVT) is the data structure that the ROM reads from the boot device supplying the program image containing the required data components to perform a successful boot.

The IVT includes the program image entry point, a pointer to Device Configuration Data (DCD) and other pointers used by the ROM during the boot process.The ROM locates the IVT at a fixed address that is determined by the boot device connected to the Chip. The IVT offset from the base address and initial load region size for each boot device type is defined in the table below. The location of the IVT is the only fixed requirement by the ROM. The remainder or the image memory map is flexible and is determined by the contents of the IVT.

Table 6-25. Image Vector Table Offset and Initial Load Region Size

Boot Device Type	Image Vector Table Offset	Initial Load Region Size
NAND	1 Kbyte = 0x4000 bytes	8 Kbyte

Table continues on the next page...

Table 6-25. Image Vector Table Offset and Initial Load Region Size (continued)

Boot Device Type	Image Vector Table Offset	Initial Load Region Size
SD/eSD/MMC/eMMC normal boot	33 Kbyte = 0x8400 bytes	8 Kbyte
eMMC Fast boot	1 Kbyte = 0x400 bytes	8 Kbyte
FlexSPI	4 Kbyte = 0x1000 bytes	8 Kbyte
ECSPI	1 Kbyte = 0x400 bytes	8 Kbyte

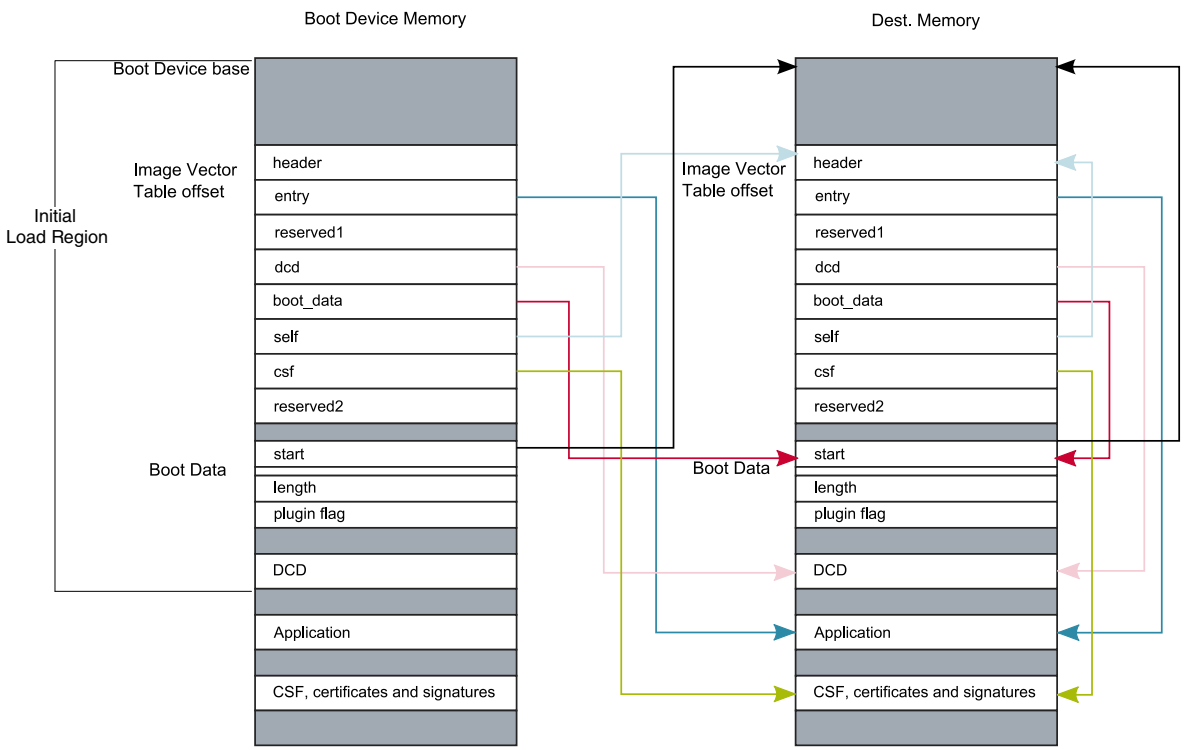


Figure 6-19. Image Vector Table

6.1.6.1.1 Image vector table structure

The IVT has the following format where each entry is a 32-bit word:

Table 6-26. IVT format

header
entry: Absolute address of the first instruction to execute from the image
reserved1: Reserved and should be zero
dcd: Absolute address of the image DCD. The DCD is optional so this field may be set to NULL if no DCD is required. See Device Configuration Data (DCD) for further details on the DCD.

Table continues on the next page...

Table 6-26. IVT format (continued)

boot data: Absolute address of the boot data
self: Absolute address of the IVT. Used internally by the ROM.
csf: Absolute address of the Command Sequence File (CSF) used by the HAB library. See High-Assurance Boot (HAB) for details on the secure boot using HAB. This field must be set to NULL when not performing a secure boot
reserved2: Reserved and should be zero

Figure 6-20 shows the IVT header format:

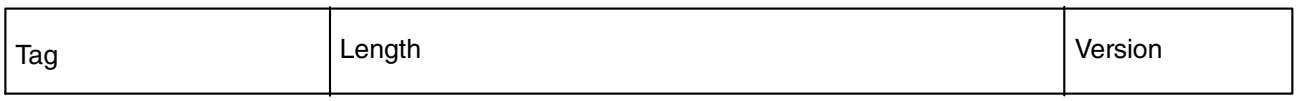


Figure 6-20. IVT header format

where:

```
Tag: A single byte field set to 0xD1
Length: a two byte field in big endian format containing the overall length of the IVT,
in bytes, including the header. (the length is fixed and must have a value of
32 bytes)
Version: A single byte field set to 0x40 or 0x41
```

6.1.6.1.2 Boot data structure

The boot data must follow the format defined in the table found here, each entry is a 32-bit word.

Table 6-27. Boot data format

start	Absolute address of the image
length	Size of the program image
plugin	Plugin flag (see Plugin image)

6.1.6.2 Device Configuration Data (DCD)

Upon reset, the chip uses the default register values for all peripherals in the system. However, these settings typically are not ideal for achieving the optimal system performance and there are even some peripherals that must be configured before they can be used.

The DCD is a configuration information contained in the program image (external to the ROM) that the ROM interprets to configure various peripherals on the chip.

For example, the EIM default settings allow the core to interface to a NOR flash device immediately after the reset. This allows the chip to interface with any NOR flash device, but has the disadvantage of slow performance. Additionally, some components (such as DDR) require some sequence of register programming as a part of the configuration before it is ready to be used. The DCD feature can be used to program the EIM registers and the DDR Controller registers to the optimal settings.

The ROM determines the location of the DCD table based on the information located in the Image Vector Table (IVT). See Image Vector Table and Boot Data for more details. The DCD table shown below is a big-endian byte array of the allowable DCD commands. The maximum size of the DCD is limited to 1768 B.

Figure 6-21. DCD data format

The DCD header is 4 B with the following format:

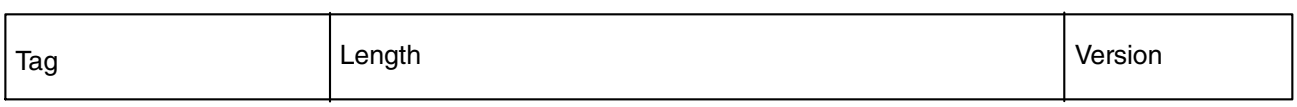


Figure 6-22. DCD header format

where:

```
Tag: A single-byte field set to 0xD2
Length: a two-byte field in the big-endian format containing the overall length of the DCD
(in bytes) including the header
Version: A single-byte field set to 0x41
```

6.1.6.2.1 Write data command

The write data command is used to write a list of given 1-, 2- or 4-byte values (or bitmasks) to a corresponding list of target addresses.

The format of the write data command (in a big-endian byte array) is shown in this table:

Table 6-28. Write data command format

Tag	Length	Parameter
Address		
Value/Mask		
[Address]		
[Value/Mask]		
...		
[Address]		
[Value/Mask]		

where:

```
Tag: a single-byte field set to 0xCC
Length: a two-byte field in a big-endian format, containing the length of the Write Data
Command (in bytes) including the header
Address: the target address to which the data must be written
Value/Mask: the data value (or bitmask) to be written to the preceding address
```

The parameter field is a single byte divided into the bitfields, as follows:

Table 6-29. Write data command parameter field

7	6	5	4	3	2	1	0
flags					bytes		

where

```
bytes: the width of the target locations in bytes (either 1, 2, or 4)
flags: control flags for the command behavior
Data Mask = bit 3: if set, only specific bits may be overwritten at the target address
(otherwise all bits may be overwritten)
Data Set = bit 4: if set, the bits at the target address are overwritten with this flag
(otherwise it is ignored)
```

One or more target address and value/bitmask pairs can be specified. The same bytes' and flags' parameters apply to all locations in the command.

When successful, this command writes to each target address in accordance with the flags as follows:

Table 6-30. Interpretation of write data command flags

"Mask"	"Set"	Action	Interpretation
0	0	*address = val_msk	Write value
0	1	*address = val_msk	Write value
1	0	*address &= ~val_msk	Clear bitmask
1	1	*address |= val_msk	Set bitmask

NOTE

If any of the target addresses does not have the same alignment as the data width indicated in the parameter field, none of the values are written.

If any of the values are larger or any of the bitmasks are wider than permitted by the data width indicated in the parameter field, none of the values are written.

If any of the target addresses do not lie within the allowed region, none of the values are written. The list of allowable blocks and target addresses for the chip are provided below.

6.1.6.2.2 Check data command

The check data command is used to test for a given 1-, 2-, or 4-byte bitmasks from a source address.

The check data command is a big-endian byte array with the format shown in this table:

Table 6-31. Check data command format

Tag	Length	Parameter
Address		
Mask		
[Count]		

where:

```
Tag: a single-byte field set to 0xCF
Length: a two-byte field in the big-endian format containing the length of the check data
command (in bytes) including the header
Address: the source address to test
Mask: the bit mask to test
Count: an optional poll count; If the count is not specified, this command polls
indefinitely
until the exit condition is met. If count = 0, this command behaves as for the NOP.
```

The parameter field is a single byte divided into bitfields, as follows:

Table 6-32. Check data command parameter field

7	6	5	4	3	2	1	0
flags					bytes		

where

```
bytes: the width of target locations in bytes (either 1, 2, or 4)
flags: control flags for the command behavior
Data Mask = bit 3: if set, only the specific bits may be overwritten at a target address
```

```
(otherwise all bits may be overwritten)
Data Set = bit 4: if set, the bits at the target address are overwritten with this flag
(otherwise it is ignored)
```

This command polls the source address until either the exit condition is satisfied, or the poll count is reached. The exit condition is determined by the flags as follows:

Table 6-33. Interpretation of check data command flags

"Mask"	"Set"	Action	Interpretation
0	0	(*address & mask) == 0	All bits clear
0	1	(*address & mask) == mask	All bits set
1	0	(*address & mask)!= mask	Any bit clear
1	1	(*address & mask)!= 0	Any bit set

NOTE

If the source address does not have the same alignment as the data width indicated in the parameter field, the value is not read.

If the bitmask is wider than permitted by the data width indicated in the parameter field, the value is not read.

6.1.6.2.3 NOP command

This command has no effect.

The format of the NOP command is a big-endian four-byte array, as shown in this table:

Table 6-34. NOP command format

Tag	Length	Undefined

where:

```
Tag: a single-byte field set to 0xC0
Length: a two-byte field in big endian containing the length of the NOP command in bytes
(fixed to a
value of 4)
Undefined: this byte is ignored and can be set to any value.
```

6.1.6.2.4 Unlock command

The unlock command is used to prevent specific engine features from being locked when exiting the ROM.

The format of the unlock command (in a big-endian byte array) is shown in this table:

Table 6-35. Unlock command format

Tag	Length	Eng
Value		
Value		
...		
Value		

where:

NOTE

This command may not be used in the DCD structure if the SEC_CONFIG is configured as closed.

6.1.7 Plugin image

The ROM supports a limited number of boot devices. When using other devices as a boot source (for example, Ethernet, CDROM, or USB), the supported boot device must be used (typically serial(SPI) NOR) as a firmware to provide the missing boot drivers. Additionally, the plugin can be customized to support boot drivers, which is more flexible when performing the device initialization, such as condition judging, delay assertion, or to apply custom settings to the boot device and memory system.

In addition to the standard images, the chip also supports plugin images. The plugin images return the execution to the ROM whereas the standard image does not.

The boot ROM detects the image type using the plugin flag of the boot data structure (see Boot data structure). If the plugin flag is 1, then the ROM uses the image as a plugin function. The function must initialize the boot device and copy the program image to the final location. At the end, the plugin function must return with the program image parameters. (See High-level boot sequence for details about the boot flow).

The boot ROM authenticates the plugin image before running the plugin function and then authenticates the program image.

The plugin function must follow the API described below:

typedef BOOLEAN (*plugin_download_f)(void **start, size_t *bytes, UINT32 *ivt_offset);

ARGUMENTS PASSED:

- start - the image load address on exit.

- bytes - the image size on exit.
- ivt_offset - the offset (in bytes) of the IVT from the image start address on exit.

RETURN VALUE:

- 1 - success
- 0 - failure

6.1.8 Serial Downloader

The Serial Downloader provides a means to download a program image to the chip over the USB serial connection.

In this mode, the ROM programs the WDOG1 for a time-out specified by the fuse WDOG Time-out Select (See fusemap for details) if the WDOG_ENABLE eFuse is 1 and continuously polls for the USB connection. If no activity is found on the USB OTG1 and the watchdog timer expires, the Arm core is reset.

NOTE

After the downloaded image is loaded, it is responsible for managing the watchdog resets properly.

This figure shows the USB boot flow:

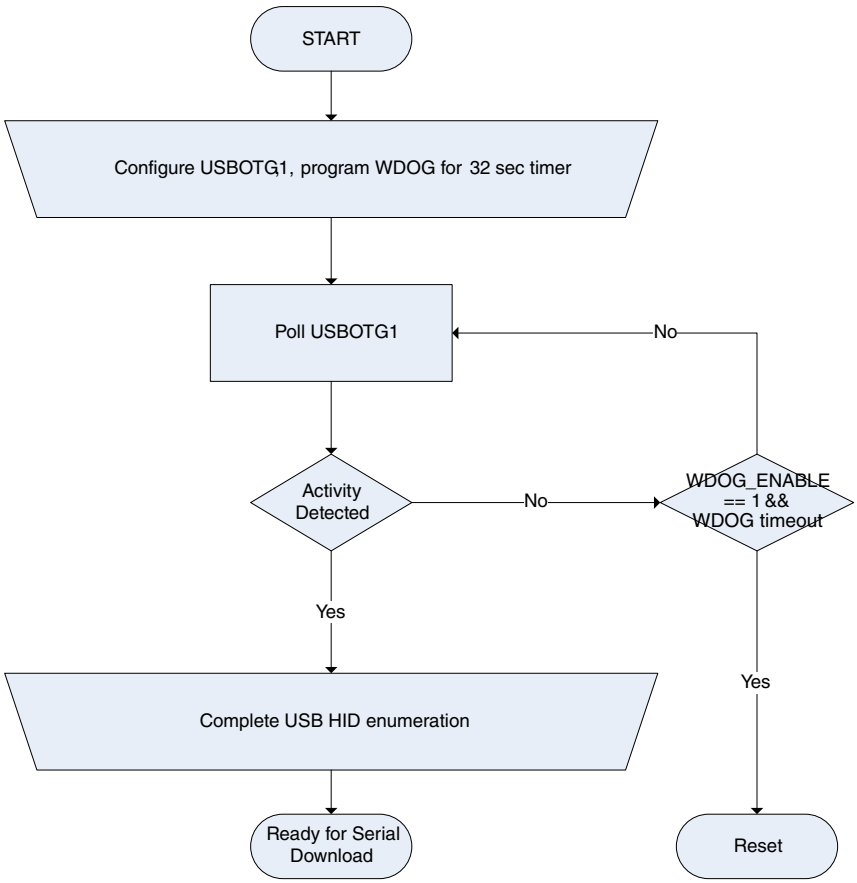


Figure 6-23. Serial Downloader boot flow

NOTE

Before going into USB serial mode, Boot ROM detect SD/MMC card on USDHC2 port. If a card is inserted, ROM will try to boot from it. This is the so-called Manufacture SD/MMC boot. This feature can be disabled by blowing fuse "Disable SD/MMC Manufacture Mode". See SD/MMC manufacture mode for details.

6.1.8.1 USB

The USB support is composed of the USB (core controller, compliant with the USB 2.0 specification) and the USBPHY (HS USB transceiver).

The ROM supports the USB OTG port for boot purposes. The other USB ports on the chip are not supported for boot purposes.

The USB Driver is implemented as a USB HID class. A collection of four HID reports are used to implement the SDP protocol for data transfers, as described in Table 6-36.

Table 6-36. USB HID reports

Report ID (first byte)	Transfer endpoint	Direction	Length	Description
1	control OUT	Host to device	17 B	SDP command from the host to the device.
2	control OUT	Host to device	Up to 1025 B	Data associated with the report 1 SDP command.
3	interrupt	Device to host	5 B	HAB security configuration. The device sends 0x12343412 in the closed mode and 0x56787856 in the open mode.
4	interrupt	Device to host	Up to 65 B	Data in response to the SDP command in report 1.

6.1.8.1.1 USB configuration details

The USB OTG function device driver supports a high speed (HS for UTMI) non-stream mode with a maximal packet size of 512 B and a low-level USB OTG function.

The VID/PID and strings for the are listed in the following table.

Table 6-37. VID/PID and strings for USB device driver

Descriptor	Value
VID	0x1FC9 (NXP vendor ID)
PID[1]	0x012B
String Descriptor1 (manufacturer)	NXP Semiconductors
String Descriptor2 (product)	SE Blank SP Blank NS Blank FR Blank
String Descriptor4	NXP Flash
String Descriptor5	NXP Flash

1. Allocation based on the BPN (Before Part Number)

6.1.8.1.2 IOMUX configuration for USB

The interface signals of the UTMI PHY are not configured in the IOMUX,, except for the USB_OTGn_ID pins. The USB ID pin function is configured using the USBNC_n_CTRL2[ID_DIG_SEL] and the IOMUXC_USB_OTGn_ID_SELECT_INPUT register. The remaining pins of the UTMI PHY interface use the dedicated contacts on the IC. See the chip data sheet for details.

6.1.8.2 Serial Download Protocol (SDP)

The 16-byte SDP command from the host to device is sent using the HID report 1.

This table describes the 16-byte SDP command data structure:

Table 6-38. 16-byte SDP command data structure

BYTE offset	Size	Name	Description
0	2	COMMAND TYPE	These commands are supported for the ROM: • 0x0101 READ_REGISTER • 0x0202 WRITE_REGISTER • 0x0404 WRITE_FILE • 0x0505 ERROR_STATUS • 0x0A0A DCD_WRITE • 0x0B0B JUMP_ADDRESS
2	4	ADDRESS	Only relevant for these commands: READ_REGISTER, WRITE_REGISTER, WRITE_FILE, DCD_WRITE, and JUMP_ADDRESS. For the READ_REGISTER and WRITE_REGISTER commands, this field is the address to a register. For the WRITE_FILE and JUMP_ADDRESS commands, this field is an address to the internal or external memory address.
6	1	FORMAT	Format of access, 0x8 for an 8-bit access, 0x10 for a 16-bit access, and 0x20 for a 32-bit access. Only relevant for the READ_REGISTER and WRITE_REGISTER commands.
7	4	DATA COUNT	Size of the data to read or write. Only relevant for the WRITE_FILE, READ_REGISTER, WRITE_REGISTER, and DCD_WRITE commands. For the WRITE_FILE and DCD_WRITE commands, the DATA COUNT is in the byte units.
11	4	DATA	The value to write. Only relevant for the WRITE_REGISTER command.
15	1	RESERVED	Reserved

6.1.8.2.1 SDP commands

The SDP commands are described in the following sections.

6.1.8.2.1.1 READ_REGISTER

The transaction for the READ_REGISTER command consists of these reports: Report1 for the command, Report3 for the security configuration, and Report4 for the response or the register value.

The register to read is specified in the ADDRESS field of the SDP command. The first device sends Report3 with the security configuration followed by the Report4 with the bytes read at a given address. If the count is greater than 64, multiple reports with the report id 4 are sent until the entire data requested by the host is sent. The STATUS is either 0x12343412 for the closed parts and 0x56787856 for the open or field return parts.

Report1, Command, Host to Device:

1	Valid values for the READ_REGISTER COMMAND, ADDRESS, FORMAT, DATA_COUNT

ID 16-byte SDP command

Report3, Response, Device to Host:

3	Four bytes indicating the security configuration

ID 4 bytes status

Report4, Response, Device to Host: first response report

4	Register value

ID 4 bytes of data containing the register value. If the number of bytes requested is less than 4, the remaining bytes must be ignored by the host.

Multiple reports of the report id 4 are sent until the entire requested data is sent.

Report4, Response, Device to Host: last response report

4	Register value

ID 64 bytes of data containing the register value. If the number of bytes requested is less than 64, the remaining bytes must be ignored by the host.

6.1.8.2.1.2 WRITE_REGISTER

The transaction for the WRITE_REGISTER command consists of these reports: Report1 for the command, Report3 for the security configuration and Report4 for the write status.

The host sends Report1 with the WRITE_REGISTER command. The register to write is specified in the ADDRESS field of the SDP command of Report1, with the FORMAT field set to the data type (number of bits to write, either 8, 16, or 32) and the value to write in the DATA field of the SDP command. The device writes the DATA to the register address and returns the WRITE_COMPLETE code using Report4 and the security configuration using Report3 to complete the transaction.

Report1, Command, Host to Device:

1	Valid values for WRITE_REGISTER COMMAND, ADDRESS, FORMAT, DATA_COUNT and DATA

ID 16-byte SDP command

Report3, Response, Device to Host:

3	4 bytes indicating the security configuration

ID 4 bytes status

Report4, Response, Device to Host:

4	WRITE_COMPLETE (0x128A8A12) status

ID 64 bytes data with the first 4 bytes to indicate that the write is completed with code 0x128A8A12. On failure, the device reports the HAB error status.

6.1.8.2.1.3 WRITE_FILE

The transaction for the WRITE_FILE command consists of these reports: Report1 for the command phase, Report2 for the data phase, Report3 for the HAB mode, and Report4 to indicate that the data are received in full.

The size of each Report2 is limited to 1024 bytes (limitation of the USB HID protocol). Hence, multiple Report2 packets are sent by the host in the data phase until the entire data is transferred to the device. When the entire data (DATA_COUNT bytes) is received, the device sends Report3 with the HAB mode and Report4 with 0x88888888, indicating that the file download completed.

Report1, Host to Device:

1	Valid values for WRITE_FILE COMMAND, ADDRESS, DATA_COUNT

ID 16-byte SDP command

==========Optional Begin==========

Host sends the ERROR_STATUS command to query if the HAB rejected the address

========== Optional End==========

Report2, Host to Device:

2	File data

ID Max 1024 bytes data per report

Report2, Host to Device:

2	File data

ID Max 1024 bytes data per report

Report3, Device to Host:

3	4 bytes indicating security configuration

ID 4 bytes status

Report4, Response, Device to Host:

4	COMPLETE (0x88888888) status

ID 64 bytes data with the first four bytes to indicate that the file download completed with code 0x88888888. On failure, the device reports the HAB error status.

6.1.8.2.1.4 ERROR_STATUS

The transaction for the SDP command ERROR_STATUS consists of three reports.

Report1 is used by the host to send the command; the device sends global error status in four bytes of Report4 after returning the security configuration in Report3. When the device receives the ERROR_STATUS command, it returns the global error status that is updated for each command. This command is useful to find out whether the last command resulted in a device error or succeeded.

Report1, Command, Host to Device:

1	ERROR_STATUS COMMAND

ID 16-byte SDP Command

Report3, Response, Device to Host:

3	Four bytes indicating the security configuration

ID 4 bytes status

Report4, Response, Device to Host:

4	Four bytes Error status

ID first 4 bytes status in 64 bytes Report4

6.1.8.2.1.5 DCD_WRITE

The SDP command DCD_WRITE is used by the host to send multiple register writes in one shot. This command is provided to speed up the process of programming the register writes (such as to configure an external RAM device).

The command goes with Report1 from the host with COMMAND TYPE set to DCD_WRITE, ADDRESS which is used as a temporary location of the DCD data, and DATA_COUNT to the number of bytes sent in the data out phase. In the data phase, the host sends the data for a number of registers using Report2. The device completes the transaction with Report3 indicating the security configuration and Report4 with the WRITE_COMPLETE code 0x12828212.

Report1, Command, Host to Device:

1	DCD_WRITE COMMAND, ADDRESS, DATA_COUNT

ID 16-byte SDP Command

Report2, Data, Host to Device:

2	DCD binary data

ID Max 1024 bytes per report

Report3, Response, Device to Host:

3	Four bytes indicating the security configuration

ID 4 bytes status

Report4, Response, Device to Host:

4	WRITE_COMPLETE (0x128A8A12) status

ID 64 bytes report with the first four bytes to indicate that the write completed with the code 0x128A8A12. On failure, the device reports the HAB error status.

See Device Configuration Data (DCD) for the DCD format description.

6.1.8.2.1.6 SKIP_DCD_HEADER

The SDP command SKIP_DCD_HEADER is used by the host to inform the device to skip the DCD configuration within the download image.

If the download image must be run on the DDR, the DCD configuration data must be built into the image. In case the host issued DCD_WRITE to push the DCD configuration data to the device for the DDR initialization, the image with the DCD information causes the ROM to initialize the DDR twice, and may cause the initialization processing to hang.

The SKIP_DCD_HEADER command informs the device to skip the DCD configuration within the download image and avoid this issue.

This command is typically sent after JUMP_ADDRESS. This command is sent by the host in the command-phase of the transaction using Report1, there is no data phase for this command. The device completes the transaction with Report3 indicating the security configuration and Report4 with the OK_ACK code 0x900DD009.

Report1, Command, Host to Device:

1	SKIP_DCD_HEADER

ID 16-byte SDP Command

Report3, Response, Device to Host:

3	Four bytes indicating the security configuration

ID 4 bytes status

Report4, Response, Device to Host:

4	OK_ACK (0x900DD009)

6.1.8.2.1.7 JUMP_ADDRESS

The SDP command JUMP_ADDRESS is the last command that the host can send to the device. After this command, the device jumps to the address specified in the ADDRESS field of the SDP command and starts to execute.

This command usually follows after the WRITE_FILE command. The command is sent by the host in the command-phase of the transaction using Report1. There is no data phase for this command, but the device sends the status Report3 to complete the transaction. If the authentication fails, it also sends Report4 with the HAB error status.

Report1, Command, Host to Device:

1	JUMP_ADDRESS COMMAND, ADDRESS

ID 16-byte SDP Command

Report3, Response, Device to Host:

3	Four bytes indicating the security configuration

ID 4 bytes status

This report is sent by the device only in case of an error jumping to the given address, or if the device reports error in Report4, Response, Device to Host:

4	Four bytes HAB error status

ID 4 bytes status, 64 bytes report length

6.1.9 Recovery devices

The chip supports recovery devices. If the primary boot device fails, the boot ROM tries to boot from the recovery device using the USDHC2 port.

6.1.10 Low-power boot

The ROM supports the low-power boot. If the LPB_BOOT fuses are blown, the chip checks if there is a low-power condition via the pad. If there is a low-power boot condition, the ROM applies division factors on the ARM, DDR, AXI, and AHB root clocks based on the LPB_BOOT fuses value (see the table below). The polarity of the low-power boot condition on the pad is set by the BT _LPB_POLARITY fuse (see the following figure).

Table 6-39. Low-power boot frequencies

LPB_BOOT	Boot Frequencies=0	Boot Frequencies=1
00	ARM_A53_CLK_ROOT= 800 MHz AHB_CLK_ROOT= 133 MHz MAIN_AXI_CLK_ROOT= 333 MHz	ARM_A53_CLK_ROOT= 400 MHz AHB_CLK_ROOT= 133 MHz MAIN_AXI_CLK_ROOT= 166 MHz
01	ARM_A53_CLK_ROOT= 800 MHz AHB_CLK_ROOT= 133 MHz MAIN_AXI_CLK_ROOT= 333 MHz	ARM_A53_CLK_ROOT= 400 MHz AHB_CLK_ROOT= 133 MHz MAIN_AXI_CLK_ROOT= 166 MHz
10	ARM_A53_CLK_ROOT= 400 MHz AHB_CLK_ROOT= 133 MHz MAIN_AXI_CLK_ROOT= 166 MHz	ARM_A53_CLK_ROOT= 200 MHz AHB_CLK_ROOT= 133 MHz MAIN_AXI_CLK_ROOT= 83.3 MHz
11	ARM_A53_CLK_ROOT= 200 MHz AHB_CLK_ROOT= 133 MHz MAIN_AXI_CLK_ROOT= 83.3 MHz	ARM_A53_CLK_ROOT= 200 MHz AHB_CLK_ROOT= 133 MHz MAIN_AXI_CLK_ROOT= 41.67 MHz

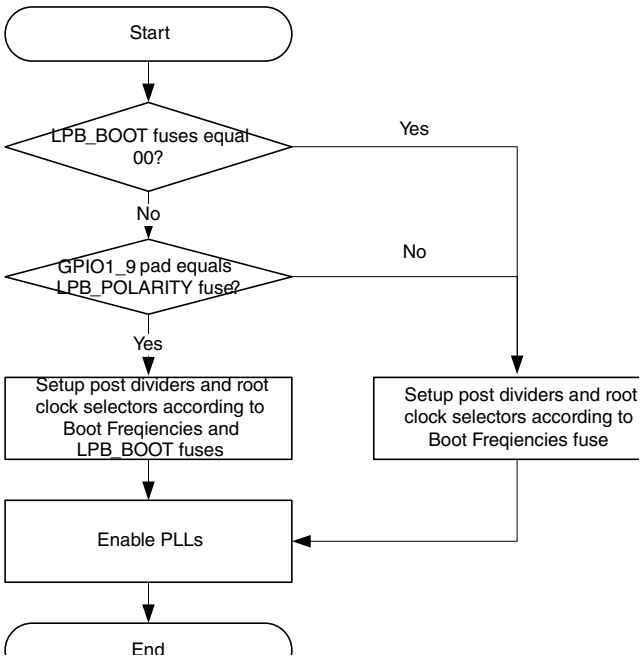


Figure 6-24. Low-power boot flow

6.1.11 SD/MMC manufacture mode

When the internal boot and recover boot (if enabled) failed , the boot goes to the SD/MMC manufacture mode before the serial download mode. In the manufacture mode, one bit bus width is used despite of the fuse setting.

By default, the SD/MMC manufacture mode is enabled. Blow the fuse of the DISABLE_SDMMC_MFG to disable it.

NOTE

A secondary boot is not supported in the SD/MMC manufacture mode.

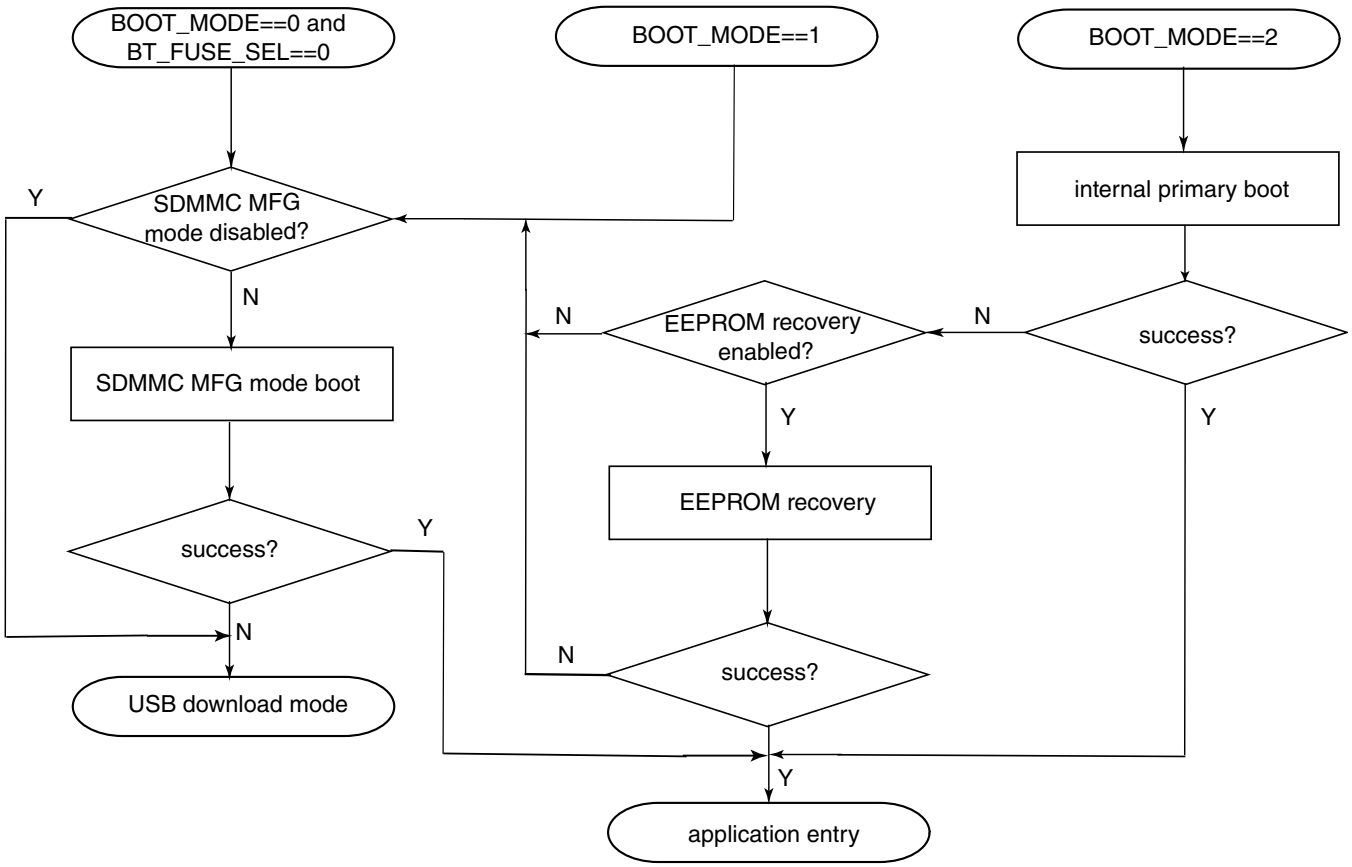


Figure 6-25. SD/MMC manufacture boot flow

6.1.12 High-Assurance Boot (HAB)

The High Assurance Boot (HAB) component of the ROM protects against the potential threat of attackers modifying the areas of code or data in the programmable memory to make it behave in an incorrect manner. The HAB also prevents the attempts to gain access to features which must not be available.

The integration of the HAB feature with the ROM code ensures that the chip does not enter an operational state if the existing hardware security blocks detected a condition that may be a security threat or if the areas of memory deemed to be important were modified. The HAB uses the RSA digital signatures to enforce these policies.

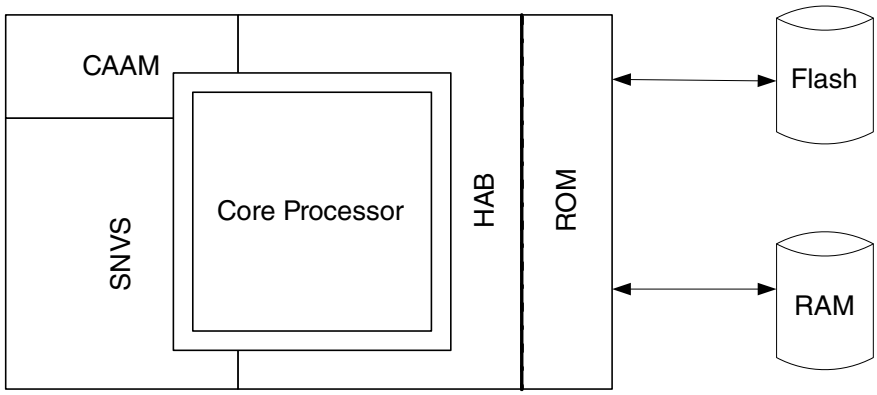


Figure 6-26. Secure boot components

The figure above illustrates the components used during a secure boot using HAB. The HAB interfaces with the SNVS to make sure that the system security state is as expected. The HAB also uses the CAAM hardware block to accelerate the SHA-256 message digest operations performed during the signature verifications and AES-128 operations for the encrypted boot operations. The HAB also includes a software implementation of SHA-256 for cases where a hardware accelerator can't be used. The RSA key sizes supported are 1024, 2048, 3072, and 4096 bits. The RSA signature verification operations are performed by a software implementation contained in the HAB library. The main features supported by the HAB are:

- X.509 public key certificate support
- CMS signature format support
- Proprietary encrypted boot support. Note that the encrypted boot depends on the CAAM hardware module. When the CAAM is disabled (when the EXPORT_CONTROL fuse is blown), the encrypted boot is not available.

NOTE

NXP provides the reference Code Signing Tool (CST) for key generation, certificate generation, and code signing for use with the HAB library. The CST can be found by searching for "IMX_CST_TOOL" at http://www.nxp.com.

NOTE

For further details on using the secure boot feature using HAB, refer to *Secure Boot on i.MX 50, i.MX 53, i.MX 6 and i.MX 7 Series using HABv4 (AN4581).*

6.1.12.1 HAB API vector table addresses

For devices that perform a secure boot, the HAB library may be called by the boot stages that execute after the ROM code.

NOTE

For additional information on the secure boot including the HAB API, refer to *HABv4 RVT Guidelines and Recommendations (AN12263)*.

6.1.13 Boot information for software

To address the requirement that the boot image may need to get the basic boot information when getting out of the boot process, the boot software information (Boot_SW_Info) is provisioned by the ROM.

The software must read the ROM address 0x9e8 to get the base address of the Boot_SW_Info data structure, and parse the Boot_SW_Info content to get the boot information.

Table 6-40. Boot_SW_Info structure

Offset	Byte3	Byte2	Byte1	Byte0
0x0	Reserved	Boot Device Type	Boot Device Instance	Reserved
0x4	Arm core frequency (in Hz)			
0x8	AXI bus frequency (in Hz)			
0xC	DDR frequency (in Hz)			
0x10	GPT1 input clock frequency (in Hz)			
0x14	Reserved			
0x18				
0x1C				

NOTE

The boot ROM sets the GPT1 in a free-running mode with a 32-kHz input clock.

Boot device type mapping:

- 0x1 - SD card or eSD chip
- 0x2 - MMC card or eMMC chip
- 0x3 - NAND chip

Boot device instance: The instance index of the boot device, starting from 0.

6.2 Fusemap

6.2.1 Boot Fusemap

The following section details the various modes and selection of the required boot devices. A separate map is given for each and every boot device. The device select is specified by BOOT_CFG[14:12] fuses listed below.

Table 6-41. Boot Device Select

Boot Device	BOOT_CFG[14]	BOOT_CFG[13]	BOOT_CFG[12]
SD/eSD	0	0	1
MMC/eMMC	0	1	0
NAND	0	1	1
FlexSPI	1	0	0
SPI/NOR	1	1	0

NOTE

Fuses marked as “Reserved” are reserved for NXP internal (and future) use only. Customers should not attempt to burn these, as the IC behavior may be unpredictable. The reserved fuses can be read as either 0 or 1.

Table 6-42. SD/eSD Boot Fusemap

Addr	7	6	5	4	3	2	1	0
0x470[15:8]	Reserved	Boot Device Select			Port Select: 00 - SD1 01 - SD2 10 - SD3		Power Cycle Enable 0 - Disable 1 - Enable	SD Loopback Clock Source SEL (SDR50 and SDR104 Only) 0 - Through SD pad 1 - Direct
0x470[7:0]	Fast Boot: 0 - Regular 1 - Fast Boot	Reserved		Bus Width: 0 - 1-bit 1 - 4-bit	Speed 000 - Normal/SDR12 001 - High/SDR25 010 - SDR50 011 - SDR104 101 - Reserved for DDR50 Others - Reserved			Reserved

Table 6-43. MMC/eMMC Boot Fusemap

Addr	7	6	5	4	3	2	1	0
0x470[15:8]	Reserved	Boot Device Select			Port Select: 00 - SD1 01 - SD2 10 - SD3		Power Cycle Enable 0 - Disable 1 - Enable	SD Loopback Clock Source SEL (SDR50 and SDR104 Only) 0 - Through SD pad 1 - Direct
0x470[7:0]	Fast Boot: 0 - Regular 1 - Fast Boot	Bus Width: 000 - 1-bit 001 - 4-bit 010 - 8-bit 101 - 4-bit DDR (MMC 4.4) 110 - 8-bit DDR (MMC 4.4) Else - Reserved			Speed 00 - Normal 01 - High 10 - Reserved for HS200 11 - Reserved		USDHC IO VOLTAGE SELECTION For Normal Boot Mode 0 - 3.3V 1 - 1.8V	USDHC IO VOLTAGE SELECTION For Manufacture Mode 0 - 3.3V 1 - 1.8V

Table 6-44. NAND Boot Fusemap

Addr	7	6	5	4	3	2	1	0
0x470[15:8]	Reserved	Boot Device Select			Pages in Block: 00 - 128 01 - 64 10 - 32 11 - 256		NAND Row Address Bytes: 00 - 3 01 - 2 10 - 4 11 - 5	
0x470[7:0]	BT_TOGGLEMODE 0 - Raw NAND 1 - Toggle mode	BOOT_SEARCH_COUNT 00 - 2 01 - 2 10 - 4 11 - 8	Toggle Mode 33MHz Preamble Delay, Read Latency 000 - 16 GPMICLK cycles 001 - 1 GPMICLK cycles 010 - 2 GPMICLK cycles 011 - 3 GPMICLK cycles 100 - 4 GPMICLK cycles 101 - 5 GPMICLK cycles 110 - 6 GPMICLK cycles 111 - 7 GPMICLK cycles 1111- 15 GPMICLK cycles					LBA Reset

Table 6-45. FlexSPI Boot Fusemap

Addr	7	6	5	4	3	2	1	0
0x470[15:8]	Reserved	Boot Device Select			Flash Auto Probe	FLASH TYPE: 000 - Device supports 3B read by default 001 - Device supports 4B read by default 010 - HyperFlash 1V8 011 - HyperFlash 3V3 100 - MXIC Octal DDR		
0x470[7:0]	HOLD TIME: 00 - 500us 01 - 1ms 10 - 3ms 11 - 10ms		FLASH Auto Probe Type		FlexSPI FLASH Dummy Cycle			

Table 6-46. SPI/NOR Boot Fusemap

Addr	7	6	5	4	3	2	1	0
0x470[15:8]	Reserved	Boot Device Select			Port Select: 000 - eCSPI1 001 - eCSPI2 010 - eCSPI3			SPI Addressing: 0 - 3 bytes (24-bit) 1 - 2 bytes (16-bit)
0x470[7:0]	CS select (SPI only) 00 - CS#0 (default) 01 - CS#1 10 - CS#2 11 - CS#3	Reserved						

Table 6-47. Boot Fusemap

Addr	7	6	5	4	3	2	1	0
0x480[7:0]	Reserved				Reserved	FlexSPI FLASH Frequency 000 - 100 MHz 001 - 133 MHz 010 - 166 MHz 011 - 200 MHz 100 - 80 MHz 101 - 20 MHz		

Table continues on the next page...

Table 6-47. Boot Fusemap (continued)

Addr	7	6	5	4	3	2	1	0
						Other - Reserved		
0x480[15:8]	LPB_BOOT (Core/DDR/ Bus) 00/01 - LPB Disable 10 - Div by 2 11 - Div by 4		BT_LPB_POLARITY (GPIO polarity)	L1 I-Cache DISABLE	TZASC_ENABLE	WDOG_ENABLE 0 - Disabled 1 - Enabled	Boot Frequencies (ARM/DDR) 0 - 800 / 800 MHz 1 - 400 / 400 MHz	L1 D-Cache DISABLE
0x480[23:16]	NOC_ID_REMAP_BYPASS	SDP_READ_DISABLE	SDP_DISABLE	FORCE_INTERNAL_BOOT	Reserved	WDOG Timeout Select 000 - 64s 001 - 32s 010 - 16s 011 - 8s 100 - 4s Others - Reserved		
0x480[31:24]	Recovery Port Select 000 - eCSPI1 001 - eCSPI2 010 - eCSPI3			Recovery SPI Addressing 0 - 3-bytes (24-bit) 1 - 2-bytes (16-bit)	Recovery CS select (SPI only) 00 - CS#0 (default)		Recovery Boot Enable 0 - Disabled 1 - Enabled	Reserved
0x490[7:0]	Reserved							
0x490[15:8]	USDHC DLL Select 0 - DLL Slave Mode for 1 - DLL Override Mode	MMC_DLL_DLY[6:0] Delay target for USDHC DLL, it is applied to slave mode target delay or override mode target delay depends on DLL Override fuse bit value.						
0x490[23:16]	Disable SDMMC Manufacture mode 0 - Enable 1 - Disable	USDHC_CMD_OE_PRE_EN (SD/MMC debug)	eMMC 4.4 - RESET TO PRE-IDLE STATE 0 - Enable 1 - Disable	USDHC_PAD_PULL_DOWN 0 - no action 1 - pull down	ENABLE_EMMC_22K_PULLUP 0 - 47K pullup 1 - 22K pullup	USDHC_IOMUX_SION_BIT_ENABLE 0 - Disable 1 - Enable	USDHC Override Pad Settings (using PAD_SETTINGS value)	USDHC DLL_ENABLE 0 - Disable DLL for SD/ eMMC 1 - Enable DLL for SD/ eMMC
0x490[31:24]	USDHCPAD_SETTINGS[7:0]							
0x4A0[7:0]	SD Calibration Step 00 - 1		uSDHC Power Cycle Interval 00 - 20ms		uSDHC Power Cycle Delay 0 - 5ms	uSDHC Power On Polarity 0 - Low	USDHC_PAD_SETTINGS[8]	Fast Boot Acknowledge Disable

Table continues on the next page...

Table 6-47. Boot Fusemap (continued)

Addr	7	6	5	4	3	2	1	0
			01 - 10ms 10 - 5ms 11 - 2.5ms		1 - 2.5ms	1 - High		0 - Boot Ack Disabled 1 - Boot Ack Enabled
0x4A0[15:8]	NAND_READ_CMD_CODE1[7:0]							
0x4A0[23:16]	NAND_READ_CMD_CODE2[7:0]							
0x4A0[31:24]	NAND_PAD_SETTINGS[7:0]							
0x4B0[7:0]	Override NAND Pad Settings (using PAD_SETTINGS value)	GPMI Read DDR DLL Target Value 0000 - 7 0001 - 1 0111 - 0 1111 - 15				NAND Reset time 1.5ms	0x4B0[1:0] NAND Number of Devices: 00 - 1 01 - 2 10 - 4 11 - Reserved	
0x4B0[15:8]	READ_RETRY_SEQ_ID[3:0] 0000 - do not use read retry (RR) sequence embedded in ROM 0001 - Micron 20nm RR sequence 0010 - Toshiba A19nm RR sequence 0011 - Toshiba 19nm RR sequence 0100 - SanDisk 19nm RR sequence 0101 - SanDisk 1ynmRR sequence 0110 - Hynix 20nm A Die RR sequence 0111 - Hynix 26nm RR sequence 1000 - Hynix 20nm B Die RR sequence 1001 - Hynix 20nm C Die RR sequence Others - Reserved				Reserved			
0x4B0[23:16]	Reserved							
0x4B0[31:24]	RNG_TRIM[7:0]							

6.2.2 Lock Fusemap

Table 6-48 describes the functions of various lock fuses.

Table 6-48. Lock Fuses

Addr	7	6	5	4	3	2	1	0
0x400[7:0]	ANALOG_LOCK 1x - OP		MEM_TRIM_LOCK 1x - OP		BOOT_CFG_LOCK 1x - OP		TESTER_LOCK 1x - OP	

Table continues on the next page...

Table 6-48. Lock Fuses (continued)

Addr	7	6	5	4	3	2	1	0
	x1 - WP		x1 - WP		x1 - WP		x1 - WP	
0x400[15:8]	MAC_ADDR_LOCK 1x - OP x1 - WP		USB_ID_LOCK 1x - WP + OP 01 - WP		Reserved	SJC_RESP_LOCK WRP,OP,RDP	SRK_LOCK 1 - WP, OP	Reserved
0x400[23:16]	GP2_LOCK 1x - OP x1 - WP		GP1_LOCK 1x - OP x1 - WP		Reserved		Reserved	MANUFACTURE_KEY_LOCK 1 - RP, WP, OP
0x400[31:24]	Reserved							

NOTE

TESTER_LOCK programmed by NXP / set at factory

6.2.3 Fusemap Descriptions Table

NOTE

Definitions for fuse settings are as follows:

- Unlock - The controlled field can be read, sensed, burned, or overridden in the corresponding OCOTP shadow register.
- Lock - The controlled field can't be read, overridden nor burned.
- Override Protect (OP) - The controlled field can be read, sensed, or burned in the corresponding OCOTP shadow register.
- Write Protect (WP) - The controlled field can be read, sensed, or overridden in the corresponding OCOTP shadow register.
- OP + WP - The controlled field can only be read or sensed in the corresponding OCOTP shadow register. It cannot be burned or overridden in the corresponding OCOTP shadow register.
- Lock - The controlled field cannot be read, burned, or overridden.

Table 6-49. Fusemap Descriptions

Fuse Address	Fuses Name	Number of Fuses	Fuses Function	Setting	Used by
0x400[1:0]	TESTER_LOCK	2	Lock for tester related fuses at 0x400-0x460.	00 - Unlock 10 - OP 01 - WP 11 - OP + WP	OCOTP
0x400[3:2]	BOOT_CFG_LOCK	2	Lock for BOOT related fuses at 0x470-4B0.	00 - Unlock 10 - OP 01 - WP 11 - OP + WP	OCOTP
0x400[8:4]	Reserved	5	Reserved	Reserved	Reserved
0x400[9]	SRK_LOCK	1	Lock for SRK_HASH[255:0] fuses.	0 - Unlock 1 - OP + WP	OCOTP
0x400[10]	SJC_RESP_LOCK	1	Lock for SJC_RESP[55:0] fuses.	0 - Unlock 1 - Lock	OCOTP
0x400[11]	Reserved	1	Reserved	Reserved	Reserved
0x400[13:12]	USB_ID_LOCK	2	Lock for USB_PID and USB_VID fuses.	00 - Unlock 01 - WP 11 - OP + WP	OCOTP
0x400[15:14]	MAC_ADDR_LOCK	2	Lock for MAC_ADDR fuses.	00 - Unlock 10 - OP 01 - WP 11 - OP + WP	OCOTP
0x400[16]	MANUFACTURE_KEY_LOCK	1	Lock for MANUFACTURE_PROTECTIONG_KEY[255:0] fuses.	0 - Unlock 1 - Lock	OCOTP
0x400[17]	Reserved	1	Reserved	Reserved	Reserved
0x400[19:18]	Reserved	2	Reserved	Reserved	Reserved
0x400[21:20]	GP1_LOCK	2	Lock for GP1[63:0] fuses.	00 - Unlock 10 - OP 01 - WP 11 - OP + WP	OCOTP
0x400[23:22]	GP2_LOCK	2	Lock for GP2 fuses.	00 - Unlock 10 - OP 01 - WP 11 - OP + WP	OCOTP
0x400[31:24]	Reserved	8	Reserved	Reserved	Reserved
0x420[31:0]	SJC_CHALL[63:0] / UNIQUE_ID[63:0]	32	The SJC CHALLENGE / Unique ID	-	SJC, SW
0x430	Reserved	32	Reserved	Reserved	Reserved

Table continues on the next page...

Table 6-49. Fusemap Descriptions (continued)

Fuse Address	Fuses Name	Number of Fuses	Fuses Function	Setting	Used by
0x440[3:0]	Reserved	4	Reserved	Reserved	Reserved
0x440[7:4]	Reserved	4	Reserved	Reserved	Reserved
0x440[9:8]	SPEED_GRADING[1:0]	2	Burned by tester program, for indicating IC core speed. (Hot burn may not be used).	<table><tr><th>FCA[5:4]</th><th>FHA[3:2]</th><th>FRAL[1:0]</th><th>MHz</th><th>P/N</th></tr><tr><td>xx</td><td>xx</td><td>00</td><td>800</td><td>08</td></tr><tr><td>xx</td><td>xx</td><td>01</td><td>1200</td><td>12</td></tr><tr><td>xx</td><td>xx</td><td>10</td><td>1600</td><td>16</td></tr><tr><td>xx</td><td>xx</td><td>11</td><td>1800</td><td>18</td></tr></table>	PROD / SW
0x440[31:10]	Reserved	22	Reserved	Reserved	Reserved
0x450[1:0]	NUM_A53_CORES	2	Number of A53 CPU cores available.	00 - 4 cores 01 - Reserved 10 - 2 cores 11 - 1 core	SRC, SJC, SW
0x450[7:2]	Reserved	6	Reserved	Reserved	Reserved
0x450[8]	M4_DISABLE	1	Disable M4 Core.	0 - enabled 1 - disabled	M4
0x450[9]	M4_MPU_DISABLE	1	Disable M4 MPU IP.	0 - enabled 1 - disabled	M4
0x450[10]	M4_FPU_DISABLE	1	Disable M4 FPU IP.	0 - enabled 1 - disabled	M4
0x450[11]	USB_OTG1_DISABLE	1	Disable USB OTG1 IP.	0 - enabled 1 - disabled	USB OTG1
0x450[12]	USB_OTG2_DISABLE	1	Disable USB OTG2 IP.	0 - enabled 1 - disabled	USB OTG2
0x450[15:13]	Reserved	3	Reserved	Reserved	Reserved
0x450[16]	EXPORT_CONTROL	1	Used for disabling CAAM and SNVS encryption	0 - Secure part 1 - Security disable (CAAM encryption disabled)	CAAM, SW(ROM)
0x450[17]	SEC_CONFIG[0]	1	Security Configuration Mode, and block off debugging of security HW, by JTAG.	Combined with SEC_CONFIG[1], provide FAB/Open/Close security states: 00 - FAB (Open)	SJC, SNVS, , SRC, TPSMP

Table continues on the next page...

Table 6-49. Fusemap Descriptions (continued)

Fuse Address	Fuses Name	Number of Fuses	Fuses Function	Setting	Used by
				01 - Open - allows any code to be flashed and executed, even if it has no valid signature. 1x - Closed (Security On) Also - used for 'blocking' debug of security modules, when burned.	
0x450[18]	VPU_G1_DISABLE	1	Disable G1 Decoder in VPU	0 - enabled 1 - disabled	VPU
0x450[19]	VPU_G2_DISABLE	1	Disable G2 Decoder in VPU	0 - enabled 1 - disabled	VPU
0x450[20]	VPU_H1_DISABLE	1	Disable H1 Encoder in VPU	0 - enabled 1 - disabled	VPU
0x450[21]	GPU2D_DISABLE	1	Disable GPU 2D IP.	0 - enabled 1 - disabled	GPU
0x450[22]	PCIE1_DISABLE	1	Disable PCIe-1 IP.	0 - enabled 1 - disabled	PCIE
0x450[23]	Reserved	1	Reserved.	Reserved	Reserved
0x450[24]	GPU3D_DISABLE	1	Disable GPU 3D IP.	0 - enabled 1 - disabled	GPU
0x450[27:25]	Reserved	3	Reserved	Reserved	Reserved
0x450[28]	MIPI_DSI_DISABLE	1	Disable MIPI DSI IP.	0 - enabled 1 - disabled	MIPI DSI
0x450[29]	ENET_DISABLE	1	Disable ENET IP.	0 - enabled 1 - disabled	ENET
0x450[30]	MIPI_CSI_DISABLE	1	Disable MIPI CSI IP.	0 - enabled 1 - disabled	MIPI CSI
0x450[31]	Reserved	1	Reserved	Reserved	Reserved
0x460	Reserved	32	Reserved	Reserved	Reserved
0x470[15:0]	BOOT_CFG	16	BOOT configuration register, Usage varies, depending on selected boot device.	See boot fusemap for details.	SRC SW(ROM)
0x470[24:16]	Reserved	9	Reserved	Reserved	Reserved
0x470[25]	SEC_CONFIG[1]	1	Security Configuration (with SEC_CONFIG[0])	00 - FAB (Open) 01 - Open - allows any code to be flashed and executed, even if it has no valid signature. 1x - Closed (Security On)	SW (ROM), SRC, SNVS, TPSMP

Table continues on the next page...

Table 6-49. Fusemap Descriptions (continued)

Fuse Address	Fuses Name	Number of Fuses	Fuses Function	Setting	Used by
0x470[26]	Reserved	1	Reserved	Reserved	Reserved
0x470[27]	Reserved	1	Reserved	Reserved	Reserved
0x470[28]	BT_FUSE_SEL	1	Determines, whether using fuses for boot configuration, or GPIO /Serial loader.	If boot_mode="00" (Development) 0=Boot mode configuration is taken from GPIOs. 1=Boot mode configuration is taken from fuses. If boot_mode="10" (Production) 0 - Boot using Serial Loader (USB) 1- Boot mode configuration is taken from fuses.	SRC SW(ROM)
0x470[29]	FORCE_COLD_BOOT(SBMR)	1	Force cold boot when A7 core come out of reset. Reflected in SBMR reg of SRC	Fuse Function: 0 – Default behavior allowing a fast recovery from low power modes. That is, the ROM is allowed to jump to the address previously programmed in the SRC persistent register. 1 – Fast recovery path in the ROM is not allowed and a cold boot is always performed. Customers wanting a higher level of security should burn this fuse.	SRC SW(ROM)
0x470[31:30]	Reserved	2	Reserved	Reserved	Reserved
0x480[31:0]	BOOT_CFG_PARAMETER	32	BOOT configuration parameters, Usage varies, depending on selected boot device.	See boot fusemap for details.	SW (ROM)
0x490[31:0]	BOOT_CFG_PARAMETER	32	BOOT configuration parameters, Usage varies, depending on selected boot device.	See boot fusemap for details.	SW (ROM)
0x4A0[31:0]	BOOT_CFG_PARAMETER	32	BOOT configuration parameters, Usage varies, depending on selected boot device.	See boot fusemap for details.	SW (ROM)
0x4B0[31:0]	BOOT_CFG_PARAMETER	32	BOOT configuration parameters, Usage varies, depending on selected boot device.	See boot fusemap for details.	SW (ROM)

Table continues on the next page...

Table 6-49. Fusemap Descriptions (continued)

Fuse Address	Fuses Name	Number of Fuses	Fuses Function	Setting	Used by
0x4C0-0x500	Reserved	384	Reserved	Reserved	Reserved
0x580[31:0]	SRK_HASH[255:0]	256	SRK key, no HW visible lines. NO HW Visible signals available	-	SW (HAB)
0x600[23:0]	SJC_RESP[55:0]	56	Response reference value for the secure JTAG controller	-	SJC
0x610[31:24]	Reserved	8	Reserved	Reserved	Reserved
0x620[15:0]	USB_VID[31:0]	16	USB VID	-	SW
0x620[31:16]	USB_PID[31:0]	16	USB PID	-	SW
0x630	Reserved	32	Reserved	Reserved	Reserved
0x640[15:0]	MAC_ADDR[47:0]	48	Reserved for customers/SW	-	SW
0x650[31:16]	Reserved	48	Reserved	Reserved	Reserved
0x670-0x6F0	Reserved	288	Reserved	Reserved	Reserved
0x700[31:0]	Reserved	256	Reserved	Reserved	Reserved
0x780[31:0]	GP1[63:0]	64	General Purpose fuse register #1	-	SW
0x7A0[31:0]	GP2[63:0]	64	General Purpose fuse register #2	-	SW
0x7C0-0x7F0	Reserved	128	Reserved	Reserved	Reserved
0x800[31:0]	GP5[383:0]	384	General Purpose fuse register #5	-	SW
0x8C0[31:0]	Reserved	128	Reserved	Reserved	Reserved
0x900-0x13F0	Reserved	5632	Reserved	Reserved	Reserved

6.3 On-Chip OTP Controller (OCOTP_CTRL)

6.3.1 Overview

This section contains information describing the requirements for the on-chip eFuse OTP controller along with details about the block functionality and implementation.

In this document, the words "eFuse" and "OTP" are interchangeable. OCOTP refers to the hardware block itself.

6.3.1.1 Features

The OCOTP provides the following features:

- Loading and housing of fuse content into shadow registers.
- Generation of HWV_FUSE (hardware visible fuse bus) and the HWV_REG bus which is made up of volatile PIO register based "fuses". The HWV_REG bits come from the SCS (Software Controllable Signals) register.
- Generation of STICKY_REG which is consist of sticky register bits.
- Provide program-protect and read-protect eFuse.
- Provide override and read protection of shadow register.

6.3.2 Top-Level Symbol and Functional Overview

The figure found here shows the OCOTP system level diagram.

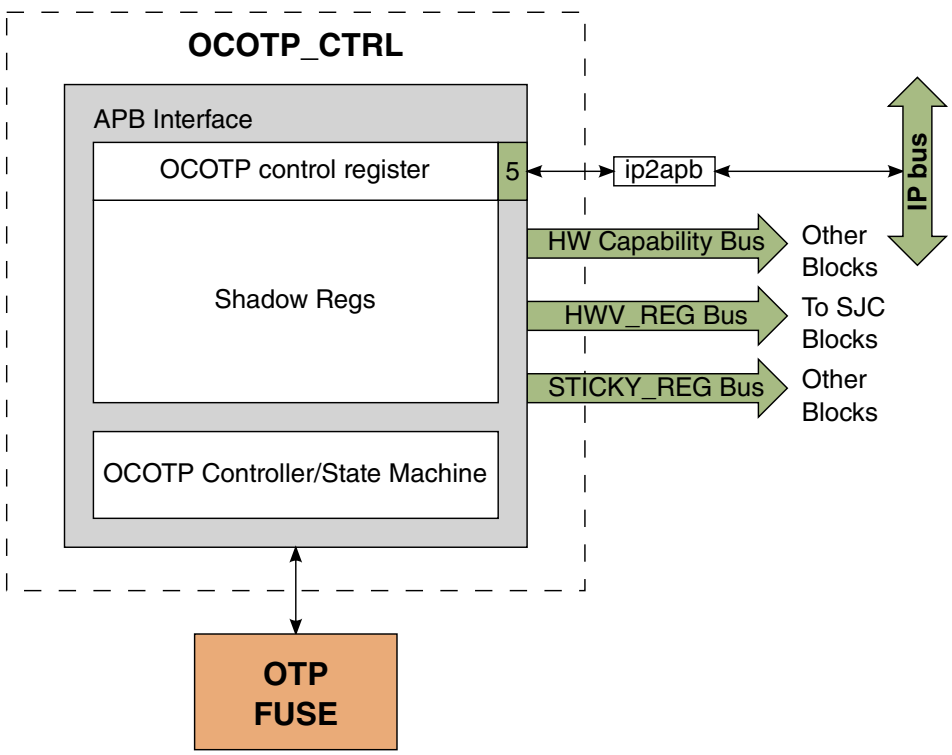


Figure 6-27. OCOTP System Level Diagram

6.3.2.1 Operation

The IP bus interface of the OCOTP provides two functions.

- Configure control registers for programming and reading fuse .
- Override and read shadow registers.

6.3.2.1.1 Shadow Register Reload

All fuse words in are shadowed. Therefore, fuse information is available through memory mapped shadow registers. If fuses are subsequently programmed, the shadow registers should be reloaded to keep them coherent with the fuse bank arrays.

The "reload shadows" feature allows the user to force a reload of the shadow registers (including HW_OCOTP_LOCK) without having to reset the device. To force a reload, complete the following steps:

1. Check that HW_OCOTP_CTRL[BUSY] and HW_OCOTP_CTRL[ERROR] are clear. Overlapped accesses are not supported by the controller. Any pending write , read or reload must be completed before a new access can be requested.
2. Set the HW_OCOTP_CTRL[RELOAD_SHADOWS] bit. OCOTP will read all the fuse one by one and put it into corresponding shadow register.
3. Wait for HW_OCOTP_CTRL[BUSY] and HW_OCOTP_CTRL[RELOAD_SHADOWS] to be cleared by the controller.

The controller will automatically clear the HW_OCOTP_CTRL[RELOAD_SHADOWS] bit after the successful completion of the operation.

6.3.2.1.2 Fuse and Shadow Register Read

All shadow registers are always readable through the APB bus except some secret keys regions. When their corresponding fuse lock bits are set, the shadow registers also become read locked. After read locking, reading from these registers will return 0xBADABADA.

In addition HW_OCOTP_CTRL[ERROR] will be set. It must be cleared by software before any new write , read or reload access can be issued. Subsequent reads to unlocked shadow locations will still work successfully however.

To read fuse words directly from correctly complete the following steps:

1. Check that HW_OCOTP_CTRL[BUSY] and HW_OCOTP_CTRL[ERROR] are clear. Overlapped accesses are not supported by the controller. Any pending write, read or reload must be completed before a read access can be requested.
2. Write the requested to HW_OCOTP_CTRL[ADDR].

6.3.2.1.3 Fuse and Shadow Register Writes

Shadow register bits can be overridden by software until the corresponding fuse lock bit for the region is set. When the lock shadow bit is set, the shadow registers for that lock region become write locked. The LOCK shadow register also has no shadow or fuse lock bits but it is always read only.

In order to avoid "rogue" code performing erroneous writes to OTP, a special unlocking sequence is required for writes to the fuse banks. To program fuse bank correctly complete the following steps:

1. Program HW_OCOTP_TIMING[STROBE_PROG] and fields with timing values to match the current frequency of the ipg_clk. OTP writes will work at maximum bus frequencies as long as the parameters are set correctly.
2. Check that HW_OCOTP_CTRL[BUSY] and HW_OCOTP_CTRL[ERROR] are clear. Overlapped accesses are not supported by the controller. Any pending write or reload must be completed before a write access can be requested.
3. Write the requested to HW_OCOTP_CTRL[ADDR] and program the unlock code into HW_OCOTP_CTRL[WR_UNLOCK]. This must be programmed for each write access. The lock code is documented in the register description. Both the unlock code and address can be written in the same operation.

It should be noted that write latencies to OTP are numbers of 10 micro-seconds. Write latencies is based on amount of bit filed which is 1. For example : program half fuse bits in one word need 10 us x 16.

For further details of OTP read/write operations see [eFUSE].

HW_OCOTP_CTRL[ERROR] will be set under the following conditions:

- A write is performed to a shadow register during a shadow reload (essentially, while HW_OCOTP_CTRL[RELOAD_SHADOWS] is set. In addition, the contents of the shadow register shall not be updated.
- A write is performed to a shadow register which has been locked.
- A read is performed to from a shadow register which has been read locked.
- A program is performed to a fuse which has been .
- A read is performed to from a fuse which has been read locked.

6.3.2.1.4 Write Postamble

Due to internal electrical characteristics of the OTP during writes, all OTP operations following a write must be separated by 2 us after the clearing of HW_OCOTP_CTRL_BUSY following the write. This guarantees programming voltages on-chip to reach a steady state when exiting a write sequence. This includes reads, shadow reloads, or other writes.

A recommended software sequence to meet the postamble requirements is as follows:

- Issue the write and poll for BUSY (as per Fuse Shadow Memory Footprint).
- After BUSY is clear, wait an additional 2 us.
- Perform the next OTP operation.

6.3.2.2 Fuse Shadow Memory Footprint

The OTP memory footprint shows in the following figure. The registers are grouped by lock region. Their names correspond to the PIO register and fusemap names.

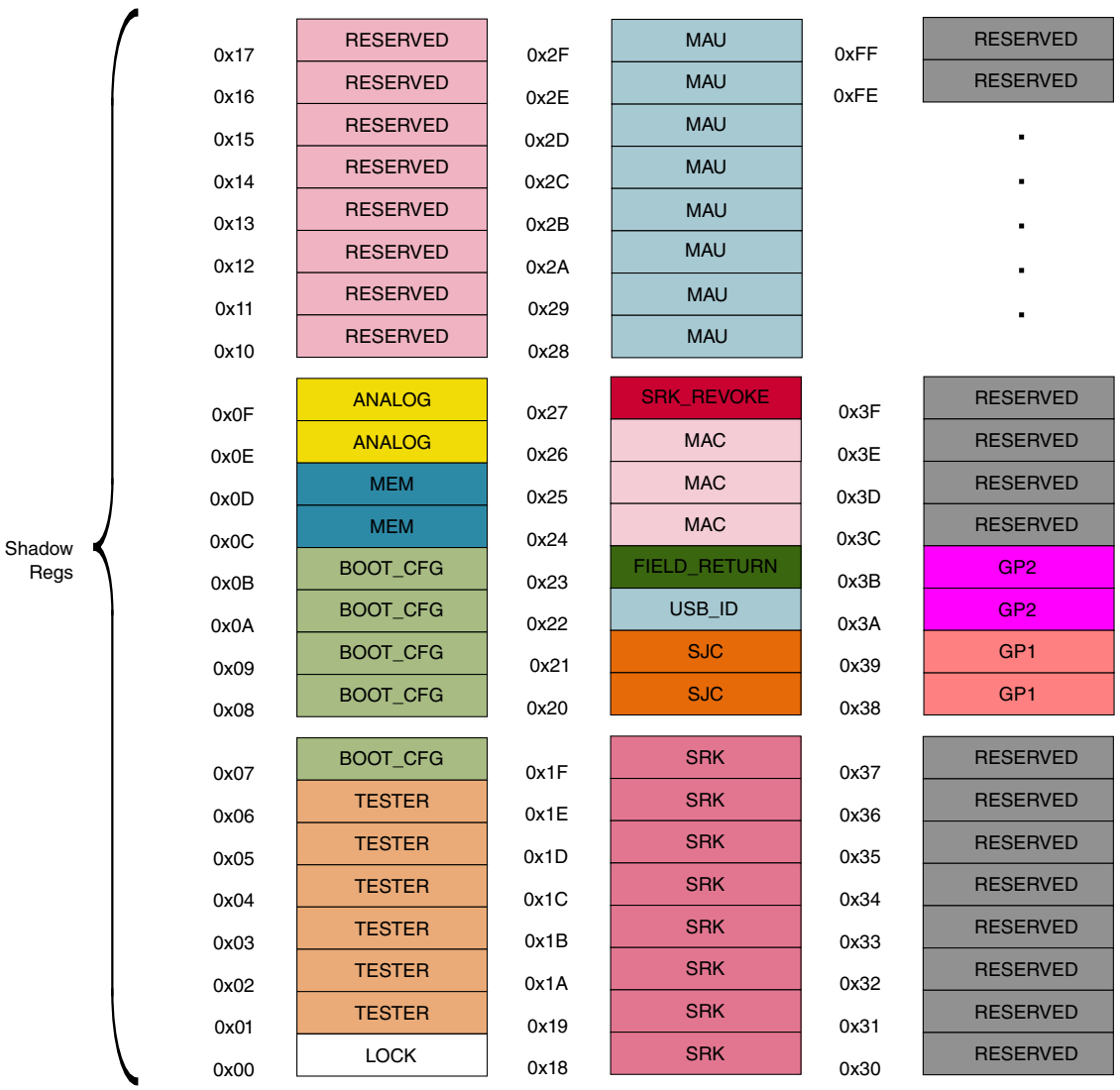


Figure 6-28. OTP Memory Footprint

6.3.2.3 OTP Read/Write Timing Parameters

There are timing fields contained in the HW_OCOTP_TIMING register that specify counter limit values, which are used to specify the signal timing.

Both two timing parameters are specified in ipg_clk cycles. Since the ipg_clk frequency can be set to a range of values, these parameters must be adjusted with the clock to yield the appropriate delay.

6.3.2.4 Hardware Visible Fuses

The hwv_fuse bus emanates from the OCOTP block and goes to various other blocks inside the chip. This bus is made up of the shadow register bits for .

Only a subset of these fuse bits are currently used by the hardware. The fuse bits are initially copied from the banks after reset is deasserted. When all fuse bits are loaded into their shadow registers, the OCOTP asserts the fuse_latched output signal.

The hwv_reg bus also comes from the OCOTP. Its source is the HW_OCOTP_SCS register. This register has 1 defined bit, the HAB_JDE bit, that is connected to the SJC block. The SCS bits are intended to be used as volatile fuse bits under software control. Additional bits will be defined as needed in future implementations.

The system-wide reset sequence must be coordinated by the system reset controller, so that the hwv_fuse and hwv_reg buses are stable and reflect the values of the fuses before they are used by the rest of the system.

6.3.2.5 Behavior During Reset

The OCOTP is always active. The shadow registers automatically load the appropriate OTP contents after reset is deasserted. During this load-time HW_OCOTP_CTRL_BUSY is set. The load time is similar to that of a "reload shadow" operation.

6.3.2.6 Secure JTAG control

The JTAG control fuses are used to allow or disallow JTAG access to secured resources.

Three JTAG security levels are envisioned, as shown in the table below.

Table 6-50. JTAG Security Level Control Bits

Security Mode	JTAG_SMODE	Description
No Debug	2'b11	The highest security level.
Secure JTAG	2'b01	Limit the JTAG access by using key based authentication mechanism.
JTAG Enable	2'b00	Low Security, all JTAG features are enabled.

6.3.3 Fuse Map

See the Fusemap chapter of this reference manual for more information.

6.3.4 OCOTP Memory Map/Register Definition

NOTE

When write/read unimplemented register address in ocotp_ctrl, ocotp_ctrl will not send error and read data will be 0.

OCOTP Hardware Register Format Summary

OCOTP memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3035_0000	OTP Controller Control Register (OCOTP_HW_OCOTP_CTRL)	32	R/W	0000_0000h	6.3.4.1/ 831
3035_0004	OTP Controller Control Register (OCOTP_HW_OCOTP_CTRL_SET)	32	R/W	0000_0000h	6.3.4.1/ 831
3035_0008	OTP Controller Control Register (OCOTP_HW_OCOTP_CTRL_CLR)	32	R/W	0000_0000h	6.3.4.1/ 831
3035_000C	OTP Controller Control Register (OCOTP_HW_OCOTP_CTRL_TOG)	32	R/W	0000_0000h	6.3.4.1/ 831
3035_0010	OTP Controller Timing Register (OCOTP_HW_OCOTP_TIMING)	32	R/W	0400_0000h	6.3.4.2/ 832
3035_0020	OTP Controller Write Data Register (OCOTP_HW_OCOTP_DATA)	32	R/W	0000_0000h	6.3.4.3/ 833
3035_0030	OTP Controller Write Data Register (OCOTP_HW_OCOTP_READ_CTRL)	32	R/W	0000_0000h	6.3.4.4/ 834
3035_0040	OTP Controller Read Data Register (OCOTP_HW_OCOTP_READ_FUSE_DATA)	32	R/W	0000_0000h	6.3.4.5/ 835
3035_0050	Sticky bit Register (OCOTP_HW_OCOTP_SW_STICKY)	32	R/W	0000_0000h	6.3.4.6/ 836

Table continues on the next page...

OCOTP memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3035_0060	Software Controllable Signals Register (OCOTP_HW_OCOTP_SCS)	32	R/W	0000_0000h	6.3.4.7/ 839
3035_0064	Software Controllable Signals Register (OCOTP_HW_OCOTP_SCS_SET)	32	R/W	0000_0000h	6.3.4.7/ 839
3035_0068	Software Controllable Signals Register (OCOTP_HW_OCOTP_SCS_CLR)	32	R/W	0000_0000h	6.3.4.7/ 839
3035_006C	Software Controllable Signals Register (OCOTP_HW_OCOTP_SCS_TOG)	32	R/W	0000_0000h	6.3.4.7/ 839
3035_0090	OTP Controller Version Register (OCOTP_HW_OCOTP_VERSION)	32	R/W	0148_1299h	6.3.4.8/ 840
3035_0400	Value of OTP Bank0 Word0 (Lock controls) (OCOTP_HW_OCOTP_LOCK)	32	R/W	0000_0000h	6.3.4.9/ 841
3035_0410	Value of OTP Bank0 Word1 (Tester Info.) (OCOTP_HW_OCOTP_TESTER0)	32	R/W	0000_0000h	6.3.4.10/ 842
3035_0420	Value of OTP Bank0 Word2 (tester Info.) (OCOTP_HW_OCOTP_TESTER1)	32	R/W	0000_0000h	6.3.4.11/ 843
3035_0430	Value of OTP Bank0 Word3 (Tester Info.) (OCOTP_HW_OCOTP_TESTER2)	32	R/W	0000_0000h	6.3.4.12/ 843
3035_0440	Value of OTP Bank1 Word0 (Tester Info.) (OCOTP_HW_OCOTP_TESTER3)	32	R/W	0000_0000h	6.3.4.13/ 844
3035_0450	Value of OTP Bank1 Word1 (Tester Info.) (OCOTP_HW_OCOTP_TESTER4)	32	R/W	0000_0000h	6.3.4.14/ 844
3035_0460	Value of OTP Bank1 Word2 (Tester Info.) (OCOTP_HW_OCOTP_TESTER5)	32	R/W	0000_0000h	6.3.4.15/ 845
3035_0470	Value of OTP Bank1 Word3 (Boot Configuration Info.) (OCOTP_HW_OCOTP_BOOT_CFG0)	32	R/W	0000_0000h	6.3.4.16/ 845
3035_0480	Value of OTP Bank2 Word0 (Boot Configuration Info.) (OCOTP_HW_OCOTP_BOOT_CFG1)	32	R/W	0000_0000h	6.3.4.17/ 846
3035_0490	Value of OTP Bank2 Word1 (Boot Configuration Info.) (OCOTP_HW_OCOTP_BOOT_CFG2)	32	R/W	0000_0000h	6.3.4.18/ 846
3035_04A0	Value of OTP Bank2 Word2 (Boot Configuration Info.) (OCOTP_HW_OCOTP_BOOT_CFG3)	32	R/W	0000_0000h	6.3.4.19/ 847
3035_04B0	Value of OTP Bank2 Word3 (BOOT Configuration Info.) (OCOTP_HW_OCOTP_BOOT_CFG4)	32	R/W	0000_0000h	6.3.4.20/ 847
3035_04C0	Value of OTP Bank3 Word0 (Memory Related Info.) (OCOTP_HW_OCOTP_MEM_TRIM0)	32	R/W	0000_0000h	6.3.4.21/ 848
3035_04D0	Value of OTP Bank3 Word1 (Memory Related Info.) (OCOTP_HW_OCOTP_MEM_TRIM1)	32	R/W	0000_0000h	6.3.4.22/ 848
3035_04E0	Value of OTP Bank3 Word2 (Analog Info.) (OCOTP_HW_OCOTP_ANA0)	32	R/W	0000_0000h	6.3.4.23/ 849
3035_04F0	Value of OTP Bank3 Word3 (Analog Info.) (OCOTP_HW_OCOTP_ANA1)	32	R/W	0000_0000h	6.3.4.24/ 849
3035_0580	Shadow Register for OTP Bank6 Word0 (SRK Hash) (OCOTP_HW_OCOTP_SRK0)	32	R/W	0000_0000h	6.3.4.25/ 850

Table continues on the next page...

OCOTP memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3035_0590	Shadow Register for OTP Bank6 Word1 (SRK Hash) (OCOTP_HW_OCOTP_SRK1)	32	R/W	0000_0000h	6.3.4.26/ 850
3035_05A0	Shadow Register for OTP Bank6 Word2 (SRK Hash) (OCOTP_HW_OCOTP_SRK2)	32	R/W	0000_0000h	6.3.4.27/ 851
3035_05B0	Shadow Register for OTP Bank6 Word3 (SRK Hash) (OCOTP_HW_OCOTP_SRK3)	32	R/W	0000_0000h	6.3.4.28/ 851
3035_05C0	Shadow Register for OTP Bank7 Word0 (SRK Hash) (OCOTP_HW_OCOTP_SRK4)	32	R/W	0000_0000h	6.3.4.29/ 852
3035_05D0	Shadow Register for OTP Bank7 Word1 (SRK Hash) (OCOTP_HW_OCOTP_SRK5)	32	R/W	0000_0000h	6.3.4.30/ 852
3035_05E0	Shadow Register for OTP Bank7 Word2 (SRK Hash) (OCOTP_HW_OCOTP_SRK6)	32	R/W	0000_0000h	6.3.4.31/ 853
3035_05F0	Shadow Register for OTP Bank7 Word3 (SRK Hash) (OCOTP_HW_OCOTP_SRK7)	32	R/W	0000_0000h	6.3.4.32/ 853
3035_0600	Value of OTP Bank8 Word0 (Secure JTAG Response Field) (OCOTP_HW_OCOTP_SJC_RESP0)	32	R/W	0000_0000h	6.3.4.33/ 854
3035_0610	Value of OTP Bank8 Word1 (Secure JTAG Response Field) (OCOTP_HW_OCOTP_SJC_RESP1)	32	R/W	0000_0000h	6.3.4.34/ 854
3035_0620	Value of OTP Bank8 Word2 (USB ID info) (OCOTP_HW_OCOTP_USB_ID)	32	R/W	0000_0000h	6.3.4.35/ 855
3035_0630	Value of OTP Bank5 Word6 (Field Return) (OCOTP_HW_OCOTP_FIELD_RETURN)	32	R/W	0000_0000h	6.3.4.36/ 855
3035_0640	Value of OTP Bank9 Word0 (MAC Address) (OCOTP_HW_OCOTP_MAC_ADDR0)	32	R/W	0000_0000h	6.3.4.37/ 856
3035_0650	Value of OTP Bank9 Word1 (MAC Address) (OCOTP_HW_OCOTP_MAC_ADDR1)	32	R/W	0000_0000h	6.3.4.38/ 856
3035_0660	Value of OTP Bank9 Word2 (MAC Address) (OCOTP_HW_OCOTP_MAC_ADDR2)	32	R/W	0000_0000h	6.3.4.39/ 857
3035_0670	Value of OTP Bank9 Word3 (SRK Revoke) (OCOTP_HW_OCOTP_SRK_REVOKE)	32	R/W	0000_0000h	6.3.4.40/ 857
3035_0680	Shadow Register for OTP Bank10 Word0 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY0)	32	R/W	0000_0000h	6.3.4.41/ 858
3035_0690	Shadow Register for OTP Bank10 Word1 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY1)	32	R/W	0000_0000h	6.3.4.42/ 858
3035_06A0	Shadow Register for OTP Bank10 Word2 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY2)	32	R/W	0000_0000h	6.3.4.43/ 859
3035_06B0	Shadow Register for OTP Bank10 Word3 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY3)	32	R/W	0000_0000h	6.3.4.44/ 859
3035_06C0	Shadow Register for OTP Bank11 Word0 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY4)	32	R/W	0000_0000h	6.3.4.45/ 860
3035_06D0	Shadow Register for OTP Bank11 Word1 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY5)	32	R/W	0000_0000h	6.3.4.46/ 860
3035_06E0	Shadow Register for OTP Bank11 Word2 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY6)	32	R/W	0000_0000h	6.3.4.47/ 861

Table continues on the next page...

OCOTP memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3035_06F0	Shadow Register for OTP Bank11 Word3 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY7)	32	R/W	0000_0000h	6.3.4.48/ 861
3035_0780	Value of OTP Bank14 Word0 () (OCOTP_HW_OCOTP_GP10)	32	R/W	0000_0000h	6.3.4.49/ 862
3035_0790	Value of OTP Bank14 Word1 () (OCOTP_HW_OCOTP_GP11)	32	R/W	0000_0000h	6.3.4.50/ 862
3035_07A0	Value of OTP Bank14 Word2 () (OCOTP_HW_OCOTP_GP20)	32	R/W	0000_0000h	6.3.4.51/ 862
3035_07B0	Value of OTP Bank14 Word3 () (OCOTP_HW_OCOTP_GP21)	32	R/W	0000_0000h	6.3.4.52/ 863

6.3.4.1 OTP Controller Control Register (OCOTP_HW_OCOTP_CTRL*n*)

The OCOTP Control and Status Register provides the necessary software interface for performing read and write operations to the On-Chip OTP (One-Time Programmable ROM). The control fields such as WR_UNLOCK, ADDR and BUSY/ERROR may be used in conjuction with the HW_OCOTP_DATA register to perform write operations. Read operations to the On-Chip OTP are involving ADDR, BUSY/ERROR bit field and HW_OCOTP_READ_CTRL register. Read value is saved in HW_OCOTP_READ_FUSE_DATA register.

Address: 3035_0000h base + 0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WR_UNLOCK															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					RELOAD_SHADOWS	ERROR	BUSY	ADDR							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

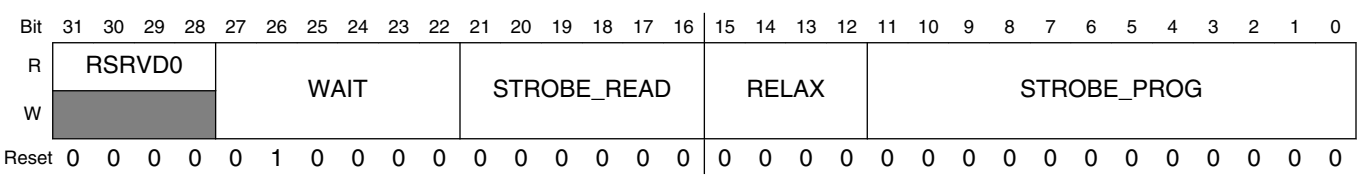
OCOTP_HW_OCOTP_CTRL*n* field descriptions

Field	Description
31–16 WR_UNLOCK	Write 0x3E77 to enable OTP write accesses. NOTE: This register must be unlocked on a write-by-write basis (a write is initiated when HW_OCOTP_DATA is written), so the UNLOCK bitfield must contain the correct key value during all writes to HW_OCOTP_DATA, otherwise a write shall not be initiated. This field is automatically cleared after a successful write completion (clearing of BUSY).
15–11 -	This field is reserved. Reserved
10 RELOAD_ SHADOWS	Set to force re-loading the shadow registers (HW/SW capability and LOCK). This operation will automatically set BUSY. Once the shadow registers have been re-loaded, BUSY and RELOAD_SHADOWS are automatically cleared by the controller.
9 ERROR	Set by the controller when an access to a locked region(OTP or shadow register) is requested. Must be cleared before any further access can be performed. This bit can only be set by the controller. This bit is also set if the Pin interface is active and software requests an access to the OTP. In this instance, the ERROR bit cannot be cleared until the Pin interface access has completed. Reset this bit by writing a one to the SCT clear address space and not by a general write.
8 BUSY	OTP controller status bit. When active, no new write access or read access to OTP(including RELOAD_SHADOWS) can be performed. Cleared by controller when access complete. After reset (or after setting RELOAD_SHADOWS), this bit is set by the controller until the HW/SW and LOCK registers are successfully copied, after which time it is automatically cleared by the controller.
ADDR	OTP write and read access address register. Specifies one of 128 word address locations (0x00 - 0x7f). If a valid access is accepted by the controller, the controller makes an internal copy of this value. This internal copy will not update until the access is complete.

6.3.4.2 OTP Controller Timing Register (OCOTP_HW_OCOTP_TIMING)

This register specifies timing parameters for programming and reading the OCOTP fuse array.

Address: 3035_0000h base + 10h offset = 3035_0010h



OCOTP_HW_OCOTP_TIMING field descriptions

Field	Description
31–28 RSRVD0	These bits always read back zero.
27–22 WAIT	This count value specifies time interval between auto read and write access in one time program. It is given in number of ipg_clk periods.

Table continues on the next page...

OCOTP_HW_OCOTP_TIMING field descriptions (continued)

Field	Description
21–16 STROBE_READ	This count value specifies the strobe period in one time read OTP. Trd = ((STROBE_READ+1)- 2*(RELAX +1)) /ipg_clk_freq. It is given in number of ipg_clk periods.
15–12 RELAX	This count value specifies the time to add to all default timing parameters other than the Tpgm and Trd. It is given in number of ipg_clk periods.
STROBE_PROG	This count value specifies the strobe period in one time write OTP. Tpgm = ((STROBE_PROG+1)- 2*(RELAX+1)) /ipg_clk_freq. It is given in number of ipg_clk periods.

6.3.4.3 OTP Controller Write Data Register (OCOTP_HW_OCOTP_DATA)

This register is used in conjuction with HW_OCOTP_CTRL to perform one-time writes to the OTP.

Address: 3035_0000h base + 20h offset = 3035_0020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

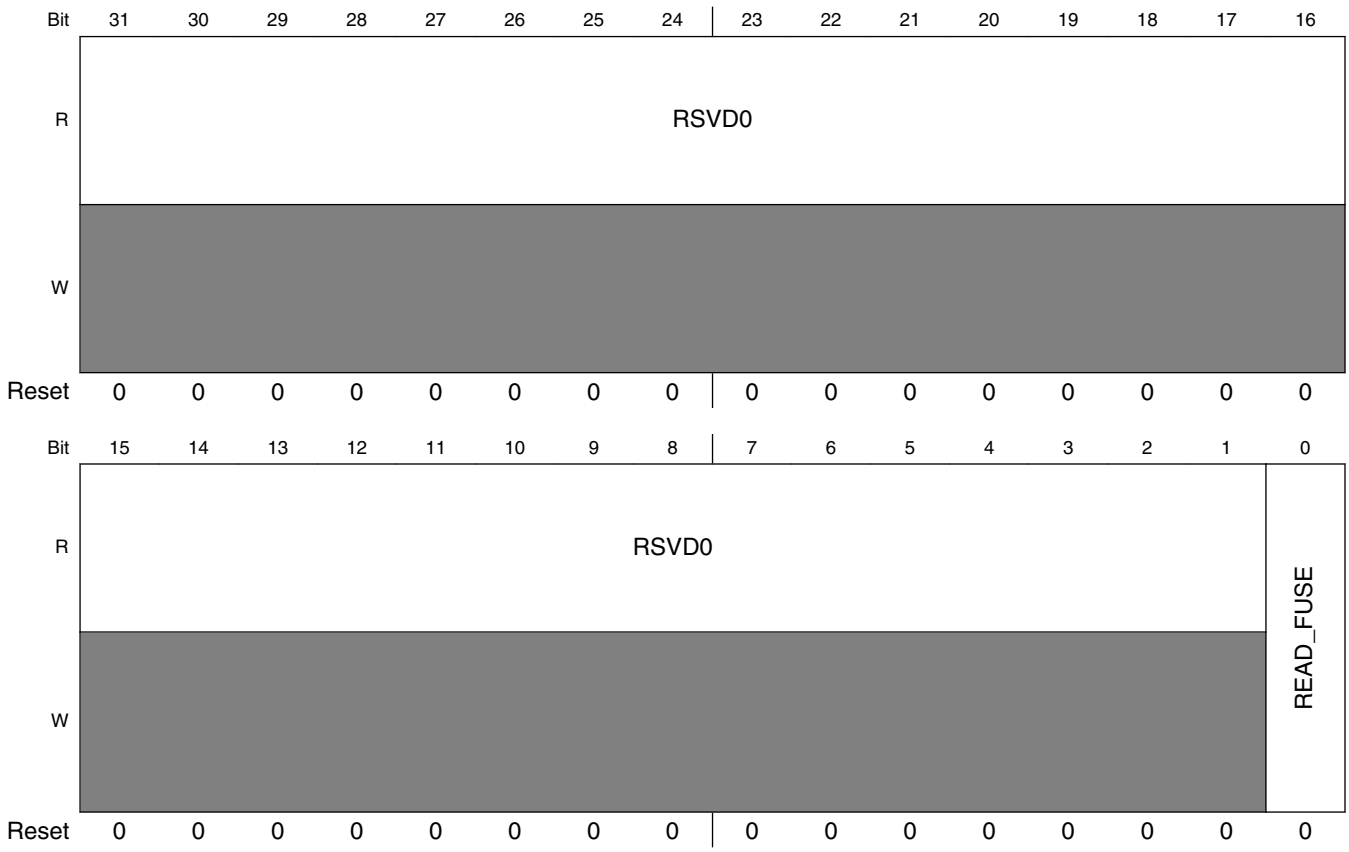
OCOTP_HW_OCOTP_DATA field descriptions

Field	Description
DATA	Used to initiate a write to OTP.

6.3.4.4 OTP Controller Write Data Register (OCOTP_HW_OCOTP_READ_CTRL)

This register is used in conjuction with HW_OCOTP_CTRL to perform one time read to the OTP.

Address: 3035_0000h base + 30h offset = 3035_0030h



OCOTP_HW_OCOTP_READ_CTRL field descriptions

Field	Description
31–1 RSVD0	Reserved
0 READ_FUSE	Used to initiate a read to OTP.

6.3.4.5 OTP Controller Read Data Register (OCOTP_HW_OCOTP_READ_FUSE_DATA)

The data read from OTP

Address: 3035_0000h base + 40h offset = 3035_0040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

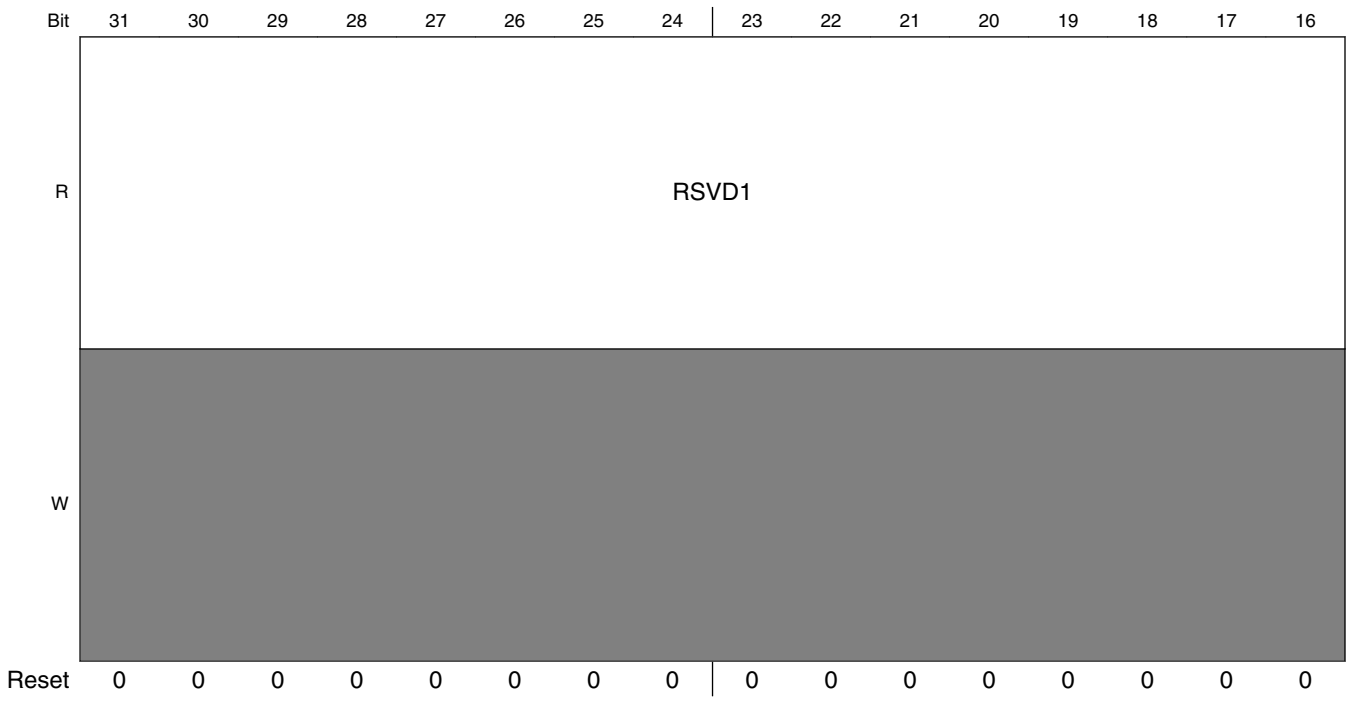
OCOTP_HW_OCOTP_READ_FUSE_DATA field descriptions

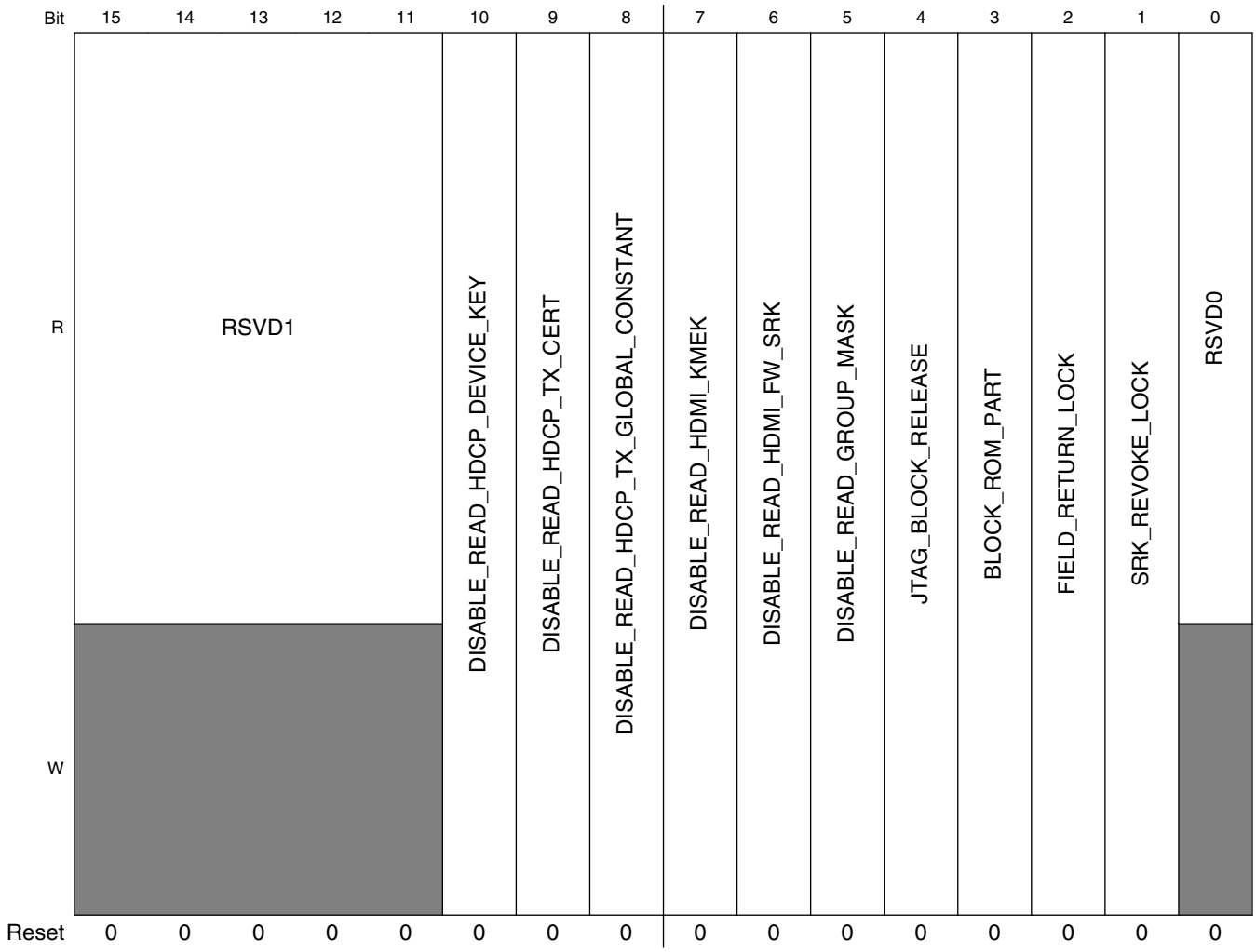
Field	Description
DATA	The data read from OTP

6.3.4.6 Sticky bit Register (OCOTP_HW_OCOTP_SW_STICKY)

Some sticky bits are used by SW to lock some fuse area , shadow registers and other features.

Address: 3035_0000h base + 50h offset = 3035_0050h





OCOTP_HW_OCOTP_SW_STICKY field descriptions

Field	Description
31–11 RSVD1	Reserved
10 DISABLE_ READ_HDCP_ DEVICE_KEY	Shadow register write and OTP write lock for HDCP_DEVICE_HDCP region. When set, the writing of this region's shadow register and OTP fuse word are blocked. Once this bit is set, it is always high unless a POR is issued.
9 DISABLE_ READ_HDCP_ TX_CERT	Shadow register write and OTP write lock for HDCP_TX_CERT region. When set, the writing of this region's shadow register and OTP fuse word are blocked. Once this bit is set, it is always high unless a POR is issued.
8 DISABLE_ READ_HDCP_ TX_GLOBAL_ CONSTANT	Shadow register write and OTP write lock for HDCP_TX_GLOBAL_CONSTANT region. When set, the writing of this region's shadow register and OTP fuse word are blocked. Once this bit is set, it is always high unless a POR is issued.

Table continues on the next page...

OCOTP_HW_OCOTP_SW_STICKY field descriptions (continued)

Field	Description
7 DISABLE_ READ_HDMI_ KMEK	Shadow register write and OTP write lock for HDMI_KMEK region. When set, the writing of this region's shadow register and OTP fuse word are blocked. Once this bit is set, it is always high unless a POR is issued.
6 DISABLE_ READ_HDMI_ FW_SRK	Shadow register write and OTP write lock for HDMI_FW_SRK region. When set, the writing of this region's shadow register and OTP fuse word are blocked. Once this bit is set, it is always high unless a POR is issued.
5 DISABLE_ READ_GROUP_ MASK	Shadow register write and OTP write lock for GROUP_MASK region. When set, the writing of this region's shadow register and OTP fuse word are blocked. Once this bit is set, it is always high unless a POR is issued.
4 JTAG_BLOCK_ RELEASE	Set by Arm during Boot after DTCP is initialized and before test mode entry. * 0 (Default) - JTAG is blocked (subject to other conditions).* 1 - JTAG block is released (subject to other controls). Once this bit is set, it is always high unless a POR is issued.
3 BLOCK_ROM_ PART	Set by Arm during Boot after DTCP is initialized and before test mode entry, if ROM_PART_LOCK=1.* 0 (Default) - Secret part of Boot ROM is not hidden (subject to other conditions).* 1 - Secret part of Boot ROM is hidden.Once this bit is set, it is always high unless a POR is issued.
2 FIELD_ RETURN_LOCK	Shadow register write and OTP write lock for FIELD_RETURN region. When set, the writing of this region's shadow register and OTP fuse word are blocked.Once this bit is set, it is always high unless a POR is issued.
1 SRK_REVOKE_ LOCK	Shadow register write and OTP write lock for SRK_REVOKE region. When set, the writing of this region's shadow register and OTP fuse word are blocked. Once this bit is set, it is always high unless a POR is issued.
0 RSVD0	Reserved

6.3.4.7 Software Controllable Signals Register (OCOTP_HW_OCOTP_SCS*n*)

This register holds volatile configuration values that can be set and locked by trusted software. All values are returned to their defualt values after POR.

Address: 3035_0000h base + 60h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LOCK	SPARE														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SPARE															HAB_JDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_SCS*n* field descriptions

Field	Description
31 LOCK	When set, all of the bits in this register are locked and can not be changed through SW programming. This bit is only reset after a POR is issued.
30–1 SPARE	Unallocated read/write bits for implementation specific software use.
0 HAB_JDE	HAB JTAG Debug Enable. This bit is used by the HAB to enable JTAG debugging, assuming that a properlay signed command to do so is found and validated by the HAB. The HAB must lock the register before passing control to the OS whether or not JTAG debugging has been enabled. Once JTAG is enabled by this bit, it can not be disabled unless the system is reset by POR. 0: JTAG debugging is not enabled by the HAB (it may still be enabled by other mechanisms). 1: JTAG debugging is enabled by the HAB (though this signal may be gated off).

6.3.4.8 OTP Controller Version Register (OCOTP_HW_OCOTP_VERSION)

This register indicates the RTL version in use.

Address: 3035_0000h base + 90h offset = 3035_0090h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								STEP															
W																																
Reset	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0	1	0	0	1	0	1	0	0	1	1	0	0	1

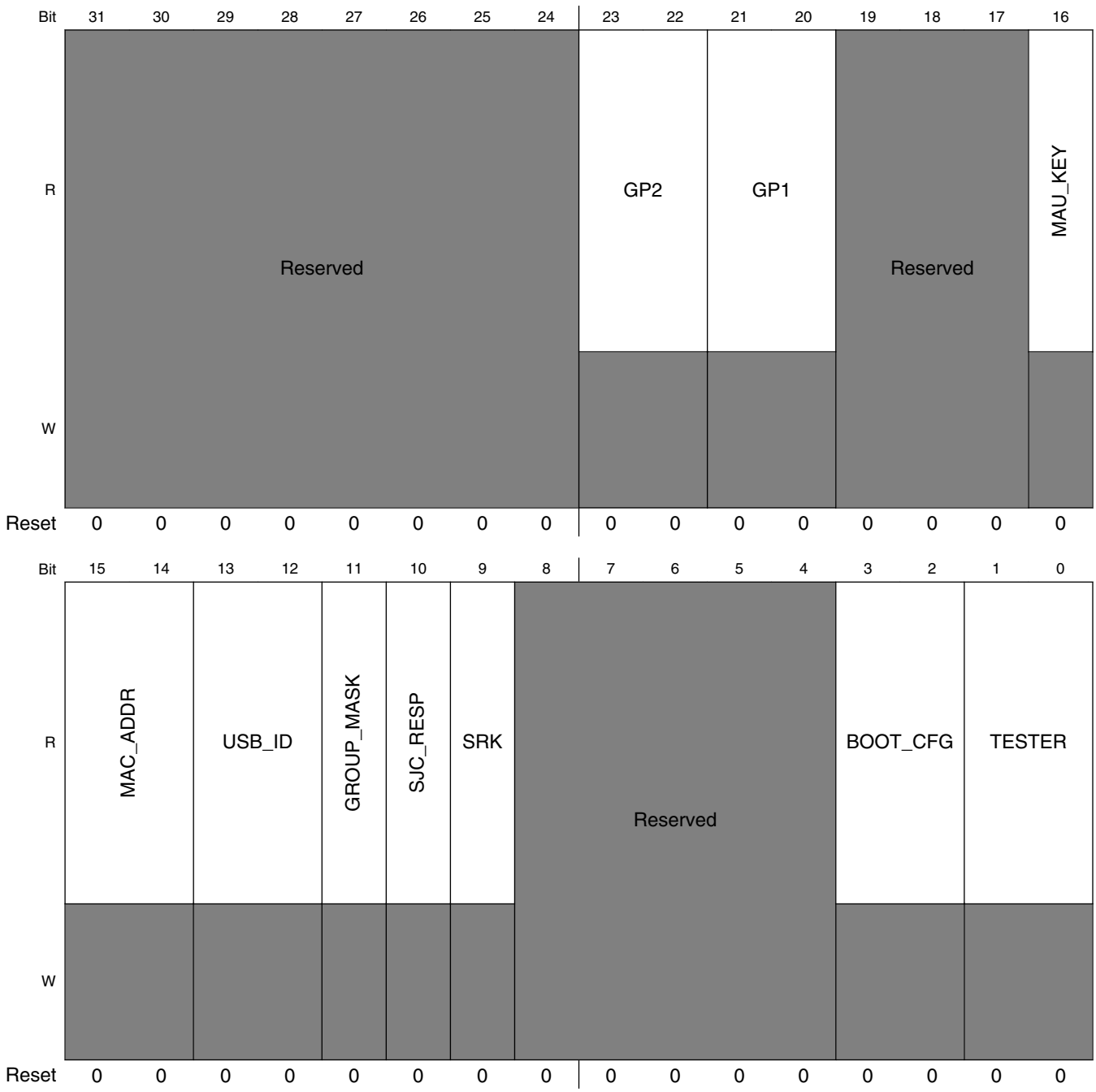
OCOTP_HW_OCOTP_VERSION field descriptions

Field	Description
31–24 MAJOR	Fixed read-only value reflecting the MAJOR field of the RTL version.
23–16 MINOR	Fixed read-only value reflecting the MINOR field of the RTL version.
STEP	Fixed read-only value reflecting the stepping of the RTL version.

6.3.4.9 Value of OTP Bank0 Word0 (Lock controls) (OCOTP_HW_OCOTP_LOCK)

Shadowed memory mapped access to OTP Bank 0, word 0.

Address: 3035_0000h base + 400h offset = 3035_0400h



OCOTP_HW_OCOTP_LOCK field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–22 GP2	Status of shadow register and OTP write lock for gp2 region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.
21–20 GP1	Status of shadow register and OTP write lock for gp1 region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.
19–17 -	This field is reserved. Reserved
16 MAU_KEY	Status of shadow register read and write, OTP read and write lock for manufacture_key region. When set, the writing of this region's shadow register and OTP fuse word are blocked. The read of this region's shadow register and OTP fuse word are also blocked.
15–14 MAC_ADDR	Status of shadow register and OTP write lock for mac_addr region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.
13–12 USB_ID	Status of shadow register and OTP write lock for usb_id region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.
11 GROUP_MASK	Status of shadow register and OTP write lock for group mask region. When set, the writing of this region's shadow register and OTP fuse word are blocked.
10 SJC_RESP	Status of shadow register read and write, OTP read and write lock for sjc_resp region. When set, the writing of this region's shadow register and OTP fuse word are blocked. The read of this region's shadow register and OTP fuse word are also blocked.
9 SRK	Status of shadow register and OTP write lock for srk region. When set, the writing of this region's shadow register and OTP fuse word are blocked.
8–4 -	This field is reserved. Reserved
3–2 BOOT_CFG	Status of shadow register and OTP write lock for boot_cfg region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.
TESTER	Status of shadow register and OTP write lock for tester region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.

6.3.4.10 Value of OTP Bank0 Word1 (Tester Info.) (OCOTP_HW_OCOTP_TESTER0)

Shadowed memory mapped access to OTP Bank 0, word 1.

Address: 3035_0000h base + 410h offset = 3035_0410h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

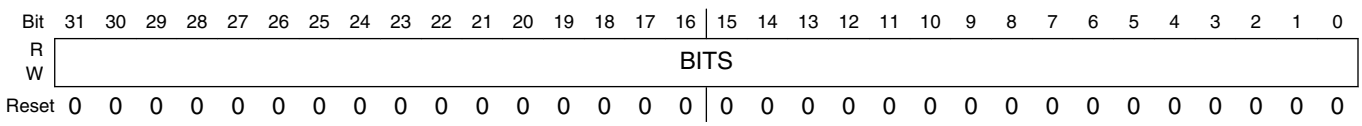
OCOTP_HW_OCOTP_TESTER0 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 0, word 1. These bits become read-only after the HW_OCOTP_LOCK_TESTER[1] bit is set.

6.3.4.11 Value of OTP Bank0 Word2 (tester Info.) (OCOTP_HW_OCOTP_TESTER1)

shadowed memory mapped access to OTP Bank 0, word 2.

Address: 3035_0000h base + 420h offset = 3035_0420h



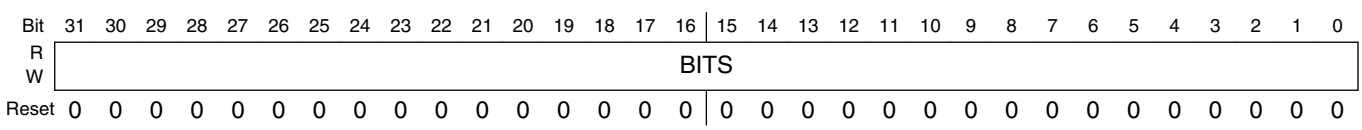
OCOTP_HW_OCOTP_TESTER1 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 0, word 2. These bits become read-only after the HW_OCOTP_LOCK_TESTER[1] bit is set.

6.3.4.12 Value of OTP Bank0 Word3 (Tester Info.) (OCOTP_HW_OCOTP_TESTER2)

Shadowed memory mapped access to OTP Bank 0, word 3.

Address: 3035_0000h base + 430h offset = 3035_0430h



OCOTP_HW_OCOTP_TESTER2 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 0, word 3. These bits become read-only after the HW_OCOTP_LOCK_TESTER[1] bit is set.

6.3.4.13 Value of OTP Bank1 Word0 (Tester Info.) (OCOTP_HW_OCOTP_TESTER3)

Non-shadowed memory mapped access to OTP Bank 1, word 0.

Address: 3035_0000h base + 440h offset = 3035_0440h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_TESTER3 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 1, word 0. These bits become read-only after the HW_OCOTP_LOCK_TESTER[1] bit is set.

6.3.4.14 Value of OTP Bank1 Word1 (Tester Info.) (OCOTP_HW_OCOTP_TESTER4)

Shadowed memory mapped access to OTP Bank 1, word 1.

Address: 3035_0000h base + 450h offset = 3035_0450h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_TESTER4 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 1, word 1. These bits become read-only after the HW_OCOTP_LOCK_TESTER[1] bit is set.

6.3.4.15 Value of OTP Bank1 Word2 (Tester Info.) (OCOTP_HW_OCOTP_TESTER5)

Shadowed memory mapped access to OTP Bank 1, word 2.

Address: 3035_0000h base + 460h offset = 3035_0460h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_TESTER5 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 1, word 2. These bits become read-only after the HW_OCOTP_LOCK_TESTER[1] bit is set.

6.3.4.16 Value of OTP Bank1 Word3 (Boot Configuration Info.) (OCOTP_HW_OCOTP_BOOT_CFG0)

Shadowed memory mapped access to OTP Bank 1, word 3.

Address: 3035_0000h base + 470h offset = 3035_0470h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_BOOT_CFG0 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 1, word 3. These bits become read-only after the HW_OCOTP_LOCK_BOOT_CFG[1] bit is set.

6.3.4.17 Value of OTP Bank2 Word0 (Boot Configuration Info.) (OCOTP_HW_OCOTP_BOOT_CFG1)

Shadowed memory mapped access to OTP bank 2, word 0.

Address: 3035_0000h base + 480h offset = 3035_0480h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_BOOT_CFG1 field descriptions

Field	Description
BITS	Reflects value of OTP bank 2, word 0. These bits become read-only after the HW_OCOTP_LOCK_BOOT_CFG[1] bit is set.

6.3.4.18 Value of OTP Bank2 Word1 (Boot Configuration Info.) (OCOTP_HW_OCOTP_BOOT_CFG2)

Shadowed memory mapped access to OTP bank 2, word 1.

Address: 3035_0000h base + 490h offset = 3035_0490h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_BOOT_CFG2 field descriptions

Field	Description
BITS	Reflects value of OTP bank 2, word 1. These bits become read-only after the HW_OCOTP_LOCK_BOOT_CFG[1] bit is set.

6.3.4.19 Value of OTP Bank2 Word2 (Boot Configuration Info.) (OCOTP_HW_OCOTP_BOOT_CFG3)

Shadowed memory mapped access to OTP bank 2, word 2.

Address: 3035_0000h base + 4A0h offset = 3035_04A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_BOOT_CFG3 field descriptions

Field	Description
BITS	Reflects value of OTP bank 2, word 2. These bits become read-only after the HW_OCOTP_LOCK_BOOT_CFG[1] bit is set.

6.3.4.20 Value of OTP Bank2 Word3 (BOOT Configuration Info.) (OCOTP_HW_OCOTP_BOOT_CFG4)

Shadowed memory mapped access to OTP bank 2, word 3.

Address: 3035_0000h base + 4B0h offset = 3035_04B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_BOOT_CFG4 field descriptions

Field	Description
BITS	Reflects value of OTP bank 2, word 3. These bits become read-only after the HW_OCOTP_LOCK_BOOT_CFG[1] bit is set.

6.3.4.21 Value of OTP Bank3 Word0 (Memory Related Info.) (OCOTP_HW_OCOTP_MEM_TRIM0)

Shadowed memory mapped access to OTP bank 3, word 0.

Address: 3035_0000h base + 4C0h offset = 3035_04C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_MEM_TRIM0 field descriptions

Field	Description
BITS	Reflects value of OTP bank 3, word 0. These bits become read-only after the HW_OCOTP_LOCK_MEM_TRIM[1] bit is set.

6.3.4.22 Value of OTP Bank3 Word1 (Memory Related Info.) (OCOTP_HW_OCOTP_MEM_TRIM1)

Shadowed memory mapped access to OTP bank 3, word 1.

Address: 3035_0000h base + 4D0h offset = 3035_04D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_MEM_TRIM1 field descriptions

Field	Description
BITS	Reflects value of OTP bank 3, word 1. These bits become read-only after the HW_OCOTP_LOCK_MEM_TRIM[1] bit is set.

6.3.4.23 Value of OTP Bank3 Word2 (Analog Info.) (OCOTP_HW_OCOTP_ANA0)

Shadowed memory mapped access to OTP bank 3, word 2.

Address: 3035_0000h base + 4E0h offset = 3035_04E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_ANA0 field descriptions

Field	Description
BITS	Reflects value of OTP bank 3, word 2. These bits become read-only after the HW_OCOTP_LOCK_ANALOG[1] bit is set.

6.3.4.24 Value of OTP Bank3 Word3 (Analog Info.) (OCOTP_HW_OCOTP_ANA1)

Shadowed memory mapped access to OTP bank 3, word 3.

Address: 3035_0000h base + 4F0h offset = 3035_04F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

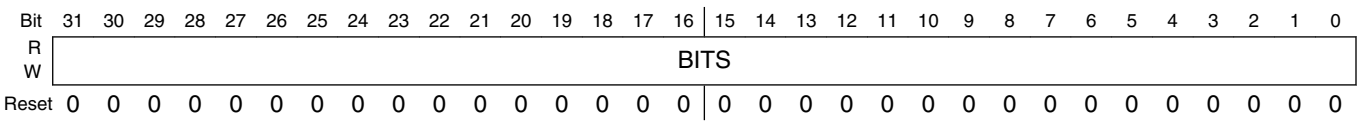
OCOTP_HW_OCOTP_ANA1 field descriptions

Field	Description
BITS	Reflects value of OTP bank 3, word 3. These bits become read-only after the HW_OCOTP_LOCK_ANALOG[1] bit is set.

6.3.4.25 Shadow Register for OTP Bank6 Word0 (SRK Hash) (OCOTP_HW_OCOTP_SRK0)

Shadowed memory mapped access to OTP Bank 6, word 0.

Address: 3035_0000h base + 580h offset = 3035_0580h



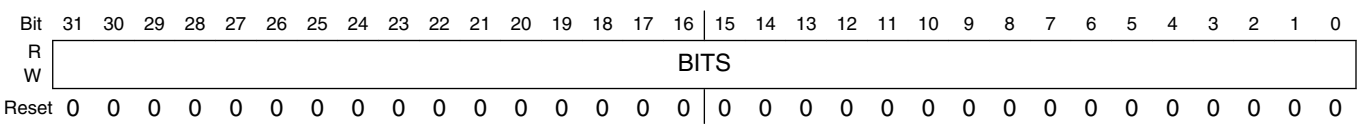
OCOTP_HW_OCOTP_SRK0 field descriptions

Field	Description
BITS	Shadow register for the hash of the Super Root Key word0 (Copy of OTP Bank 6, word 0). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

6.3.4.26 Shadow Register for OTP Bank6 Word1 (SRK Hash) (OCOTP_HW_OCOTP_SRK1)

Shadowed memory mapped access to OTP Bank 6, word 1.

Address: 3035_0000h base + 590h offset = 3035_0590h



OCOTP_HW_OCOTP_SRK1 field descriptions

Field	Description
BITS	Shadow register for the hash of the Super Root Key word1 (Copy of OTP Bank 6, word 1). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

6.3.4.27 Shadow Register for OTP Bank6 Word2 (SRK Hash) (OCOTP_HW_OCOTP_SRK2)

Shadowed memory mapped access to OTP Bank 6, word 2.

Address: 3035_0000h base + 5A0h offset = 3035_05A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_SRK2 field descriptions

Field	Description
BITS	Shadow register for the hash of the Super Root Key word2 (Copy of OTP Bank 6, word 2). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

6.3.4.28 Shadow Register for OTP Bank6 Word3 (SRK Hash) (OCOTP_HW_OCOTP_SRK3)

Shadowed memory mapped access to OTP Bank 6, word 3.

Address: 3035_0000h base + 5B0h offset = 3035_05B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

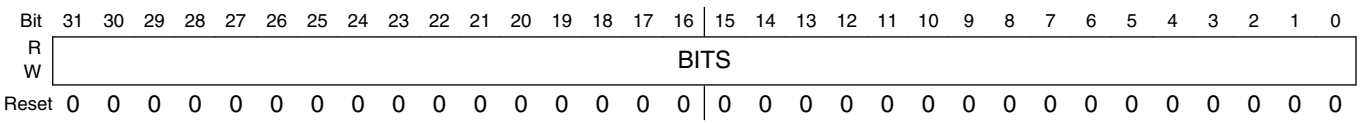
OCOTP_HW_OCOTP_SRK3 field descriptions

Field	Description
BITS	Shadow register for the hash of the Super Root Key word3 (Copy of OTP Bank 6, word 3). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

6.3.4.29 Shadow Register for OTP Bank7 Word0 (SRK Hash) (OCOTP_HW_OCOTP_SRK4)

Shadowed memory mapped access to OTP Bank 7, word 0.

Address: 3035_0000h base + 5C0h offset = 3035_05C0h



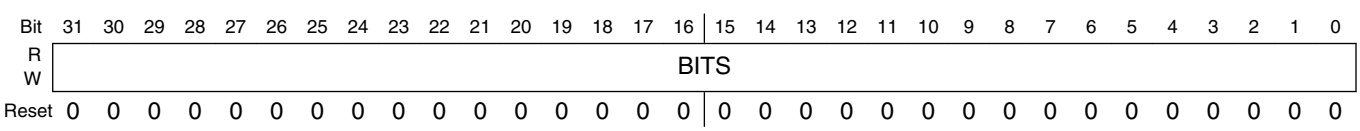
OCOTP_HW_OCOTP_SRK4 field descriptions

Field	Description
BITS	Shadow register for the hash of the Super Root Key word4 (Copy of OTP Bank 7, word 0). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

6.3.4.30 Shadow Register for OTP Bank7 Word1 (SRK Hash) (OCOTP_HW_OCOTP_SRK5)

Shadowed memory mapped access to OTP Bank 7, word 1.

Address: 3035_0000h base + 5D0h offset = 3035_05D0h



OCOTP_HW_OCOTP_SRK5 field descriptions

Field	Description
BITS	Shadow register for the hash of the Super Root Key word5 (Copy of OTP Bank 7, word 1). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

6.3.4.31 Shadow Register for OTP Bank7 Word2 (SRK Hash) (OCOTP_HW_OCOTP_SRK6)

Shadowed memory mapped access to OTP Bank 7, word 2.

Address: 3035_0000h base + 5E0h offset = 3035_05E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_SRK6 field descriptions

Field	Description
BITS	Shadow register for the hash of the Super Root Key word6 (Copy of OTP Bank 7, word 2). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

6.3.4.32 Shadow Register for OTP Bank7 Word3 (SRK Hash) (OCOTP_HW_OCOTP_SRK7)

Shadowed memory mapped access to OTP Bank 7, word 3.

Address: 3035_0000h base + 5F0h offset = 3035_05F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_SRK7 field descriptions

Field	Description
BITS	Shadow register for the hash of the Super Root Key word7 (Copy of OTP Bank 7, word 3). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

6.3.4.33 Value of OTP Bank8 Word0 (Secure JTAG Response Field) (OCOTP_HW_OCOTP_SJC_RESP0)

Shadowed memory mapped access to OTP Bank 8, word 0.

Address: 3035_0000h base + 600h offset = 3035_0600h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_SJC_RESP0 field descriptions

Field	Description
BITS	Shadow register for the SJC_RESP Key word0 (Copy of OTP Bank 8, word 0). These bits can be not read and wrotten after the HW_OCOTP_LOCK_SJC_RESP bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

6.3.4.34 Value of OTP Bank8 Word1 (Secure JTAG Response Field) (OCOTP_HW_OCOTP_SJC_RESP1)

Shadowed memory mapped access to OTP Bank 8, word 1.

Address: 3035_0000h base + 610h offset = 3035_0610h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_SJC_RESP1 field descriptions

Field	Description
BITS	Shadow register for the SJC_RESP Key word1 (Copy of OTP Bank 8, word 1). These bits can be not read and wrotten after the HW_OCOTP_LOCK_SJC_RESP bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

6.3.4.35 Value of OTP Bank8 Word2 (USB ID info) (OCOTP_HW_OCOTP_USB_ID)

Shadowed memory mapped access to OTP Bank 8, word 2.

Address: 3035_0000h base + 620h offset = 3035_0620h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_USB_ID field descriptions

Field	Description
BITS	Reflects value of OTP Bank 8, word 2.

6.3.4.36 Value of OTP Bank5 Word6 (Field Return) (OCOTP_HW_OCOTP_FIELD_RETURN)

Shadowed memory mapped access to OTP Bank 8, word 3.

Address: 3035_0000h base + 630h offset = 3035_0630h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

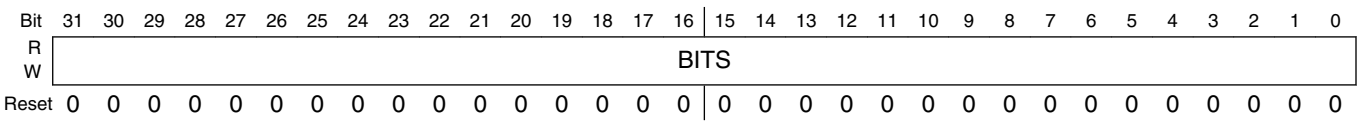
OCOTP_HW_OCOTP_FIELD_RETURN field descriptions

Field	Description
BITS	Reflects value of OTP Bank 8, word 3.

6.3.4.37 Value of OTP Bank9 Word0 (MAC Address) (OCOTP_HW_OCOTP_MAC_ADDR0)

Shadowed memory mapped access to OTP Bank 9, word 0.

Address: 3035_0000h base + 640h offset = 3035_0640h



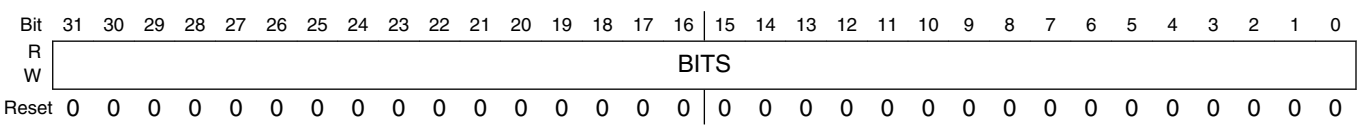
OCOTP_HW_OCOTP_MAC_ADDR0 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 9, word 0.

6.3.4.38 Value of OTP Bank9 Word1 (MAC Address) (OCOTP_HW_OCOTP_MAC_ADDR1)

Shadowed memory mapped access to OTP Bank 9, word 1.

Address: 3035_0000h base + 650h offset = 3035_0650h



OCOTP_HW_OCOTP_MAC_ADDR1 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 9, word 1.

6.3.4.39 Value of OTP Bank9 Word2 (MAC Address) (OCOTP_HW_OCOTP_MAC_ADDR2)

Shadowed memory mapped access to OTP Bank 9, word 2.

Address: 3035_0000h base + 660h offset = 3035_0660h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_MAC_ADDR2 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 9, word 2.

6.3.4.40 Value of OTP Bank9 Word3 (SRK Revoke) (OCOTP_HW_OCOTP_SRK_REVOKE)

Shadowed memory mapped access to OTP Bank 9, word 3.

Address: 3035_0000h base + 670h offset = 3035_0670h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_SRK_REVOKE field descriptions

Field	Description
BITS	Reflects value of OTP Bank 9, word 3.

6.3.4.41 Shadow Register for OTP Bank10 Word0 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY0)

Shadowed memory mapped access to OTP Bank 10, word 0.

Address: 3035_0000h base + 680h offset = 3035_0680h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_MAU_KEY0 field descriptions

Field	Description
BITS	Shadow register for the MAU Key word0 (Copy of OTP Bank 10, word 0). These bits can be not read and wrotten after the HW_OCOTP_LOCK_MAU_KEY bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

6.3.4.42 Shadow Register for OTP Bank10 Word1 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY1)

Shadowed memory mapped access to OTP Bank 10, word 1.

Address: 3035_0000h base + 690h offset = 3035_0690h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_MAU_KEY1 field descriptions

Field	Description
BITS	Shadow register for the MAU Key word1 (Copy of OTP Bank 10, word 1). These bits can be not read and wrotten after the HW_OCOTP_LOCK_MAU_KEY bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

6.3.4.43 Shadow Register for OTP Bank10 Word2 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY2)

Shadowed memory mapped access to OTP Bank 10, word 2.

Address: 3035_0000h base + 6A0h offset = 3035_06A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_MAU_KEY2 field descriptions

Field	Description
BITS	Shadow register for the MAU Key word2 (Copy of OTP Bank 10, word 2). These bits can be not read and wrotten after the HW_OCOTP_LOCK_MAU_KEY bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

6.3.4.44 Shadow Register for OTP Bank10 Word3 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY3)

Shadowed memory mapped access to OTP Bank 10, word 3.

Address: 3035_0000h base + 6B0h offset = 3035_06B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

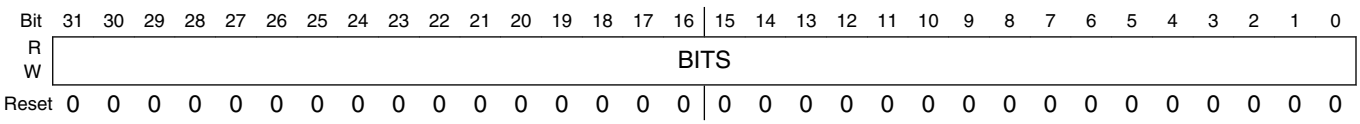
OCOTP_HW_OCOTP_MAU_KEY3 field descriptions

Field	Description
BITS	Shadow register for the MAU Key word3 (Copy of OTP Bank 10, word 3). These bits can be not read and wrotten after the HW_OCOTP_LOCK_MAU_KEY bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

6.3.4.45 Shadow Register for OTP Bank11 Word0 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY4)

Shadowed memory mapped access to OTP Bank 11, word 0.

Address: 3035_0000h base + 6C0h offset = 3035_06C0h



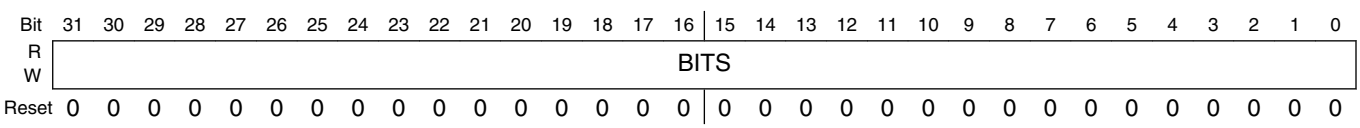
OCOTP_HW_OCOTP_MAU_KEY4 field descriptions

Field	Description
BITS	Shadow register for the MAU Key word4 (Copy of OTP Bank 11, word 0). These bits can be not read and wrotten after the HW_OCOTP_LOCK_MAU_KEY bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

6.3.4.46 Shadow Register for OTP Bank11 Word1 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY5)

Shadowed memory mapped access to OTP Bank 11, word 1.

Address: 3035_0000h base + 6D0h offset = 3035_06D0h



OCOTP_HW_OCOTP_MAU_KEY5 field descriptions

Field	Description
BITS	Shadow register for the MAU Key word5 (Copy of OTP Bank 11, word 1). These bits can be not read and wrotten after the HW_OCOTP_LOCK_MAU_KEY bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

6.3.4.47 Shadow Register for OTP Bank11 Word2 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY6)

Shadowed memory mapped access to OTP Bank 11, word 2.

Address: 3035_0000h base + 6E0h offset = 3035_06E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_MAU_KEY6 field descriptions

Field	Description
BITS	Shadow register for the MAU Key word6 (Copy of OTP Bank 11, word 2). These bits can be not read and wrotten after the HW_OCOTP_LOCK_MAU_KEY bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

6.3.4.48 Shadow Register for OTP Bank11 Word3 (MAU Key) (OCOTP_HW_OCOTP_MAU_KEY7)

Shadowed memory mapped access to OTP Bank 11, word 3.

Address: 3035_0000h base + 6F0h offset = 3035_06F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

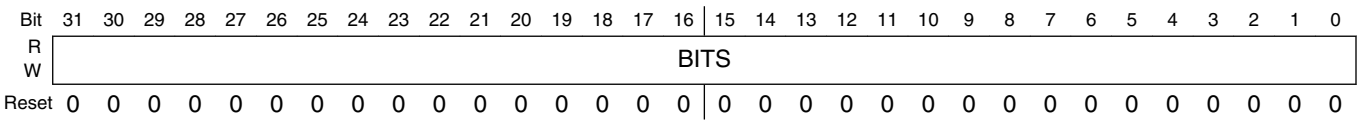
OCOTP_HW_OCOTP_MAU_KEY7 field descriptions

Field	Description
BITS	Shadow register for the MAU Key word7 (Copy of OTP Bank 11, word 3). These bits can be not read and wrotten after the HW_OCOTP_LOCK_MAU_KEY bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

6.3.4.49 Value of OTP Bank14 Word0 () (OCOTP_HW_OCOTP_GP10)

Shadowed memory mapped access to OTP Bank 14, word 0.

Address: 3035_0000h base + 780h offset = 3035_0780h



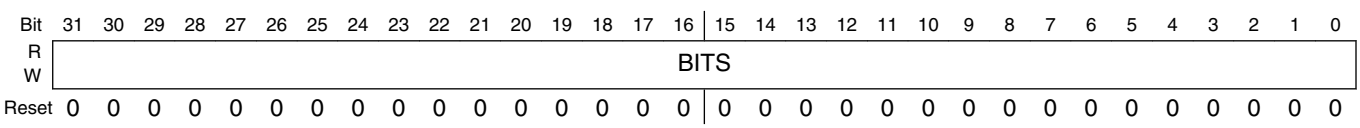
OCOTP_HW_OCOTP_GP10 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 14, word 0.

6.3.4.50 Value of OTP Bank14 Word1 () (OCOTP_HW_OCOTP_GP11)

Shadowed memory mapped access to OTP Bank 14, word 1.

Address: 3035_0000h base + 790h offset = 3035_0790h



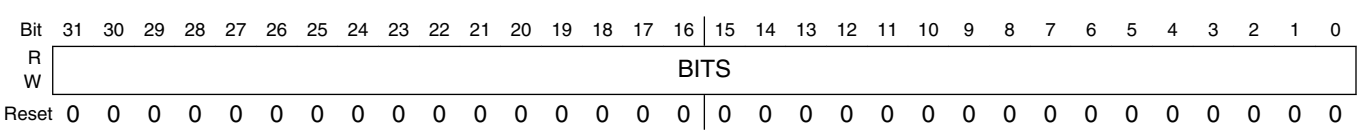
OCOTP_HW_OCOTP_GP11 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 14, word 1.

6.3.4.51 Value of OTP Bank14 Word2 () (OCOTP_HW_OCOTP_GP20)

Shadowed memory mapped access to OTP Bank 14, word 2.

Address: 3035_0000h base + 7A0h offset = 3035_07A0h



OCOTP_HW_OCOTP_GP20 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 14, word 2.

6.3.4.52 Value of OTP Bank14 Word3 () (OCOTP_HW_OCOTP_GP21)

Shadowed memory mapped access to OTP Bank 14, word 3.

Address: 3035_0000h base + 7B0h offset = 3035_07B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OCOTP_HW_OCOTP_GP21 field descriptions

Field	Description
BITS	Reflects value of OTP Bank 14, word 3.

6.4 Secure Non-Volatile Storage (SNVS)

6.4.1 SNVS introduction

SNVS is a companion module to the CAAM module.

SNVS incorporates both security and non-security functionality. The SNVS non-security functionality is described in this document, but the SNVS security functionality is described only in the Security Reference Manual.

SNVS non-security functions:

- Realtime Counter (RTC) - a software accessible realtime counter
  - RTC can be set to the value in the SRTC
- General Purpose Register - a set of registers used to hold 128 bits of data specified by software

  - If the SNVS_LP power input is connected to an uninterrupted power supply, the GPR value is maintained when main SoC is powered off
- Chip power-on/power-off - If the SNVS_LP power input is connected to an uninterrupted power supply and the Power On button input signal is connected to a power button external to the chip, logic within SNVS_LP can be used to wake the chip from a power down.

6.4.1.1 SNVS feature list

The following table summarizes the features of SNVS:

Table 6-51. SNVS feature list

Feature	Description	Links for Further Information
Real time counter (RTC)	• The RTC is driven by a dedicated clock, which is off when the system power is down. • Programmable time alarm interrupt	SNVS_HP Real Time Counter
General-purpose register	• The general-purpose register is available to software to store 128 bits of data. • The general-purpose register is zeroized when a security violation is detected. • If the SNVS_LP power input is connected to an uninterrupted power supply (see SNVS power domains), the general-purpose register value is retained even if the main chip is powered down.	Using the General-Purpose Register
Register access restrictions	• Some registers/values can be written only once per boot cycle.	privileged and non-privileged registers
Wakeup from power off	• Input signal from off chip requests SNVS_LP to power on the main SoC (Assuming that the SNVS_LP power input is connected to an uninterrupted power supply (see SNVS power domains). • Hardware debounces the input signal using software-specified signal bounce characteristics	LP Wake-Up Interrupt Enable

6.4.1.2 SNVS functional description

SNVS implements several non-security features that involve software interaction:

- reading or writing the Realtime Counter (RTC) (This is a non-privileged operation.) - software can also instruct SNVS to load the current SRTC value into the RTC
- reading or writing the General Purpose Register (GPR) (Note that there may be a significant delay when reading or writing registers in the LP section if the LP clock is different from the HP clock.)

The following sections describe in more detail the operation of SNVS.

6.4.2 SNVS Structure

SNVS is organized as two major sub-modules:

- Low-Power Section of SNVS (SNVS_LP)

  The SNVS_LP section provides hardware that enables secure storage and protection of sensitive data. The SNVS module is designed to safely hold security-related data such as cryptographic key, time counter, monotonic counter, and general purpose security information.

  The SNVS_LP block implements the following functional units:
  - Control and Status Registers
  - General Purpose Registers

  When the LP section is connected to an uninterrupted power supply the state of these registers is maintained even when the main chip power is off. (see SNVS power domains)

- High-Power Section of SNVS (SNVS_HP)

  The SNVS_HP section contains all SNVS status and configuration registers. It implements all features that enable system communication and provisioning of the SNVS_LP section.

  The SNVS_HP provides an interface between SNVS_LP and the rest of the system.

  The SNVS_HP block implements the following functional units:
  - IP Bus Interface
  - SNVS_LP Interface
  - Zeroizable Master Key Programming Mechanism
  - Real Time Counter with Alarm Control and Status Registers
  - Control and status registers

  SNVS_HP is in the chip's power supply domain and thus receives power along with the rest of the chip.

The following figure illustrates the structure of SNVS.

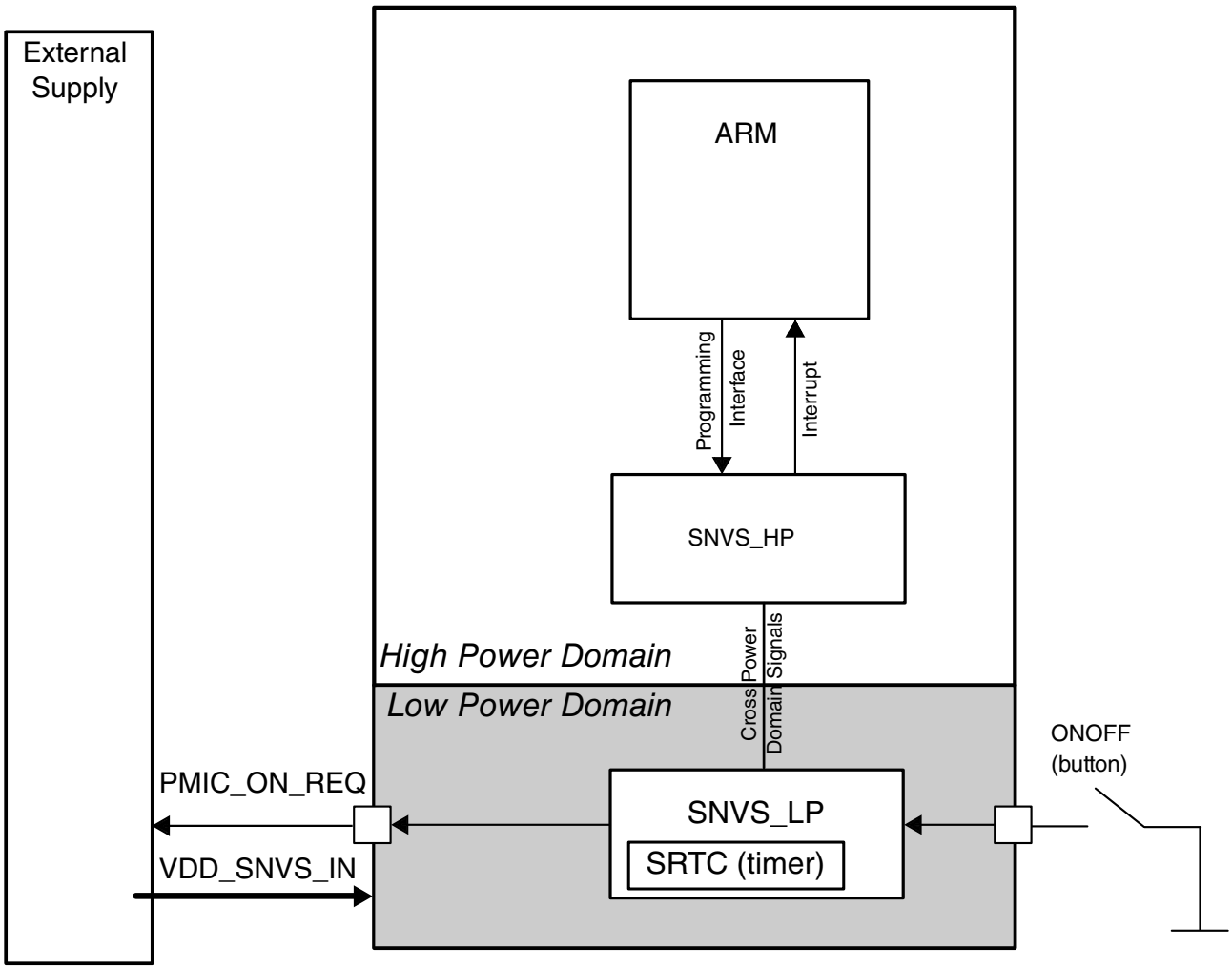


Figure 6-29. SNVS Block Diagram

6.4.2.1 SNVS power domains

In some versions of SNVS (including this version), the LP (Low Power) section is implemented in an independent power domain from the HP (High Power) section, and most other logic on the chip. Throughout the SNVS documentation whenever mention is made of "always-on" logic, this assumes a version of SNVS that implements an independent power domain for the LP section, and that the power for this section is supplied by an uninterrupted power supply. The purpose for the independent power domain is so that data can be retained and certain logic can remain functional even when the main chip logic is powered down. But this is possible only if the LP domain remains powered via an uninterrupted power supply when the main chip power domain is powered off. Usually this uninterrupted power supply would be a battery, with possibly some power management logic to power the LP section from main power (and perhaps

recharge the battery) when main power is on, and switch to battery power when the main power is off. In versions of SNVS with an independent LP power domain the LP section can be electrically isolated from the rest of the chip logic to ensure that its logic does not get corrupted when the main chip is powered down. If the battery runs down or is removed, an LP POR will occur when the LP section next powers up. Note that some OEMs may choose to connect LP power to HP/main chip power and dispense with a battery. In that case the SNVS will operate the same as an SNVS without an independent LP power domain. No state will be retained in the LP section when the chip is powered down, and an LP POR will occur whenever there is an HP POR.

6.4.2.2 Power glitch detector (PGD)

SNVS_LP incorporates a mechanism to detect glitches on the SNVS_LP power supply that might cause the LP control, status, and secure counter values to change. The mechanism works as follows:

1. The PGD register (LPPGDR) is loaded with the known specific value 4173_6166h as part of the SNVS initialization process.
2. Subsequently, this register's value is continuously compared to the hardwired value 4173_6166h.
3. If the comparison indicates that any bit has changed, a power glitch violation is asserted.

Power glitch detection is always enabled and cannot be disabled. At LP POR this register is reset to all 0's, so the hardwired comparison fails and a power glitch violation is reported. Therefore, before programming any feature in the SNVS the power glitch violation should be cleared. The initialization software should write the proper value (4173_6166h) into LPPGDR (see SNVS_LP Power Glitch Detector Register (LPPGDR)) and should then clear the power glitch record in the LP status register (see SNVS_LP Status Register (LPSR)).

The following figure shows the PGD mechanism.

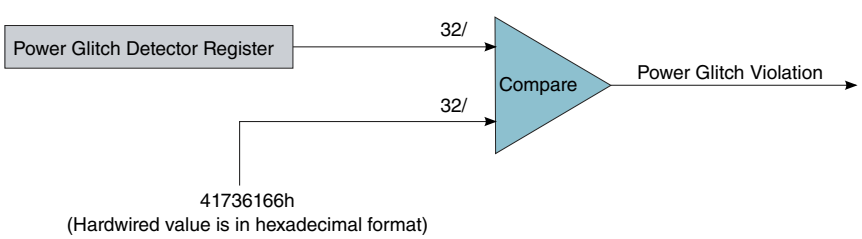


Figure 6-30. Power glitch detector

6.4.2.3 SNVS clock sources

The SNVS has the following clock sources:

- System peripheral clock input. This clock is used by the SNVS's internal logic, for example, the Security State Machine. This clock can be gated outside of the module when the SNVS indicates that it is not in use.
- HP RTC clock. This clock is used by SNVS_HP real-time counter. This clock does not need to be synchronous with other clocks.

6.4.3 Runtime Procedures

SNVS implements a number of features that are intended to be accessed by software at runtime (as opposed to accessed at boot time). These features include:

- Real Time Clock (see SNVS_HP Real Time Counter)
- General Purpose Register (see Using the General-Purpose Register)

Procedures for using these features are described in the following sections.

6.4.3.1 Using SNVS Timer Facilities

SNVS incorporates timer facilities that can optionally generate an interrupt at a specified time. As described in the following sections, SNVS_HP incorporates a Real Time Counter that is available for general use, and SNVS_LP incorporates a Secure Real Time Counter intended for security applications.

6.4.3.1.1 SNVS_HP Real Time Counter

SNVS_HP implements a real time counter that can be read or written by any application; it has no privileged software access restrictions. When the chip is powered down the RTC is not active and it is reset at chip POR. The RTC can be used to generate a functional interrupt request either at a specific time, or at a specific frequency, or both. To generate an interrupt request at a specific time HPTA_EN is set to 0, the desired time is written to HPTA_MS and HPTA_LS and then HPTA_EN is set to 1. HPTA_EN, HPTA_MS and HPTA_LS can be written by any software that has access to SNVS registers; there are no privileged access restrictions. The counter can be synchronized to the SNVS_LP SRTC by writing to the HP_TS bit of SNVS_HP Control Register. This is particularly useful if the SNVS_LP is powered from an uninterrupted power source because the RTC can then be set from a chip-internal time source.

6.4.3.1.2 RTC/SRTC control bits setting

All SNVS registers are programmed from the register bus, consequently any software-initiated changes are synchronized with the IP clock. Several registers can also change synchronously with the RTC/SRTC clock after they are programmed. To avoid IP clock and RTC/SRTC clock synchronization issues, the following values can be changed only when the corresponding function is disabled.

Table 6-52. RTC/SRTC synchronized values list

Function	Value/register	Control bit setting
HP section		
HP Real Time Counter	HPRTCMR and HPRTCLR Registers	RTC_EN = 0 : HPRTCMR/HPRTCLR **can** be programmed RTC_EN = 1 : HPRTCMR/HPRTCLR **cannot** be programmed
HP Time Alarm	HPTAMR and HPTALR Registers	HPTA_EN = 0 : HPTAMR/HPTALR **can** be programmed HPTA_EN = 1 : HPTAMR/HPTALR **cannot** be programmed
LP section		
LP Secure Real Time Counter	LPRTCMR and LPRTCLR Registers	SRTC_ENV = 0 : LPRTCMR/LPRTCLR **can** be programmed SRTC_ENV = 1 : LPRTCMR/LPRTCLR **cannot** be programmed
LP Time Alarm	LPTAR Register	LPTA_EN = 0 : LPTAR **can** be programmed LPTA_EN = 1 : LPTAR **cannot** be programmed

Use the following steps to program synchronized values:

1. Check the enable bit value. If set, clear it.

2. Verify that the enable bit is cleared. There are two reasons to verify the enable bit's setting:

- Enable bit clearing does not happen immediately; it takes three IP clock cycles and two RTC/SRTC clock cycles to change the enable bit's value.
- If the enable bit is locked for programming, it cannot be cleared.

3. Program the desired value.

4. Set the enable bit; it takes three IP clock cycles and two RTC/SRTC clock cycles for the bit to set.

NOTE

Incrementing the value programmed into RTC/SRTC registers by two compensates for the two RTC/SRTC clock cycle delay that is required to enable the counter.

6.4.3.1.3 Reading RTC and SRTC values

Software should follow the following procedure to ensure that it has read correct data from the RTC (HPRTCMR and HPRTCLR) and SRTC (LPSRTCMR and LPSRTCLR) registers:

- Read the most-significant half and the least-significant half of the RTC/SRTC and then read both halves again. If the values read are the same both times, the value is correct.
- If the two consecutive pairs of reads yield different results, perform two more reads.

The worst case scenario may require three sessions of two consecutive pairs of reads. There are several reasons that the values may be incorrectly read initially:

- Synchronization issues between the RTC/SRTC clock and the system clock
- Since the counter continues to increment, there may be a carry from the least-significant 32-bits to the most-significant bits in between reading the two halves of the counter

6.4.3.2 Using Other SNVS Registers

The sections below describe how to use the General Purpose Register.Monotonic Counter.The sections below describe how to use the General Purpose Register and the Monotonic Counter.

6.4.3.2.1 Using the General-Purpose Register

SNVS implements a 128-bit general-purpose register allows software to store a small amount of data. To maintain backward compatibility with versions of SNVS that implement only a 32-bit general purpose register, the most-significant word of the general purpose register is aliased to the original legacy address, and to maintain backward compatibility with versions of snvs_module_name that implement a 128-bit general purpose register, the most-significant half of the general purpose register is aliased to the previous legacy address address. The data in the GPR will be retained during system power-down mode as long as the SNVS_LP remains powered by an uninterrupted power source.

6.4.4 Reset and Initialization of SNVS

SNVS is implemented in two sections (HP and LP) that both must be initialized by software. If the SNVS_LP is powered by an uninterrupted power source that is separate from main SoC power, then SNVS can operate in either of two modes, depending upon whether the main SoC power is on or off. During main SoC power-down SNVS_HP is powered-down, but SNVS_LP is powered from the backup power supply and is electrically isolated from the rest of the chip. In this mode SNVS_LP keeps its registers' values but the LP registers cannot be read or written. During main SoC power-up the isolation of SNVS_LP is disabled and both SNVS_HP and SNVS_LP are powered from the main SoC power. Both LP and HP registers can be read and written (locks and privilege modes permitting). Signals between the SNVS_HP and SNVS_LP sections are enabled and all SNVS functions are operational.

Since the HP and LP sections reside in different power domains, the POR for the two sections can occur at different times. If the SNVS_LP section remains powered by an uninterrupted power source when the main SoC power is off, SNVS_LP is initialized rarely, typically once when the device is first powered on and again whenever the battery is replaced. During main SoC power-up the isolation of SNVS_LP is disabled and both SNVS_HP and SNVS_LP are powered from the main SoC power. Signals between the SNVS_HP and SNVS_LP sections are enabled and all SNVS functions are operational. The SNVS_HP section is powered from the main SoC power, so it must be initialized after the device is powered on.

6.4.5 SNVS register descriptions

This section contains detailed register descriptions for the SNVS registers. Each description includes a standard register diagram and register table. The register table provides detailed descriptions of the register bit and field functions, in bit order.

SNVS registers consist of two types:

- Privileged read/write accessible
- Non-privileged read/write accessible

Privileged read/write accessible registers can only be accessed for read/write by privileged software. Unauthorized write accesses are ignored, and unauthorized read accesses return zero. Non-privileged software can access privileged access registers when the non-privileged software access enable bit is set in the SNVS_HP Command Register.

- Non-Secure
- Trusted
- Secure

Non-privileged read/write accessible registers are read/write accessible by any software.

The LP register values are set only on LP POR and are unaffected by System (HP) POR. The HP registers are set only on System POR and are unaffected by LP POR.

The following table shows the SNVS main memory map.

NOTE

For more information on security-related bitfields, see the Security Reference Manual.

6.4.5.1 SNVS Memory map

SNVS base address: 3037_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	SNVS_HP Command Register (HPCOMR)	32	RW	0000_0000h
8h	SNVS_HP Control Register (HPCR)	32	RW	0000_0000h
14h	SNVS_HP Status Register (HPSR)	32	RW	8000_0000h
24h	SNVS_HP Real Time Counter MSB Register (HPRTCMR)	32	RW	0000_0000h
28h	SNVS_HP Real Time Counter LSB Register (HPRTCLR)	32	RW	0000_0000h
2Ch	SNVS_HP Time Alarm MSB Register (HPTAMR)	32	RW	0000_0000h
30h	SNVS_HP Time Alarm LSB Register (HPTALR)	32	RW	0000_0000h
34h	SNVS_LP Lock Register (LPLR)	32	RW	0000_0000h
38h	SNVS_LP Control Register (LPCR)	32	RW	0000_0020h
4Ch	SNVS_LP Status Register (LPSR)	32	RW	0000_0008h
5Ch	SNVS_LP Secure Monotonic Counter MSB Register (LPSMCMR)	32	RW	0000_0000h
60h	SNVS_LP Secure Monotonic Counter LSB Register (LPSMCLR)	32	RW	0000_0000h
64h	SNVS_LP Power Glitch Detector Register (LPPGDR)	32	RW	0000_0000h
68h	SNVS_LP General Purpose Register 0 (legacy alias) (LPGPR0_legacy_alias)	32	RW	0000_0000h
90h - 9Ch	SNVS_LP General Purpose Registers 0 .. 3 (LPGPR0_alias - LPGPR3_alias)	32	RW	0000_0000h
100h - 10Ch	SNVS_LP General Purpose Registers 0 .. 3 (LPGPR0 - LPGPR3)	32	RW	0000_0000h
BF8h	SNVS_HP Version ID Register 1 (HPVIDR1)	32	RO	003E_0103h
BFCh	SNVS_HP Version ID Register 2 (HPVIDR2)	32	RO	0600_0300h

6.4.5.2 SNVS_HP Command Register (HPCOMR)

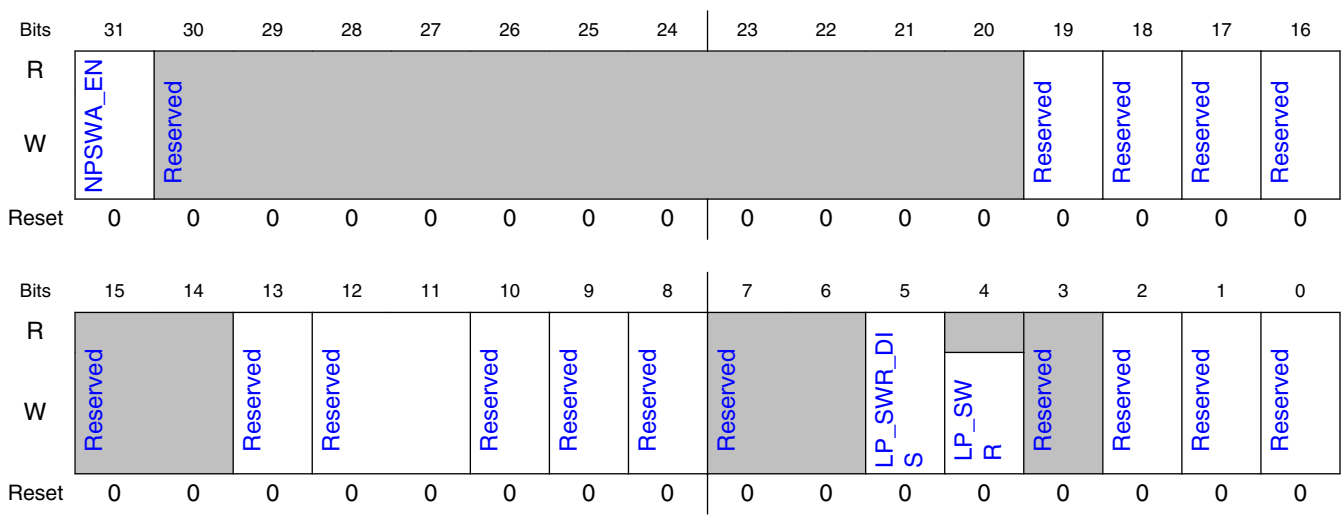
6.4.5.2.1 Offset

Register	Offset
HPCOMR	4h

6.4.5.2.2 Function

The SNVS_HP Command Register contains the command, configuration, and control bits for the SNVS block. This is a privileged write register.

6.4.5.2.3 Diagram



6.4.5.2.4 Fields

Field	Function
31 NPSWA_EN	Non-Privileged Software Access Enable When set, allows non-privileged software to access all SNVS registers, including those that are privileged software read/write access only. 0 Only privileged software can access privileged registers 1 Any software can access privileged registers
30-20	Reserved

Table continues on the next page...

Field	Function
—	
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15-14 —	Reserved
13 —	Reserved
12-11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7-6 —	Reserved
5 LP_SWR_DIS	LP Software Reset Disable When set, disables the LP software reset. Once set, this bit can only be reset by the system reset. 0b - LP software reset is enabled 1b - LP software reset is disabled
4 LP_SWR	LP Software Reset When set to 1, the registers in the SNVS_LP section are reset. 0b - No Action 1b - Reset LP section
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

6.4.5.3 SNVS_HP Control Register (HPCR)

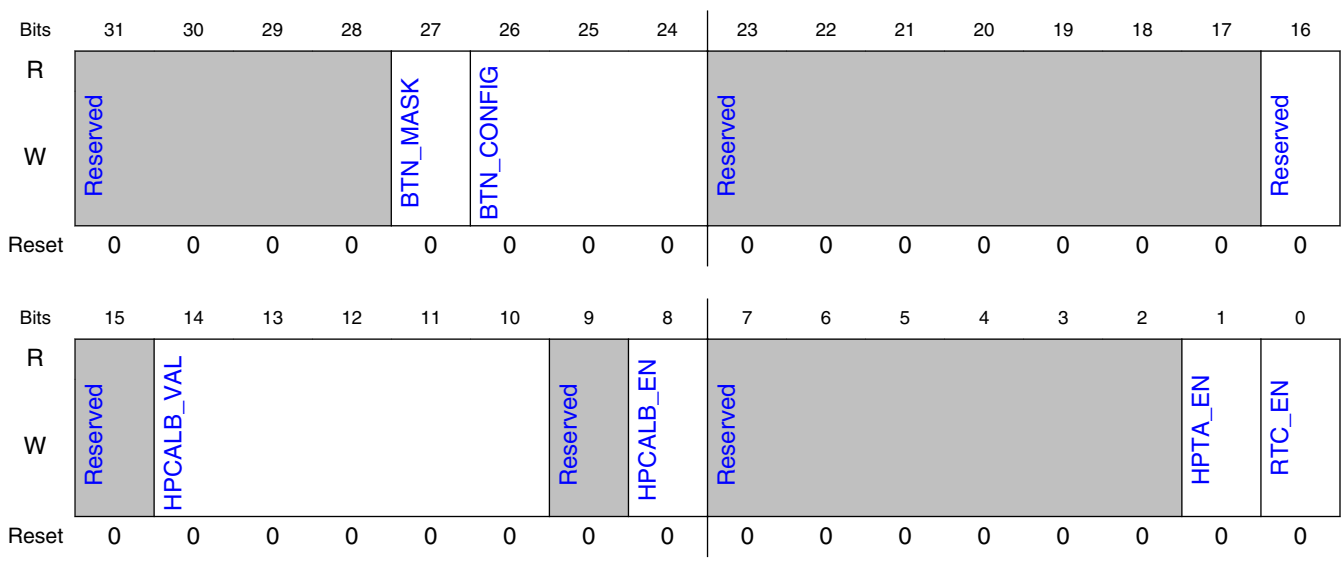
6.4.5.3.1 Offset

Register	Offset
HPCR	8h

6.4.5.3.2 Function

The SNVS_HP Control Register contains various control bits of the HP section of SNVS. This is *not* a privileged write register.

6.4.5.3.3 Diagram



6.4.5.3.4 Fields

Field	Function
31-28 —	Reserved
27 BTN_MASK	Button interrupt mask. This bit is used to mask the ipi_snvs_btn_int_b (button) interrupt request. 0: Interrupt disabled

Table continues on the next page...

Field	Function
	1: Interrupt enabled
26-24 BTN_CONFIG	Button Configuration. This field is used to configure which feature of the button (BTN) input signal constitutes "active". 000: Button signal is active high 001: Button signal is active low 010: Button signal is active on the falling edge 011: Button signal is active on the rising edge 100: Button signal is active on any edge All other patterns are Reserved
23-17 —	Reserved
16 —	Reserved
15 —	Reserved
14-10 HPCALB_VAL	HP Calibration Value Defines signed calibration value for the HP Real Time Counter. This field can be programmed only when RTC Calibration is disabled (HPCALB_EN is not set). This is a 5-bit 2's complement value, hence the allowable calibration values are in the range from -16 to +15 counts per 32768 ticks of the counter. 00000b - +0 counts per each 32768 ticks of the counter 00001b - +1 counts per each 32768 ticks of the counter 00010b - +2 counts per each 32768 ticks of the counter 01111b - +15 counts per each 32768 ticks of the counter 10000b - -16 counts per each 32768 ticks of the counter 10001b - -15 counts per each 32768 ticks of the counter 11110b - -2 counts per each 32768 ticks of the counter 11111b - -1 counts per each 32768 ticks of the counter
9 —	Reserved
8 HPCALB_EN	HP Real Time Counter Calibration Enabled Indicates that the time calibration mechanism is enabled. 0b - HP Timer calibration disabled 1b - HP Timer calibration enabled
7-2 —	Reserved
1 HPTA_EN	HP Time Alarm Enable When set, the time alarm interrupt is generated if the value in the HP Time Alarm Registers is equal to the value of the HP Real Time Counter. 0b - HP Time Alarm Interrupt is disabled 1b - HP Time Alarm Interrupt is enabled
0 RTC_EN	HP Real Time Counter Enable. This bit syncs with the 32KHz clock. It won't update with the bus clock. 0b - RTC is disabled 1b - RTC is enabled

6.4.5.4 SNVS_HP Status Register (HPSR)

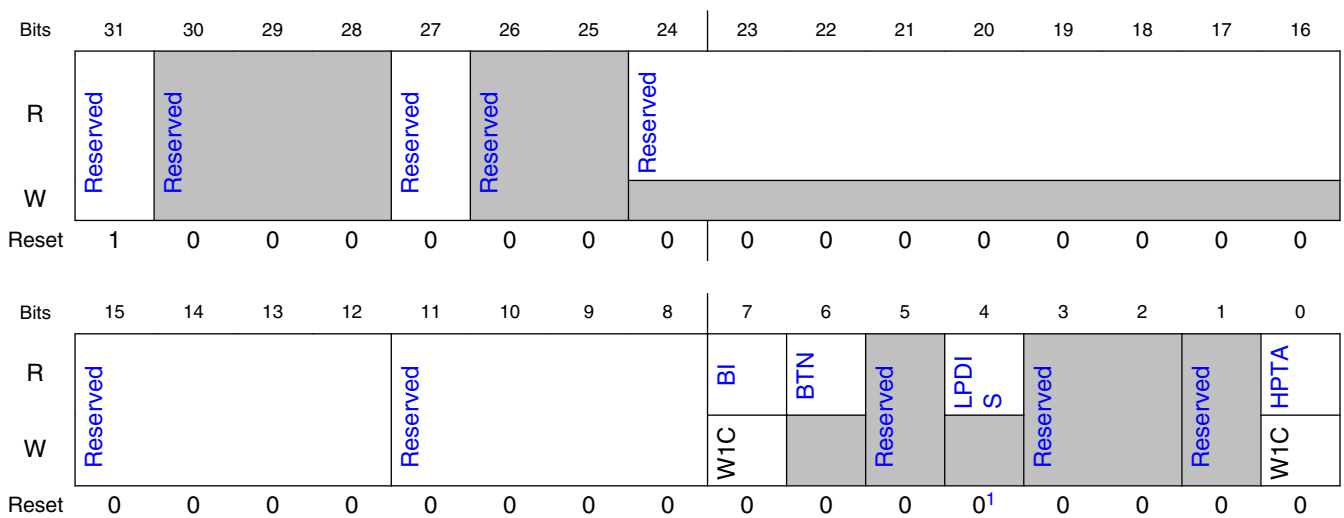
6.4.5.4.1 Offset

Register	Offset
HPSR	14h

6.4.5.4.2 Function

The HP Status Register reflects the internal state of the SNVS. This is *not* a privileged write register.

6.4.5.4.3 Diagram



1. The value of Low Power Disable is determined by the *no_battery* input signal to SNVS.

6.4.5.4.4 Fields

Field	Function
31 —	Reserved
30-28 —	Reserved
27	Reserved

Table continues on the next page...

Field	Function
—	
26-25 —	Reserved
24-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7 BI	Button Interrupt Signal ipi_snvs_btn_int_b was asserted.
6 BTN	Button Value of the BTN input. This is the external button used for PMIC control. 0: BTN not pressed 1: BTN pressed
5 —	Reserved
4 LPDIS	Low Power Disable If 1, the low power section has been disabled by means of an input signal to SNVS.
3-2 —	Reserved
1 —	Reserved
0 HPTA	HP Time Alarm Indicates that the HP Time Alarm has occurred since this bit was last cleared. 0b - No time alarm interrupt occurred. 1b - A time alarm interrupt occurred.

6.4.5.5 SNVS_HP Real Time Counter MSB Register (HPRTCMR)

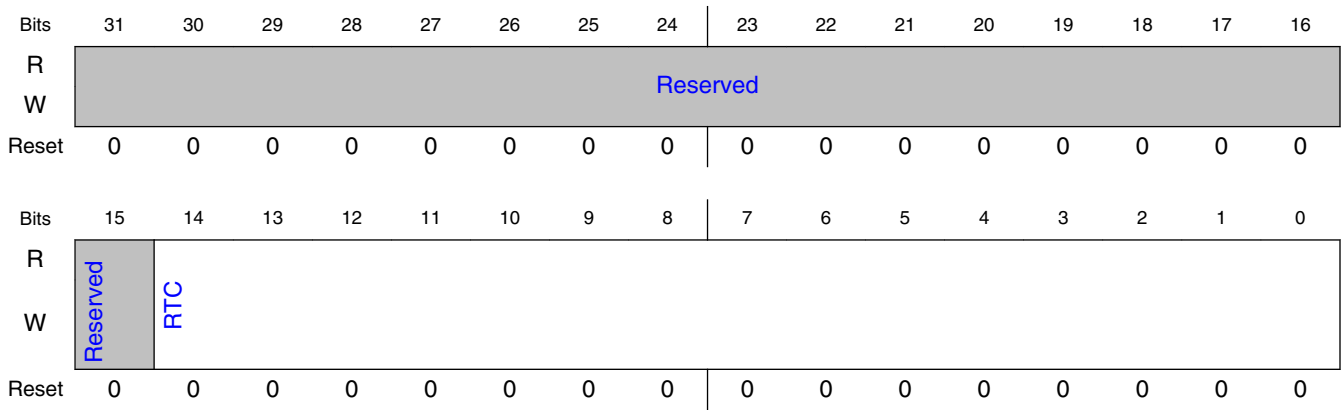
6.4.5.5.1 Offset

Register	Offset
HPRTCMR	24h

6.4.5.5.2 Function

The SNVS_HP Real Time Counter MSB register contains the 15 most-significant bits of the HP Real Time Counter. This is *not* a privileged write register.

6.4.5.5.3 Diagram



6.4.5.5.4 Fields

Field	Function
31-15 —	Reserved
14-0 RTC	HP Real Time Counter The most-significant 15 bits of the RTC. This register can be programmed only when RTC is not active (RTC_EN bit is not set).

6.4.5.6 SNVS_HP Real Time Counter LSB Register (HPRTCLR)

6.4.5.6.1 Offset

Register	Offset
HPRTCLR	28h

6.4.5.6.2 Function

The SNVS_HP Real Time Counter LSB register contains the 32 least-significant bits of the HP real time counter. This is *not* a privileged write register.

6.4.5.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RTC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RTC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.4.5.6.4 Fields

Field	Function
31-0 RTC	HP Real Time Counter least-significant 32 bits. This register can be programmed only when RTC is not active (RTC_EN bit is not set).

6.4.5.7 SNVS_HP Time Alarm MSB Register (HPTAMR)

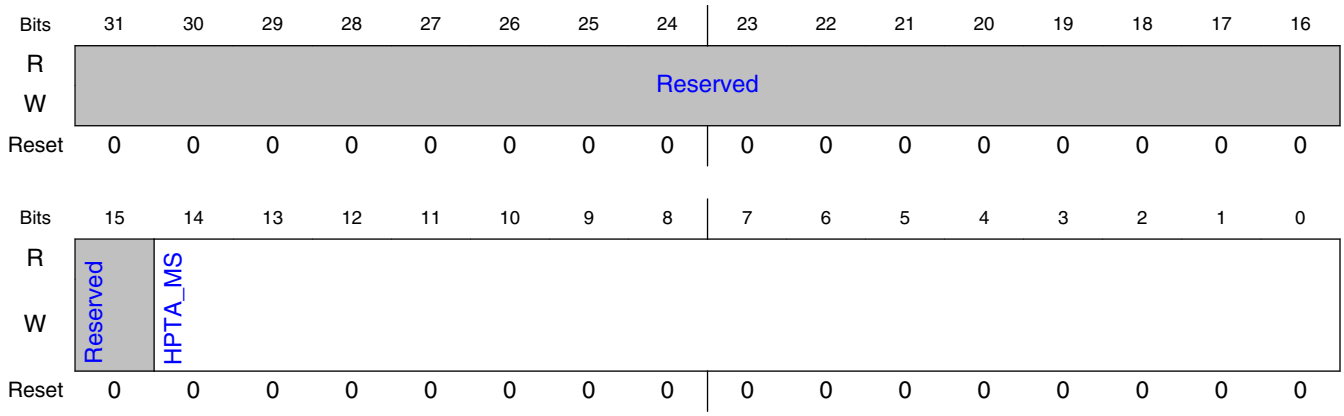
6.4.5.7.1 Offset

Register	Offset
HPTAMR	2Ch

6.4.5.7.2 Function

The SNVS_HP Time Alarm MSB register contains the most-significant bits of the SNVS_HP Time Alarm value. This is *not* a privileged write register.

6.4.5.7.3 Diagram



6.4.5.7.4 Fields

Field	Function
31-15 —	Reserved
14-0 HPTA_MS	HP Time Alarm, most-significant 15 bits. This register can be programmed only when HP time alarm is disabled (HPTA_EN bit is not set).

6.4.5.8 SNVS_HP Time Alarm LSB Register (HPTALR)

6.4.5.8.1 Offset

Register	Offset
HPTALR	30h

6.4.5.8.2 Function

The SNVS_HP Time Alarm LSB register contains the 32 least-significant bits of the SNVS_HP Time Alarm value. This is *not* a privileged write register.

6.4.5.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HPTA_LS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HPTA_LS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.4.5.8.4 Fields

Field	Function
31-0 HPTA_LS	HP Time Alarm, 32 least-significant bits. This register can be programmed only when HP time alarm is disabled (HPTA_EN bit is not set).

6.4.5.9 SNVS_LP Lock Register (LPLR)

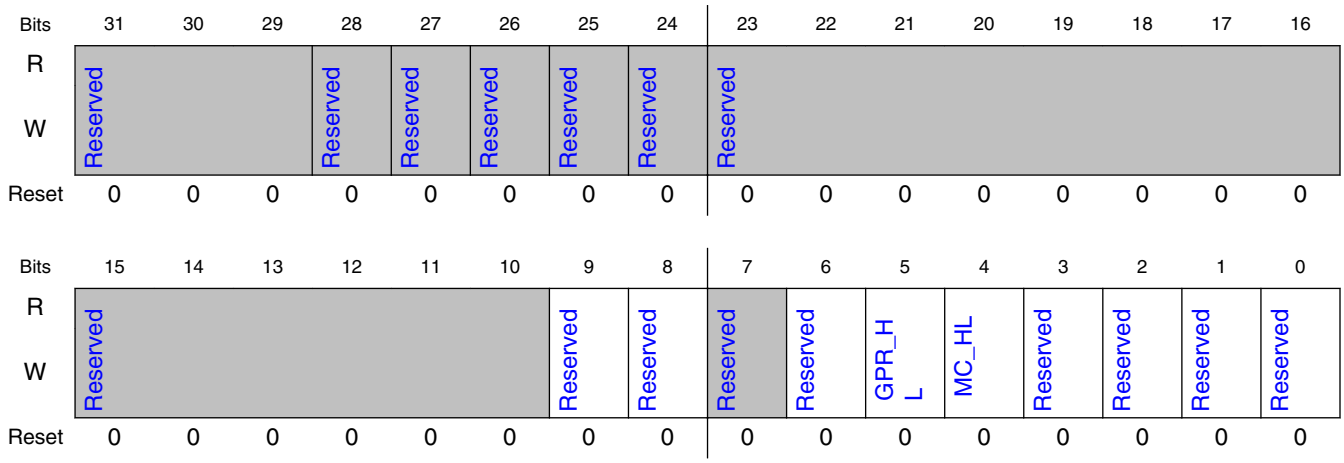
6.4.5.9.1 Offset

Register	Offset
LPLR	34h

6.4.5.9.2 Function

The SNVS_LP Lock Register contains lock bits for the SNVS_LP registers. This is a privileged write register.

6.4.5.9.3 Diagram



6.4.5.9.4 Fields

Field	Function
31-29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5	General Purpose Register Hard Lock

Table continues on the next page...

Field	Function
GPR_HL	When set, prevents any writes to the GPR. Once set, this bit can only be reset by the LP POR. 0b - Write access is allowed. 1b - Write access is not allowed.
4 MC_HL	Monotonic Counter Hard Lock When set, prevents any writes (increments) to the MC Registers and MC_ENV bit. Once set, this bit can only be reset by the LP POR. 0b - Write access (increment) is allowed. 1b - Write access (increment) is not allowed.
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

6.4.5.10 SNVS_LP Control Register (LPCR)

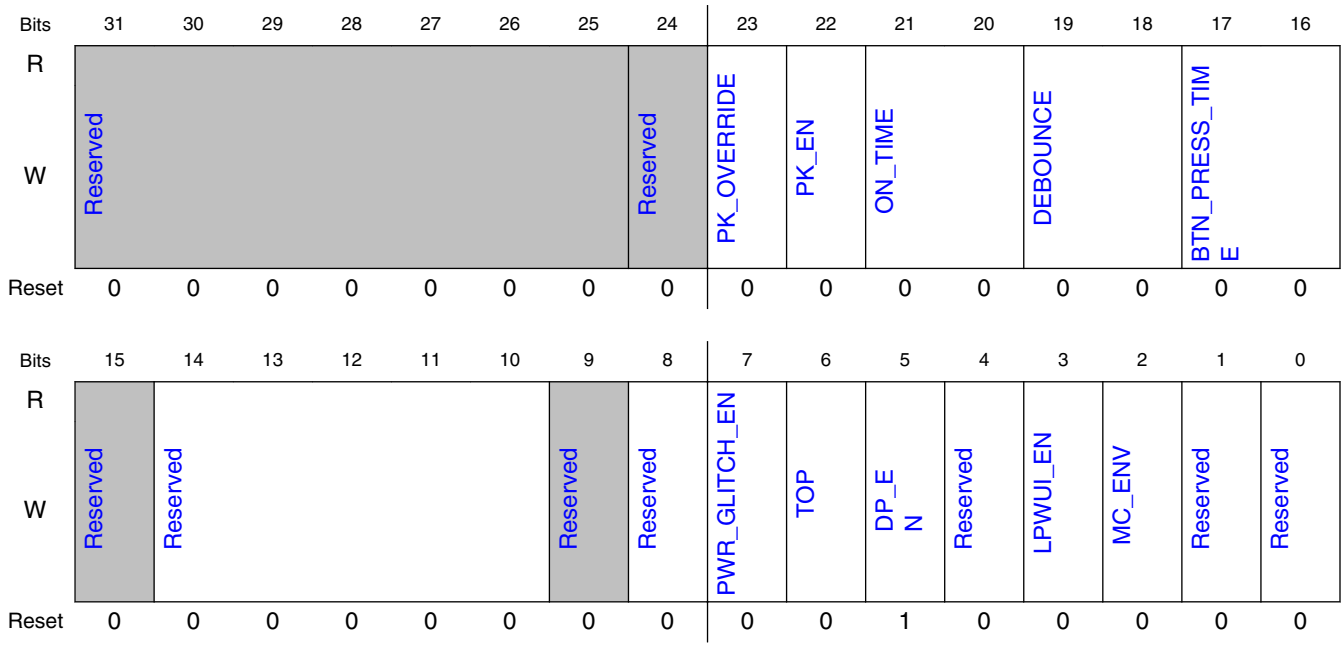
6.4.5.10.1 Offset

Register	Offset
LPCR	38h

6.4.5.10.2 Function

The SNVS_LP Control Register contains various control bits of the LP section of SNVS. This is a privileged write register.

6.4.5.10.3 Diagram



6.4.5.10.4 Fields

Field	Function
31-25 —	Reserved
24 —	Reserved
23 PK_OVERRIDE	PMIC On Request Override The value written to PK_OVERRIDE will be asserted on output signal snvs_lp_pk_override. That signal is used to override the IOMUX control for the PMIC I/O pad.
22 PK_EN	PMIC On Request Enable The value written to PK_EN will be asserted on output signal snvs_lp_pk_en. That signal is used to turn off the pullup/pulldown circuitry in the PMIC I/O pad.
21-20 ON_TIME	The ON_TIME field is used to configure the period of time after BTN is asserted before pmic_en_b is asserted to turn on the SoC power. 00: 500msec off->on transition time 01: 50msec off->on transition time 10: 100msec off->on transition time 11: 0msec off->on transition time
19-18 DEBOUNCE	This field configures the amount of debounce time for the BTN input signal. 00: 50msec debounce 01: 100msec debounce

Table continues on the next page...

Field	Function
	10: 500msec debounce 11: 0msec debounce
17-16 BTN_PRESS_TIME	This field configures the button press time out values for the PMIC Logic. 00 : 5 secs 01 : 10 secs 10 : 15 secs 11 : long press disabled (pmic_en_b will not be asserted regardlessof how long BTN is asserted)
15 —	Reserved
14-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 PWR_GLITCH_EN	Power Glitch Enable By default the detection of a power glitch does not cause the pmic_en_b signal to be asserted. Setting the Power Glitch Enable bit to 1 enables the power glitch event for the PMIC. 0 - disabled 1 - enabled
6 TOP	Turn off System Power Asserting this bit causes a signal to be sent to the Power Management IC to turn off the system power. This bit will clear once power is off. This bit is only valid when the Dumb PMIC is enabled. 0b - Leave system power on. 1b - Turn off system power.
5 DP_EN	Dumb PMIC Enabled When set, software can control the system power. When cleared, the system requires a Smart PMIC to automatically turn power off. 0b - Smart PMIC enabled. 1b - Dumb PMIC enabled.
4 —	Reserved
3 LPWUI_EN	LP Wake-Up Interrupt Enable This interrupt line should be connected to the external pin and is intended to inform the external chip about an SNVS_LP event (MC rollover, SRTC rollover, or time alarm). This wake-up signal can be asserted only when the chip (HP section) is powered down, and the LP section is isolated. 0 LP wake-up interrupt is disabled. 1 LP wake-up interrupt is enabled.
2 MC_ENV	Monotonic Counter Enabled and Valid When set, the MC can be incremented (by write transaction to the LPSMCMR or LPSMCLR). Once MC_SL or MC_HL bit is set this bit can be changed only by LP software reset or LP POR. 0b - MC is disabled or invalid.

Table continues on the next page...

Field	Function
	1b - MC is enabled and valid.
1 —	Reserved
0 —	Reserved

6.4.5.11 SNVS_LP Status Register (LPSR)

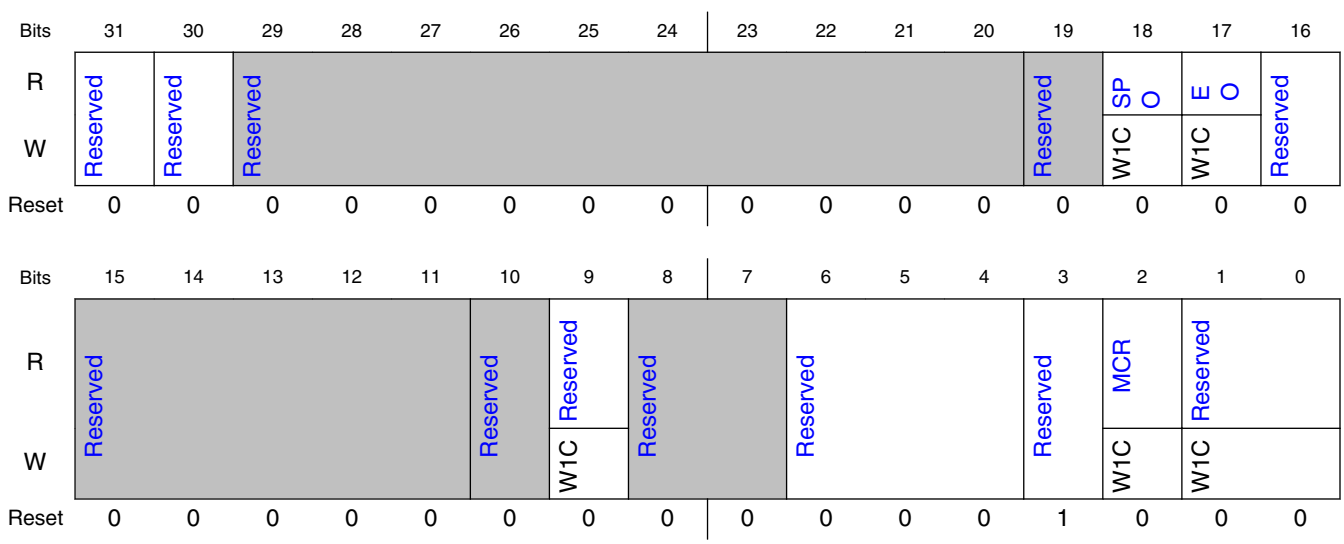
6.4.5.11.1 Offset

Register	Offset
LPSR	4Ch

6.4.5.11.2 Function

The SNVS_LP Status Register reflects the internal state and behavior of the SNVS_LP. This is a privileged write register.

6.4.5.11.3 Diagram



6.4.5.11.4 Fields

Field	Function
31 —	Reserved
30 —	Reserved
29-20 —	Reserved
19 —	Reserved
18 SPO	Set Power Off The SPO bit is set when the power button is pressed longer than the configured debounce time. Writing to the SPO bit will clear the set_pwr_off_irq interrupt. 0b - Set Power Off was not detected. 1b - Set Power Off was detected.
17 EO	Emergency Off This bit is set when a power off is requested. 0b - Emergency off was not detected. 1b - Emergency off was detected.
16 —	Reserved
15-11 —	Reserved
10 —	Reserved
9 —	Reserved
8-7 —	Reserved
6-4 —	Reserved
3 —	Reserved
2 MCR	Monotonic Counter Rollover 0b - MC has not reached its maximum value. 1b - MC has reached its maximum value.
1-0 —	Reserved

6.4.5.12 SNVS_LP Secure Monotonic Counter MSB Register (LPSMCMR)

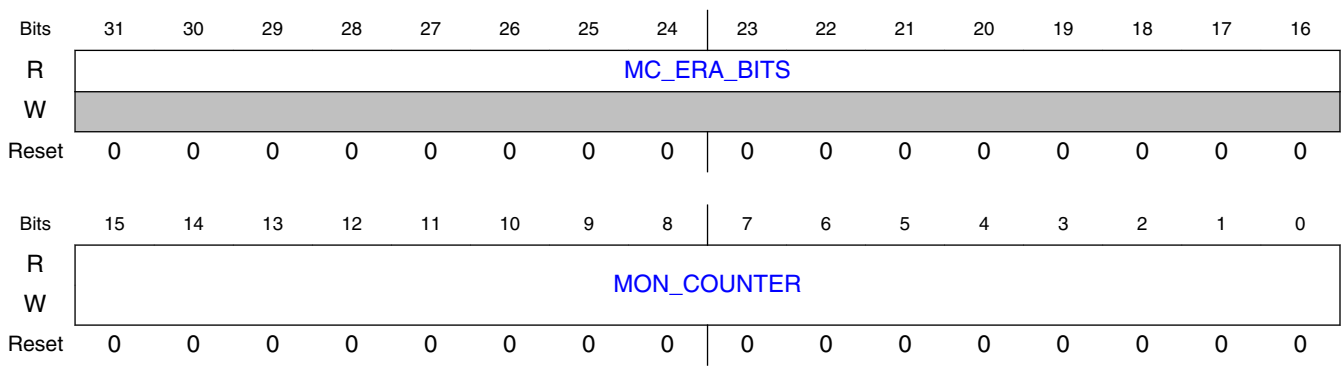
6.4.5.12.1 Offset

Register	Offset
LPSMCMR	5Ch

6.4.5.12.2 Function

The SNVS_LP Secure Monotonic Counter MSB Register contains the monotonic counter era bits and the most-significant 16 bits of the monotonic counter. The monotonic counter is incremented by one if there is a write command to the LPSMCMR or LPSMCLR register. This is a non-privileged read-only register.

6.4.5.12.3 Diagram



6.4.5.12.4 Fields

Field	Function
31-16 MC_ERA_BITS	Monotonic Counter Era Bits These bits are inputs to the module and typically connect to fuses. When the Monotonic Counter is in use (i.e. enabled and valid and powered by an uninterrupted power source), and the boot software detects that the Monotonic Counter most-significant 16 Bits and Monotonic Counter LSB Register have been reset (MC_ENV=0), the boot software can take action to ensure that the value in the monotonic counter remains monotonic (i.e. never decreasing). The action is to blow an additional MC_ERA_BITS fuse. Since the MC_ERA_BITS field forms the most-significant field of the monotonic counter, blowing an additional fuse guarantees that the new monotonic counter value is higher than any previous value. Since the Monotonic Counter is reset on an LP Software Reset, an excessive number of MC_ERA_BITS fusez may be consumed if LP Software Reset is used repeatedly.
15-0	Monotonic Counter most-significant 16 Bits

Field	Function
MON_COUNTER	**Note that writing to this register does *not* change the value of this field to the value that was written.** **The 48-bit monotonic counter value (consisting of LPSMCMR[MON_COUNTER] prepended to LPSMCLR[MON_COUNTER]) is incremented by one when:** • A write transaction to the LPSMCMR or LPSMCLR register is detected. • The MC_ENV bit is set. • MC_SL and MC_HL bits are not set. This value can be reset only by LP software reset or LP POR.

6.4.5.13 SNVS_LP Secure Monotonic Counter LSB Register (LPSMCLR)

6.4.5.13.1 Offset

Register	Offset
LPSMCLR	60h

6.4.5.13.2 Function

The SNVS_LP Secure Monotonic Counter LSB Register contains the 32 least-significant bits of the monotonic counter. The MC is incremented by one if there is a write command to the LPSMCMR or LPSMCLR register. This is a non-privileged read-only register.

6.4.5.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MON_COUNTER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MON_COUNTER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.4.5.13.4 Fields

Field	Function
31-0 MON_COUNTER	Monotonic Counter bits **Note that writing to this register does *not* change the value of this field to the value that was written.** **The 48-bit monotonic counter value (consisting of LPSMCMR[MON_COUNTER] prepended to LPSMCLR[MON_COUNTER]) is incremented by one when:** • A write transaction to the LPSMCMR or LPSMCLR register is detected. • The MC_ENV bit is set. • MC_SL and MC_HL bits are not set. This value can be reset only by LP software reset or LP POR.

6.4.5.14 SNVS_LP Power Glitch Detector Register (LPPGDR)

6.4.5.14.1 Offset

Register	Offset
LPPGDR	64h

6.4.5.14.2 Function

The SNVS_LP Power Glitch Detector Register is a 32-bit read/write register that is used for storing the power glitch detector value, as described in Power glitch detector (PGD). This is a privileged write register.

6.4.5.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PGD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PGD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.4.5.14.4 Fields

Field	Function
31-0 PGD	Power Glitch Detector Value

6.4.5.15 SNVS_LP General Purpose Register 0 (legacy alias) (LPGPR0_legacy_alias)

6.4.5.15.1 Offset

Register	Offset
LPGPR0_legacy_alias	68h

6.4.5.15.2 Function

See register SNVS_LP General Purpose Registers 0 .. 3 (LPGPR0 - LPGPR3).

6.4.5.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.4.5.15.4 Fields

Field	Function
31-0 GPR	General Purpose Register When GPR_SL or GPR_HL bit is set, the register cannot be programmed.

6.4.5.16 SNVS_LP General Purpose Registers 0 .. 3 (LPGPR0_alias - LPGPR3_alias)

6.4.5.16.1 Offset

Register	Offset
LPGPR0_alias	90h
LPGPR1_alias	94h
LPGPR2_alias	98h
LPGPR3_alias	9Ch

6.4.5.16.2 Function

See register SNVS_LP General Purpose Registers 0 .. 3 (LPGPR0 - LPGPR3).

6.4.5.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.4.5.16.4 Fields

Field	Function
31-0 GPR	General Purpose Register When GPR_SL or GPR_HL bit is set, the register cannot be programmed.

6.4.5.17 SNVS_LP General Purpose Registers 0 .. 3 (LPGPR0 - LPGPR3)

6.4.5.17.1 Offset

Register	Offset
LPGPR0	100h
LPGPR1	104h
LPGPR2	108h
LPGPR3	10Ch

6.4.5.17.2 Function

The SNVS_LP General Purpose Register is a 128-bit read/write register located in SNVS_LP, which can be used by any application for retaining data during an SoC power-down mode. This is a privileged read/write register. The full GPR register is accessed as 4 32-bit registers located in successive word addresses starting at offset 100h. For backward compatibility with earlier versions of SNVS, LPGPR0..LPGPR3 are aliased at the earlier offset of 90h and LPGPR0 is also aliased at its original offset of 68h. New software should access the GPR register at the preferred offset of 100h. The GPR will be automatically zeroized when an enabled security event occurs, unless GPR zeroization is disabled via the GPR_Z_DIS bit in the LP Control Register.

6.4.5.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.4.5.17.4 Fields

Field	Function
31-0	General Purpose Register

Field	Function
GPR	When GPR_SL or GPR_HL bit is set, the register cannot be programmed.

6.4.5.18 SNVS_HP Version ID Register 1 (HPVIDR1)

6.4.5.18.1 Offset

Register	Offset
HPVIDR1	BF8h

6.4.5.18.2 Function

The SNVS_HP Version ID Register 1 is a non-privileged read-only register that contains the current version of the SNVS. The version consists of a module ID, a major version number, and a minor version number.

6.4.5.18.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IP_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR_REV								MINOR_REV							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1

6.4.5.18.4 Fields

Field	Function
31-16 IP_ID	SNVS block ID
15-8 MAJOR_REV	SNVS block major version number
7-0 MINOR_REV	SNVS block minor version number

6.4.5.19 SNVS_HP Version ID Register 2 (HPVIDR2)

6.4.5.19.1 Offset

Register	Offset
HPVIDR2	BFCh

6.4.5.19.2 Function

The SNVS_HP Version ID Register 2 is a non-privileged read-only register that indicates the current version of the SNVS. Version ID register 2 consists of the following fields: integration options, ECO revision, and configuration options.

6.4.5.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IP_ERA								INTG_OPT							
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ECO_REV								CONFIG_OPT							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

6.4.5.19.4 Fields

Field	Function
31-24 IP_ERA	IP Era 00h - Era 1 or 2 03h - Era 3 04h - Era 4 05h - Era 5
23-16 INTG_OPT	SNVS Integration Options
15-8	SNVS ECO Revision

Table continues on the next page...

Field	Function
ECO_REV	
7-0 CONFIG_OPT	SNVS Configuration Options

6.5 System Reset Controller (SRC)

6.5.1 SRC Overview

The System Reset Controller (SRC) controls the reset and boot operation of the SoC.

It is responsible for the generation of all reset signals and boot decoding.

The reset controller determines the source and the type of reset, such as POR, COLD, and performs the necessary reset qualification and stretching sequences. Based on the type of reset, the reset logic generates the reset sequence for the entire IC. Whenever the chip is powered on, the reset is issued through SRC_ONOFF signal and the entire chip is reset.

6.5.1.1 Features

The SRC includes the following features.

- Receives and handles the resets from all the reset sources
- Resets the appropriate domains based upon the resets sources and the nature of the reset
- Latches the SRC_BOOT_MODE pins and common configuration signals from the internal fuse

6.5.2 Top-level resets, power-up sequence and external supply integration

Information found here defines chip resets, power-up sequence, and external supply integration.

6.5.2.1 Reset and Power-up Flow

The chip presumes the following reset and power-up flow:

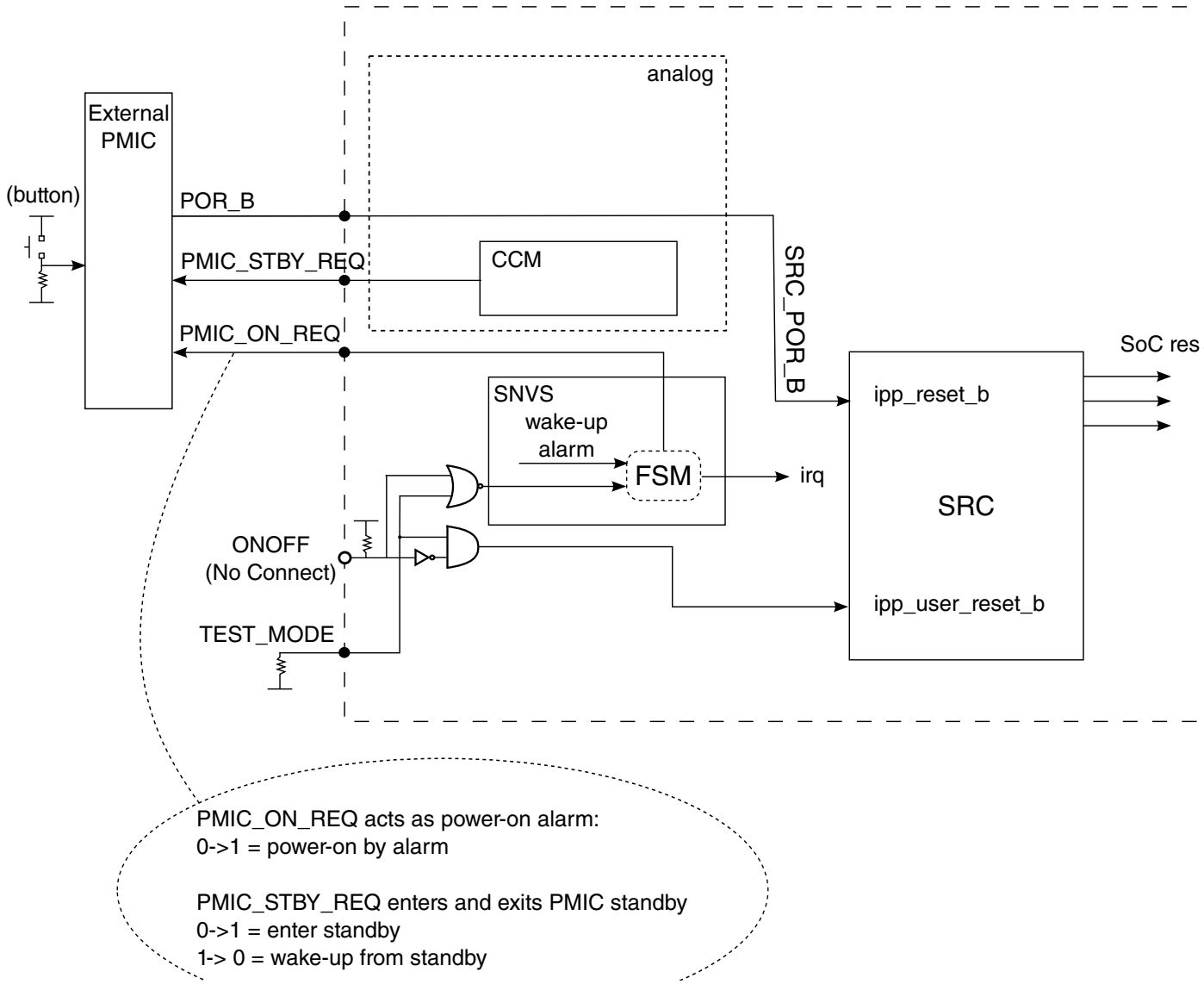


Figure 6-31. Chip reset scheme under external PMIC control

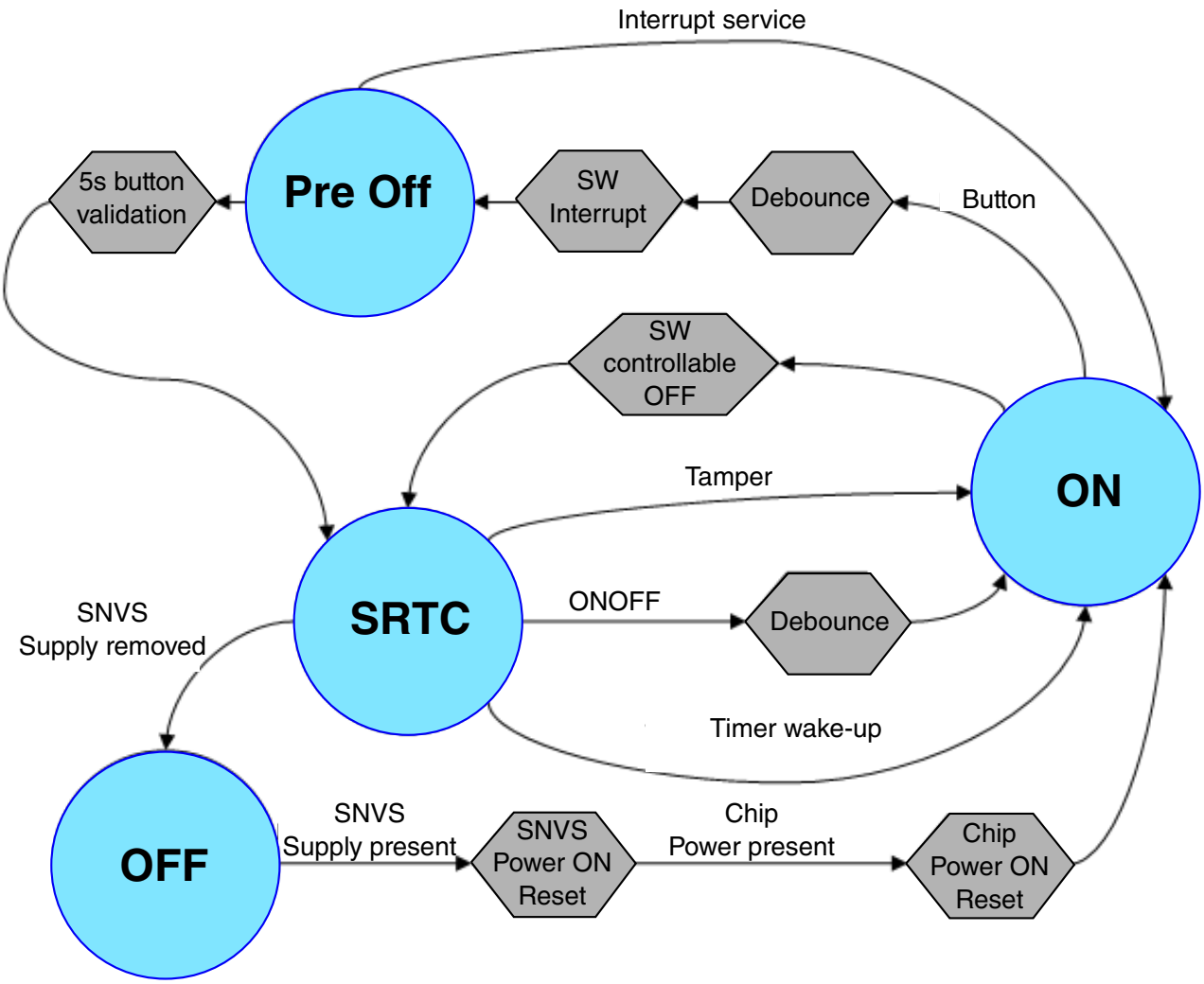


Figure 6-32. Chip on/off state flow diagram

6.5.2.2 Finite-State Machine (FSM)

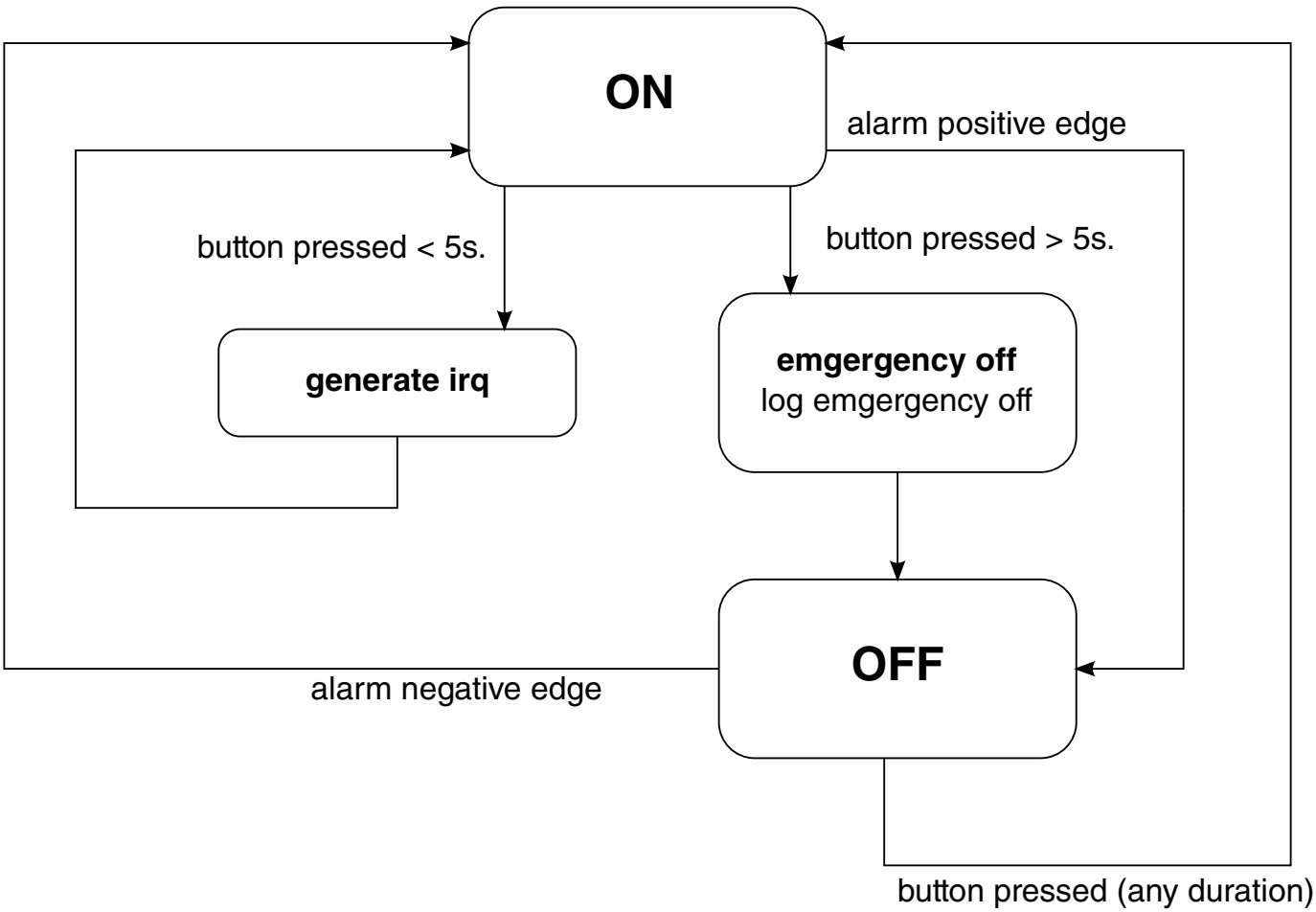


Figure 6-33. FSM

6.5.2.3 Power mode transitions

Table 6-53. Power mode transitions

Power mode	Configuration with external PMIC	Configuration with internal PMIC
ON, first time	1. SoC power supply is connected to SNVS. 2. When button is pressed, PMIC powers on.	1. SoC power supply is connected to SNVS. 2. When button is pressed, 'state' goes ON, PMIC_ON_REQ goes '1'. 3. External regulator is enabled.
Normal ON to OFF, by button	1. Button is pressed for a short duration on the external PMIC. 2. Interrupt request (irq) is sent to SoC from external PMIC. 3. SoC is programming PMIC for power off when standby is asserted.	1. SoC button is pressed for a short duration. 2. Interrupt request (irq) is sent to SoC from FSM. 3. Alarm timer is set up by software routine and started. 4. Upon alarm_in assertion to '1', PMIC_ON_REQ goes '0'. 5. External regulator goes OFF.

Table continues on the next page...

Table 6-53. Power mode transitions (continued)

Power mode	Configuration with external PMIC	Configuration with internal PMIC
	4. In CCM STOP mode, Standby is asserted, PMIC gates SoC supplies.	
Emergency ON to OFF, by button	1. Button is pressed for an extended time on the external PMIC. 2. PMIC is powering off.	1. Button is pressed for longer than 5 seconds on the SoC. 2. FSM validates button pressed for 5 seconds. 3. Emergency power off is logged, PMIC_ON_REQ goes '0', alarm_mask goes '1'. 4. External regulator goes OFF.
OFF to ON, by button	1. Button is pressed on the external PMIC. 2. PMIC powers ON.	1. Button is pressed on the SoC. 2. PMIC_ON_REQ goes '1', alarm_mask goes '0'. 3. External regulator powers ON.
OFF to ON, by timer alarm	1. Timer alarm in SNVS is programmed by software before SoC goes OFF. 2. SoC enters OFF mode. 3. Upon timer limit, wake up alarm goes '0'. PMIC_ON_REQ goes '1'. 4. PMIC receives assertion of PMIC_ON_REQ and wakes up.	1. Timer alarm in SNVS is programmed by software before SoC goes OFF. 2. SoC enters OFF mode. 3. Upon timer limit, wake up alarm goes '0'. PMIC_ON_REQ goes '1'. 4. External regulator is enabled by PMIC_ON_REQ = 1.

6.5.3 Power-On Reset and power sequencing

This module generates an internal POR_B signal that is logically AND'ed with any externally applied SRC_POR_B signal. The internal POR_B signal will be held low until all of the following conditions are met:

- 4ms after the external power supply VDDHIGH_IN is valid
- 1ms after the VDD_SOC_CAP supply is valid

The 4ms and 1ms delays are derived from counting the 32 kHz RTC clock cycles; the accuracy depends on the accuracy of the RTC. When the RTC crystal is either absent or in the process of powering up, an internal ring oscillator will be the source of RTC, which is not as accurate as the crystal.

6.5.3.1 External POR using SRC_POR_B

If the external SRC_POR_B signal is used to control the processor POR, SRC_POR_B must remain low (asserted) until the VDD_ARM_CAP and VDD_SOC_CAP supplies are stable.

6.5.3.2 Internal POR

If the external SRC_POR_B signal is not used (always held high or left unconnected), the processor defaults to the internal POR function (PMU controls generation of the POR based on the power supplies).

If the internal POR function is used, the following power supply requirements must be met:

- VDD_ARM_IN and VDD_SOC_IN may be supplied from the same source, or
- VDD_SOC_IN can be supplied before VDD_ARM_IN with a maximum delay of 1 ms.

6.5.4 Functional Description

6.5.4.1 Reset Control

This section details the reset control of this device.

6.5.4.1.1 Reset inputs and outputs

The reset control logic receives reset requests from all potential reset sources. All the immediate sources of reset are directly passed to the reset stretching block, whereas the resets requiring qualification are passed on to the reset qualification logic before they are sent to the reset stretching block.

All reset inputs and outputs are described in the following figure:

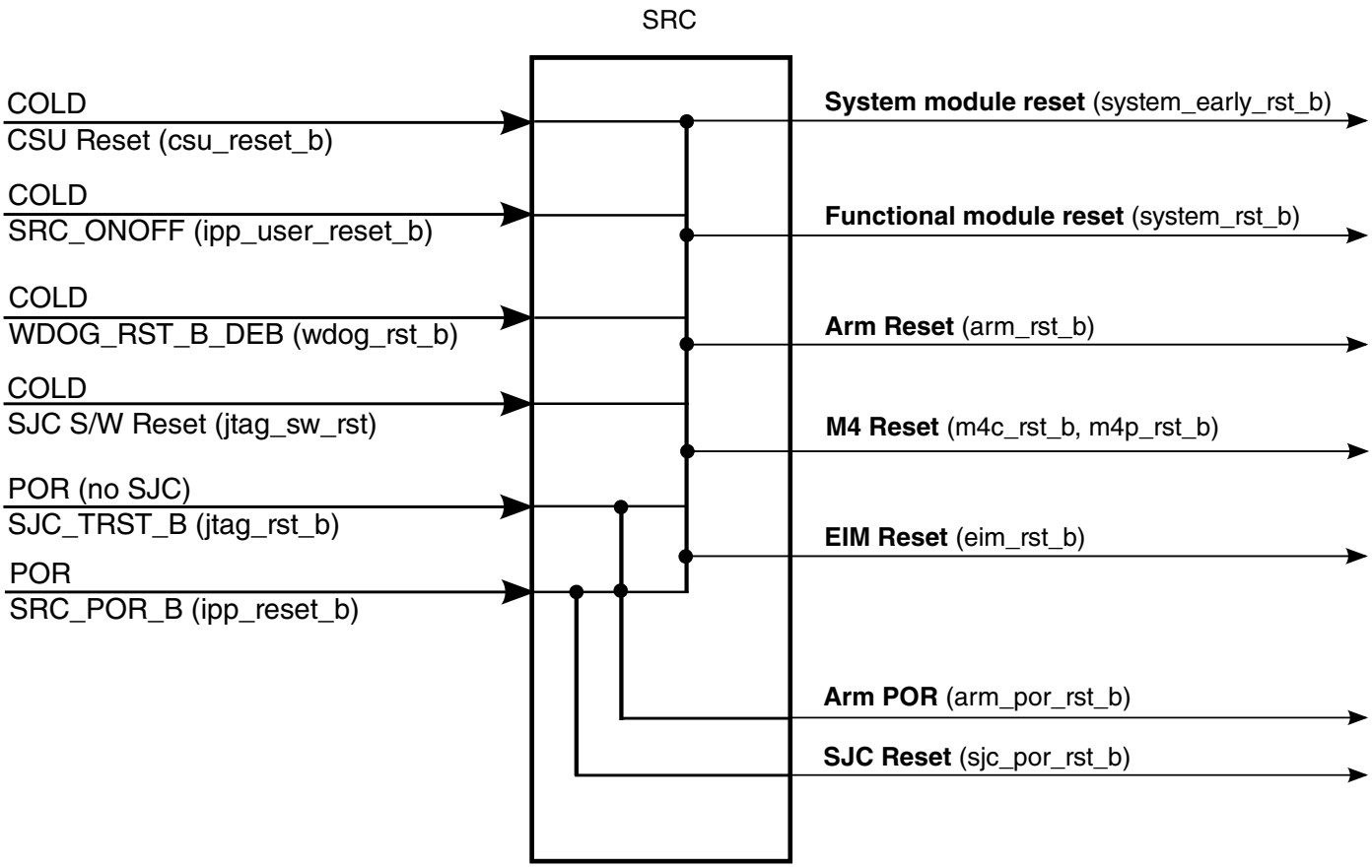


Figure 6-34. SRC inputs and outputs

The reset types and modules they affect are shown in Table 6-54. As there is no chip POR, the POR_B is used to reset the entire chip including test logic and JTAG modules.

NOTE

All resets are expected to be active low except jtag_sw_rst.

Table 6-54. SRC reset functionality

SoC Modules	POR	COLD
System modules (PLLs, fuses, etc)	yes	yes
Functional modules	yes	yes
Arm	yes	yes
Arm SoC	yes	yes
M4 Core	yes	yes
M4 Platform	yes	yes
Arm POR	yes	no
Arm debug	yes	no
SJC	yes	no
SRTC	yes	no

The reset priorities are POR (strongest) and COLD (weakest). If a stronger reset is asserted during the sequence of a weaker reset, then the weaker sequence will be overridden, and the stronger reset sequence will commence. There is no priority within a reset type (POR, etc). If a reset is asserted during the reset sequence of the same type, the reset sequence will be interrupted and restarted.

The following lists the functionality of each of these reset outputs:

- system_early_rst_b - Resets the system modules that need to start first as CCM, OCOTP_CTRL, FUSEBOX, etc.
- system_rst_b - Resets functional modules
- arm_rst_b - Resets Arm module (on regular system reset)
- arm_por_rst_b - Resets Arm POR input
- arm_soc_rst_b - Reset for Arm SOC
- m4c_rst_b - Reset for M4 core
- m4p_rst_b - Reset for M4 platform
- arm_dbg_rst_b - Reset debug logic of Arm
- test_logic_rst_b - Reset test logic (IOMUXC, DAP)
- sjc_por_rst_b - Reset to SJC
- srtc_rst_b - Resets SRTC

NOTE

It is assumed that each reset source will deassert after its assertion, either due to reset generated to the system from SRC, or by negation of the reset source (if it came from an external source to the chip). In the latter case, the reset source is assumed to be held for at least 2 XTALI clocks so it can be sampled by SRC.

6.5.4.1.2 Reset Handling

6.5.4.1.2.1 POR (SRC_POR_B)

SRC_POR_B is an external reset signal. When the chip is powered up, the reset signal is passed through the POR_B pin indicating power-up sequence. The SRC resets the entire chip including the JTAG (SJC) module. All SRC registers will be reset during the POR sequence.

As soon as SRC_POR_B occurs, all resets are asserted and the entire chip is reset by SRC. The SRC_POR_B is stretched for 2 XTALI cycles and the stretching sequence takes place after 2 XTALI clocks of POR_B pin deassertion.

The srtc_rst_b signal is deasserted together with SRC_POR_B signal. The output are also deasserted after the stretching of SRC_POR_B has deasserted.

The sjc_por_rst_b signal is deasserted together with SRC_POR_B signal. The output is also deasserted after the stretching of SRC_POR_B has deasserted.

Once the above resets deassert, system_early_rst_b reset is deasserted after 2 XTALI clocks. The system_early_rst_b is used for the CCM and PLL-IPs to start generating PLL clock ouputs and the system root clocks.

When the system root clocks are ready, the CCM will assert system_clk_ready signal. This signal is generated during the start sequence in the CCM and it involves the preparation of the PLLs to generate clock roots for functional operation.

SRC then enables OCOTP_CTRL and fusebox clocks, so that fuses can be loaded to OCOTP_CTRL.

- SRC will prepare the boot information
- After 8 ipg cycles, resets to all modules will be de-asserted
- After 8 ipg cycles, system clocks will be enabled (en_system_clk).

6.5.4.1.2.2 COLD RESET

The sequence is similar to SRC_POR_B except the memory repair operation is not performed.

Once the reset source deasserts, system_early_rst_b reset is deasserted after at least 2 XTALI clocks. The system_early_rst_b is used for the CCM and PLL-IPs to start generating PLL clock outputs and the system root clocks.

Once the system root clocks are ready, the CCM will assert system_clk_ready signal. This signal is generated during the start sequence in the CCM and it involves the preparation of the PLLs to generate clock roots for functional operation. See CCM for more information.

Once system_clk_ready arrives at the SRC, it will enable OCOTP_CTRL and fusebox clocks, so that fuses can be loaded to OCOTP_CTRL. OCOTP_CTRL will notify with iim_ready_flag once the fusebox loading finishes.

- SRC will prepare the boot information
- After 8 ipg cycles resets to all modules will be deasserted
- After 8 ipg cycles, system clocks will be enabled (en_system_clk).

6.5.4.2 Parallel Reset Requests

SRC will follow the following rules in the case of parallel reset requests:

1. The order of strength of resets is POR - strongest, COLD - weakest
2. If a stronger reset is asserted during weaker reset sequence, then the stronger reset will take over and the stronger reset process will commence. The following cases fall into this category:
   - POR reset request in the middle of cold reset process - the cold will be stopped and the POR sequence will start.
3. If a weaker reset is asserted during stronger reset sequence, then the stronger reset sequence will continue without interference. If at the end of the stronger reset process the weaker request is still asserted then the weaker sequence will commence. The following cases fall into this category:
   - COLD reset requests in the middle of POR reset process - the POR process will continue without interference.
4. If a similar reset request is asserted during the process of reset handling, then the process of reset handling will start over (with the same process). The following cases fall into this category:
   - POR reset request in the middle of POR reset process - the POR process will start over.
   - COLD reset request in the middle of COLD reset process - the COLD process will start over.

6.5.4.3 Boot Mode Control

6.5.4.3.1 BOOT_MODE Pin Latching

The exact boot sequence is controlled by the values of the BOOT_MODE pins on this device.

The value of the BOOT_MODE pins will be latched after the OCOTP_CTRL asserts the fuse read completion flag. After latching, the values of the BOOT_MODE pins are used to determine the booting options of the core as described in the SRC_SBMRx registers.

The boot mode general purpose bits can be provided to the SRC from either e-fuses or GPIO signals. The gpio_bt_sel e-fuse defines the source to be used to derive the boot information. When gpio_bt_sel is set, e-fuses are used. When cleared, GPIO signals are used.

The boot information is provided in SRC_SBMR1 register. The figure below shows the selection of boot mode information.

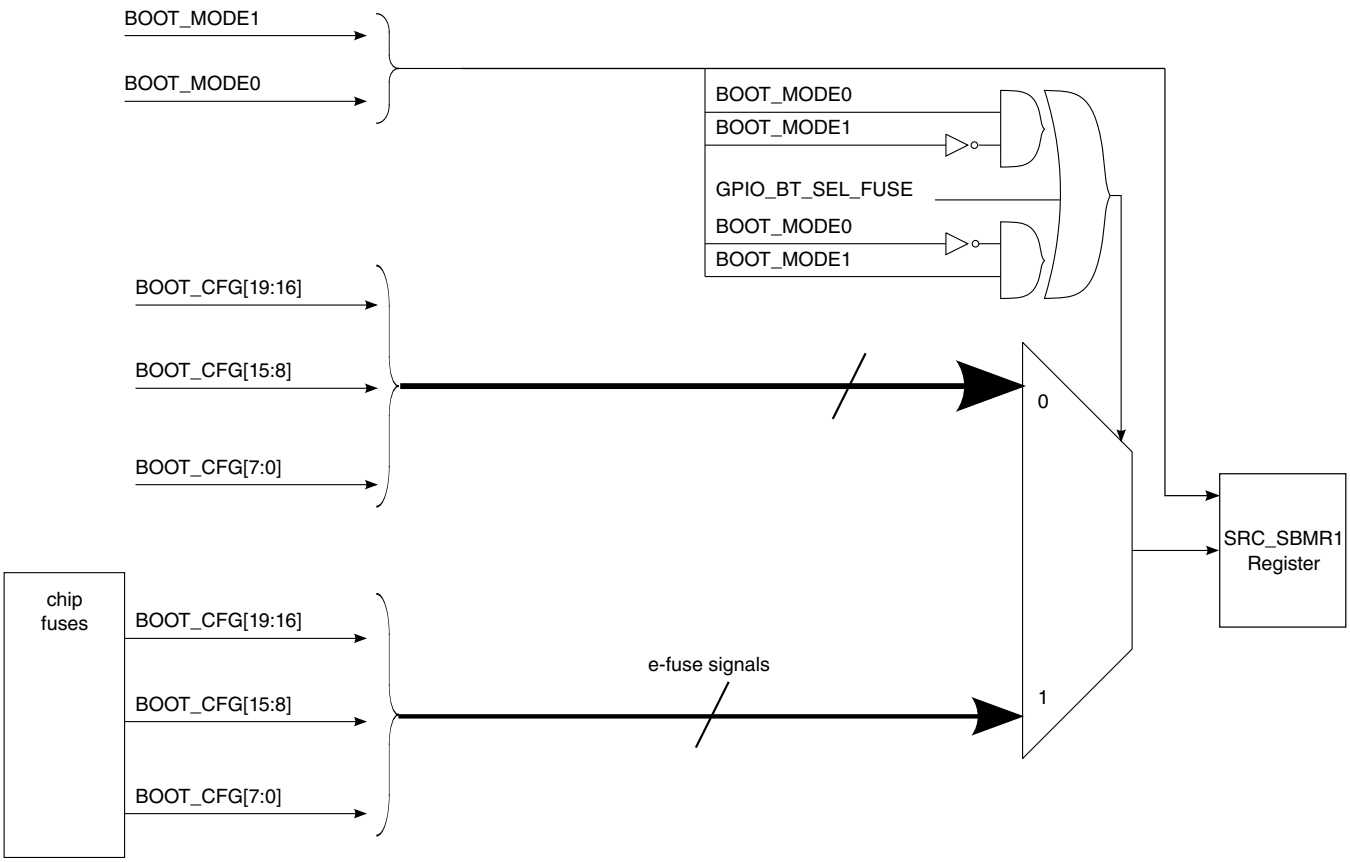


Figure 6-35. Boot mode information

6.5.5 SRC Memory Map/Register Definition

SRC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3039_0000	SRC Reset Control Register (SRC_SCR)	32	R/W	0000_00A0h	6.5.5.1/ 909
3039_0004	A53 Reset Control Register (SRC_A53RCR0)	32	R/W	000A_0000h	6.5.5.2/ 910
3039_0008	A53 Reset Control Register (SRC_A53RCR1)	32	R/W	0000_0001h	6.5.5.3/ 915
3039_000C	M4 Reset Control Register (SRC_M4RCR)	32	R/W	0000_00A8h	6.5.5.4/ 917
3039_0020	USB OTG PHY1 Reset Control Register (SRC_USBOPHY1_RCR)	32	R/W	0000_0000h	6.5.5.5/ 920
3039_0024	USB OTG PHY2 Reset Control Register (SRC_USBOPHY2_RCR)	32	R/W	0000_0000h	6.5.5.6/ 921

Table continues on the next page...

SRC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3039_0028	MIPI PHY Reset Control Register (SRC_MIPIPHY_RCR)	32	R/W	0000_0000h	6.5.5.7/ 923
3039_002C	PCIE PHY Reset Control Register (SRC_PCIEPHY_RCR)	32	R/W	0000_000Ah	6.5.5.8/ 924
3039_0034	DISPLAY Reset Control Register (SRC_DISP_RCR)	32	R/W	0000_0000h	6.5.5.9/ 927
3039_0040	GPU Reset Control Register (SRC_GPU_RCR)	32	R/W	0000_0000h	6.5.5.10/ 928
3039_0044	VPU Reset Control Register (SRC_VPU_RCR)	32	R/W	0000_0000h	6.5.5.11/ 930
3039_0058	SRC Boot Mode Register 1 (SRC_SBMR1)	32	R	0000_0000h	6.5.5.12/ 931
3039_005C	SRC Reset Status Register (SRC_SRSR)	32	R/W	0000_0001h	6.5.5.13/ 931
3039_0068	SRC Interrupt Status Register (SRC_SISR)	32	R/W	0000_0000h	6.5.5.14/ 934
3039_006C	SRC Interrupt Mask Register (SRC_SIMR)	32	R/W	0000_03FFh	6.5.5.15/ 936
3039_0070	SRC Boot Mode Register 2 (SRC_SBMR2)	32	R	0000_0000h	6.5.5.16/ 938
3039_0074	SRC General Purpose Register 1 (SRC_GPR1)	32	R/W	0000_0000h	6.5.5.17/ 939
3039_0078	SRC General Purpose Register 2 (SRC_GPR2)	32	R/W	0000_0000h	6.5.5.18/ 939
3039_007C	SRC General Purpose Register 3 (SRC_GPR3)	32	R/W	0000_0000h	6.5.5.19/ 940
3039_0080	SRC General Purpose Register 4 (SRC_GPR4)	32	R/W	0000_0000h	6.5.5.20/ 940
3039_0084	SRC General Purpose Register 5 (SRC_GPR5)	32	R/W	0000_0000h	6.5.5.21/ 941
3039_0088	SRC General Purpose Register 6 (SRC_GPR6)	32	R/W	0000_0000h	6.5.5.22/ 941
3039_008C	SRC General Purpose Register 7 (SRC_GPR7)	32	R/W	0000_0000h	6.5.5.23/ 941
3039_0090	SRC General Purpose Register 8 (SRC_GPR8)	32	R/W	0000_0000h	6.5.5.24/ 942
3039_0094	SRC General Purpose Register 9 (SRC_GPR9)	32	R/W	0000_0000h	6.5.5.25/ 942
3039_0098	SRC General Purpose Register 10 (SRC_GPR10)	32	R/W	0000_0000h	6.5.5.26/ 943
3039_1000	SRC DDR Controller Reset Control Register (SRC_DDRC_RCR)	32	R/W	0000_000Fh	6.5.5.27/ 943

6.5.5.1 SRC Reset Control Register (SRC_SCR)

The reset control register (SCR), contains bits that control operation of the reset controller.

Address: 3039_0000h base + 0h offset = 3039_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DOM_EN	LOCK	Reserved		DOMAIN3	DOMAIN2	DOMAIN1	DOMAIN0	Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								MASK_TEMPSENSE_RESET				Reserved			
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0

SRC_SCR field descriptions

Field	Description
31 DOM_EN	Domain Control enable for this register. **NOTE:** [31:30] and [27:24] areas are not controlled by this bit. [31:30] and [27:24] area can be modified by any masters from any domains when Lock bit is not set. 0 Disables domain control. All of this register's bits except [31:30] and [27:24] can be modified by any masters 1 Enables domain control. All of this register's bits except [31:30] and [27:24] can only be modified by the masters from the domains specified in [27:24] area.
30 LOCK	Domain control bits lock **NOTE:** Lock bit is a write-once register, once it is set to 1, it can't be write to 0 0 [31] and [27:24] bits can be modified 1 [31] and [27:24] bits cannot be modified
29–28 -	This field is reserved. Reserved
27 DOMAIN3	Domain3 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain2. The master from domain2 cannot write to this register. 1 This register is assigned to domain2. The master from domain2 can write to this register

Table continues on the next page...

SRC_SCR field descriptions (continued)

Field	Description
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain1 cannot write to this register. 1 This register is assigned to domain1. The master from domain1 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain3 cannot write to this register. 1 This register is assigned to domain0. The master from domain3 can write to this register
23–8 -	This field is reserved. Reserved
7–4 MASK_ TEMPSENSE_ RESET	Mask tempsense_reset source. If these 4 bits are coded from A to 5 then, the tempsense_reset input to SRC will be masked and the tempsense_reset will not create a reset to the chip. 0101 tempsense_reset is masked 1010 tempsense_reset is not masked
-	This field is reserved. Reserved

6.5.5.2 A53 Reset Control Register (SRC_A53RCR0)

The A53 Reset Control Register (A53RCR), contains bits that control the A53 reset generation.

Address: 3039_0000h base + 4h offset = 3039_0004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DOM_EN	LOCK	Reserved		DOMAIN3	DOMAIN2	DOMAIN1	DOMAIN0	Reserved		A53_L2RESET	A53_SOC_DBG_RESET	MASK_WDOG1_RST			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	A53_ETM_RESET3	A53_ETM_RESET2	A53_ETM_RESET1	A53_ETM_RESET0	A53_DBG_RESET3	A53_DBG_RESET2	A53_DBG_RESET1	A53_DBG_RESET0	A53_CORE_RESET3	A53_CORE_RESET2	A53_CORE_RESET1	A53_CORE_RESET0	A53_CORE_POR_RESET3	A53_CORE_POR_RESET2	A53_CORE_POR_RESET1	A53_CORE_POR_RESET0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRC_A53RCR0 field descriptions

Field	Description
31 DOM_EN	Domain Control enable for this register. **NOTE:** [31:30] and [27:24] areas are not controlled by this bit. [31:30] and [27:24] area can be modified by any masters from any domains when Lock bit is not set. 0 Disables domain control. All of this register's bits except [31:30] and [27:24] can be modified by any masters 1 Enables domain control. All of this register's bits except [31:30] and [27:24] can only be modified by the masters from the domains specified in [27:24] area.
30 LOCK	Domain control bits lock **NOTE:** Lock bit is a write-once register, once it is set to 1, it can't be write to 0 0 [31] and [27:24] bits can be modified 1 [31] and [27:24] bits cannot be modified
29–28 -	This field is reserved. Reserved
27 DOMAIN3	Domain3 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain2. The master from domain2 cannot write to this register. 1 This register is assigned to domain2. The master from domain2 can write to this register
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain1 cannot write to this register. 1 This register is assigned to domain1. The master from domain1 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain3 cannot write to this register. 1 This register is assigned to domain0. The master from domain3 can write to this register
23–22 -	This field is reserved. Reserved
21 A53_L2RESET	Software reset for A53 Snoop Control Unit (SCU). **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert SCU reset 1 assert SCU reset
20 A53_SOC_DBG_RESET	Software reset for system level debug reset. It initializes the shared Debug APB, the CTI, and the CTM. It also causes: • A53_dbgreset[3:0] and A53_etmreset[3:0] to be asserted • debug logic in the processor power domain and in the debug power domain to be reset

Table continues on the next page...

SRC_A53RCR0 field descriptions (continued)

Field	Description
	NOTE: This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert system level debug reset 1 assert system level debug reset
19–16 MASK_WDOG1_RST	Mask wdog1_rst_b source. If these 4 bits are coded from A to 5 then, the wdog1_rst_b input to SRC will be masked and the wdog1_rst_b will not create a reset to the chip. **NOTE:** During the time the WDOG event is masked using SRC logic, it is likely that the WDOG Reset Status Register (WRSR) bit 1 (which indicates a WDOG timeout event) will get asserted. software / OS developer must prepare for this case. Re-enabling the WDOG is possible, by un-masking it in SRC, though it must be preceded by servicing the WDOG. However, for the case that the event has been asserted, the status bit (WRSR bit-1) will remain asserted, regardless of servicing the WDOG module. (Hardware reset is the only way to cause the de-assertion of that bit). Any other code will be coded to 1010 i.e. wdog1_rst_b is not masked 0101 wdog1_rst_b is masked 1010 wdog1_rst_b is not masked
15 A53_ETM_RESET3	Software reset for core3 ETM only. **NOTE:** This is a self clearing bit. Once it is set to 1, the rest process will begin, and once it finished, this bit will be self-cleared. 0 do not assert core3 ETM reset 1 assert core3 ETM reset
14 A53_ETM_RESET2	Software reset for core2 ETM only. **NOTE:** This is a self clearing bit. Once it is set to 1, the rest process will begin, and once it finished, this bit will be self-cleared. 0 do not assert core2 ETM reset 1 assert core2 ETM reset
13 A53_ETM_RESET1	Software reset for core1 ETM only. **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core1 ETM reset 1 assert core1 ETM reset
12 A53_ETM_RESET0	Software reset for core0 ETM only. **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core0 ETM reset 1 assert core0 ETM reset
11 A53_DBG_RESET3	Software reset for core3 debug only. It initialize the debug, and breakpoint and watchpoint logic in the core3 processor power domain. It also reset the debug logic for core1 processor, which is in the debug power domain.

Table continues on the next page...

SRC_A53RCR0 field descriptions (continued)

Field	Description
	NOTE: This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core3 debug reset 1 assert core3 debug reset
10 A53_DBG_RESET2	Software reset for core2 debug only. It initialize the debug, and breakpoint and watchpoint logic in the core2 processor power domain. It also reset the debug logic for core1 processor, which is in the debug power domain. **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core2 debug reset 1 assert core2 debug reset
9 A53_DBG_RESET1	Software reset for core1 debug only. It initialize the debug, and breakpoint and watchpoint logic in the core1 processor power domain. It also reset the debug logic for core1 processor, which is in the debug power domain. **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core1 debug reset 1 assert core1 debug reset
8 A53_DBG_RESET0	Software reset for core0 debug only. It initialize the debug, and breakpoint and watchpoint logic in the core1 processor power domain. It also reset the debug logic for core1 processor, which is in the debug power domain. **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core0 debug reset 1 assert core0 debug reset
7 A53_CORE_RESET3	Software reset for core3 only. It initializes the processor logic in the core1 processor power domains, not including the debug, breakpoint and watchpoint logic. **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core3 reset 1 assert core3 reset
6 A53_CORE_RESET2	Software reset for core2 only. It initializes the processor logic in the core1 processor power domains, not including the debug, breakpoint and watchpoint logic. **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core2 reset 1 assert core2 reset

Table continues on the next page...

SRC_A53RCR0 field descriptions (continued)

Field	Description
5 A53_CORE_ RESET1	Software reset for core1 only. It initializes the processor logic in the core1 processor power domains, not including the debug, breakpoint and watchpoint logic. **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core1 reset 1 assert core1 reset
4 A53_CORE_ RESET0	Software reset for core0 only. It initializes the processor logic in the core0 processor power domains, not including the debug, breakpoint and watchpoint logic. **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core0 reset 1 assert core0 reset
3 A53_CORE_ POR_RESET3	POR reset for A53 core3 only. It initializes all the core1 processor logic, including CPU Debug, and breakpoint and watchpoint logic in the core3 processor power domains **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core3 reset 1 assert core3 reset
2 A53_CORE_ POR_RESET2	POR reset for A53 core2 only. It initializes all the core1 processor logic, including CPU Debug, and breakpoint and watchpoint logic in the core2 processor power domains **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core2 reset 1 assert core2 reset
1 A53_CORE_ POR_RESET1	POR reset for A53 core1 only. It initializes all the core1 processor logic, including CPU Debug, and breakpoint and watchpoint logic in the core1 processor power domains **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core1 reset 1 assert core1 reset
0 A53_CORE_ POR_RESET0	POR reset for A53 core0 only. It initializes all the core0 processor logic, including CPU Debug, and breakpoint and watchpoint logic in the core0 processor power domains **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core0 reset 1 assert core0 reset

6.5.5.3 A53 Reset Control Register (SRC_A53RCR1)

The A53 Reset Control Register (A53RCR), contains bits that control the A53 reset generation.

Address: 3039_0000h base + 8h offset = 3039_0008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DOM_EN	LOCK	Reserved		DOMAIN3	DOMAIN2	DOMAIN1	DOMAIN0	Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								A53_RST_SLOW				A53_CORE3_ENABLE	A53_CORE2_ENABLE	A53_CORE1_ENABLE	A53_CORE0_ENABLE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

SRC_A53RCR1 field descriptions

Field	Description
31 DOM_EN	Domain Control enable for this register. **NOTE:** [31:30] and [27:24] areas are not controlled by this bit. [31:30] and [27:24] area can be modified by any masters from any domains when Lock bit is not set. 0 Disables domain control. All of this register's bits except [31:30] and [27:24] can be modified by any masters 1 Enables domain control. All of this register's bits except [31:30] and [27:24] can only be modified by the masters from the domains specified in [27:24] area.
30 LOCK	Domain control bits lock **NOTE:** Lock bit is a write-once register, once it is set to 1, it can't be write to 0 0 [31] and [27:24] bits can be modified 1 [31] and [27:24] bits cannot be modified
29–28 -	This field is reserved. Reserved
27 DOMAIN3	Domain3 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain2. The master from domain2 cannot write to this register. 1 This register is assigned to domain2. The master from domain2 can write to this register
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain1 cannot write to this register. 1 This register is assigned to domain1. The master from domain1 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain3 cannot write to this register. 1 This register is assigned to domain0. The master from domain3 can write to this register
23–7 -	This field is reserved. Reserved
6–4 A53_RST_SLOW	A53_RST_SLOW
3 A53_CORE3_ENABLE	core 3 enable 0 core3 is disabled 1 core3 is enabled
2 A53_CORE2_ENABLE	core 2 enable 0 core2 is disabled 1 core2 is enabled
1 A53_CORE1_ENABLE	core 1 enable

Table continues on the next page...

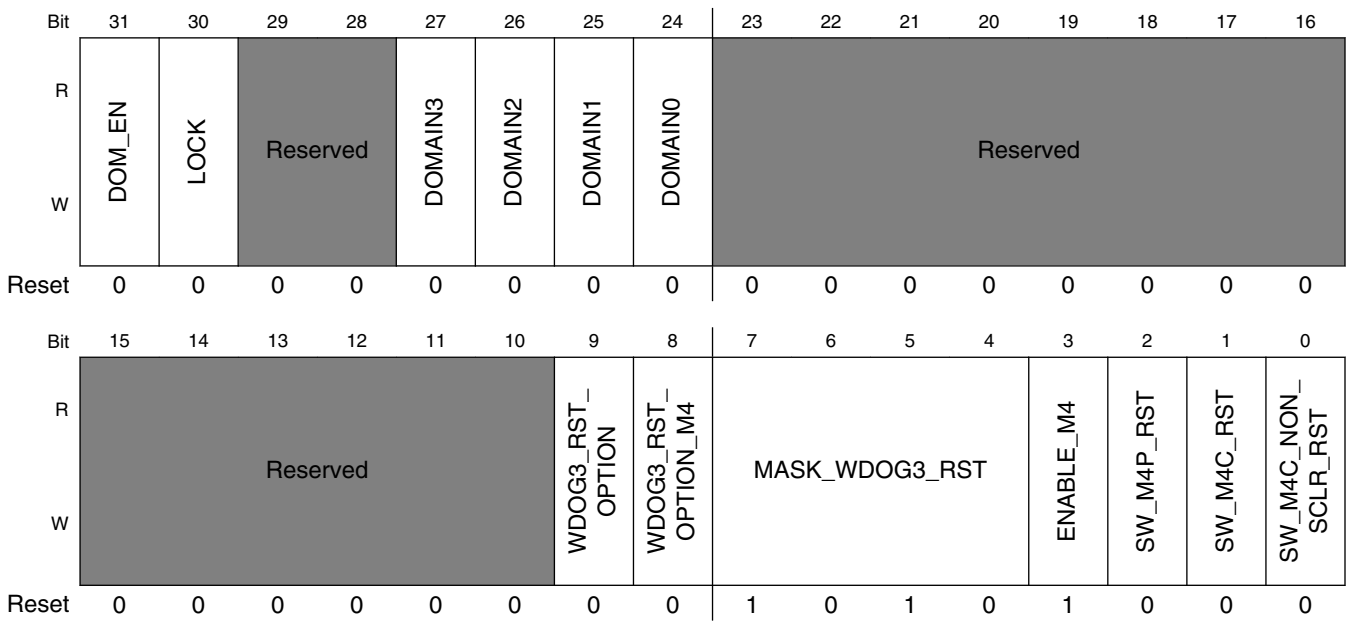
SRC_A53RCR1 field descriptions (continued)

Field	Description
	0 core1 is disabled 1 core1 is enabled
0 A53_CORE0_ENABLE	Always 1, can't be changed.

6.5.5.4 M4 Reset Control Register (SRC_M4RCR)

The M4 Reset Control Register (M4RCR), contains bits that control the M4 reset generation.

Address: 3039_0000h base + Ch offset = 3039_000Ch



SRC_M4RCR field descriptions

Field	Description
31 DOM_EN	Domain Control enable for this register. **NOTE:** [31:30] and [27:24] areas are not controlled by this bit. [31:30] and [27:24] area can be modified by any masters from any domains when Lock bit is not set. 0 Disables domain control. All of this register's bits except [31:30] and [27:24] can be modified by any masters 1 Enables domain control. All of this register's bits except [31:30] and [27:24] can only be modified by the masters from the domains specified in [27:24] area.
30 LOCK	Domain control bits lock

Table continues on the next page...

SRC_M4RCR field descriptions (continued)

Field	Description
	NOTE: Lock bit is a write-once register, once it is set to 1, it can't be write to 0 0 [31] and [27:24] bits can be modified 1 [31] and [27:24] bits cannot be modified
29–28 -	This field is reserved. Reserved
27 DOMAIN3	Domain3 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain2. The master from domain2 cannot write to this register. 1 This register is assigned to domain2. The master from domain2 can write to this register
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain1 cannot write to this register. 1 This register is assigned to domain1. The master from domain1 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain3 cannot write to this register. 1 This register is assigned to domain0. The master from domain3 can write to this register
23–10 -	This field is reserved. Reserved
9 WDOG3_RST_ OPTION	Wdog3_rst_b option 0 Wdog3_rst_b asserts M4 reset 1 Wdog3_rst_b asserts global reset
8 WDOG3_RST_ OPTION_M4	Wdog3_rst_b option for M4. This bit is only effective when wdog3_rst_option is set to 1. 0 wdgo3_rst_b Reset M4 core only 1 Reset both M4 core and platform
7–4 MASK_WDOG3_ RST	Mask wdog3_rst_b source. If these 4 bits are coded from A to 5 then, the wdog3_rst_b input to SRC will be masked and the wdog3_rst_b will not create a reset to the chip. **NOTE:** During the time the WDOG3 event is masked using SRC logic, it is likely that the WDOG3 Reset Status Register (WRSR) bit 1 (which indicates a WDOG3 timeout event) will get asserted. Software / OS developer must prepare for this case. Re-enabling the WDOG3 is possible, by un-masking it in SRC, though it must be preceded by servicing the WDOG3. However, for the case that the event has been asserted, the status bit (WRSR bit-1) will remain asserted, regardless of servicing the WDOG3 module. (Hardware reset is the only way to cause the de-assertion of that bit). Any other code will be coded to 1010 i.e. wdog3_rst_b is not masked 0101 wdog3_rst_b is masked 1010 wdog3_rst_b is not masked
3 ENABLE_M4	Enable M4 0 M4 is disabled 1 M4 is enabled

Table continues on the next page...

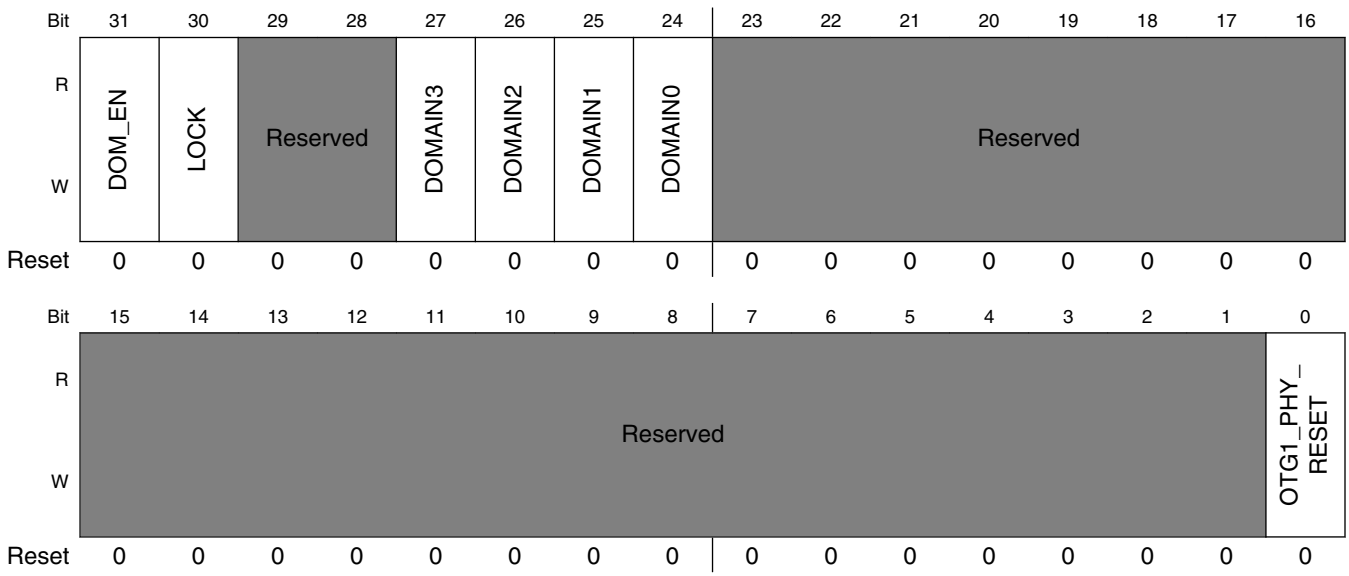
SRC_M4RCR field descriptions (continued)

Field	Description
2 SW_M4P_RST	Self-clearing SW reset for M4 platform **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. Software can determine that the reset has finished once this bit is cleared. Software can also configure SRC to generate interrupt once the software has finished. Please refer to SRC_SISR register for details. 0 do not assert M4 platform reset 1 assert M4 platform reset
1 SW_M4C_RST	Self-clearing SW reset for M4 core **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. Software can determine that the reset has finished once this bit is cleared. Software can also configure SRC to generate interrupt once the software has finished. Please refer to SRC_SISR register for details. 0 do not assert M4 core reset 1 assert M4 core reset
0 SW_M4C_NON_SCLR_RST	Non-self-clearing SW reset for M4 core 0 do not assert M4 core reset 1 assert M4 core reset

6.5.5.5 USB OTG PHY1 Reset Control Register (SRC_USBOPHY1_RCR)

The USB OTG PHY1 Reset Control Register (SRC_IP_RCR2), contains bits that control the USB OTG PHY1 reset generation.

Address: 3039_0000h base + 20h offset = 3039_0020h



SRC_USBOPHY1_RCR field descriptions

Field	Description
31 DOM_EN	Domain Control enable for this register. **NOTE:** [31:30] and [27:24] areas are not controlled by this bit. [31:30] and [27:24] area can be modified by any masters from any domains when Lock bit is not set. 0 Disables domain control. All of this register's bits except [31:30] and [27:24] can be modified by any masters 1 Enables domain control. All of this register's bits except [31:30] and [27:24] can only be modified by the masters from the domains specified in [27:24] area.
30 LOCK	Domain control bits lock **NOTE:** Lock bit is a write-once register, once it is set to 1, it can't be write to 0 0 [31] and [27:24] bits can be modified 1 [31] and [27:24] bits cannot be modified
29–28 -	This field is reserved. Reserved
27 DOMAIN3	Domain3 assignment control. Effective when dom_en is set to 1.

Table continues on the next page...

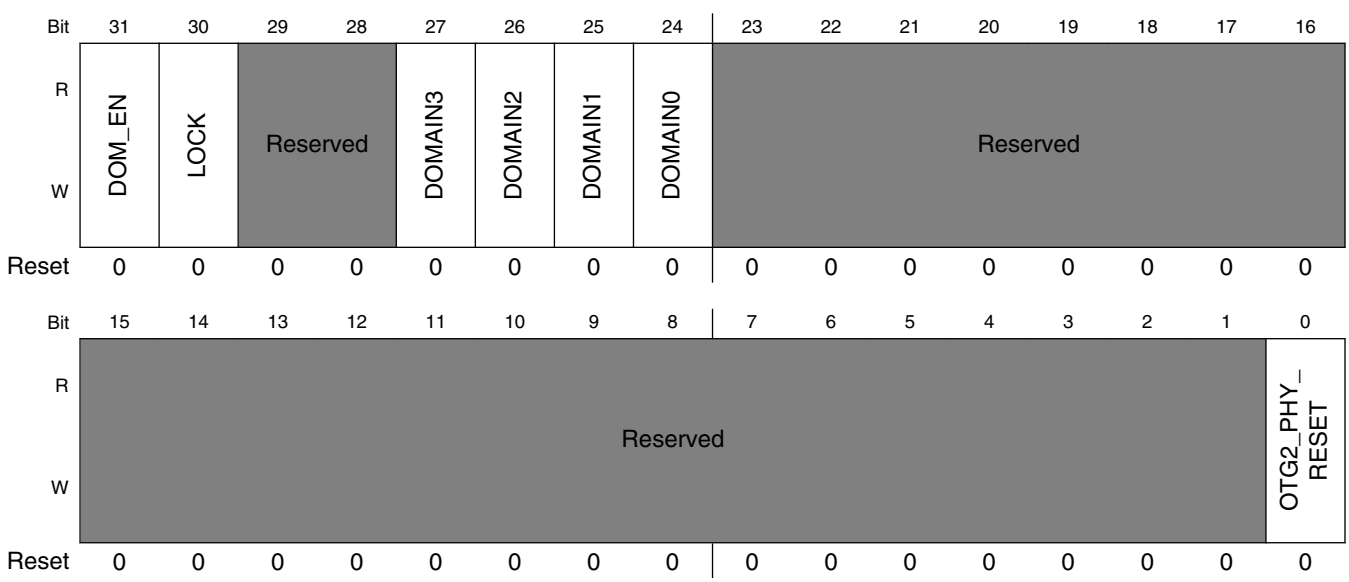
SRC_USBOPHY1_RCR field descriptions (continued)

Field	Description
	0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain2. The master from domain2 cannot write to this register. 1 This register is assigned to domain2. The master from domain2 can write to this register
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain1 cannot write to this register. 1 This register is assigned to domain1. The master from domain1 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain3 cannot write to this register. 1 This register is assigned to domain0. The master from domain3 can write to this register
23–1 -	This field is reserved. Reserved
0 OTG1_PHY_RESET	0 Don't reset USB OTG1 PHY 1 Reset USB OTG1 PHY

6.5.5.6 USB OTG PHY2 Reset Control Register (SRC_USBOPHY2_RCR)

The USB OTG PHY2 Reset Control Register (SRC_IP_RCR2), contains bits that control the USB OTG PHY2 reset generation.

Address: 3039_0000h base + 24h offset = 3039_0024h



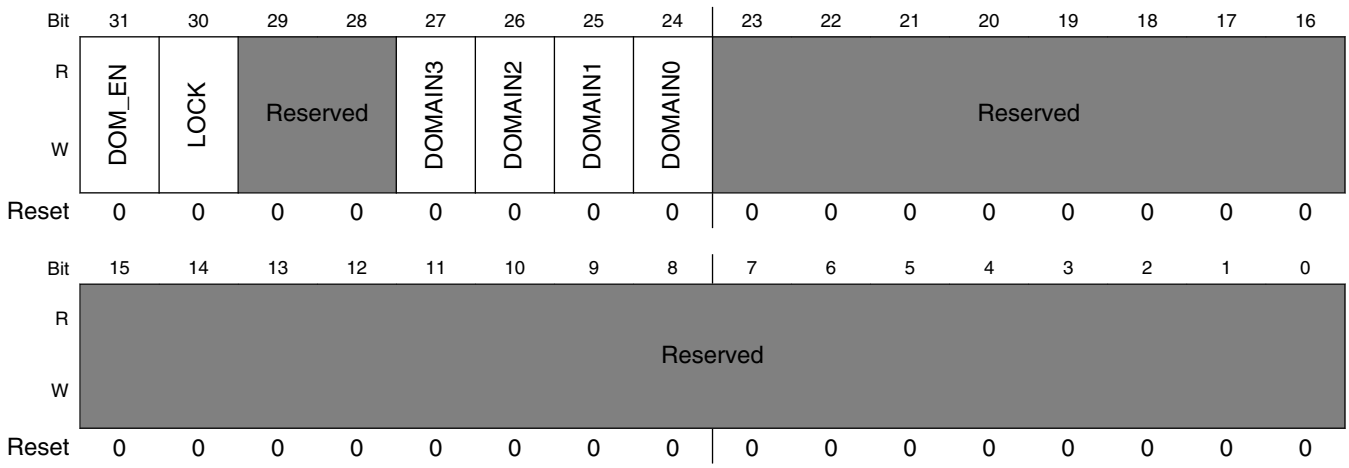
SRC_USBOPHY2_RCR field descriptions

Field	Description
31 DOM_EN	Domain Control enable for this register. **NOTE:** [31:30] and [27:24] areas are not controlled by this bit. [31:30] and [27:24] area can be modified by any masters from any domains when Lock bit is not set. 0 Disables domain control. All of this register's bits except [31:30] and [27:24] can be modified by any masters 1 Enables domain control. All of this register's bits except [31:30] and [27:24] can only be modified by the masters from the domains specified in [27:24] area.
30 LOCK	Domain control bits lock **NOTE:** Lock bit is a write-once register, once it is set to 1, it can't be write to 0 0 [31] and [27:24] bits can be modified 1 [31] and [27:24] bits cannot be modified
29–28 -	This field is reserved. Reserved
27 DOMAIN3	Domain3 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain2. The master from domain2 cannot write to this register. 1 This register is assigned to domain2. The master from domain2 can write to this register
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain1 cannot write to this register. 1 This register is assigned to domain1. The master from domain1 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain3 cannot write to this register. 1 This register is assigned to domain0. The master from domain3 can write to this register
23–1 -	This field is reserved. Reserved
0 OTG2_PHY_ RESET	0 Don't reset USB OTG2 PHY 1 Reset USB OTG2 PHY

6.5.5.7 MIPI PHY Reset Control Register (SRC_MIPIPHY_RCR)

The MIPI PHY Reset Control Register (SRC_MIPIPHY_RCR), contains bits that control the MIPI PHYreset generation.

Address: 3039_0000h base + 28h offset = 3039_0028h



SRC_MIPIPHY_RCR field descriptions

Field	Description
31 DOM_EN	Domain Control enable for this register. **NOTE:** [31:30] and [27:24] areas are not controlled by this bit. [31:30] and [27:24] area can be modified by any masters from any domains when Lock bit is not set. 0 Disables domain control. All of this register's bits except [31:30] and [27:24] can be modified by any masters 1 Enables domain control. All of this register's bits except [31:30] and [27:24] can only be modified by the masters from the domains specified in [27:24] area.
30 LOCK	Domain control bits lock **NOTE:** Lock bit is a write-once register, once it is set to 1, it can't be write to 0 0 [31] and [27:24] bits can be modified 1 [31] and [27:24] bits cannot be modified
29–28 -	This field is reserved. Reserved
27 DOMAIN3	Domain3 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain2. The master from domain2 cannot write to this register. 1 This register is assigned to domain2. The master from domain2 can write to this register

Table continues on the next page...

SRC_MIPIPHY_RCR field descriptions (continued)

Field	Description
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain1 cannot write to this register. 1 This register is assigned to domain1. The master from domain1 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain3 cannot write to this register. 1 This register is assigned to domain0. The master from domain3 can write to this register
-	This field is reserved. Reserved

6.5.5.8 PCIE PHY Reset Control Register (SRC_PCIEPHY_RCR)

The PCIE PHY Control Register (SRC_PCIEPHY_RCR), contains bits that control the PCIE PHY reset generation.

Address: 3039_0000h base + 2Ch offset = 3039_002Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DOM_EN	LOCK	Reserved		DOMAIN3	DOMAIN2	DOMAIN1	DOMAIN0	Reserved							PCIE_CTRL_APP_XFER_PENDING
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PCIE_CTRL_APP_UNLOCK_MSG	PCIE_CTRL_SYS_INT	Reserved	PCIE_CTRL_CFG_L1_AUX	PCIE_CTRL_APPS_TURNOFF	PCIE_CTRL_APPS_PME	PCIE_CTRL_APPS_EXIT	PCIE_CTRL_APPS_ENTER	PCIE_CTRL_APPS_READY	PCIE_CTRL_APPS_EN	PCIE_CTRL_APPS_RST	PCIE_CTRL_APPS_CLK_REQ	PCIEPHY_PERST	PCIEPHY_BTNRST	Reserved	PCIE_PHY_POWER_ON_RESET
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

SRC_PCIEPHY_RCR field descriptions

Field	Description
31 DOM_EN	Domain Control enable for this register.

Table continues on the next page...

SRC_PCIEPHY_RCR field descriptions (continued)

Field	Description
	NOTE: [31:30] and [27:24] areas are not controlled by this bit. [31:30] and [27:24] area can be modified by any masters from any domains when Lock bit is not set. 0 Disables domain control. All of this register's bits except [31:30] and [27:24] can be modified by any masters 1 Enables domain control. All of this register's bits except [31:30] and [27:24] can only be modified by the masters from the domains specified in [27:24] area.
30 LOCK	Domain control bits lock **NOTE:** Lock bit is a write-once register, once it is set to 1, it can't be write to 0 0 [31] and [27:24] bits can be modified 1 [31] and [27:24] bits cannot be modified
29–28 -	This field is reserved. Reserved
27 DOMAIN3	Domain3 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain2. The master from domain2 cannot write to this register. 1 This register is assigned to domain2. The master from domain2 can write to this register
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain1 cannot write to this register. 1 This register is assigned to domain1. The master from domain1 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain3 cannot write to this register. 1 This register is assigned to domain0. The master from domain3 can write to this register
23–17 -	This field is reserved. Reserved
16 PCIE_CTRL_ APP_XFER_ PENDING	PCIE_CTRL_APP_XFER_PENDING
15 PCIE_CTRL_ APP_UNLOCK_ MSG	PCIE_CTRL_APP_UNLOCK_MSG
14 PCIE_CTRL_ SYS_INT	PCIE_CTRL_SYS_INT
13 -	This field is reserved.
12 PCIE_CTRL_ CFG_L1_AUX	Pcie_ctrl_cfg_l1_aux_clk_switch_core_clk_gate_en

Table continues on the next page...

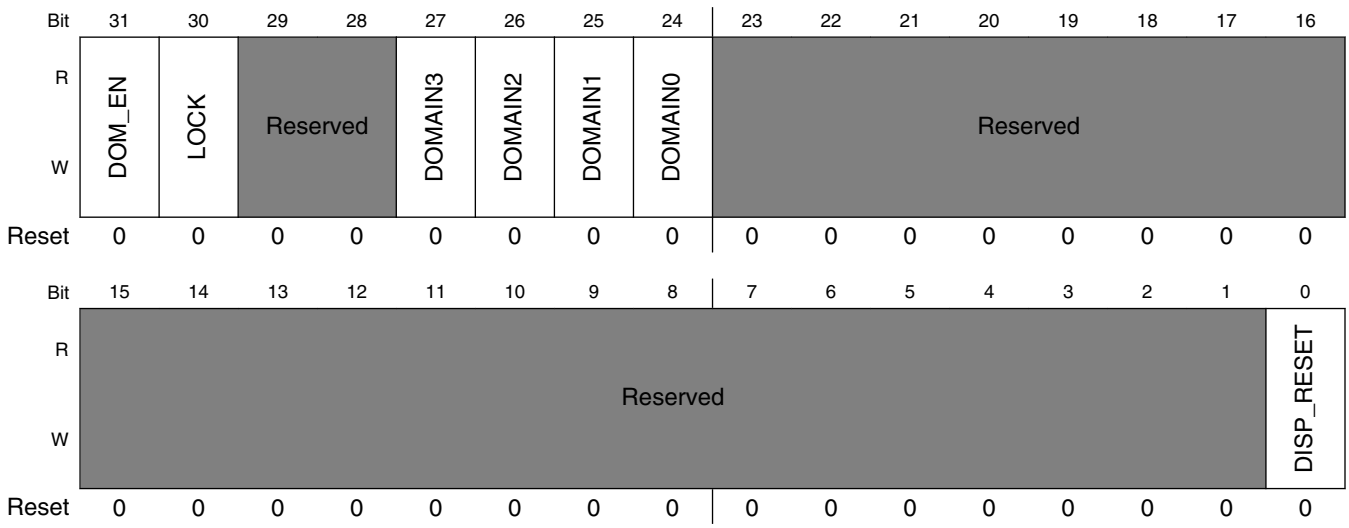
SRC_PCIEPHY_RCR field descriptions (continued)

Field	Description
11 PCIE_CTRL_APPS_TURNOFF	Pcie_ctrl_apps_pm_xmt_turnoff
10 PCIE_CTRL_APPS_PME	Pcie_ctrl_apps_pm_xmt_pme
9 PCIE_CTRL_APPS_EXIT	Pcie_ctrl_app_req_exit_l1
8 PCIE_CTRL_APPS_ENTER	Pcie_ctrl_app_req_entr_l1
7 PCIE_CTRL_APPS_READY	Pcie_ctrl_app_ready_entr_l23
6 PCIE_CTRL_APPS_EN	Pcie_ctrl_app_ltssm_enable
5 PCIE_CTRL_APPS_RST	Pcie_ctrl_app_init_rst
4 PCIE_CTRL_APPS_CLK_REQ	Pcie_ctrl_app_clk_req_n
3 PCIEPHY_PERST	Pciephy_perst
2 PCIEPHY_BTNRST	PCIE PHY button
1 -	This field is reserved. Reserved
0 PCIE_PHY_POWER_ON_RESET	PCIE_PHY_POWER_ON_RESET

6.5.5.9 DISPLAY Reset Control Register (SRC_DISP_RCR)

The DISPLAY Control Register contains bits that control the DISPLAY reset generation.

Address: 3039_0000h base + 34h offset = 3039_0034h



SRC_DISP_RCR field descriptions

Field	Description
31 DOM_EN	Domain Control enable for this register. **NOTE:** [31:30] and [27:24] areas are not controlled by this bit. [31:30] and [27:24] area can be modified by any masters from any domains when Lock bit is not set. 0 Disables domain control. All of this register's bits except [31:30] and [27:24] can be modified by any masters 1 Enables domain control. All of this register's bits except [31:30] and [27:24] can only be modified by the masters from the domains specified in [27:24] area.
30 LOCK	Domain control bits lock **NOTE:** Lock bit is a write-once register, once it is set to 1, it can't be write to 0 0 [31] and [27:24] bits can be modified 1 [31] and [27:24] bits cannot be modified
29–28 -	This field is reserved. Reserved
27 DOMAIN3	Domain3 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1.

Table continues on the next page...

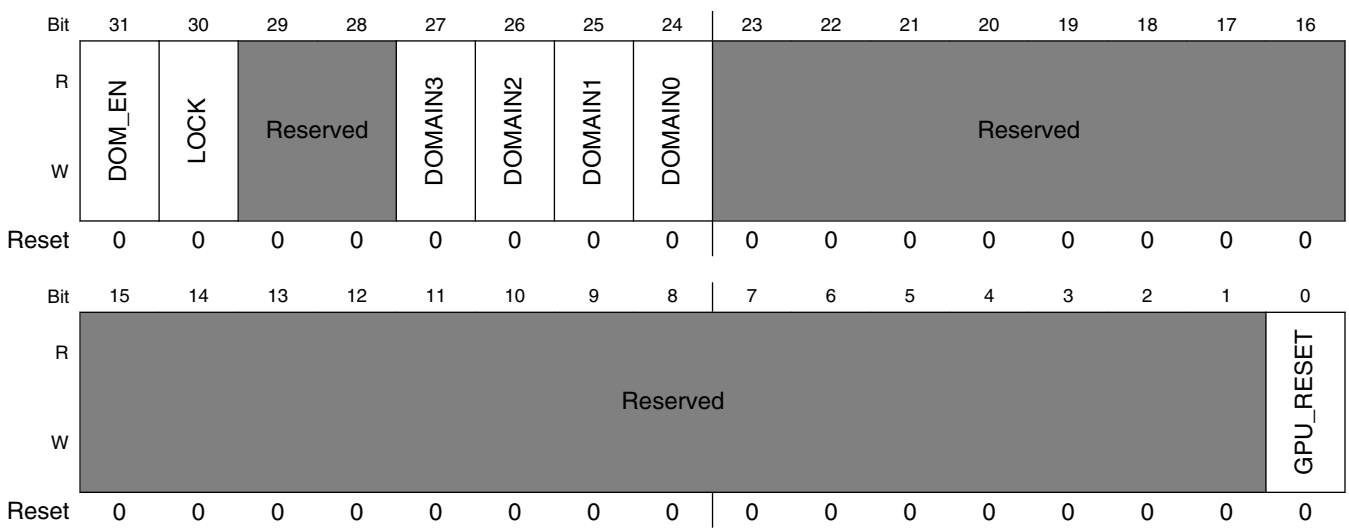
SRC_DISP_RCR field descriptions (continued)

Field	Description
	0 This register is not assigned to domain2. The master from domain2 cannot write to this register. 1 This register is assigned to domain2. The master from domain2 can write to this register
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain1 cannot write to this register. 1 This register is assigned to domain1. The master from domain1 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain3 cannot write to this register. 1 This register is assigned to domain0. The master from domain3 can write to this register
23–1 -	This field is reserved. Reserved
0 DISP_RESET	0 Don't reset dispmix 1 Reset dispmix

6.5.5.10 GPU Reset Control Register (SRC_GPU_RCR)

The GPU Control Register contains bits that control the GPU reset generation.

Address: 3039_0000h base + 40h offset = 3039_0040h



SRC_GPU_RCR field descriptions

Field	Description
31 DOM_EN	Domain Control enable for this register. **NOTE:** [31:30] and [27:24] areas are not controlled by this bit. [31:30] and [27:24] area can be modified by any masters from any domains when Lock bit is not set.

Table continues on the next page...

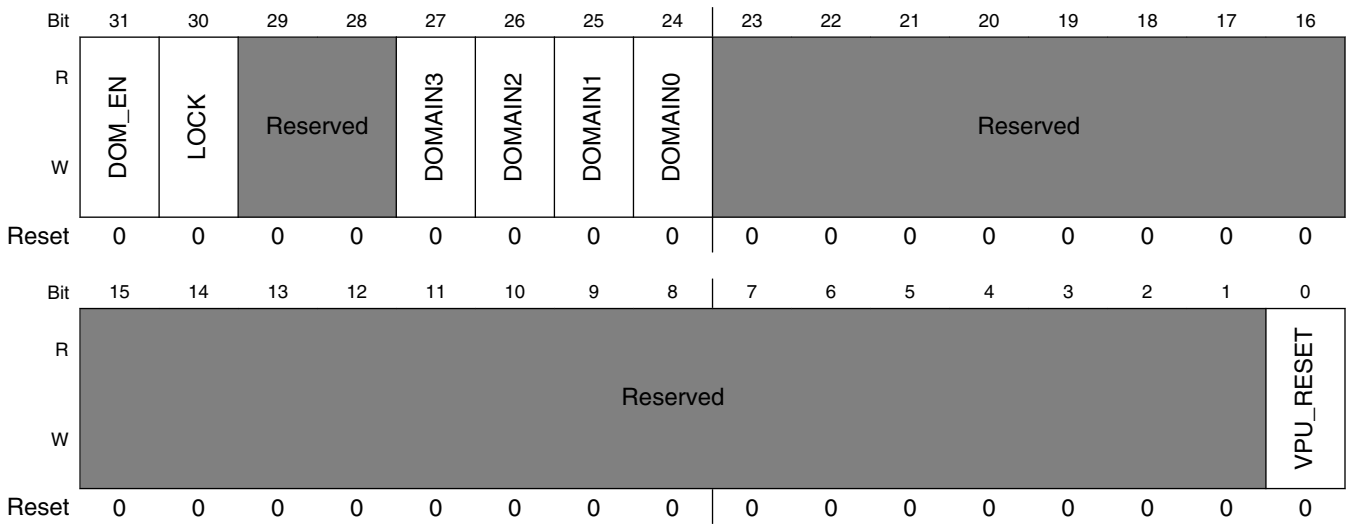
SRC_GPU_RCR field descriptions (continued)

Field	Description
	0 Disables domain control. All of this register's bits except [31:30] and [27:24] can be modified by any masters 1 Enables domain control. All of this register's bits except [31:30] and [27:24] can only be modified by the masters from the domains specified in [27:24] area.
30 LOCK	Domain control bits lock **NOTE:** Lock bit is a write-once register, once it is set to 1, it can't be write to 0 0 [31] and [27:24] bits can be modified 1 [31] and [27:24] bits cannot be modified
29–28 -	This field is reserved. Reserved
27 DOMAIN3	Domain3 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain2. The master from domain2 cannot write to this register. 1 This register is assigned to domain2. The master from domain2 can write to this register
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain1 cannot write to this register. 1 This register is assigned to domain1. The master from domain1 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain3 cannot write to this register. 1 This register is assigned to domain0. The master from domain3 can write to this register
23–1 -	This field is reserved. Reserved
0 GPU_RESET	GPU_RESET

6.5.5.11 VPU Reset Control Register (SRC_VPU_RCR)

The VPU Control Register contains bits that control the VPU reset generation.

Address: 3039_0000h base + 44h offset = 3039_0044h



SRC_VPU_RCR field descriptions

Field	Description
31 DOM_EN	Domain Control enable for this register. **NOTE:** [31:30] and [27:24] areas are not controlled by this bit. [31:30] and [27:24] area can be modified by any masters from any domains when Lock bit is not set. 0 Disables domain control. All of this register's bits except [31:30] and [27:24] can be modified by any masters 1 Enables domain control. All of this register's bits except [31:30] and [27:24] can only be modified by the masters from the domains specified in [27:24] area.
30 LOCK	Domain control bits lock **NOTE:** Lock bit is a write-once register, once it is set to 1, it can't be write to 0 0 [31] and [27:24] bits can be modified 1 [31] and [27:24] bits cannot be modified
29–28 -	This field is reserved. Reserved
27 DOMAIN3	Domain3 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1.

Table continues on the next page...

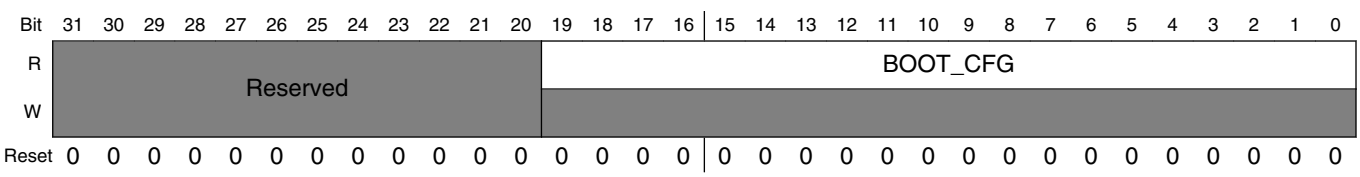
SRC_VPU_RCR field descriptions (continued)

Field	Description
	0 This register is not assigned to domain2. The master from domain2 cannot write to this register. 1 This register is assigned to domain2. The master from domain2 can write to this register
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain1 cannot write to this register. 1 This register is assigned to domain1. The master from domain1 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain3 cannot write to this register. 1 This register is assigned to domain0. The master from domain3 can write to this register
23–1 -	This field is reserved. Reserved
0 VPU_RESET	VPU_RESET

6.5.5.12 SRC Boot Mode Register 1 (SRC_SBMR1)

The Boot Mode register (SBMR) contains bits that reflect the status of Boot Mode Pins of the chip. The reset value is configuration dependent (depending on boot/fuses/IO pads).

Address: 3039_0000h base + 58h offset = 3039_0058h



SRC_SBMR1 field descriptions

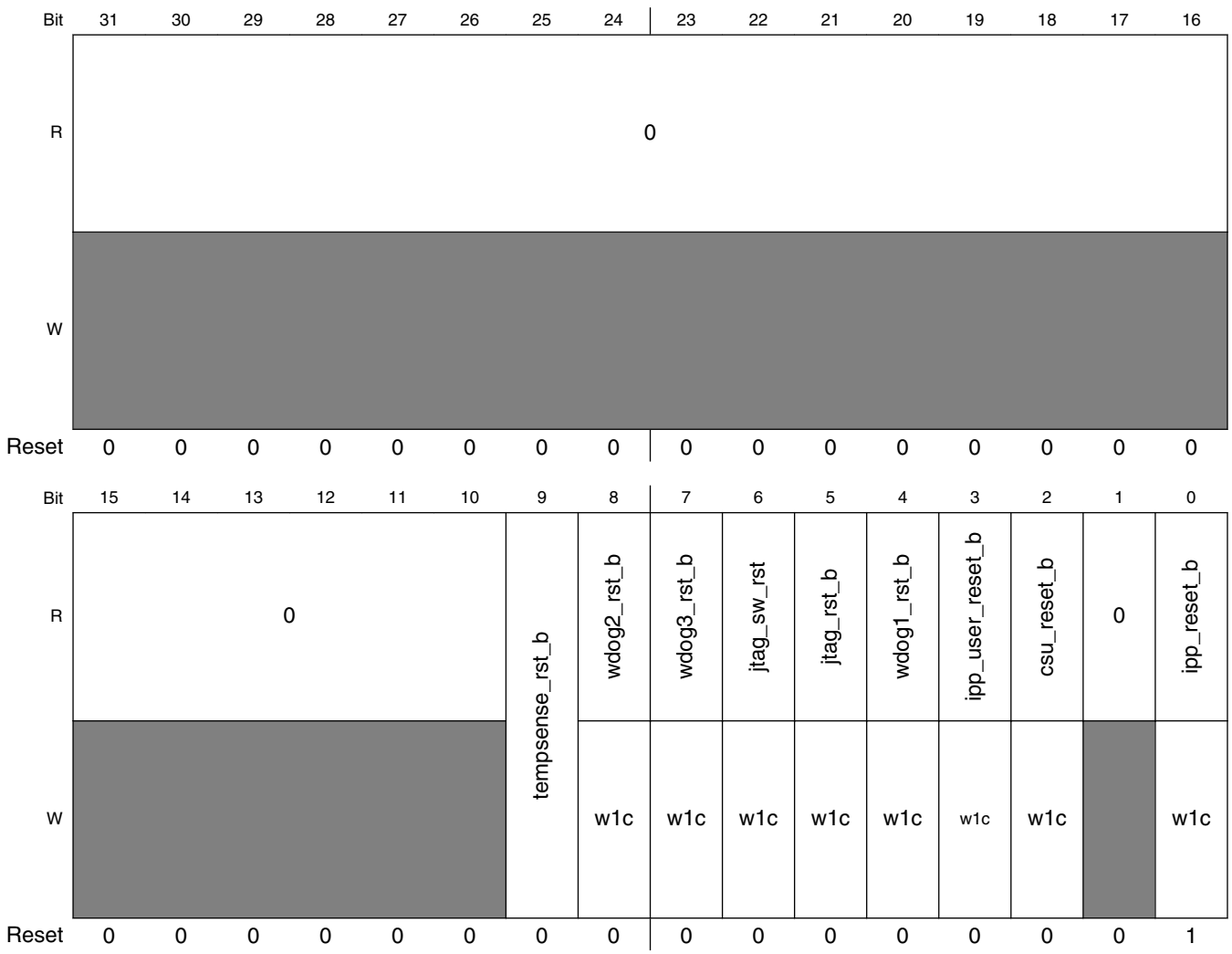
Field	Description
31–20 -	This field is reserved. Reserved
BOOT_CFG	Refer to fusemap.

6.5.5.13 SRC Reset Status Register (SRC_SRSR)

The SRSR is a write to one clear register which records the source of the reset events for the chip. The SRC reset status register will capture all the reset sources that have occurred. This register is reset on ipp_reset_b. This is a read-write register.

For bit[9-0] - writing zero does not have any effect. Writing one will clear the corresponding bit. The individual bits can be cleared by writing one to that bit. When the system comes out of reset, this register will have bits set corresponding to all the reset sources that occurred during system reset. Software has to take care to clear this register by writing one after every reset that occurs so that the register will contain the information of recently occurred reset.

Address: 3039_0000h base + 5Ch offset = 3039_005Ch



SRC_SRSR field descriptions

Field	Description
31–10 Reserved	This read-only field is reserved and always has the value 0.
9 tempsense_rst_b	Temper Sensor software reset. Indicates whether the reset was the result of software reset from on-chip Temperature Sensor.

Table continues on the next page...

SRC_SRSR field descriptions (continued)

Field	Description
	0 Reset is not a result of software reset from Temperature Sensor. 1 Reset is a result of software reset from Temperature Sensor.
8 wdog2_rst_b	IC Watchdog2 Time-out reset. Indicates whether the reset was the result of the watchdog2 time-out event. 0 Reset is not a result of the watchdog4 time-out event. 1 Reset is a result of the watchdog4 time-out event.
7 wdog3_rst_b	IC Watchdog3 Time-out reset. Indicates whether the reset was the result of the watchdog3 time-out event. 0 Reset is not a result of the watchdog3 time-out event. 1 Reset is a result of the watchdog3 time-out event.
6 jtag_sw_rst	JTAG software reset. Indicates whether the reset was the result of software reset from JTAG. 0 Reset is not a result of software reset from JTAG. 1 Reset is a result of software reset from JTAG.
5 jtag_rst_b	HIGH - Z JTAG reset. Indicates whether the reset was the result of HIGH-Z reset from JTAG. 0 Reset is not a result of HIGH-Z reset from JTAG. 1 Reset is a result of HIGH-Z reset from JTAG.
4 wdog1_rst_b	IC Watchdog1 Time-out reset. Indicates whether the reset was the result of the watchdog time-out event. 0 Reset is not a result of the watchdog1 time-out event. 1 Reset is a result of the watchdog1 time-out event.
3 ipp_user_reset_b	Indicates whether the reset was the result of the ipp_user_reset_b qualified reset. 0 Reset is not a result of the ipp_user_reset_b qualified as COLD reset event. 1 Reset is a result of the ipp_user_reset_b qualified as COLD reset event.
2 csu_reset_b	Indicates whether the reset was the result of the csu_reset_b input. **NOTE:** If case the csu_reset_b occurred during a WARM reset process, during the phase that ipg_clk is not available yet, then the occurrence of CSU reset will not be reflected in this bit. 0 Reset is not a result of the csu_reset_b event. 1 Reset is a result of the csu_reset_b event.
1 Reserved	This read-only field is reserved and always has the value 0.
0 ipp_reset_b	Indicates whether reset was the result of ipp_reset_b pin (Power-up sequence) 0 Reset is not a result of ipp_reset_b pin. 1 Reset is a result of ipp_reset_b pin.

6.5.5.14 SRC Interrupt Status Register (SRC_SISR)

Address: 3039_0000h base + 68h offset = 3039_0068h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

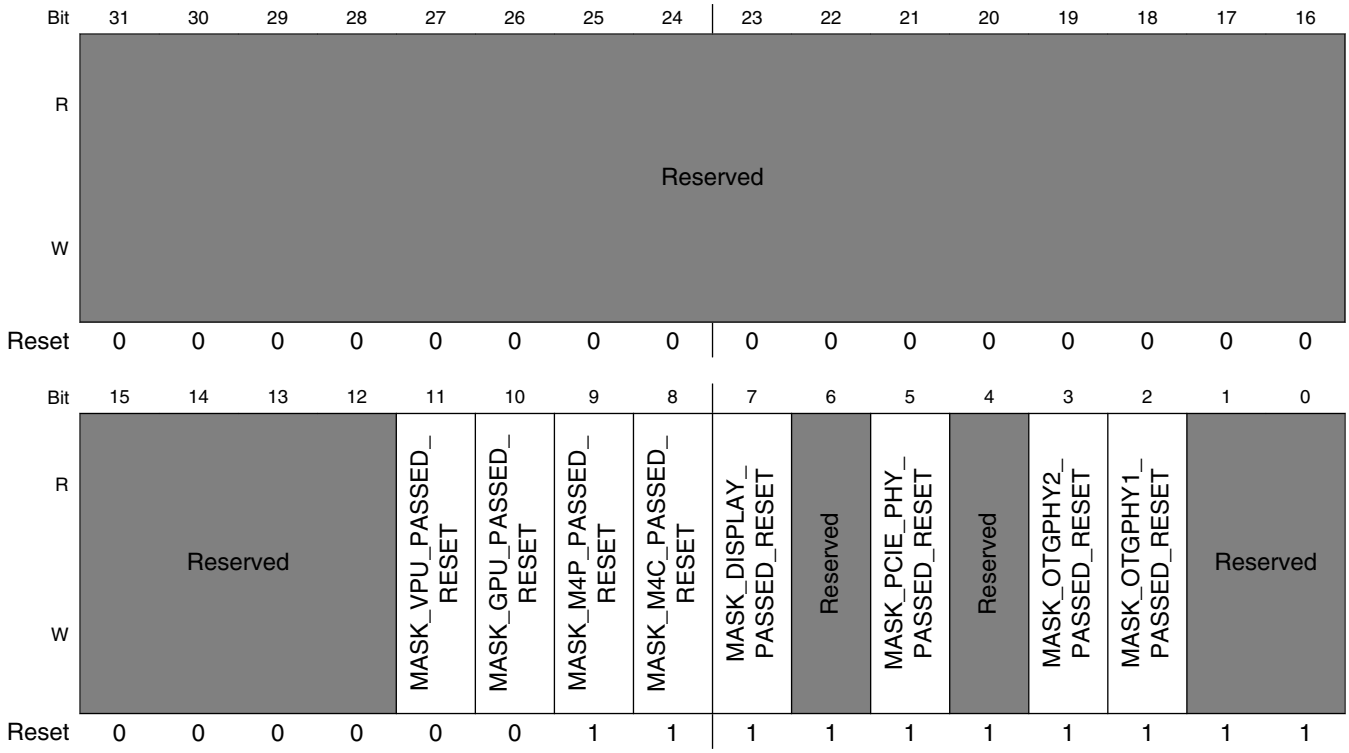
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				VPU_PASSED_RESET	GPU_PASSED_RESET	M4P_PASSED_RESET	M4C_PASSED_RESET	DISPLAY_PASSED_RESET	Reserved	PCIE1_PHY_PASSED_RESET	Reserved	OTGPHY2_PASSED_RESET	OTGPHY1_PASSED_RESET	Reserved	Reserved
W					w1c	w1c	w1c	w1c	w1c		w1c		w1c	w1c		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRC_SISR field descriptions

Field	Description
31–12 -	This field is reserved. Reserved
11 VPU_PASSED_ RESET	Interrupt generated to indicate that VPU passed software reset and is ready to be used 0 interrupt generated not due to VPU reset 1 interrupt generated due to VPU reset
10 GPU_PASSED_ RESET	Interrupt generated to indicate that GPU passed software reset and is ready to be used 0 interrupt generated not due to GPU reset 1 interrupt generated due to GPU reset
9 M4P_PASSED_ RESET	Interrupt generated to indicate that m4 platform passed software reset and is ready to be used 0 interrupt generated not due to m4 platform reset 1 interrupt generated due to m4 platform reset
8 M4C_PASSED_ RESET	Interrupt generated to indicate that m4 core passed software reset and is ready to be used 0 interrupt generated not due to m4 core reset 1 interrupt generated due to m4 core reset
7 DISPLAY_ PASSED_ RESET	Interrupt generated to indicate that DISPLAY passed software reset and is ready to be used 0 Interrupt generated not due to DISPLAY passed reset 1 Interrupt generated due to DISPLAY passed reset
6 -	This field is reserved. Reserved
5 PCIE1_PHY_ PASSED_ RESET	Interrupt generated to indicate that PCIE1 PHY passed software reset and is ready to be used 0 Interrupt generated not due to PCIE1 PHY passed reset 1 Interrupt generated due to PCIE1 PHY passed reset
4 -	This field is reserved. Reserved
3 OTGPHY2_ PASSED_ RESET	Interrupt generated to indicate that OTG PHY2 passed software reset and is ready to be used 0 Interrupt generated not due to OTG PHY2 passed reset 1 Interrupt generated due to OTG PHY2 passed reset
2 OTGPHY1_ PASSED_ RESET	Interrupt generated to indicate that OTG PHY1 passed software reset and is ready to be used 0 Interrupt generated not due to OTG PHY1 passed reset 1 Interrupt generated due to OTG PHY1 passed reset
1 -	This field is reserved. Reserved
0 -	This field is reserved. Reserved

6.5.5.15 SRC Interrupt Mask Register (SRC_SIMR)

Address: 3039_0000h base + 6Ch offset = 3039_006Ch



SRC_SIMR field descriptions

Field	Description
31–12 -	This field is reserved. Reserved
11 MASK_VPU_PASSED_RESET	Mask interrupt generation due to VPU passed reset 0 do not mask interrupt due to VPU passed reset - interrupt will be created 1 mask interrupt due to VPU passed reset
10 MASK_GPU_PASSED_RESET	Mask interrupt generation due to GPU passed reset 0 do not mask interrupt due to GPU passed reset - interrupt will be created 1 mask interrupt due to GPU passed reset
9 MASK_M4P_PASSED_RESET	mask interrupt generation due to m4 platform passed reset 0 do not mask interrupt due to m4 platform passed reset - interrupt will be created 1 mask interrupt due to m4platform passed reset
8 MASK_M4C_PASSED_RESET	mask interrupt generation due to m4 core passed reset 0 do not mask interrupt due to m4 core passed reset - interrupt will be created 1 mask interrupt due to m4 core passed reset

Table continues on the next page...

SRC_SIMR field descriptions (continued)

Field	Description
7 MASK_ DISPLAY_ PASSED_ RESET	Mask interrupt generation due to display passed reset 0 do not mask interrupt due to display passed reset - interrupt will be created 1 mask interrupt due to display passed reset
6 -	This field is reserved. Reserved
5 MASK_PCIE_ PHY_PASSED_ RESET	Mask interrupt generation due to PCIE PHY passed reset 0 do not mask interrupt due to PCIE PHY passed reset - interrupt will be created 1 mask interrupt due to PCIE PHY passed reset
4 -	This field is reserved. Reserved
3 MASK_ OTGPHY2_ PASSED_ RESET	mask interrupt generation due to OTG PHY2 passed reset 0 do not mask interrupt due to OTG PHY2 passed reset - interrupt will be created 1 mask interrupt due to OTG PHY2 passed reset
2 MASK_ OTGPHY1_ PASSED_ RESET	mask interrupt generation due to OTG PHY1 passed reset 0 do not mask interrupt due to OTG PHY1 passed reset - interrupt will be created 1 mask interrupt due to OTG PHY1 passed reset
-	This field is reserved. Reserved

6.5.5.16 SRC Boot Mode Register 2 (SRC_SBMR2)

The Boot Mode register (SBMR), contains bits that reflect the status of Boot Mode Pins of the chip. The default values for those bits depends on the values of pins/fuses during reset sequence.

Address: 3039_0000h base + 70h offset = 3039_0070h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						IPP_BOOT_MODE		0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								FORCE_COLD_BOOT			BT_FUSE_SEL	DIR_BT_DIS	0	SEC_CONFIG[1:0]	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRC_SBMR2 field descriptions

Field	Description
31–26 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

SRC_SBMR2 field descriptions (continued)

Field	Description
25–24 IPP_BOOT_MODE	IPP_BOOT_MODE shows the latched state of the BOOT_MODE1 and BOOT_MODE0 signals on the rising edge of POR_B. See the Boot mode pin settings section of System Boot.
23–8 Reserved	This read-only field is reserved and always has the value 0.
7–5 FORCE_COLD_BOOT	See Fusemap for additional information.
4 BT_FUSE_SEL	BT_FUSE_SEL (connected to gpio bt_fuse_sel) shows the state of the BT_FUSE_SEL fuse. See Fusemap for additional information on this fuse.
3 DIR_BT_DIS	DIR_BT_DIS shows the state of the DIR_BT_DIS fuse. See the fusemap for additional information on this fuse.
2 Reserved	This read-only field is reserved and always has the value 0.
SEC_CONFIG[1:0]	SEC_CONFIG[1] shows the state of the SEC_CONFIG[1] fuse and SEC_CONFIG[0] shows the state of the SEC_CONFIG[0] fuse. See Fusemap for additional information on this fuse.

6.5.5.17 SRC General Purpose Register 1 (SRC_GPR1)

Address: 3039_0000h base + 74h offset = 3039_0074h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																C0_START_ADDRH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRC_GPR1 field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
C0_START_ADDRH	Core0 start reset address: RVBARADDR0 = {SRC_GPR1[15:0], SRC_GPR2[21:2]}

6.5.5.18 SRC General Purpose Register 2 (SRC_GPR2)

Address: 3039_0000h base + 78h offset = 3039_0078h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										C0_START_ADDRL																					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRC_GPR2 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
C0_START_ADDRL	Core0 start reset address: RVBARADDR0 = {SRC_GPR1[15:0], SRC_GPR2[21:2]}

6.5.5.19 SRC General Purpose Register 3 (SRC_GPR3)

Address: 3039_0000h base + 7Ch offset = 3039_007Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																C1_START_ADDRH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRC_GPR3 field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
C1_START_ADDRH	Core1 start reset address: RVBARADDR1 = {SRC_GPR3[15:0], SRC_GPR4[21:2]}

6.5.5.20 SRC General Purpose Register 4 (SRC_GPR4)

Address: 3039_0000h base + 80h offset = 3039_0080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										C1_START_ADDRL																					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRC_GPR4 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
C1_START_ADDRL	Core1 start reset address: RVBARADDR1 = {SRC_GPR3[15:0], SRC_GPR4[21:2]}

6.5.5.21 SRC General Purpose Register 5 (SRC_GPR5)

Address: 3039_0000h base + 84h offset = 3039_0084h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																C2_START_ADDRH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRC_GPR5 field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
C2_START_ADDRH	Core2 start reset address: RVBARADDR2 = {SRC_GPR5[15:0], SRC_GPR6[21:2]}

6.5.5.22 SRC General Purpose Register 6 (SRC_GPR6)

Address: 3039_0000h base + 88h offset = 3039_0088h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										C2_START_ADDRL																					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRC_GPR6 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
C2_START_ADDRL	Core2 start reset address: RVBARADDR2 = {SRC_GPR5[15:0], SRC_GPR6[21:2]}

6.5.5.23 SRC General Purpose Register 7 (SRC_GPR7)

Address: 3039_0000h base + 8Ch offset = 3039_008Ch

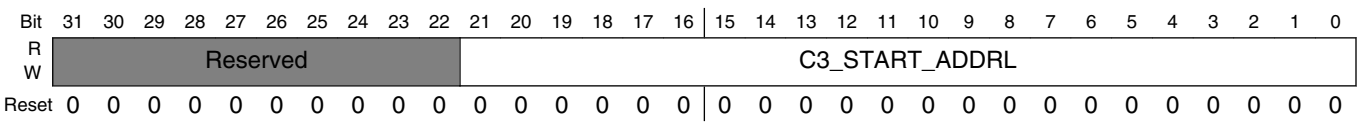
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																C3_START_ADDRH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRC_GPR7 field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
C3_START_ADDRH	Core3 start reset address: RVBARADDR3 = {SRC_GPR7[15:0], SRC_GPR8[21:2]}

6.5.5.24 SRC General Purpose Register 8 (SRC_GPR8)

Address: 3039_0000h base + 90h offset = 3039_0090h



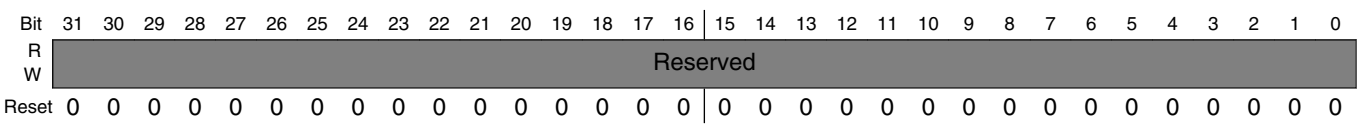
SRC_GPR8 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
C3_START_ADDRL	Core3 start reset address: RVBARADDR3 = {SRC_GPR7[15:0], SRC_GPR8[21:2]}

6.5.5.25 SRC General Purpose Register 9 (SRC_GPR9)

Reserved for Internal Use.

Address: 3039_0000h base + 94h offset = 3039_0094h



SRC_GPR9 field descriptions

Field	Description
-	This field is reserved. Reserved.

6.5.5.26 SRC General Purpose Register 10 (SRC_GPR10)

Address: 3039_0000h base + 98h offset = 3039_0098h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRC_GPR10 field descriptions

Field	Description
-	This field is reserved. Reserved.

6.5.5.27 SRC DDR Controller Reset Control Register (SRC_DDRC_RCR)

DDR Controller Reset Control Register

Address: 3039_0000h base + 1000h offset = 3039_1000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DOM_EN	LOCK	Reserved		DOMAIN3	DOMAIN2	DOMAIN1	DOMAIN0	Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										DDRC1_PHY_WRST	DDRC1_SYS_RST	DDRC1_PHY_PWROKIN	DDRC1_PHY_RESET	DDRC1_CORE_RST	DDRC1_PRST
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

SRC_DDRC_RCR field descriptions

Field	Description
31 DOM_EN	Domain Control enable for this register. **NOTE:** [31:30] and [27:24] areas are not controlled by this bit. [31:30] and [27:24] area can be modified by any masters from any domains when Lock bit is not set.

Table continues on the next page...

SRC_DDRC_RCR field descriptions (continued)

Field	Description
	0 Disables domain control. All of this register's bits except [31:30] and [27:24] can be modified by any masters 1 Enables domain control. All of this register's bits except [31:30] and [27:24] can only be modified by the masters from the domains specified in [27:24] area.
30 LOCK	Domain control bits lock **NOTE:** Lock bit is a write-once register, once it is set to 1, it can't be write to 0 0 [31] and [27:24] bits can be modified 1 [31] and [27:24] bits cannot be modified
29–28 -	This field is reserved. Reserved
27 DOMAIN3	Domain3 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain2. The master from domain2 cannot write to this register. 1 This register is assigned to domain2. The master from domain2 can write to this register
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain1 cannot write to this register. 1 This register is assigned to domain1. The master from domain1 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain0 cannot write to this register. 1 This register is assigned to domain0. The master from domain0 can write to this register
23–6 -	This field is reserved. Reserved
5 DDRC1_PHY_WRST	Active 1
4 DDRC1_SYS_RST	Active 1
3 DDRC1_PHY_PWROKIN	0 De-assert DDR controller 1 Assert DDR Controller
2 DDRC1_PHY_RESET	0 De-assert DDR controller 1 Assert DDR Controller
1 DDRC1_CORE_RST	DDR Controller core_ddrc_rstn and aresetn. 0 De-assert DDR controller aresetn and core_ddrc_rstn 1 Assert DDR Controller preset and DDR PHY reset
0 DDRC1_PRST	DDR Controller preset and DDR PHY reset.

Table continues on the next page...

SRC_DDRC_RCR field descriptions (continued)

Field	Description
	NOTE: This reset can only be released when DDR Controller clock inputs are active and stable for 30 cycles 0 De-assert DDR Controller preset and DDR PHY reset reset 1 Assert DDR Controller preset and DDR PHY reset

6.6 Watchdog Timer (WDOG)

6.6.1 Overview

The Watchdog Timer (WDOG) protects against system failures by providing a method by which to escape from unexpected events or programming errors.

Once the WDOG is activated, it must be serviced by the software on a periodic basis. If servicing does not take place, the timer times out. Upon timeout, the WDOG asserts the internal system reset signal, WDOG_RESET_B_DEB to the System Reset Controller (SRC).

There is also a provision for WDOG signal assertion by timeout counter expiration. There is an option of programmable interrupt generation before the counter actually times out. The time at which the interrupt needs to be generated prior to counter timeout is programmable. There is a power down counter which is enabled out of any reset (POR, Warm/Cold). This counter has a fixed timeout period of 16 seconds, upon which it asserts the WDOG signal.

Flow diagrams for the timeout counter, power down counter and interrupt operations are are shown in Flow Diagrams.

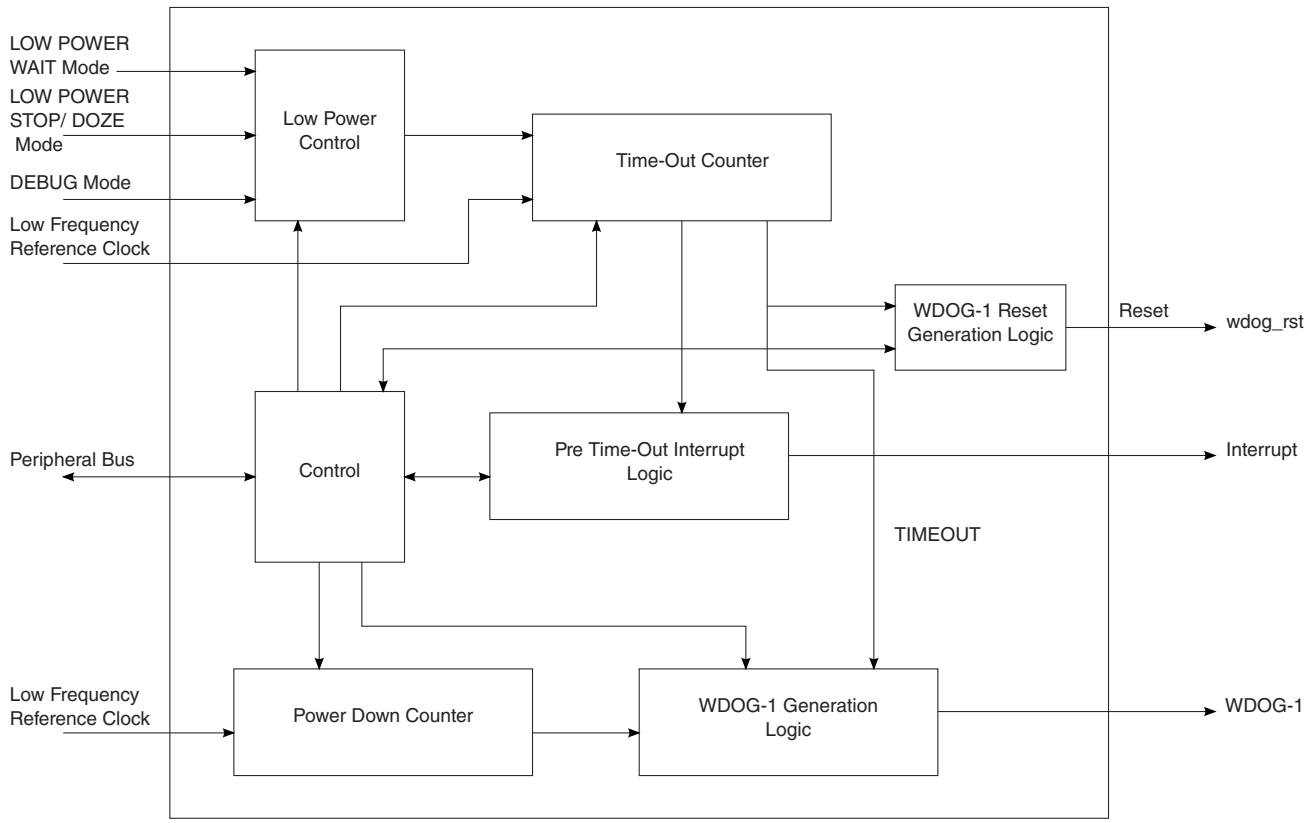


Figure 6-36. WDOG Diagram

6.6.1.1 Features

The WDOG features are listed below:

- Configurable timeout counter with timeout periods from 0.5 to 128 seconds which, after timeout expiration, result in the assertion of WDOG_RESET_B_DEB reset signal .
- Time resolution of 0.5 seconds
- Configurable timeout counter that can be programmed to run or stop during low-power modes
- Configurable timeout counter that can be programmed to run or stop during DEBUG mode
- Programmable interrupt generation prior to timeout
- The duration between interrupt and timeout events can be programmed from 0 to 127.5 seconds in steps of 0.5 seconds.
- Power down counter with fixed timeout period of 16 seconds, which if not disabled after reset will assert WDOG_B signal low

- Power down counter will be enabled out of any reset (POR, Warm / Cold reset) by default.

6.6.2 Functional description

This section provides a complete functional description of the block.

6.6.2.1 Timeout event

The WDOG provides timeout periods from 0.5 to 128 seconds with a time resolution of 0.5 seconds.

The user can determine the timeout period by writing to the WDOG timeout field (WT[7:0]) in the Watchdog Control Register (WDOG_WCR). The WDOG must be enabled by setting the WDE bit of Watchdog Control Register (WDOG_WCR) for the timeout counter to start running. After the WDOG is enabled, the counter is activated, loads the timeout value and begins to count down from this programmed value. The timer will time out when the counter reaches zero and the WDOG outputs a system reset signal, WDOG_RESET_B_DEB and asserts WDOG_B (WDT bit should be set in Watchdog Control Register (WDOG_WCR)).

However, the timeout condition can be prevented by reloading the counter with the new timeout value (WT[7:0] of WDOG_WCR) if a service routine (see Servicing WDOG to reload the counter) is performed before the counter reaches zero. If any system errors occur which prevent the software from servicing the Watchdog Service Register (WDOG_WSR), the timeout condition occurs. By performing the service routine, the WDOG reloads its counter to the timeout value indicated by bits WT[7:0] of the Watchdog Control Register (WDOG_WCR) and it restarts the countdown.

A system reset will reset the counter and place it in the idle state at any time during the countdown. The counter flow diagram is shown in Flow Diagrams.

NOTE

The timeout value is reloaded to the counter either at the time WDOG is enabled or after the service routine has been performed.

6.6.2.1.1 Servicing WDOG to reload the counter

To reload a timeout value to the counter the proper service sequence begins by writing 0x_5555 followed by 0x_AAAA to the Watchdog Service Register (WDOG_WSR). Any number of instructions can be executed between the two writes. If the WDOG_WSR is

not loaded with 0x_5555 prior to writing 0x_AAAA to the WDOG_WSR, the counter is not reloaded. If any value other than 0x_AAAA is written to the WDOG_WSR after 0x_5555, the counter is not reloaded. This service sequence will reload the counter with the timeout value WT[7:0] of Watchdog Control Register (WDOG_WCR). The timeout value can be changed at any point; it is reloaded when WDOG is serviced by the core.

6.6.2.2 Interrupt event

Prior to timeout, the WDOG can generate an interrupt which can be considered a warning that timeout will occur shortly.

The duration between interrupt event and timeout event can be controlled by writing to the WICT field of Watchdog Interrupt Control Register (WDOG_WICR). It can vary between 0 and 127.5 seconds. If the WDOG is serviced (Servicing WDOG to reload the counter)before the interrupt generation, the counter will be reloaded with the timeout value WT[7:0] of Watchdog Control Register (WDOG_WCR) and the interrupt will not be triggered.

6.6.2.3 Power-down counter event

The power-down counter inside WDOG will be enabled out of reset. This counter has a fixed timeout value of 16 seconds, after which it will drive the WDOG_B signal low.

To prevent this, the software must disable this counter by clearing the PDE bit of Watchdog Miscellaneous Control Register (WDOG_WMCR) within 16 seconds of reset deassertion. Once disabled, this counter can't be enabled again until the next system reset occurs. This feature is intended to prevent the hanging up of cores after reset, as WDOG is not enabled out of reset.

6.6.2.4 Low power modes

6.6.2.4.1 STOP and DOZE mode

If the WDOG timer disable bit for low power STOP and DOZE mode (WDZST) bit in the Watchdog Control Register (WDOG_WCR), is cleared, the WDOG timer continues to operate using the low frequency reference clock. If the low power enable (WDZST) bit is set, the WDOG timer operation will be suspended in low power STOP or DOZE mode. Upon exiting low power STOP or DOZE mode, the WDOG operation returns to what it was prior to entering the STOP or DOZE mode.

6.6.2.4.2 WAIT mode

If the WDOG timer disable bit for low power WAIT mode (WDW) bit in the Watchdog Control Register (WDOG_WCR), is cleared, the WDOG timer continues to operate using the low frequency reference clock . If the low power WAIT enable (WDW) bit is set, the WDOG timer operation will be suspended. Upon exiting low power WAIT mode, the WDOG operation returns to what it was prior to entering the WAIT mode.

NOTE

The WDOG timer won't be able to detect events that happen for periods shorter than one low frequency reference clock cycle. For example, in repeated WAIT mode entry or exit, if the RUN mode time is less than one low frequency reference clock cycle and if the WDW bit is set, the WDOG timer may never time out, even though the system is in RUN mode for a finite duration; WDOG may not see a low frequency reference clock edge during its wake time.

6.6.2.5 Debug mode

The WDOG timer can be configured for continual operation, or for suspension during debug mode. If the WDOG debug enable (WDBG) bit is set in the Watchdog Control Register (WDOG_WCR), the WDOG timer operation is suspended in debug mode. If the WDBG bit is set and the debug mode is entered, WDOG timer operation is suspended after two low frequency reference clocks. Similarly, WDOG timer operation continues after two low frequency reference clocks of debug mode exit. Register read and write accesses in debug mode continue to function normally. Also, while in debug mode, the WDE bit of Watchdog Control Register (WDOG_WCR) can be enabled/disabled directly. If the WDOG debug enable (WDBG) bit is cleared then WDOG timer operation is not suspended. The power-down counter is not affected by debug mode entry/exit.

NOTE

If the WDE bit of Watchdog Control Register (WDOG_WCR) is set/cleared while in debug mode, it remains set/cleared even after exiting debug mode.

6.6.2.6 Operations

6.6.2.6.1 Watchdog reset generation

The WDOG generated reset signal WDOG_RESET_B_DEB is asserted by the following operations:

- A software write to the Software Reset Signal (SRS) bit of the Watchdog Control Register (WDOG_WCR).
- WDOG timeout. See Timeout event.

The $\overline{\text{wdog\_rst}}$ will be asserted for one clock cycle of low frequency reference clock for both a timeout condition and a software write occurrence. It remains asserted for 1 clock cycle of low frequency reference clock even if a system reset is asserted in between. Figure 6-38 shows the timing diagram of this signal due to a timeout condition.

6.6.2.6.2 WDOG_B generation

The WDOG asserts WDOG_B in the following scenarios:

- Software write to WDA bit of Watchdog Control Register (WDOG_WCR). WDOG_B signal remains asserted as long as the WDA bit is "0".
- WDOG timeout condition, WDT bit of Watchdog Control Register (WDOG_WCR) must be set for this scenario. A description of the timeout condition can be found in the Timeout event. WDOG_B signal remains asserted until a power-on reset (POR) occurs. It gets cleared after the POR occurs (not due to any other system reset). Figure 6-39 shows the timing diagram of WDOG_B due to timeout condition.
- WDOG power-down counter timeout, PDE bit of Watchdog Miscellaneous Control Register (WDOG_WMCR) should not be cleared for this scenario. A description of this counter can be found in the Power-down counter event. WDOG_B signal remains asserted for one clock cycle of low frequency reference clock.

Figure 6-37 shows the scenarios under which WDOG_B gets asserted.

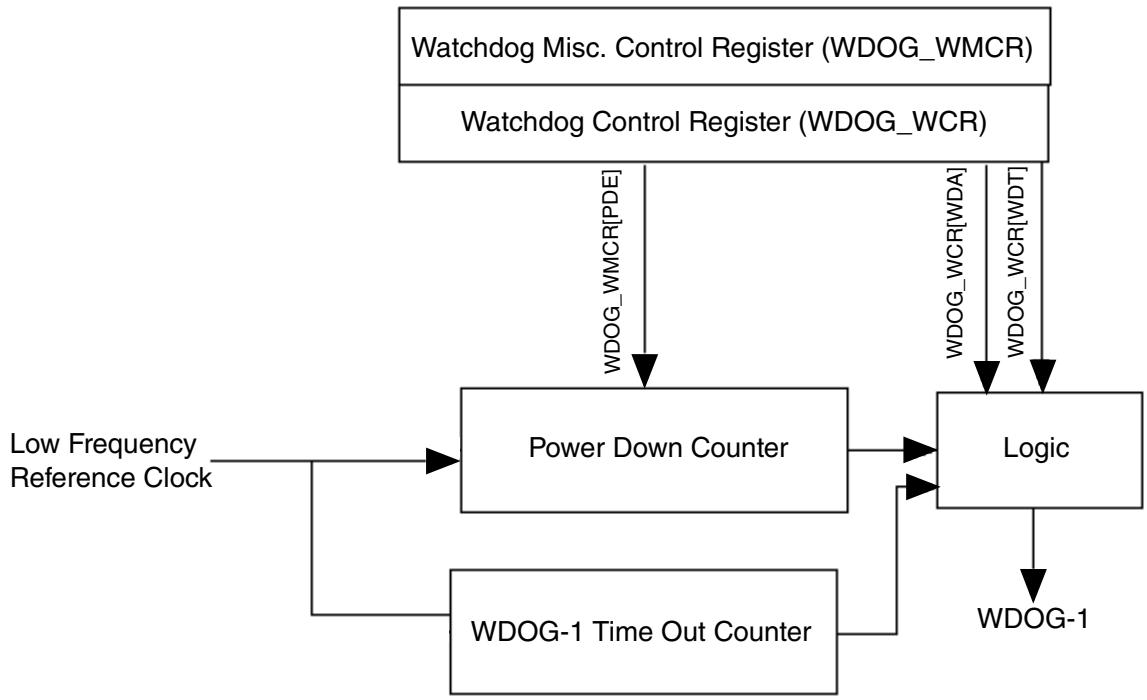


Figure 6-37. WDOG_B generation

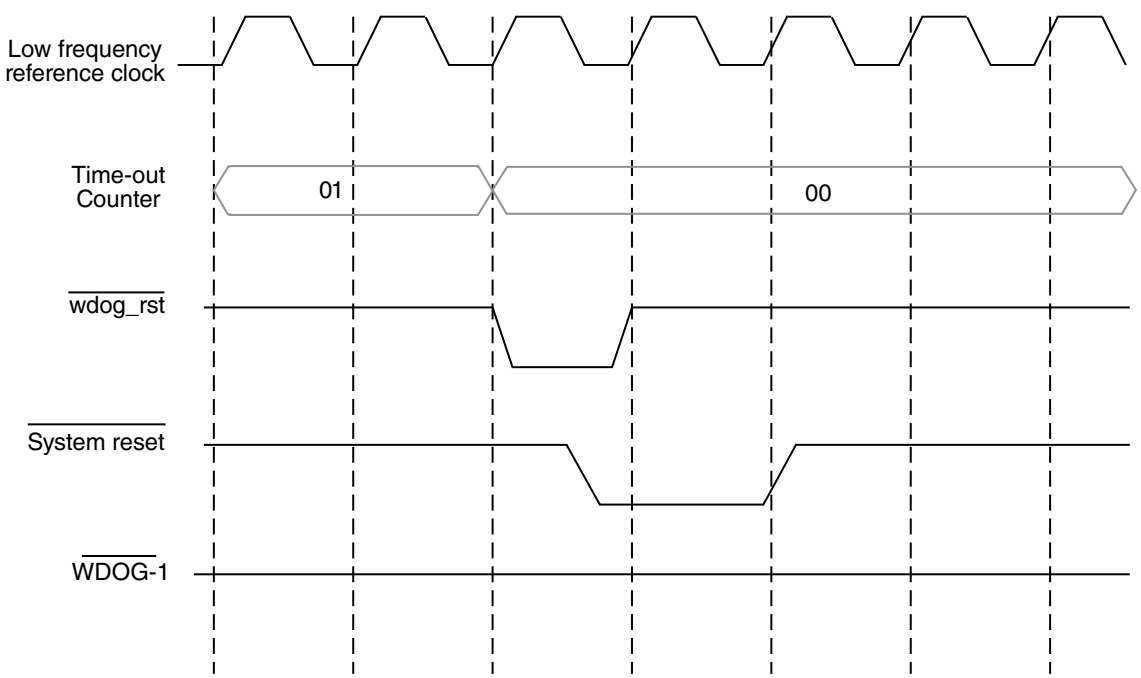


Figure 6-38. WDOG timeout condition/WDT bit is not set

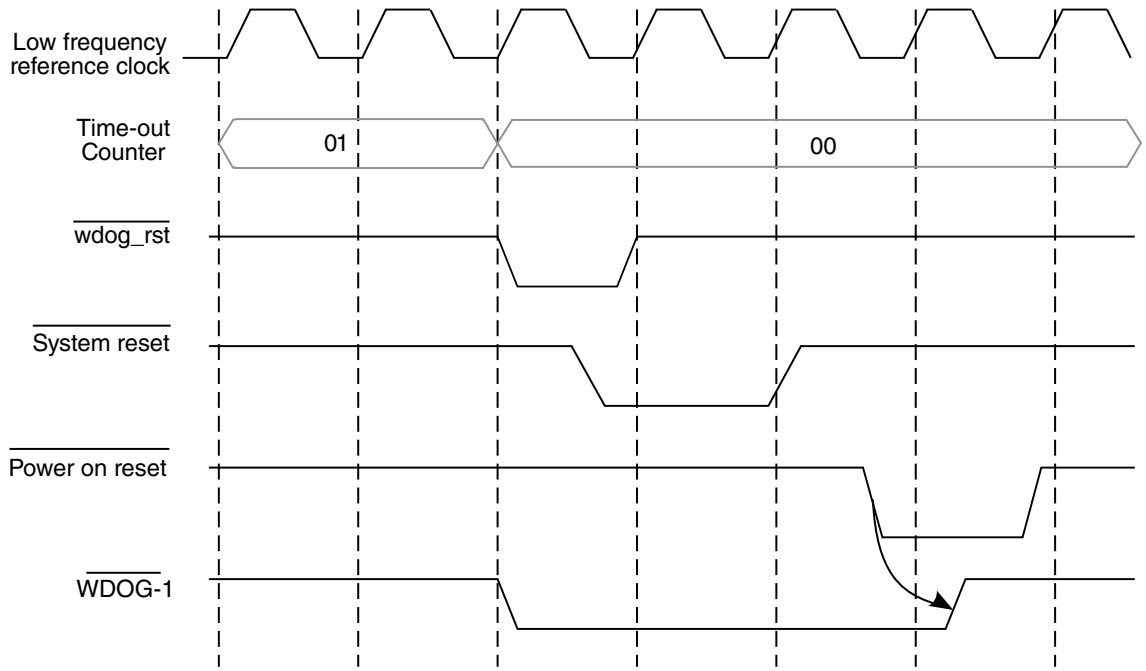


Figure 6-39. WDOG timeout condition/WDT bit is set

6.6.2.7 Reset

The block is reset by a system reset and the WDOG counter will be disabled. The power-down counter is enabled and starts counting.

6.6.2.8 Interrupt

The WDOG has the feature of Interrupt generation before timeout.

The interrupt will be generated only if the WIE bit in Watchdog Interrupt Control Register (WDOG_WICR) is set. The exact time at which the interrupt should occur (prior to timeout) depends on the value of WICT field of Watchdog Interrupt Control Register (WDOG_WICR). For example, if the WICT field has a value 0x04, then the interrupt will be generated two seconds prior to timeout. Once the interrupt is triggered the WTIS bit in Watchdog Interrupt Control Register (WDOG_WICR) will be set. The software needs to clear this bit to deassert the interrupt. If the WDOG is serviced before the interrupt generation then the counter will be reloaded with the timeout value WT[7:0] of Watchdog Control Register (WDOG_WCR) and interrupt would not be triggered.

6.6.2.9 Flow Diagrams

A flow diagram of WDOG operation is shown below.

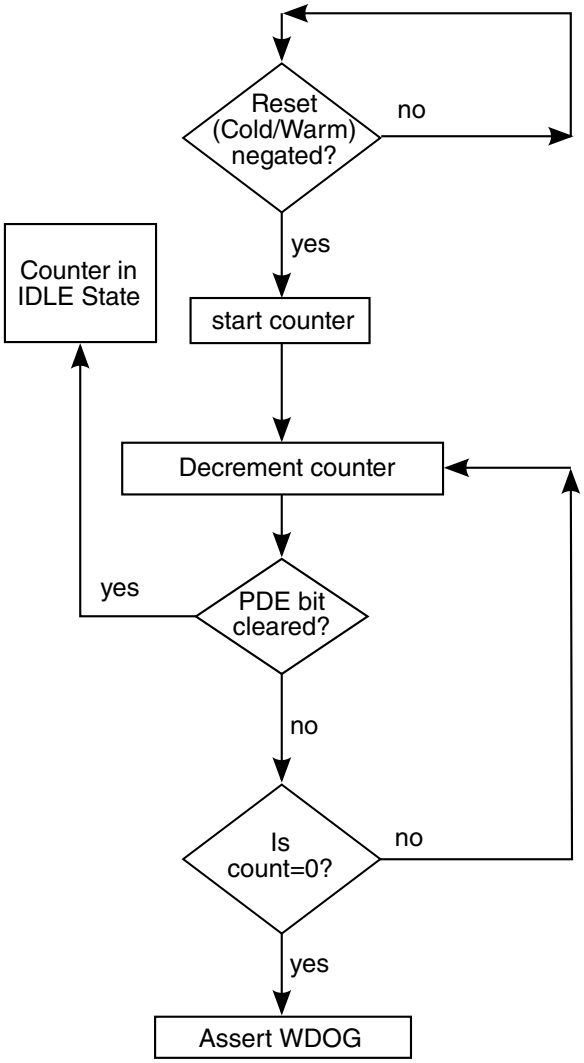


Figure 6-40. Power-Down Counter Flow Diagram

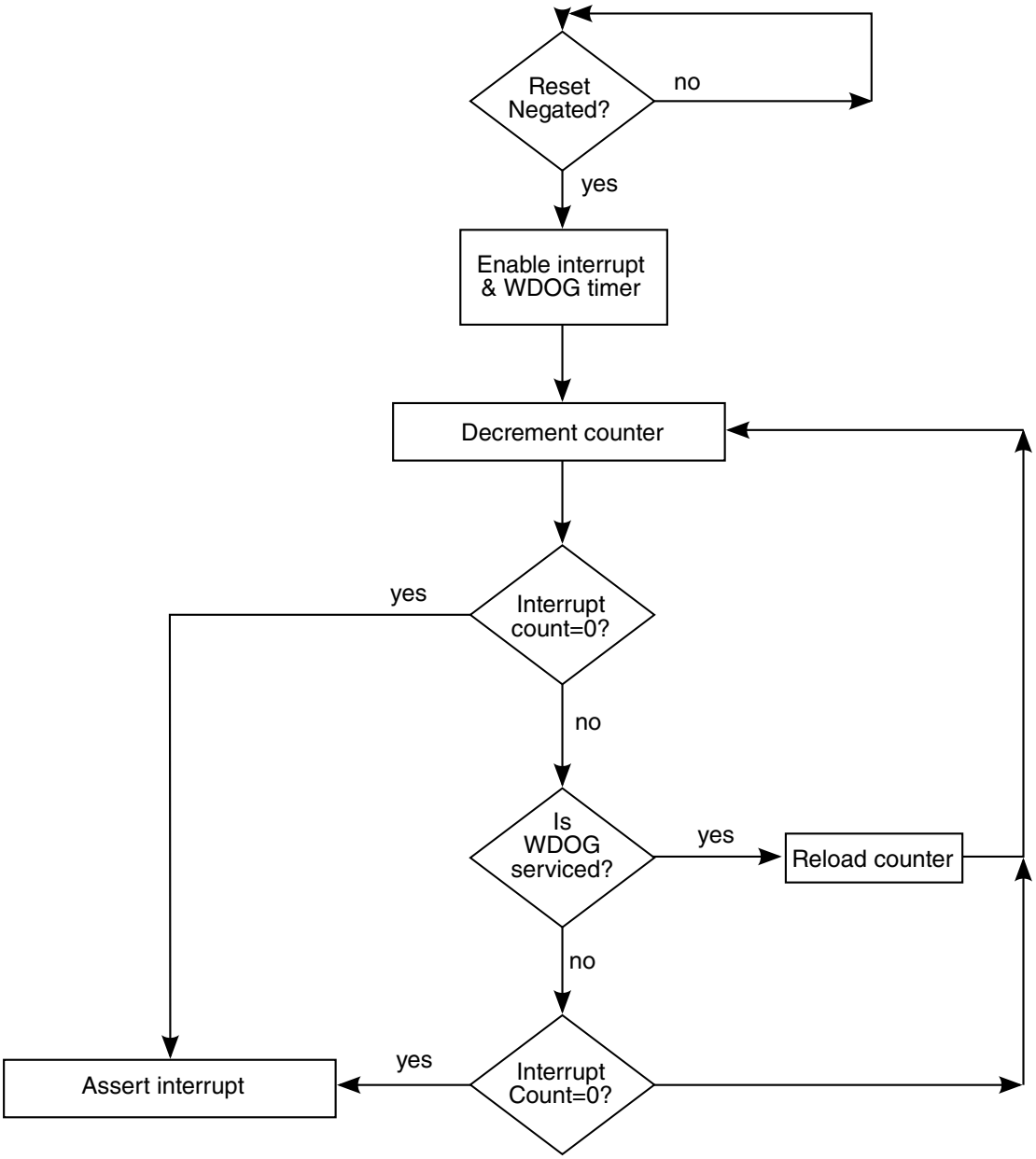


Figure 6-41. Interrupt Generation Flow Diagram

6.6.3 Initialization

The following sequence should be performed for WDOG initialization.

- PDE bit of Watchdog Miscellaneous Control Register (WDOG_WMCR) should be cleared to disable the power down counter.

- WT field of Watchdog Control Register (WDOG_WCR) should be programmed for sufficient timeout value.
- WDOG should be enabled by setting WDE bit of Watchdog Control Register (WDOG_WCR) so that the timeout counter loads the WT field value of Watchdog Control Register (WDOG_WCR) and starts counting.

6.6.4 WDOG Memory Map/Register Definition

The WDOG Memory Map/Register Definition can be found here.

The WDOG has user-accessible, 16-bit registers used to configure, operate, and monitor the state of the Watchdog Timer. Byte operations can be performed on these registers. If a 32-bit access is performed,the WDOG will not generate a peripheral bus error but will behave normally, like a 16-Bit access, making read/write possible. A 32-Bit access should be avoided, as the system may go to an unknown state.

WDOG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3028_0000	Watchdog Control Register (WDOG1_WCR)	16	R/W	0030h	6.6.4.1/ 956
3028_0002	Watchdog Service Register (WDOG1_WSR)	16	R/W	0000h	6.6.4.2/ 958
3028_0004	Watchdog Reset Status Register (WDOG1_WRSR)	16	R	0000h	6.6.4.3/ 958
3028_0006	Watchdog Interrupt Control Register (WDOG1_WICR)	16	R/W	0004h	6.6.4.4/ 959
3028_0008	Watchdog Miscellaneous Control Register (WDOG1_WMCR)	16	R/W	0001h	6.6.4.5/ 960
3029_0000	Watchdog Control Register (WDOG2_WCR)	16	R/W	0030h	6.6.4.1/ 956
3029_0002	Watchdog Service Register (WDOG2_WSR)	16	R/W	0000h	6.6.4.2/ 958
3029_0004	Watchdog Reset Status Register (WDOG2_WRSR)	16	R	0000h	6.6.4.3/ 958
3029_0006	Watchdog Interrupt Control Register (WDOG2_WICR)	16	R/W	0004h	6.6.4.4/ 959
3029_0008	Watchdog Miscellaneous Control Register (WDOG2_WMCR)	16	R/W	0001h	6.6.4.5/ 960
302A_0000	Watchdog Control Register (WDOG3_WCR)	16	R/W	0030h	6.6.4.1/ 956

Table continues on the next page...

WDOG memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302A_0002	Watchdog Service Register (WDOG3_WSR)	16	R/W	0000h	6.6.4.2/ 958
302A_0004	Watchdog Reset Status Register (WDOG3_WRSR)	16	R	0000h	6.6.4.3/ 958
302A_0006	Watchdog Interrupt Control Register (WDOG3_WICR)	16	R/W	0004h	6.6.4.4/ 959
302A_0008	Watchdog Miscellaneous Control Register (WDOG3_WMCR)	16	R/W	0001h	6.6.4.5/ 960

6.6.4.1 Watchdog Control Register (WDOGx_WCR)

The Watchdog Control Register (WDOG_WCR) controls the WDOG operation.

- WDZST, WDBG and WDW are write-once only bits. Once the software does a write access to these bits, they will be locked and cannot be reprogrammed until the next system reset assertion.
- WDE is a write one once only bit. Once software performs a write "1" operation to this bit it cannot be reset/cleared until the next system reset.
- WDT is also a write one once only bit. Once software performs a write "1" operation to this bit it cannot be reset/cleared until the next POR. This bit does not get reset/ cleared due to any system reset.

Address: Base address + 0h offset

Bit	15	14	13	12	11	10	9	8
Read Write	WT							
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read Write	WDW	SRE	WDA	SRS	WDT	WDE	WDBG	WDZST
Reset	0	0	1	1	0	0	0	0

WDOGx_WCR field descriptions

Field	Description
15–8 WT	Watchdog Time-out Field. This 8-bit field contains the time-out value that is loaded into the Watchdog counter after the service routine has been performed or after the Watchdog is enabled. After reset, WT[7:0] must have a value written to it before enabling the Watchdog otherwise count value of zero which is 0.5 seconds is loaded into the counter. **NOTE:** The time-out value can be written at any point of time but it is loaded to the counter at the time when WDOG is enabled or after the service routine has been performed. For more information see Timeout event .

Table continues on the next page...

WDOGx_WCR field descriptions (continued)

Field	Description
	0x00 - 0.5 Seconds (Default). 0x01 - 1.0 Seconds. 0x02 - 1.5 Seconds. 0x03 - 2.0 Seconds. 0xff - 128 Seconds.
7 WDW	Watchdog Disable for Wait. This bit determines the operation of WDOG during Low Power WAIT mode. This is a write once only bit. 0 Continue WDOG timer operation (Default). 1 Suspend WDOG timer operation.
6 SRE	Software Reset Extension. Required to be set to 1 when used in conjunction with the Software Reset Signal (SRS). 0 Reserved 1 This bit must be set to 1.
5 WDA	WDOG_B assertion. Controls the software assertion of the WDOG_B signal. 0 Assert WDOG_B output. 1 No effect on system (Default).
4 SRS	Software Reset Signal. Controls the software assertion of the WDOG-generated reset signal WDOG_RESET_B_DEB. This bit automatically resets to 1 after it has been asserted to 0. For proper operation of this function, the SRE bit in this register must be set to 1. **NOTE:** This bit does not generate the software reset to the block. 0 Assert system reset signal. 1 No effect on the system (Default).
3 WDT	WDOG_B Time-out assertion. Determines if the WDOG_B gets asserted upon a Watchdog Time-out Event. This is a write-one once only bit. **NOTE:** There is no effect on WDOG_RESET_B_DEB (WDOG Reset) upon writing on this bit. WDOG_B gets asserted along with WDOG_RESET_B_DEB if this bit is set. 0 No effect on WDOG_B (Default). 1 Assert WDOG_B upon a Watchdog Time-out event.
2 WDE	Watchdog Enable. Enables or disables the WDOG block. This is a write one once only bit. It is not possible to clear this bit by a software write, once the bit is set. **NOTE:** This bit can be set/reset in debug mode (exception). 0 Disable the Watchdog (Default). 1 Enable the Watchdog.
1 WDBG	Watchdog DEBUG Enable. Determines the operation of the WDOG during DEBUG mode. This bit is write once only. 0 Continue WDOG timer operation (Default). 1 Suspend the watchdog timer.
0 WDZST	Watchdog Low Power. Determines the operation of the WDOG during low-power modes. This bit is write once-only. **NOTE:** The WDOG can continue/suspend the timer operation in the low-power modes (STOP and DOZE mode).

Table continues on the next page...

WDOGx_WCR field descriptions (continued)

Field	Description
	0 Continue timer operation (Default). 1 Suspend the watchdog timer.

6.6.4.2 Watchdog Service Register (WDOGx_WSR)

When enabled, the WDOG requires that a service sequence be written to the Watchdog Service Register (WSR) to prevent the timeout condition.

NOTE

Executing the service sequence will reload the WDOG timeout counter.

Address: Base address + 2h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	WSR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

WDOGx_WSR field descriptions

Field	Description
WSR	Watchdog Service Register. This 16-bit field contains the Watchdog service sequence. Both writes must occur in the order listed prior to the time-out, but any number of instructions can be executed between the two writes. The service sequence must be performed as follows: 0x5555 Write to the Watchdog Service Register (WDOG_WSR). 0xAAAA Write to the Watchdog Service Register (WDOG_WSR).

6.6.4.3 Watchdog Reset Status Register (WDOGx_WRSR)

The WRSR is a read-only register that records the source of the output reset assertion. It is not cleared by a hard reset. Therefore, only one bit in the WRSR will always be asserted high. The register will always indicate the source of the last reset generated due to WDOG. Read access to this register is with one wait state. Any write performed on this register will generate a Peripheral Bus Error .

A reset can be generated by the following sources, as listed in priority from highest to lowest:

- Watchdog Time-out
- Software Reset

Address: Base address + 4h offset

Bit	15	14	13	12	11	10	9	8
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	0			POR	0		TOUT	SFTW
Write								
Reset	0	0	0	0	0	0	0	0

WDOGx_WRSR field descriptions

Field	Description
15–5 Reserved	This read-only field is reserved and always has the value 0.
4 POR	Power On Reset. Indicates whether the reset is the result of a power on reset. 0 Reset is not the result of a power on reset. 1 Reset is the result of a power on reset.
3–2 Reserved	This read-only field is reserved and always has the value 0.
1 TOUT	Timeout. Indicates whether the reset is the result of a WDOG timeout. 0 Reset is not the result of a WDOG timeout. 1 Reset is the result of a WDOG timeout.
0 SFTW	Software Reset. Indicates whether the reset is the result of a WDOG software reset by asserting SRS bit 0 Reset is not the result of a software reset. 1 Reset is the result of a software reset.

6.6.4.4 Watchdog Interrupt Control Register (WDOGx_WICR)

The WDOG_WICR controls the WDOG interrupt generation.

Address: Base address + 6h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	WIE	WTIS	0						WICT							
Write		w1c														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

WDOGx_WICR field descriptions

Field	Description
15 WIE	Watchdog Timer Interrupt enable bit. Reset value is 0.

Table continues on the next page...

WDOGx_WICR field descriptions (continued)

Field	Description
	NOTE: This bit is a write once only bit. Once the software does a write access to this bit, it will get locked and cannot be reprogrammed until the next system reset assertion 0 Disable Interrupt (Default). 1 Enable Interrupt.
14 WTIS	Watchdog Timer Interrupt Status bit will reflect the timer interrupt status, whether interrupt has occurred or not. Once the interrupt has been triggered software must clear this bit by writing 1 to it. 0 No interrupt has occurred (Default). 1 Interrupt has occurred
13–8 Reserved	This read-only field is reserved and always has the value 0.
WICT	Watchdog Interrupt Count Time-out (WICT) field determines, how long before the counter time-out must the interrupt occur. The reset value is 0x04 implies interrupt will occur 2 seconds before time-out. The maximum value that can be programmed to WICT field is 127.5 seconds with a resolution of 0.5 seconds. **NOTE:** This field is write once only. Once the software does a write access to this field, it will get locked and cannot be reprogrammed until the next system reset assertion. 0x00 WICT[7:0] = Time duration between interrupt and time-out is 0 seconds. 0x01 WICT[7:0] = Time duration between interrupt and time-out is 0.5 seconds. 0x04 WICT[7:0] = Time duration between interrupt and time-out is 2 seconds (Default). 0xff WICT[7:0] = Time duration between interrupt and time-out is 127.5 seconds.

6.6.4.5 Watchdog Miscellaneous Control Register (WDOGx_WMCR)

WDOG_WMCR Controls the Power Down counter operation.

Address: Base address + 8h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0															PDE
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

WDOGx_WMCR field descriptions

Field	Description
15–1 Reserved	This read-only field is reserved and always has the value 0.
0 PDE	Power Down Enable bit. Reset value of this bit is 1, which means the power down counter inside the WDOG is enabled after reset. The software must write 0 to this bit to disable the counter within 16 seconds of reset de-assertion. Once disabled this counter cannot be enabled again. See Power-down counter event for operation of this counter. **NOTE:** This bit is write-one once only bit. Once software sets this bit it cannot be reset until the next system reset.

Table continues on the next page...

WDOGx_WMCR field descriptions (continued)

Field	Description
	0 Power Down Counter of WDOG is disabled. 1 Power Down Counter of WDOG is enabled (Default).

Chapter 7
Interrupts and DMA

7.1 Interrupts and DMA Events

7.1.1 Overview

This chapter provides the interrupt assignments of the ARM domain in A53 Interrupts, CM4 interrupts, and the DMA events in SDMA event mapping

7.1.2 A53 Interrupts

The Global Interrupt Controller (GIC) collects up to 128 interrupt requests from all the chip sources and provides an interface to the Cortex A53 CPU.

Each interrupt can be configured as a normal or a secure interrupt. Software force registers and software priority masking are also supported. The following table describes the A53 interrupt sources.

Table 7-1. ARM Domain Interrupt Summary

IRQ	Module	Logic	Interrupt Description
0	boot	-	Used to notify cores on exception condition while boot
1	dap	-	DAP Interrupt
2	sdma1	-	AND of all 48 SDMA1 interrupts (events) from all the channels
3	gpu3d	-	GPU3D Interrupt
4	snvs_lp_wrapper	OR	ON-OFF button press shorter than 5 secs (pulse event)
4	snvs_hp_wrapper	OR	ON-OFF button press shorter than 5 secs (pulse event)
5	lcdif	-	LCDIF Interrupt
6	spdif1	OR	SPDIF Rx interrupt

Table continues on the next page...

Table 7-1. ARM Domain Interrupt Summary (continued)

IRQ	Module	Logic	Interrupt Description
6	spdif1	OR	SPDIF Tx interrupt
7	vpu	-	VPU G1 Decoder Interrupt
8	vpu	-	VPU G2 Decoder Interrupt
9	qos	-	QOS Interrupt
10	wdog3	-	Watchdog Timer reset
11	hs	-	HS Interrupt Request
12	apbhdma	OR4	GPMI operation channel 0 description complete interrupt
12	apbhdma	OR4	GPMI operation channel 1 description complete interrupt
12	apbhdma	OR4	GPMI operation channel 2 description complete interrupt
12	apbhdma	OR4	GPMI operation channel 3 description complete interrupt
13	-	OR	-
13	-	OR	-
14	rawnand	-	BCH operation complete interrupt
15	rawnand	-	GPMI operation TIMEOUT ERROR interrupt
16	csi1	-	CSI Interrupt
17	mipi_csi1	-	MIPI CSI Interrupt
18	mipi_dsi	-	MIPI DSI Interrupt
19	snvs_hp_wrapper	-	SRTC Consolidated Interrupt. Non TZ.
20	snvs_hp_wrapper	-	SRTC Security Interrupt. TZ.
21	csu	-	CSU Interrupt Request. Indicates to the processor that one or more alarm inputs were asserted
22	usdhc1	-	uSDHC1 Enhanced SDHC Interrupt Request
23	usdhc2	-	uSDHC2 Enhanced SDHC Interrupt Request
24	usdhc3	-	uSDHC3 Enhanced SDHC Interrupt Request
25	gpu2d	-	GPU3D interrupt
26	uart1	-	UART-1 ORed interrupt
27	uart2	-	UART-2 ORed interrupt
28	uart3	-	UART-3 ORed interrupt
29	uart4	-	UART-4 ORed interrupt
30	vpu	-	VPU H1 Encoder Interrupt
31	ecspi1	-	eCSPI1 interrupt request line to the core.
32	ecspi2	-	eCSPI2 interrupt request line to the core.
33	ecspi3	-	eCSPI3 interrupt request line to the core.
34	sdma3	-	AND of all 48 SDMA3 interrupts (events) from all the channels
35	i2c1	-	I2C-1 Interrupt

Table continues on the next page...

Table 7-1. ARM Domain Interrupt Summary (continued)

IRQ	Module	Logic	Interrupt Description
36	i2c2	-	I2C-2 Interrupt
37	i2c3	-	I2C-3 Interrupt
38	i2c4	-	I2C-4 Interrupt
39	rdc	-	RDC interrupt
40	usb1	-	USB-1 Interrupt
41	usb2	-	USB-2 Interrupt
42	-	-	-
43	-	-	-
44	micfil	-	Digital Microphone interface voice activity detector event interrupt
45	micfil	-	Digital Microphone interface voice activity detector error interrupt
46	gpt6	-	OR of GPT Rollover interrupt line, Input Capture 1 & 2 lines, Output Compare 1,2 &3 Interrupt lines
47	sctr	-	System Counter Interrupt [0]
48	sctr	-	System Counter Interrupt [1]
49	anamix	OR	TempSensor (Temperature alarm).
49	anamix	OR	TempSensor (Temperature critical alarm).
49	-	-	-
50	sai3	OR4	SAI3 Receive Interrupt
50	sai3	OR4	SAI3 Receive Async Interrupt
50	sai3	OR4	SAI3 Transmit Interrupt
50	sai3	OR4	SAI3 Transmit Async Interrupt
51	gpt5	-	OR of GPT Rollover interrupt line, Input Capture 1 & 2 lines, Output Compare 1,2 &3 Interrupt lines
52	gpt4	-	OR of GPT Rollover interrupt line, Input Capture 1 & 2 lines, Output Compare 1,2 &3 Interrupt lines
53	gpt3	-	OR of GPT Rollover interrupt line, Input Capture 1 & 2 lines, Output Compare 1,2 &3 Interrupt lines
54	gpt2	-	OR of GPT Rollover interrupt line, Input Capture 1 & 2 lines, Output Compare 1,2 &3 Interrupt lines
55	gpt1	-	OR of GPT Rollover interrupt line, Input Capture 1 & 2 lines, Output Compare 1,2 &3 Interrupt lines
56	gpio1	-	Active HIGH Interrupt from INT7 from GPIO
57	gpio1	-	Active HIGH Interrupt from INT6 from GPIO
58	gpio1	-	Active HIGH Interrupt from INT5 from GPIO
59	gpio1	-	Active HIGH Interrupt from INT4 from GPIO

Table continues on the next page...

Table 7-1. ARM Domain Interrupt Summary (continued)

IRQ	Module	Logic	Interrupt Description
60	gpio1	-	Active HIGH Interrupt from INT3 from GPIO
61	gpio1	-	Active HIGH Interrupt from INT2 from GPIO
62	gpio1	-	Active HIGH Interrupt from INT1 from GPIO
63	gpio1	-	Active HIGH Interrupt from INT0 from GPIO
64	gpio1	-	Combined interrupt indication for GPIO1 signal 0 throughout 15
65	gpio1	-	Combined interrupt indication for GPIO1 signal 16 throughout 31
66	gpio2	-	Combined interrupt indication for GPIO2 signal 0 throughout 15
67	gpio2	-	Combined interrupt indication for GPIO2 signal 16 throughout 31
68	gpio3	-	Combined interrupt indication for GPIO3 signal 0 throughout 15
69	gpio3	-	Combined interrupt indication for GPIO3 signal 16 throughout 31
70	gpio4	-	Combined interrupt indication for GPIO4 signal 0 throughout 15
71	gpio4	-	Combined interrupt indication for GPIO4 signal 16 throughout 31
72	gpio5	-	Combined interrupt indication for GPIO5 signal 0 throughout 15
73	gpio5	-	Combined interrupt indication for GPIO5 signal 16 throughout 31
74	-	-	-
75	-	-	-
76	-	-	-
77	-	-	-
78	wdog1	-	Watchdog 1 Timer reset
79	wdog2	-	Watchdog 2 Timer reset
80	-	-	-
81	pwm1	-	Cumulative interrupt line. OR of Rollover Interrupt line, Compare Interrupt line and FIFO Waterlevel crossing interrupt line
82	pwm2	-	Cumulative interrupt line. OR of Rollover Interrupt line, Compare Interrupt line and FIFO Waterlevel crossing interrupt line
83	pwm3	-	Cumulative interrupt line. OR of Rollover Interrupt line, Compare Interrupt line and FIFO Waterlevel crossing interrupt line
84	pwm4	-	Cumulative interrupt line. OR of Rollover Interrupt line, Compare Interrupt line and FIFO Waterlevel crossing interrupt line
85	ccmsrcgpcmix	-	CCM, Interrupt Request 1
86	ccmsrcgpcmix	-	CCM, Interrupt Request 2

Table continues on the next page...

Table 7-1. ARM Domain Interrupt Summary (continued)

IRQ	Module	Logic	Interrupt Description
87	ccmsrcgpcmix	-	GPC, Interrupt Request 1
88	mu	-	Interrupt to A53
89	ccmsrcgpcmix	-	SRC interrupt request
90	sai5	OR4	SAI5 Receive Interrupt
90	sai5	OR4	SAI5 Receive Async Interrupt
90	sai5	OR4	SAI5 Transmit Interrupt
90	sai5	OR4	SAI5 Transmit Async Interrupt
90	sai6	OR4	SAI6 Receive Interrupt
90	sai6	OR4	SAI6 Receive Async Interrupt
90	sai6	OR4	SAI6 Transmit Interrupt
90	sai6	OR4	SAI6 Transmit Async Interrupt
91	rtic	-	RTIC Interrupt
92	cpu	OR	Performance Unit Interrupts from Quad-A53 platform (interrnally: PMUIRQ[0])
92	cpu	OR	Performance Unit Interrupts from Quad-A53 platform (interrnally: PMUIRQ[1])
92	cpu	OR	Performance Unit Interrupts from Quad-A53 platform (interrnally: PMUIRQ[2])
92	cpu	OR	Performance Unit Interrupts from Quad-A53 platform (interrnally: PMUIRQ[3])
93	cpu	OR	CTI trigger outputs from Quad-A53 platform (internal: nCTIIRQ[0])
93	cpu	OR	CTI trigger outputs from Quad-A53 platform (internal: nCTIIRQ[1])
93	cpu	OR	CTI trigger outputs from Quad-A53 platform (internal: nCTIIRQ[2])
93	cpu	OR	CTI trigger outputs from Quad-A53 platform (internal: nCTIIRQ[3])
94	ccmsrcgpcmix	-	Combined CPU wdog interrupts (4x) out of SRC.
95	sai1	OR4	SAI1 Receive Interrupt
95	sai1	OR4	SAI1 Receive Async Interrupt
95	sai1	OR4	SAI1 Transmit Interrupt
95	sai1	OR4	SAI1 Transmit Async Interrupt
96	sai2	OR4	SAI2 Receive Interrupt
96	sai2	OR4	SAI2 Receive Async Interrupt
96	sai2	OR4	SAI2 Transmit Interrupt
96	sai2	OR4	SAI2 Transmit Async Interrupt
97	mu	-	Interrupt to M4
98	ddr	-	Interrupt for performance monitor in DRAM controller
99	ddr	-	-
100	-	OR4	-

Table continues on the next page...

Table 7-1. ARM Domain Interrupt Summary (continued)

IRQ	Module	Logic	Interrupt Description
100	-	OR4	-
100	-	OR4	-
100	-	OR4	-
101	cpu	-	Error indicator for AXI transaction with a write response error condition.
102	cpu	-	Error indicator for L2 RAM double-bit ECC error.
103	sdma2	-	AND of all 48 SDMA2 interrupts (events) from all the channels
104	sjc	-	Interrupt triggered by SJC register
105	caam_wrapper	-	CAAM interrupt queue for JQ
106	caam_wrapper	-	CAAM interrupt queue for JQ
107	qspi	-	QSPI Interrupt
108	tzasc	-	TZASC (PL380) interrupt
109	micfil	-	Digital Microphone interface interrupt
110	micfil	-	Digital Microphone interface error interrupt
111	-	-	-
112	perfmon1	-	General interrupt
113	perfmon2	-	General interrupt
114	caam_wrapper	-	CAAM interrupt queue for JQ
115	caam_wrapper	-	Recoverable error interrupt
116	hs	-	HS Interrupt Request
117	-	-	-
118	enet1	OR4	MAC 0 Receive Buffer Done
118	enet1	OR4	MAC 0 Receive Frame Done
118	enet1	OR4	MAC 0 Transmit Buffer Done
118	enet1	OR4	MAC 0 Transmit Frame Done
119	enet1	OR4	MAC 0 Receive Buffer Done
119	enet1	OR4	MAC 0 Receive Frame Done
119	enet1	OR4	MAC 0 Transmit Buffer Done
119	enet1	OR4	MAC 0 Transmit Frame Done
120	enet1	OR	MAC 0 Periodic Timer Overflow
120	enet1	OR	MAC 0 Time Stamp Available
120	enet1	OR	MAC 0 Payload Receive Error
120	enet1	OR	MAC 0 Transmit FIFO Underrun
120	enet1	OR	MAC 0 Collision Retry Limit
120	enet1	OR	MAC 0 Late Collision
120	enet1	OR	MAC 0 Ethernet Bus Error
120	enet1	OR	MAC 0 MII Data Transfer Done
120	enet1	OR	MAC 0 Receive Buffer Done
120	enet1	OR	MAC 0 Receive Frame Done

Table continues on the next page...

Table 7-1. ARM Domain Interrupt Summary (continued)

IRQ	Module	Logic	Interrupt Description
120	enet1	OR	MAC 0 Transmit Buffer Done
120	enet1	OR	MAC 0 Transmit Frame Done
120	enet1	OR	MAC 0 Graceful Stop
120	enet1	OR	MAC 0 Babbling Transmit Error
120	enet1	OR	MAC 0 Babbling Receive Error
120	enet1	OR	MAC 0 Receive Flush Frame0
120	enet1	OR	MAC 0 Receive Flush Frame1
120	enet1	OR	MAC 0 Receive Flush Frame2
120	enet1	OR	MAC 0 Wakeup Request (sync)
120	enet1	OR	MAC 0 Babbling Receive Error
120	enet1	OR	MAC 0 Wakeup Request (sync)
121	enet1	-	MAC 0 1588 Timer Interrupt – synchronous
122	pcie_ctrl1	-	Coming from GLUE logic, of set/reset FF, driven by PCIE signals.
123	pcie_ctrl1	-	Coming from GLUE logic, of set/reset FF, driven by PCIE signals.
124	pcie_ctrl1	-	Coming from GLUE logic, of set/reset FF, driven by PCIE signals.
125	pcie_ctrl1	-	Coming from GLUE logic, of set/reset FF, driven by PCIE signals.
126	-	-	-
127	pcie_ctrl1	-	Channels [63:32] interrupts requests

7.1.3 CM4 interrupts

The Nested Vectored Interrupt Controller (NVIC) collects up to 128 interrupt requests from all chip sources and provides an interface to the Cortex M4 Core.

The following table describes the M4 interrupt sources.

Table 7-2. CM4 Interrupt Summary

IRQ	Module	Logic	Interrupt Description
0	GPR_IRQ_IRQn	-	Used to notify cores on exception condition while boot.
1	dap	-	DAP Interrupt
2	sdma1	-	AND of all 48 SDMA1 interrupts (events) from all the channels
3	gpu3d	-	GPU3D Interrupt
4	snvs_lp_wrapper	OR	ON-OFF button press shorter than 5 secs (pulse event)

Table continues on the next page...

Table 7-2. CM4 Interrupt Summary (continued)

IRQ	Module	Logic	Interrupt Description
4	snvs_hp_wrapper	OR	ON-OFF button press shorter than 5 secs (pulse event)
5	lcdif	-	LCDIF Interrupt
6	spdif1	OR	SPDIF Rx interrupt
6	spdif1	OR	SPDIF Tx interrupt
7	vpu	-	VPU G1 Decoder Interrupt
8	vpu	-	VPU G2 Decoder Interrupt
9	qos	-	QOS Interrupt
10	wdog3	-	Watchdog Timer reset
11	hs	-	HS Interrupt Request
12	apbhdma	OR4	GPMI operation channel 0 description complete interrupt
12	apbhdma	OR4	GPMI operation channel 1 description complete interrupt
12	apbhdma	OR4	GPMI operation channel 2 description complete interrupt
12	apbhdma	OR4	GPMI operation channel 3 description complete interrupt
13	-	OR	-
13	-	OR	-
14	rawnand	-	BCH operation complete interrupt
15	rawnand	-	GPMI operation TIMEOUT ERROR interrupt
16	csi1	-	CSI Interrupt
17	mipi_csi1	-	MIPI CSI Interrupt
18	mipi_dsi	-	MIPI DSI Interrupt
19	snvs_hp_wrapper	-	SRTC Consolidated Interrupt. Non TZ.
20	snvs_hp_wrapper	-	SRTC Security Interrupt. TZ.
21	csu	-	CSU Interrupt Request. Indicates to the processor that one or more alarm inputs were asserted
22	usdhc1	-	uSDHC1 Enhanced SDHC Interrupt Request
23	usdhc2	-	uSDHC2 Enhanced SDHC Interrupt Request
24	usdhc3	-	uSDHC3 Enhanced SDHC Interrupt Request
25	gpu2d	-	GPU3D interrupt
26	uart1	-	UART-1 ORed interrupt
27	uart2	-	UART-2 ORed interrupt
28	uart3	-	UART-3 ORed interrupt
29	uart4	-	UART-4 ORed interrupt
30	vpu	-	VPU H1 Encoder Interrupt
31	ecspi1	-	eCSPI1 interrupt request line to the core.
32	ecspi2	-	eCSPI2 interrupt request line to the core.

Table continues on the next page...

Table 7-2. CM4 Interrupt Summary (continued)

IRQ	Module	Logic	Interrupt Description
33	ecspi3	-	eCSPI3 interrupt request line to the core.
34	sdma3	-	AND of all 48 SDMA3 interrupts (events) from all the channels
35	i2c1	-	I2C-1 Interrupt
36	i2c2	-	I2C-2 Interrupt
37	i2c3	-	I2C-3 Interrupt
38	i2c4	-	I2C-4 Interrupt
39	rdc	-	RDC interrupt
40	usb1	-	USB-1 Interrupt
41	usb2	-	USB-2 Interrupt
42	-	-	-
43	-	-	-
44	micfil	-	Digital Microphone interface voice activity detector event interrupt
45	micfil	-	Digital Microphone interface voice activity detector error interrupt
46	gpt6	-	OR of GPT Rollover interrupt line, Input Capture 1 & 2 lines, Output Compare 1,2 &3 Interrupt lines
47	sctr	-	System Counter Interrupt [0]
48	sctr	-	System Counter Interrupt [1]
49	anamix	OR	TempSensor (Temperature alarm).
49	anamix	OR	TempSensor (Temperature critical alarm).
49	-	-	-
50	sai3	OR4	SAI3 Receive Interrupt
50	sai3	OR4	SAI3 Receive Async Interrupt
50	sai3	OR4	SAI3 Transmit Interrupt
50	sai3	OR4	SAI3 Transmit Async Interrupt
51	gpt5	-	OR of GPT Rollover interrupt line, Input Capture 1 & 2 lines, Output Compare 1,2 &3 Interrupt lines
52	gpt4	-	OR of GPT Rollover interrupt line, Input Capture 1 & 2 lines, Output Compare 1,2 &3 Interrupt lines
53	gpt3	-	OR of GPT Rollover interrupt line, Input Capture 1 & 2 lines, Output Compare 1,2 &3 Interrupt lines
54	gpt2	-	OR of GPT Rollover interrupt line, Input Capture 1 & 2 lines, Output Compare 1,2 &3 Interrupt lines
55	gpt1	-	OR of GPT Rollover interrupt line, Input Capture 1 & 2 lines, Output Compare 1,2 &3 Interrupt lines
56	gpio1	-	Active HIGH Interrupt from INT7 from GPIO

Table continues on the next page...

Table 7-2. CM4 Interrupt Summary (continued)

IRQ	Module	Logic	Interrupt Description
57	gpio1	-	Active HIGH Interrupt from INT6 from GPIO
58	gpio1	-	Active HIGH Interrupt from INT5 from GPIO
59	gpio1	-	Active HIGH Interrupt from INT4 from GPIO
60	gpio1	-	Active HIGH Interrupt from INT3 from GPIO
61	gpio1	-	Active HIGH Interrupt from INT2 from GPIO
62	gpio1	-	Active HIGH Interrupt from INT1 from GPIO
63	gpio1	-	Active HIGH Interrupt from INT0 from GPIO
64	gpio1	-	Combined interrupt indication for GPIO1 signal 0 throughout 15
65	gpio1	-	Combined interrupt indication for GPIO1 signal 16 throughout 31
66	gpio2	-	Combined interrupt indication for GPIO2 signal 0 throughout 15
67	gpio2	-	Combined interrupt indication for GPIO2 signal 16 throughout 31
68	gpio3	-	Combined interrupt indication for GPIO3 signal 0 throughout 15
69	gpio3	-	Combined interrupt indication for GPIO3 signal 16 throughout 31
70	gpio4	-	Combined interrupt indication for GPIO4 signal 0 throughout 15
71	gpio4	-	Combined interrupt indication for GPIO4 signal 16 throughout 31
72	gpio5	-	Combined interrupt indication for GPIO5 signal 0 throughout 15
73	gpio5	-	Combined interrupt indication for GPIO5 signal 16 throughout 31
74	-	-	-
75	-	-	-
76	-	-	-
77	-	-	-
78	wdog1	-	Watchdog 1 Timer reset
79	wdog2	-	Watchdog 2 Timer reset
80	-	-	-
81	pwm1	-	Cumulative interrupt line. OR of Rollover Interrupt line, Compare Interrupt line and FIFO Waterlevel crossing interrupt line
82	pwm2	-	Cumulative interrupt line. OR of Rollover Interrupt line, Compare Interrupt line and FIFO Waterlevel crossing interrupt line
83	pwm3	-	Cumulative interrupt line. OR of Rollover Interrupt line, Compare Interrupt line and FIFO Waterlevel crossing interrupt line

Table continues on the next page...

Table 7-2. CM4 Interrupt Summary (continued)

IRQ	Module	Logic	Interrupt Description
84	pwm4	-	Cumulative interrupt line. OR of Rollover Interrupt line, Compare Interrupt line and FIFO Waterlevel crossing interrupt line
85	ccmsrcgpcmix	-	CCM, Interrupt Request 1
86	ccmsrcgpcmix	-	CCM, Interrupt Request 2
87	ccmsrcgpcmix	-	GPC, Interrupt Request 1
88	mu	-	Interrupt to A53
89	ccmsrcgpcmix	-	SRC interrupt request
90	sai5	OR4	SAI5 Receive Interrupt
90	sai5	OR4	SAI5 Receive Async Interrupt
90	sai5	OR4	SAI5 Transmit Interrupt
90	sai5	OR4	SAI5 Transmit Async Interrupt
90	sai6	OR4	SAI6 Receive Interrupt
90	sai6	OR4	SAI6 Receive Async Interrupt
90	sai6	OR4	SAI6 Transmit Interrupt
90	sai6	OR4	SAI6 Transmit Async Interrupt
91	rtic	-	RTIC Interrupt
92	cpu	OR	Performance Unit Interrupts from Quad-A53 platform (interrnally: PMUIRQ[0])
92	cpu	OR	Performance Unit Interrupts from Quad-A53 platform (interrnally: PMUIRQ[1])
92	cpu	OR	Performance Unit Interrupts from Quad-A53 platform (interrnally: PMUIRQ[2])
92	cpu	OR	Performance Unit Interrupts from Quad-A53 platform (interrnally: PMUIRQ[3])
93	cpu	OR	CTI trigger outputs from Quad-A53 platform (internal: nCTIIRQ[0])
93	cpu	OR	CTI trigger outputs from Quad-A53 platform (internal: nCTIIRQ[1])
93	cpu	OR	CTI trigger outputs from Quad-A53 platform (internal: nCTIIRQ[2])
93	cpu	OR	CTI trigger outputs from Quad-A53 platform (internal: nCTIIRQ[3])
94	ccmsrcgpcmix	-	Combined CPU wdog interrupts (4x) out of SRC.
95	sai1	OR4	SAI1 Receive Interrupt
95	sai1	OR4	SAI1 Receive Async Interrupt
95	sai1	OR4	SAI1 Transmit Interrupt
95	sai1	OR4	SAI1 Transmit Async Interrupt
96	sai2	OR4	SAI2 Receive Interrupt
96	sai2	OR4	SAI2 Receive Async Interrupt
96	sai2	OR4	SAI2 Transmit Interrupt
96	sai2	OR4	SAI2 Transmit Async Interrupt

Table continues on the next page...

Table 7-2. CM4 Interrupt Summary (continued)

IRQ	Module	Logic	Interrupt Description
97	mu	-	Interrupt to M4
98	ddr	-	Interrupt for performance monitor in DRAM controller
99	ddr	-	-
100	-	OR4	-
100	-	OR4	-
100	-	OR4	-
100	-	OR4	-
101	cpu	-	Error indicator for AXI transaction with a write response error condition.
102	cpu	-	Error indicator for L2 RAM double-bit ECC error.
103	sdma2	-	AND of all 48 SDMA2 interrupts (events) from all the channels
104	sjc	-	Interrupt triggered by SJC register
105	caam_wrapper	-	CAAM interrupt queue for JQ
106	caam_wrapper	-	CAAM interrupt queue for JQ
107	qspi	-	QSPI Interrupt
108	tzasc	-	TZASC (PL380) interrupt
109	micfil	-	Digital Microphone interface interrupt
110	micfil	-	Digital Microphone interface error interrupt
111	-	-	-
112	perfmon1	-	General interrupt
113	perfmon2	-	General interrupt
114	caam_wrapper	-	CAAM interrupt queue for JQ
115	caam_wrapper	-	Recoverable error interrupt
116	hs	-	HS Interrupt Request
117	-	-	-
118	enet1	OR4	MAC 0 Receive Buffer Done
118	enet1	OR4	MAC 0 Receive Frame Done
118	enet1	OR4	MAC 0 Transmit Buffer Done
118	enet1	OR4	MAC 0 Transmit Frame Done
119	enet1	OR4	MAC 0 Receive Buffer Done
119	enet1	OR4	MAC 0 Receive Frame Done
119	enet1	OR4	MAC 0 Transmit Buffer Done
119	enet1	OR4	MAC 0 Transmit Frame Done
120	enet1	OR	MAC 0 Periodic Timer Overflow
120	enet1	OR	MAC 0 Time Stamp Available
120	enet1	OR	MAC 0 Payload Receive Error
120	enet1	OR	MAC 0 Transmit FIFO Underrun
120	enet1	OR	MAC 0 Collision Retry Limit

Table continues on the next page...

Table 7-2. CM4 Interrupt Summary (continued)

IRQ	Module	Logic	Interrupt Description
120	enet1	OR	MAC 0 Late Collision
120	enet1	OR	MAC 0 Ethernet Bus Error
120	enet1	OR	MAC 0 MII Data Transfer Done
120	enet1	OR	MAC 0 Receive Buffer Done
120	enet1	OR	MAC 0 Receive Frame Done
120	enet1	OR	MAC 0 Transmit Buffer Done
120	enet1	OR	MAC 0 Transmit Frame Done
120	enet1	OR	MAC 0 Graceful Stop
120	enet1	OR	MAC 0 Babbling Transmit Error
120	enet1	OR	MAC 0 Babbling Receive Error
120	enet1	OR	MAC 0 Receive Flush Frame0
120	enet1	OR	MAC 0 Receive Flush Frame1
120	enet1	OR	MAC 0 Receive Flush Frame2
120	enet1	OR	MAC 0 Wakeup Request (sync)
120	enet1	OR	MAC 0 Babbling Receive Error
120	enet1	OR	MAC 0 Wakeup Request (sync)
121	enet1	-	MAC 0 1588 Timer Interrupt – synchronous
122	pcie_ctrl1	-	Coming from GLUE logic, of set/reset FF, driven by PCIE signals.
123	pcie_ctrl1	-	Coming from GLUE logic, of set/reset FF, driven by PCIE signals.
124	pcie_ctrl1	-	Coming from GLUE logic, of set/reset FF, driven by PCIE signals.
125	pcie_ctrl1	-	Coming from GLUE logic, of set/reset FF, driven by PCIE signals.
126	-	-	-
127	pcie_ctrl1	-	Channels [63:32] interrupts requests

7.1.4 SDMA event mapping

The following table shows the DMA request signals for peripherals in the chip.

Table 7-3. SDMA1 event mapping

SDMA	Module	Description
0	ecspi1	eCSPI1 Rx request
1	ecspi1	eCSPI1 Tx request
2	ecspi2	eCSPI2 Rx request
3	ecspi2	eCSPI2 Tx request

Table continues on the next page...

Table 7-3. SDMA1 event mapping (continued)

SDMA	Module	Description
4	ecspi3	eCSPI3 Rx request
5	ecspi3	eCSPI3 Tx request
6	-	Reserved
7	-	Reserved
8		SPDIF1 Rx DMA request
9		SPDIF1 Tx DMA request
10		SAI-2 receive DMA request
11		SAI-2 transmit DMA request
12		SAI-3 receive DMA request
13		SAI-3 transmit DMA request
14	iomux	external DMA from pad through IOMUX #1
15	iomux	external DMA from pad through IOMUX #2
16		SPDIF2 Rx DMA request
17		SPDIF2 Tx DMA request
18	i2c1	I2C1 DMA event
18	-	Reserved
19	i2c2	I2C2 DMA event
19	-	Reserved
20	i2c3	I2C3 DMA event
20	-	Reserved
21	i2c4	I2C4 DMA event
21	-	Reserved
22	uart1	Rx FIFO
23	uart1	Tx FIFO
24	uart2	Rx FIFO
25	uart2	Tx FIFO
26	uart3	Rx FIFO
27	uart3	Tx FIFO
28	uart4	Rx FIFO
29	uart4	Tx FIFO
30	-	Reserved
31	-	Reserved
32	-	Reserved
33	-	Reserved
34	-	Reserved
35	-	Reserved
36	qspi1	QSPI DMA TX request
37	qspi1	QSPI DMA RX request
38	gpt1	GPT1 counter event

Table continues on the next page...

Table 7-3. SDMA1 event mapping (continued)

SDMA	Module	Description
39	gpt2	GPT2 counter event
40	gpt3	GPT3 counter event
41		PDM Digital Microphone Interface DMA request
42	-	Reserved
43	-	Reserved
44	enet1	ENET1 1588 Event 2
45	enet1	ENET1 1588 Event 0
46	enet1	ENET1 1588 Event 3
47	enet1	ENET1 1588 Event 1

Table 7-4. SDMA2/SDMA3 event mapping

SDMA	Module	Description
0	sai1	SAI-1 receive DMA request
1	sai1	SAI-1 transmit DMA request
2	sai2	SAI-2 receive DMA request
3	sai2	SAI-2 transmit DMA request
4	sai3	SAI-3 receive DMA request
5	sai3	SAI-3 transmit DMA request
6	-	Reserved
7	-	Reserved
8	sai5	SAI-5 receive DMA request
9	sai5	SAI-5 transmit DMA request
10	sai6	SAI-6 receive DMA request
11	sai6	SAI-6 transmit DMA request
12	-	Reserved
13	-	Reserved
14	iomux	external DMA from pad through IOMUX #1
15	iomux	external DMA from pad through IOMUX #2
16	-	Reserved
17	-	Reserved
18	-	Reserved
18	-	Reserved
19	-	Reserved
19	-	Reserved
20	-	Reserved
20	-	Reserved
21	-	Reserved
21	-	Reserved

Table continues on the next page...

Table 7-4. SDMA2/SDMA3 event mapping (continued)

SDMA	Module	Description
22	-	Reserved
23	-	Reserved
24	micfil	PDM Digital Microphone Interface DMA request
25	-	Reserved
26	-	Reserved
27	-	Reserved
28	spdif1	SPDIF1 Rx DMA request
29	spdif1	SPDIF1 Tx DMA request
30	-	Reserved
31	-	Reserved
32	-	Reserved
33	-	Reserved
34	-	Reserved
35	-	Reserved
36	-	Reserved
37	-	Reserved
38	gpt4	GPT4 counter event
39	gpt5	GPT5 counter event
40	gpt6	GPT6 counter event
40	-	Reserved
41	-	Reserved
41	-	Reserved
42	-	Reserved
43	-	Reserved
44	-	Reserved
45	-	Reserved
46	-	Reserved
47	-	Reserved
47	-	Reserved

7.2 Smart Direct Memory Access Controller (SDMA)

7.2.1 Overview

The Smart Direct Memory Access (SDMA) controller offers highly-competitive DMA features combined with software-based virtual-DMA flexibility. It enables data transfers between peripheral I/O devices and internal/external memories.

The SDMA controller helps maximize system performance by off-loading the Arm core in dynamic data routing.

7.2.1.1 Block Diagram

The figure below shows a block diagram of the SDMA controller. It includes the custom RISC core along with its RAM, ROM, DMA units, and the scheduler.

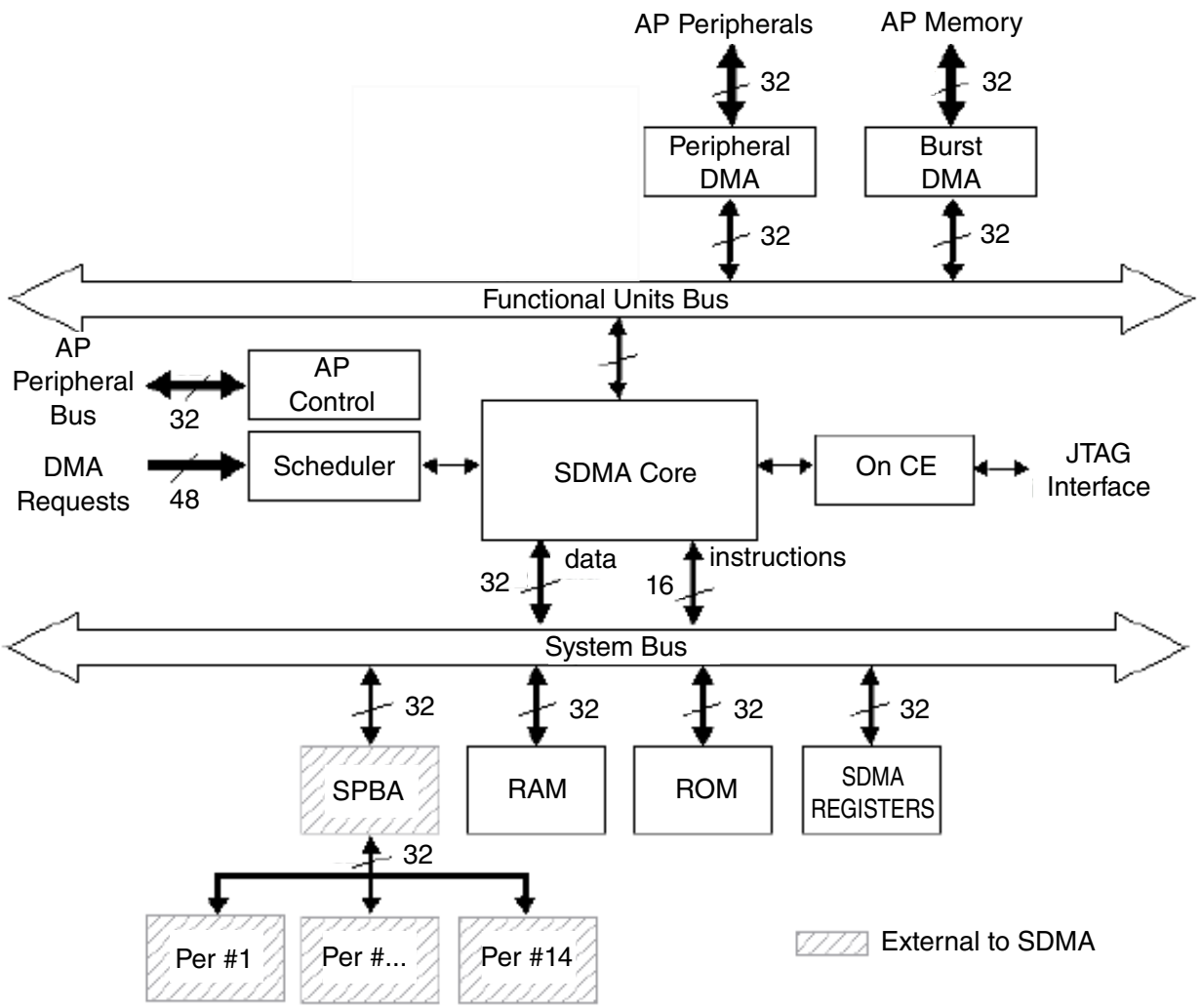


Figure 7-1. SDMA Block Diagram

The SDMA core executes short routines that perform DMA transfers; these routines are called *scripts*. The SDMA core interfaces to its own memory via the SDMA system bus. The SDMA system bus supports a 32-bit data path and a 16-bit address bus. The system bus datapath is used for both 16-bit instruction (program) memory access and 32-bit data access. DMA units interface to the core via the Functional Unit Bus and use dedicated registers to perform DMA transfers.

The SDMA memory contains a ROM and a RAM. The ROM contains startup scripts (for example, boot code) and other common utilities, which are referenced by the scripts that reside in the RAM. The internal RAM is divided into a context area and a script area (more details about this mapping are available in Instruction Memory Map and Data Memory Map).

Every transfer channel requires one context area to keep the contents of all the core and unit registers while inactive. Channel scripts are downloaded into the internal RAM by the SDMA using a dedicated channel that is started during the boot sequence. Downloads are invoked using commands and pointers provided by the Arm platform. Every channel contains a corresponding channel script located in RAM and/or ROM that can be reconfigured independently as-needed. Channel scripts can be stored in an external memory and downloaded when needed. The SDMA can be configured with any mixture of scripts to enable an endless combination of supported services.

The scheduler monitors and detects DMA requests, mapping them to channels, and mapping individual channels to a pre-configured priority. At any given point, the scheduler presents the highest priority channel that requires service to the SDMA core. A special SDMA core instruction is used to "conditionally yield" the current channel being executed to an eligible channel that requires service. If (and only if) there is an eligible channel pending, will the current channel execution be preempted.

There are two yield instructions that differently determine the eligible channels: In the first version, eligible channels are pending channels with a strictly higher priority than the current channel priority. In the second version (yieldge), eligible channels are pending channels with a priority that is greater or equal to the current channel priority. The scheduler detects devices that need service through its 48 DMA request inputs. After a request is detected, the scheduler determines the channel(s) that is (are) triggered by this request and marks it (them) as pending in the "Channel Pending (EP)" register. The priorities of all the pending channels are continuously evaluated in order to update the highest pending priority. The channel pending flag is cleared by the channel script when the transfer has completed.

The Arm platform control block contains the control registers used to configure the 32 individual channels. There are 48 Channel Enable registers, and every register maps one DMA request to any desired combination of channels. The 32 Priority registers are used to assign a programmable 1-of-7 level priority to every possible channel. This block also contains all other control registers that the Arm platform can access.

The 48 DMA requests that are connected to the scheduler come from a variety of sources. The "receive register full" and "transmit register empty" signals found in the UART and USB ports are typical examples of DMA requests that can be connected to the SDMA. These requests can be used to trigger a specific SDMA channel, or several channels.

There is an OnCE compatible debug port for product development. The OnCE includes support for setting breakpoints, single-step and trace, and register dump capability. In addition, all memory locations are accessible from the debug port.

7.2.1.2 Features

The following are the SDMA features:

- Multi-channel DMA supporting up to 32 time-division multiplexed DMA channels
- Hardware or software driven triggers for each channel
- 48 hardware driven triggers that can be mapped to any channel.
- Memory accesses including linear addressing, FIFO addressing and 2D addressing
- Fast context-switching with two-level, priority-based preemptive multi-tasking
- 16-bit instruction-set micro-RISC engine (the SDMA core)
- Two DMA units with some or all the following features:
  - Auto-flush and prefetch capability
  - Flexible address management (increment, decrement, and no address changes on source and destination address)
  - Misaligned data-transfer support
  - Uni-directional and bi-directional flows (copy mode)
  - Up to eight-word buffers for configurable burst transfers
- Support of byte-swapping
- An available API and library of scripts
- Little-Endian and Big-Endian modes
- Hardware handshakes for low-power entry sequence
- Security support to lock contents of the SDMA script RAM.
- 4-Kbyte ROM containing startup scripts (for example, boot code) and other common utilities that can be referenced by RAM-located scripts
- 8-Kbyte RAM area is divided into a processor context area and a code space area used to store channel scripts that are downloaded from the system memory
- Debug support, including a OnCE port, real-time monitors, and embedded cross-trigger events
- Supported clock frequencies in process:
  - Configurable clock options for the SDMA core and the Arm platform DMA units
    - 1:2 ratio with maximum of SDMA core running at Arm platform Peripheral Bus speed and DMA running at max DMA frequency.
    - 1:1 ratio when both SDMA core and Arm platform DMA clocks are set to the Arm platform Peripheral Bus speed.
- Peripheral bus interface for configuration register programming by the Arm platform

- The SDMA RISC engine (arithmetic and logic operations), which is referred to as the "SDMA core."
- An internal peripheral bus connected to the Shared Peripherals Bus Interface (SPBA) that enables access to up to 14 shared peripherals. SDMA supports 32-bit accesses to word peripherals and 16-bit accesses to half-word peripherals.
- The peripheral DMA unit that is hooked-up to the Arm platform Crossbar Switch to service Arm peripherals
- The burst DMA unit is able to perform burst accesses to the external memory
- All the DMA units are 32-bit AHB masters. They are connected to different buses, thus allowing concurrent accesses.

7.2.2 Functional Description

The figure below shows the SDMA topology, and is composed of the following components:

- SDMA Core (SDMA Core)
- SDMA Scheduler (Scheduler)
- Functional Units:
  - Burst DMA (Burst DMA Unit)
  - Peripheral DMA (Peripheral DMA Unit)
- Arm platform Control for Arm control register access.
- Internal RAM and ROM Memory (SDMA Programming Model)
- OnCE debug Port (The OnCE Controller)

The functional unit bus provides access by the SDMA core to the DMA units. The system bus provides access to SDMA internal memory and also supports up to 14 peripherals.

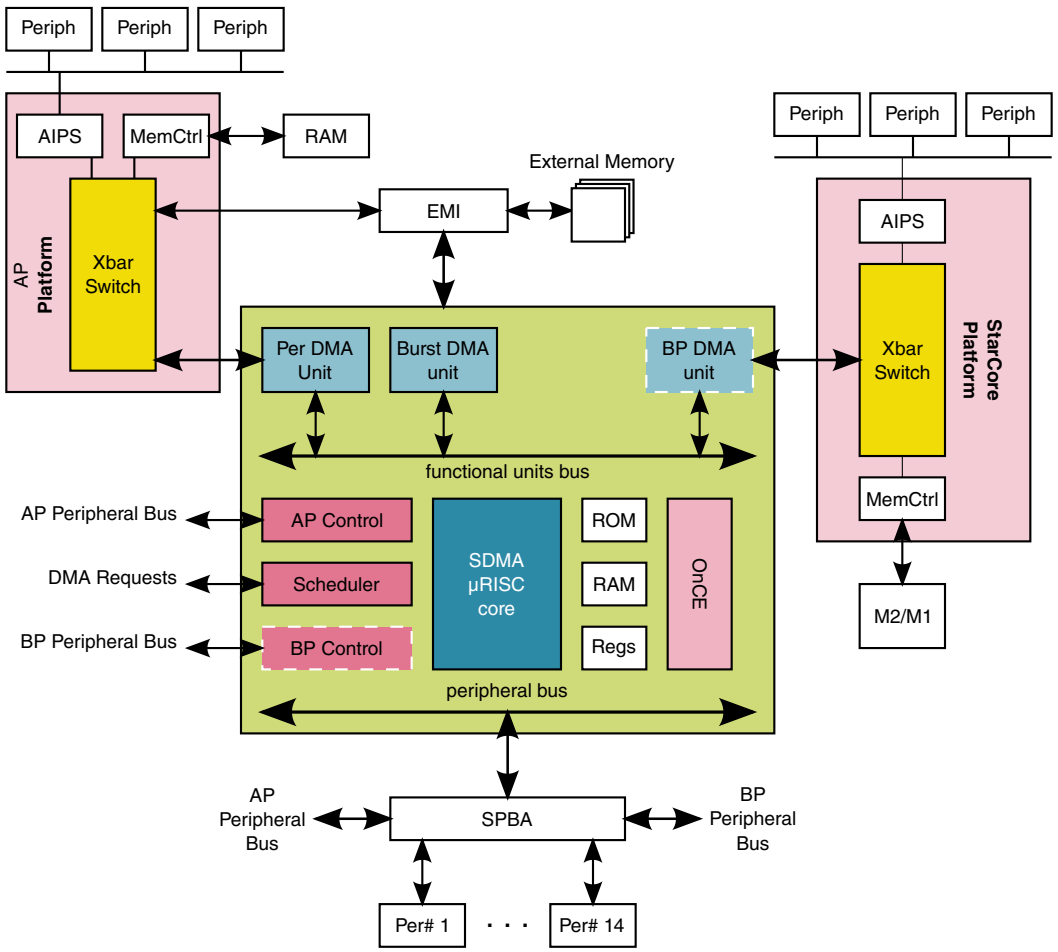


Figure 7-2. SDMA Connections

7.2.2.1 SDMA Core

The SDMA core is a customized RISC-like processor that is specifically developed to control DMA units and perform L1 tasks like byte-stuffing or framing.

The SDMA core incorporates on-chip debug capability using the OnCE.

The SDMA core is based on a 32-bit register architecture with 16-bit instructions. There are eight general purpose 32-bit registers, four flags (T, LM, SF, and DF), and four PCU registers (PC, RPC, SPC, and EPC) that can address 16,384 16-bit instructions.

7.2.2.1.1 SDMA Core Structure

The figure found here shows the structure of the SDMA core. It also shows the different registers, calculation resources, and possible data movements.

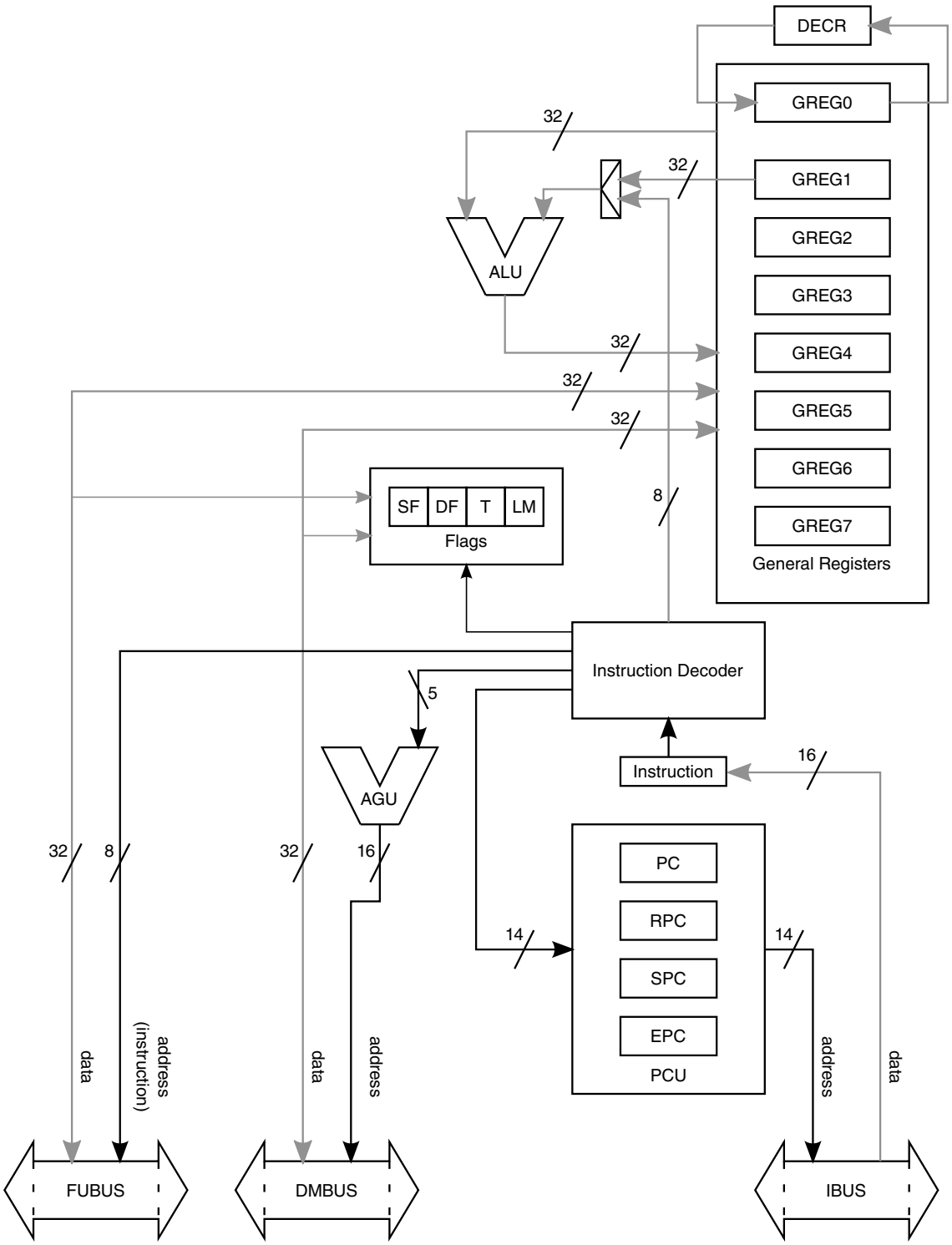


Figure 7-3. SDMA Core

- The Program Control Unit (PCU) is described in Program Control Unit (PCU). It handles the state of the core and generates the instruction fetch addresses. Instructions are retrieved from the Instruction Bus (IBUS) and stored in the SDMA

core instruction register prior to their decoding. The PCU contains the following registers:
    - The Program Counter (PC) contains the address of the current instruction.
    - The Return Program Counter (RPC) contains the address of the instruction that follows a jump to the subroutine.
    - The Start Program Counter (SPC) contains the address of the first instruction of the current hardware loop.
    - End Program Counter (EPC) contains the address of the last instruction of the current hardware loop.
- The other core registers are the general purpose registers (GREGn) and the flags.
    - The general purpose registers can be used to hold data and addresses. They can be loaded with immediate values (for example, 8-bit data that are encoded in the instruction), results of calculations that were performed with the ALU, 32-bit data that comes from the memory or peripherals via the Data Memory Bus (DMBUS), 32-bit data that comes from the DMAs via the Functional Units Bus (FUBUS) or another general purpose register. Their content can be the operands of the ALU, the data to send on either bus (DMBUS or FUBUS), or a pointer to memory (DMBUS address).
    - The general register 0 (GREG0) is also the hardware loop counter. In hardware loops, it cannot be used for any other purpose. This register uses a dedicated decrement unit (DECR) shown in Figure 7-3.
    - The flags reflect the status of operations:
        - SF and DF are set when the last load or store on either bus (FUBUS or DMBUS) received an error response.
        - LM is set when the core is executing instructions inside a hardware loop.
        - T is set when the ALU operation result was 0 or the loop counter reaches 0 (the latter is preponderant when an ALU operation is the last instruction of a hardware loop).
- The ALU has two operands: any general register and either a second general register or an immediate value. The result is always stored into the first general register. A NOP function can be utilized by moving a register's contents into itself (For example, the instruction: mov R0,R0).
- The 16-bit instructions are fetched via the instruction bus (IBUS) whose address is driven by the PC. The SDMA RAM and ROM are visible to the core as 16-bit devices through this interface.
- The memory (RAM and ROM), memory mapped registers, and external peripherals are accessed via the DMBUS. The address is always taken from a general register whose content is added to a 5-bit immediate value. This is the only available addressing mode. The DMBUS is a 32-bit data bus. Except for the peripherals that

are external to the SDMA, the address accuracy is the 32-bit word (for example, adding 1 to an address points to the next word, not the next byte).
- The functional units are accessed via the FUBUS connection. The data is exchanged with any general register, but the address (which in fact is the instruction and the selector of the functional unit) comes from an 8-bit field of the corresponding load or store.

7.2.2.1.2 Program Control Unit (PCU)

This part of the SDMA core is dedicated to the control of the RISC engine, as implied by the instructions that are executed. Its behavior is determined by the instruction type and the inputs of the SDMA.

It contains the PC, RPC, SPC, and EPC registers that are described in SDMA Core Structure.

7.2.2.1.2.1 Instruction Types

The state sequence and the delay of execution vary according to the type of the instruction. There are six possible categories of instructions, as follows:

1. Standard: Most of the instructions belong to this category, and always last 1 cycle.
2. ldf/stf: These are respectively the load and store instructions that access the functional units. They last 1+n cycles where n is the number of wait-states of the targeted functional unit.
3. ld/st: These are the load and store instructions that access the memory and peripherals. They last 1+n cycles where n is the number of wait-states of the targeted device (1 for the ROM, RAM, and memory mapped registers, 1 + the external peripheral wait-states). These instructions always last at least two cycles, but the core is able to handle them in one cycle. The first wait-state is inserted outside the core.
4. Branch: These are all the instructions that cause the Program Counter to point to another instruction other than the following one (for example, one that breaks the sequential flow). There are the absolute jumps, the conditional branches, the jump to the sub-routines, and the return from the sub-routine.
5. Loop,Modified Load or Store: The hardware loop instruction modifies the potential behavior of any load or store inside the loop (for example, when the LM flag is set). A jump may be implied after any such load or store if it received an error. The error causes an early exit of the loop, which means a jump to the instruction that follows the one that is pointed to by EPC. An additional cycle is required by the PCU to perform the jump (+1 to the ld/st/ldf/stf original execution delay). Although there is usually an implicit jump after the last instruction of the loop when the PC goes back to SPC, this is performed at no cycle cost.

6. Done: The done, yield, or yieldge instructions are used to control channel switching. When no channel switching is performed, these instructions last a single cycle. When there is a change of channel or context switch, the delay is variable and depends on many factors (as detailed in Context Switching).

7.2.2.1.2.2 PCU States

The PCU state is visible through outputs of the SDMA (see Real-Time Debug Outputs) or the OnCE status register(see OnCE Status Register (OSTAT)).

The PCU state is a four-bit field that can take the values shown in the following table. Figure 7-4 shows the possible state transitions and the corresponding conditions.

Table 7-5. PCU States

Value	State	Description
0	Program	This is the usual instruction cycle.
1	Data	This state is inserted when there are wait-states during a load or a store on the data bus (ld/st type).
2	Change of Flow	This is the second cycle of any instruction that breaks the sequence of instructions (branch and done types). This state lasts only a single cycle; it is always followed by the Program state.
3	Error in Loop	This state is used when an error causes a hardware loop exit (loop-modified load or store type). This state only lasts a single cycle; it is always followed by the Program state.
4	Debug	The SDMA is stopped in debug mode.
5	Functional Unit	This state is inserted when there are wait-states during a load or a store on the functional units bus (ldf/stf type).
6	Sleep	No script is running: The core is idle after saving the last channel context.
7	Save	The context switch FSM is saving the current channel.
8	Program in Sleep	Same as Program except there is no associated channel, this state is used when instructions are executed after entering debug mode, whereas the core was in either Sleep mode.
9	Data in Sleep	This is the same as Data except there is no associated channel.
10	Change of Flow in Sleep	This is the same as Change of Flow except there is no associated channel. This state only lasts a single cycle, and is always followed by the Program in Sleep state.
11	Error in Loop in Sleep	This is the same as Error in Loop except there is no associated. channel. This state only lasts a single cycle, and is always followed by the Program in Sleep state.
12	Debug in Sleep	This is the same as Debug except the core was put in debug mode when no channel was active.
13	Functional Unit in Sleep	This is the same as Functional Unit except there is no associated channel.
14	Sleep after Reset	This shows that no script is running, and the core is idle after a reset. When a channel becomes active, no context is restored but the core starts its boot program located at address 0 (or the address available in register in Channel 0 Boot Address (SDMAARM_CHN0ADDR)).
15	Restore	The context switch FSM is restoring the next channel context.

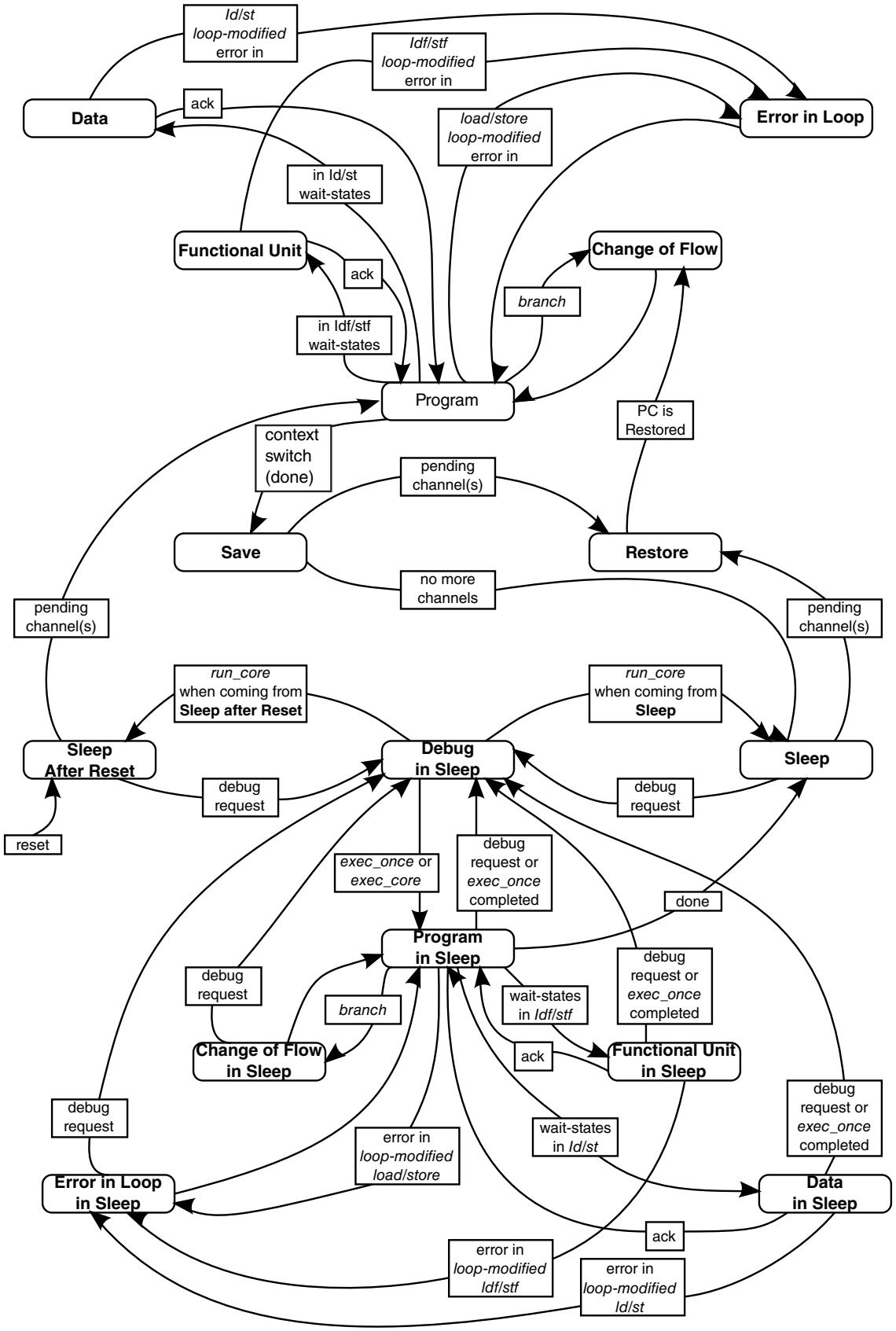


Figure 7-4. PCU State Diagram

7.2.2.1.3 SDMA Core Memory

The SDMA has two memory spaces: one for the instructions and one for the data. As both spaces share the same resources (ROM and RAM devices), the system bus manages possible conflicts when the core accesses the same resource for both an instruction read and a data read or write.

Program and data memory is further described in Address Space.

Instructions of 16-bit width are stored in 32-bit wide devices and can be accessed as data. The mapping is Big Endian: an even instruction address (terminated by 0) accesses the most significant part of the 32-bit data (bits [31:16]), and an odd instruction address (terminated by 1) accesses the least significant part of the 32-bit data (bits [15:0]). Instructions can be fetched out of internal ROM or RAM.

Data can be read from ROM, RAM, memory mapped registers, and external peripherals, and written to the same devices (except the ROM).

The ROM contains bootload scripts, channel scripts, and common subroutines which may be referenced by channel scripts elsewhere in the ROM or RAM.

The RAM is divided into a context area and a code space area which may be used to store channel scripts. The RAM contains undefined values after a hardware reset. Channel scripts and initial context values are downloaded into RAM using channel 0 which is reserved for bootload functions.

7.2.2.2 Scheduler

All channel scheduling hardware is included in the Scheduler.

7.2.2.2.1 Primary Functions

The scheduler is a hardware-based design used to coordinate the timely execution of 32 virtual DMA channels by the SDMA core on the basis of channel status and priority.

The scheduler performs the following functions:

- Monitors, detects, and registers the occurrence of any one of the 48 DMA requests
- Links a specific request to a channel or group of channels (channel mapping)
- Ignores requests that are not mapped to a previously configured channel
- Maintains a list of all the channels that are requesting service
- Assigns a pre-programmed priority level (1 of 7) to every channel requesting service
- Detects and flags overrun/underrun conditions

7.2.2.2.2 Channels and DMA Requests

7.2.2.2.2.1 Channels

A Virtual Channel (hereafter simply called a channel) manages a flow of data through the SDMA. Flows are typically unidirectional.

The SDMA can have up to 32 simultaneously operating channels, numbered from 0 to 31. Channel 0 is usually dedicated to control the SDMA script downloading. All the channels can be assigned by the Arm platform software.

7.2.2.2.2.2 DMA Requests

A DMA request is caused by externally (for example, external to the SDMA) controlled conditions (for example, UART receive FIFO reaches a threshold). The SDMA currently supports up to 48 DMA requests.

7.2.2.2.2.3 Mapping from DMA Requests to Channels and Priorities

A channel can stall waiting on a single DMA request. A single DMA request can awake more than one channel (in fact, any request can awake any combination of channels).

The mapping between DMA requests and channels is program-controlled. There is a storage element assigned for each of the 48 requests that contains a bitmap table of the channels that are awakened by the event.

Every channel also has a three-bit register that indicates its priority.

7.2.2.2.3 Scheduler Functional Description

Scheduler Overview describes the behavior of the SDMA scheduler-from the channel enabling conditions to the highest priority pending channel selection.

7.2.2.2.3.1 Scheduler Overview

The scheduler algorithm is built in hardware. It is provided with possibilities for the Arm platform to control its behavior.

The scheduler processes incoming DMA requests, maps detected requests to 0, one, or several channels, maintains a list of channels that are requesting service (pending channels), identifies the top priority and its associated channel, and selects the next active channel when the current channel yields.

The following figure shows a functional overview.

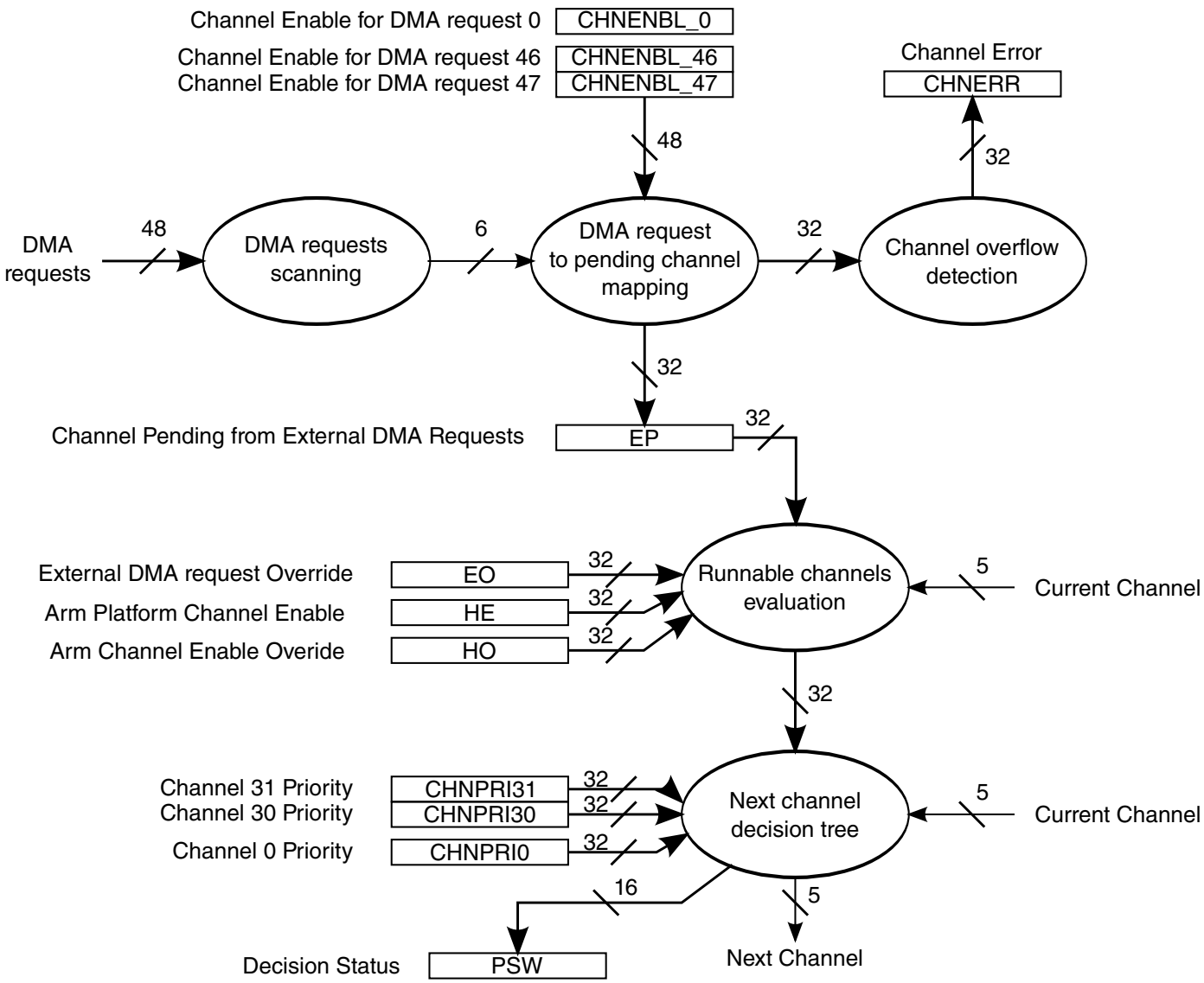


Figure 7-5. SDMA Hardware Scheduler

7.2.2.2.3.2 DMA Requests Scanning

The scheduler contains a 48-bit edge detection device that detects the rising edge of every DMA request and transmits the request number to the next stage.

The DMA requests are assumed to be generated on the same reference clock as the SDMA core clock; they are detected as soon as the signal goes from a 1-to-n-cycles low state to a 1-to-m-cycles high state.

This system is able to detect single-cycle pulses as well as level-based DMA requests such as a FIFO threshold crossing. In this case, the SDMA provides a memory mapped register that can be used by the channel script to monitor the DMA requests lines, and thus determines whether the data transfer is done or not done, and then continues with the transfer or closes the channel.

When several DMA requests are detected at the same time, they are forwarded to the next scheduler stage at the rate of one request per cycle. No request is lost.

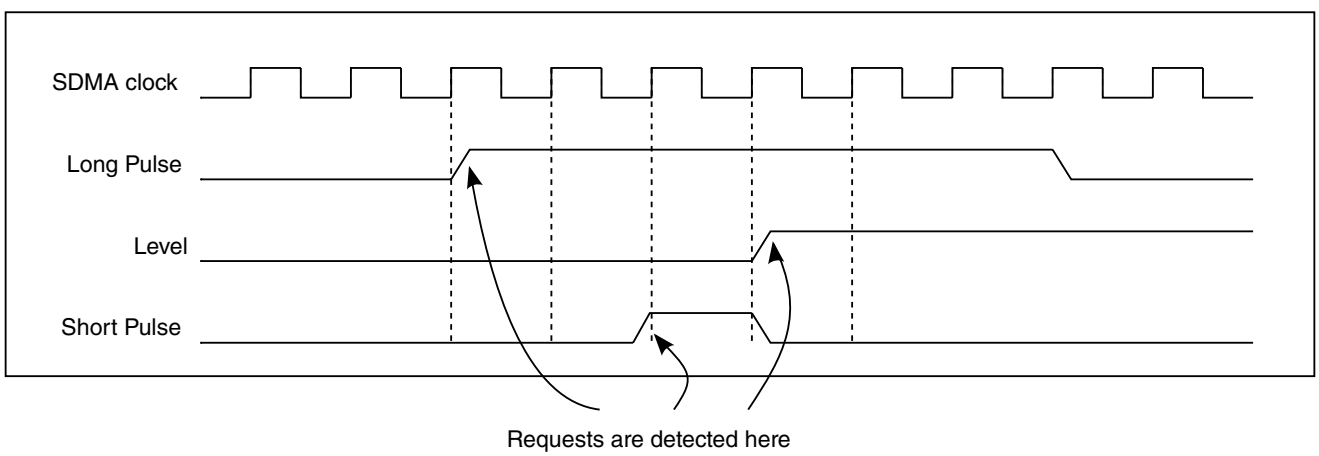


Figure 7-6. Examples of Valid DMA Requests

The DMA request inputs are connected to various sources that depend on the SoC. The exact list of DMA request inputs and their associated number is available in each respective project-specific chapter.

7.2.2.2.3.3 Mapping DMA Requests to Pending Channels

Whenever a DMA request is detected by the first stage, its number is used in the second stage to determine the channels that have to be activated.

This is performed with an array of 48 registers that are 32 bits wide: There are 48 Channel Enable Registers (CHNENBLn), one register per DMA request. The DMA request number selects the Channel Enable Registers, and every bit of this 32-bit register indicates that the corresponding channel must be activated when it is a 1.

This information is passed on the EP register. For every bit of the Channel Enable Register that is set, the corresponding bit of the EP register is also set, and the remaining bits of EP are left unchanged. The transformation of EP is summarized by the following equation:

EP = EP or CHNENBLn

The EP register is used to know which channels require service because they received a DMA request.

Typical contents of the CHNENBLn registers are all 0s, except for a single bit set. For example, a DMA request triggers one channel, but all 0s or several 1s are possible. One DMA request could activate several channels, and the channel execution sequence can be controlled by the channel priorities and numbers, as explained in the next sections. The following table illustrates an example configuration.

NOTE

From the table, the DMA request 0 is programmed to simultaneously trigger channels 0, 1, and 31. Also, DMA requests 30-47 are not used in this example. The remaining channels 2 to 30, are configured to be triggered by DMA requests 29 to 1, respectively.

Table 7-6. Channel Enable RAM Programming Example

DMA Request Number	Channel																															
	31																															0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table continues on the next page...

Table 7-6. Channel Enable RAM Programming Example (continued)

DMA Request Number	Channel																														
	31																														0
7	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table continues on the next page...

Table 7-6. Channel Enable RAM Programming Example (continued)

DMA Request Number	Channel																														
	31																														0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

7.2.2.2.3.4 Channel Overflow

A channel overflow occurs when a DMA request requires service from channel *n* by setting bit *n* of the register EP, but this bit is already set, meaning channel *n* is already pending. This can come from an overrun/underrun condition.

This detection is possible only when the DMA requests are pulses, because a level-based DMA request stays high until it is serviced, even though an underrun or overrun condition occurs, thus preventing another edge detection of the DMA request.

The channel overflow information is saved in the 32-bit CHNERR register (1 bit per channel). You can configure the SDMA to trigger an interrupt to the Arm platform when there are 1s in CHNERR. Every bit of CHNERR is masked with the corresponding bit of INTRMASK and if it gives a 1, the corresponding bit of INTR is set, triggering the interrupt.

7.2.2.2.3.5 Runnable Channels Evaluation

The EP register is used in conjunction with several other 32-bit registers to determine the channels that are runnable.

Registers EO, DO, HO and HE, are controlled by the Arm platform. EP is controlled by the DMA requests and their mapping to channels.

Several channels may be runnable at any given time. The i^{th} channel is runnable if (and only if) the condition below is true:

(HE[i] or HO[i]) and (DO[i]) and (EP[i] or EO[i])

After reset, the HE[i], HO[i], EP[i], and EO[i] bits are all cleared whereas the DO[i] bits are all set. The functions associated with DO are not available for this device. When DO[i] is set, the scheduler condition becomes:

(HE[i] or HO[i]) and (EP[i] or EO[i])

The registers in these equations are controlled as follows:

- Arm platform (host) channel enable flag HE[i] may be set or cleared by the Arm platform with the HSTART and STOP_STAT registers. It can also be cleared by the i^{th} channel script.

  Typical usage is for the Arm platform to set this flag to activate the channel. The flag is cleared by the SDMA core when the transfer is done.

- Externally triggered channel pending flag EP[i] is set by the scheduler when the channel was activated by a DMA request. It can be cleared by the i^{th} channel script.
- The Arm platform channel override flag HO[i] may be set or cleared by the Arm platform. When set, it enables the i^{th} channel to run without the involvement of the Arm platform.

  Typical usage is for the Arm platform to set this flag for channels that do not need Arm platform supervision such as channels that are controlled by DMA request events (EP).

- DO should always be set to 1 so that the runnable channel evaluation considers only HO, HE, EP, and EO.
- Externally triggered channel override flag EO[i] may be set or cleared by the Arm platform. When set, it prevents the i^{th} channel from stopping and stalling on incoming peripheral DMA requests. This is the case when the channel is not handling data transfers with peripherals (for example, a memory to memory transfer).

The SDMA can clear the HE[i], and EP[i] bits by means of a done or notify instruction. The done instruction causes a reschedule; thus, enabling another channel to preempt the current one, while the notify instruction does not. The done and notify instructions can clear either HE[i] or EP[i] (never more than one at a time).

Table 7-7. Runnable Channel Selection Control

Register	Set by	Cleared By
HO	Write to HOSTOVR register	Write to HOSTOVR register

Table continues on the next page...

Table 7-7. Runnable Channel Selection Control (continued)

Register	Set by	Cleared By
HE	Write to HSTART register	Write to STOP_STAT register or by the channel script with the done or notify instructions.
DO	Write to DSPOVR register	Write to DSPOVR register
EO	Write to EVTOVER register	Write to EVTOVER register
EP	Set by external DMA request event input.	By the channel script with the done or notify instructions

7.2.2.2.3.6 Next Channel Decision Tree

The next channel number is computed from the runnable channels list, the current channel number, and their respective priorities.

It is re-evaluated every cycle, but is only used when the current channel yields or terminates by executing a yield, yieldge, or done instruction.

The decision tree is based on the selection of the runnable channel that has the highest priority.

The highest priority channel is selected according to the following rules:

- Runnable channels are sorted by priority.
- If one of the channels with the highest priority had been preempted by a channel with a higher priority, but did not want to yield to a channel of the same priority (for example, it executed a yield, not a yieldge), it is elected as the next channel.
- The channels that belong to the highest priority group are sorted by their number and the channel that has the highest number in this group becomes the next channel. For example, if priorities are the same, channel 31 will be selected before channel 30.

When the current channel requires a reschedule with a yield(ge) or a done instruction, the context switch decision is based on the instruction parameter, the current channel number and priority, and the next channel number and priority. The possible cases are all listed in the following table. The grayed cells correspond to unusual cases that should not occur with a typical usage of the SDMA.

Table 7-8. Channel Switching Decision with a yield, yield(ge), or done

Instruction	Current Channel	Next Channel	Priorities Comparison	New Running Channel/Comments
yield (done 0)	Runnable	Not runnable	none	Current
	Runnable	Runnable	Current > Next	Current
			Current = Next	Current
			Current < Next	Next[1]

Table continues on the next page...

Table 7-8. Channel Switching Decision with a yield, yield(ge), or done (continued)

Instruction	Current Channel	Next Channel	Priorities Comparison	New Running Channel/Comments
	Not runnable	Not runnable	none	none[2] (occurs when the channel was disabled by the Arm platform)
	Not runnable	Runnable	none	Next[1] (occurs when the channel was disabled by the Arm platform)
yieldge (done 1)	Runnable	Not runnable	none	Current
	Runnable	Runnable	Current > Next	Current
			Current = Next	Next[1]
			Current < Next	Next[1]
	Not runnable	Not runnable	none	none[2] (occurs when the channel was disabled by the Arm platform)
	Not runnable	Runnable	none	Next[1] (occurs when the channel was disabled by the Arm platform)
done (done>1)	Not runnable	Not runnable	none	none[2]
	Runnable	Not runnable	none	Current[3] (occurs when the done instruction does not disable the channel runnable condition)
	Not runnable	Runnable	none	Next[1]
	Runnable	Runnable	none	Current[3] (occurs when the done instruction does not disable the channel runnable condition)

1. Current channel script execution is stopped, its context is saved; the next channel context is restored and its script execution resumes
2. Current channel context is saved and SDMA enters IDLE mode
3. Current channel context is saved, then restored, and the current channel script resumes execution

Finally, when the SDMA is in IDLE mode and a runnable channel is elected as the next channel, its context is immediately restored and the script execution resumes.

The *combinatorial-decision* tree supports dynamic modifications of the EP, EO, HE, HO, and DO flags as well as dynamic modifications of the channel priorities. The propagation times are detailed in Scheduler Pipeline Timing Diagram.

The decision tree status is available in the PSW register, which is continuously updated. It contains the next channel priority, the next channel number, the current channel priority, and the current channel number. When a priority is read as 0, it means the channel is not runnable.

A few examples of decisions are presented below:

- Channel 31 is running with priority 5, channels 13 and 24 are pending with the same priority 5; channel 24 is eligible as the next channel since 24 > 13.
- Channel 31 is running with priority 7, channels 13 and 24 are pending with priority 5; channel 31 is the next channel because its priority is greater than the other pending channels.
- Channels 7, 23, and 29 are pending with the same priority. Channel 7 is active and runs a yieldge; it is preempted by channel 29. After a period of time, channel 29 runs a yieldge, it is then preempted by channel 23 that is the selected channel since channel 29 is the current channel. Later, channel 23 runs a yieldge and is preempted by channel 29. Channels 23 and 29 will go on switching after every yieldge until one of them terminates. It is only at that point that channel 7 becomes eligible again.
- Channel 11 is running with priority 3, and channel 15 is pending with priority 4. When the channel 31 script executes a yield instruction, it gets preempted by channel 15; then channels 6 and 18 with priority 3 become pending. Because channel 11 was preempted after executing a yield and there is no pending channel with a strictly greater priority, it is eligible as the next channel (although its number 11 < 18).

7.2.2.2.3.7 Scheduler State Diagram

The Figure 7-7 summarizes the behavior of the SDMA scheduler with details about the exact mechanism of the priority decision tree. It is important to understand the scheduler is a hardwired pipeline, which means all the stages are performed simultaneously every cycle, but a change on any given stage is reflected on the next stage after the delays presented in Scheduler Pipeline Timing Diagram.

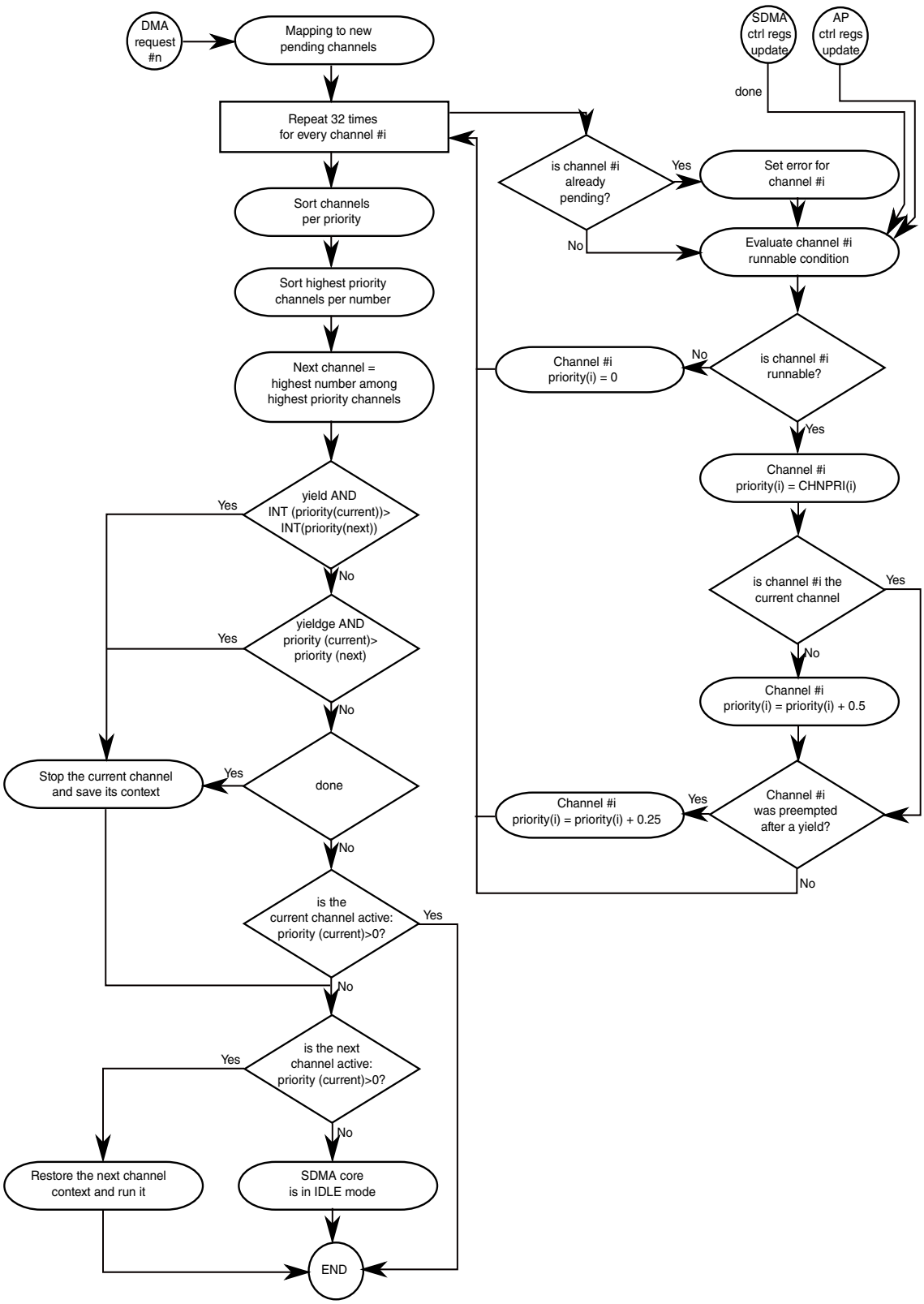


Figure 7-7. Scheduler State Diagram

7.2.2.2.3.8 Scheduler Pipeline Timing Diagram

The SDMA scheduler process of DMA-request and control-register modifications is not immediate.

The figure below shows the exact delays of all the tasks. The reference clock is the SDMA core clock.

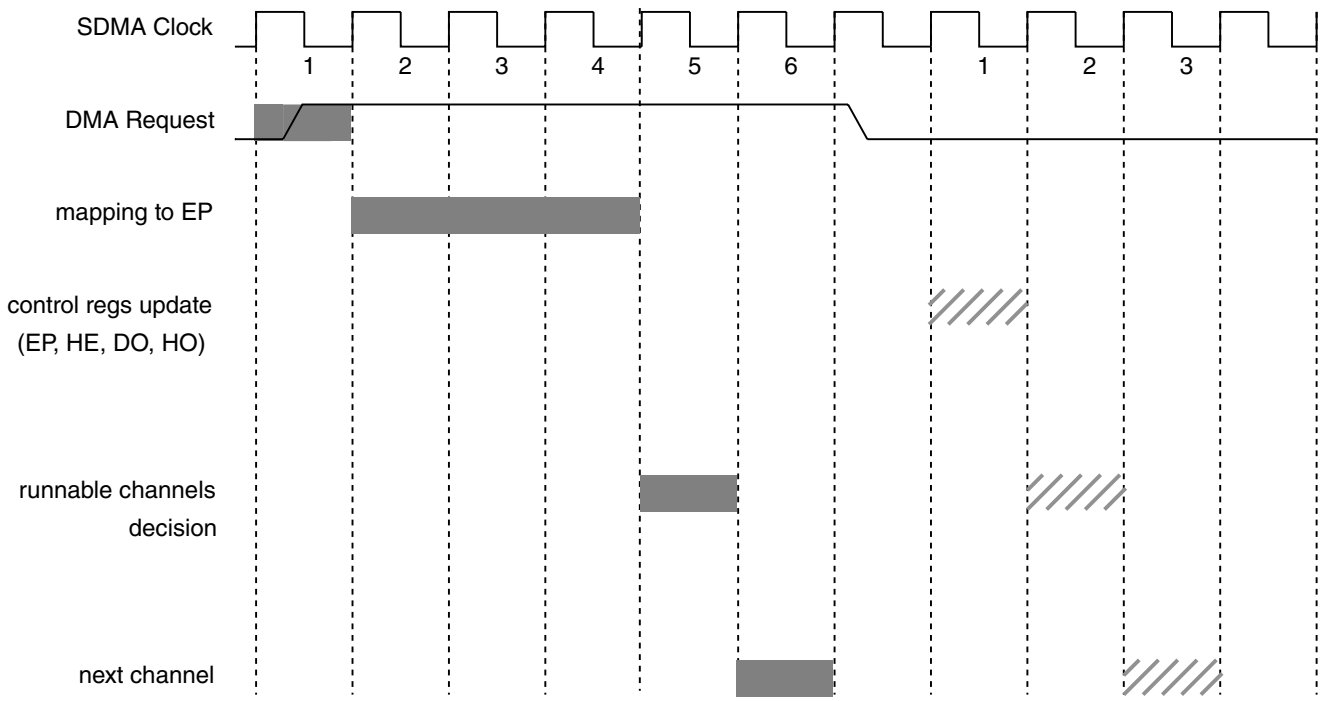


Figure 7-8. Scheduler Timing Diagram

Two numbers can be inferred from this timing diagram. First, it takes six SDMA core clock cycles to update the next channel from a DMA request. Second, it takes three SDMA core clock cycles to update the next channel from a direct modification of the condition registers (EP, DO, HE, or HO) by any processor. The processors that can modify these bits include SDMA with a done instruction or the Arm platform with a write access through the corresponding control port on their respective peripheral bus).

7.2.2.2.3.9 Channel-DMA Request Mapping

The 48 DMA request inputs to the SDMA scheduler are listed in project-specific chapters. Refer to the respective chapters for this information.

7.2.2.2.3.10 Examples: How to Start a Channel

A channel can be started when the following equation is true for channel *i*:

(HE[i] or HO[i]) and (DO[i]) and (EP[i] or EO[i])

Once this equation is true, the scheduler can start this channel according to the priority of all pending channels. Several examples of configuration are listed below:

1. To start a channel triggered by Arm platform software:
   - Initially, configure HO[i]=0, DO[i]=1, and EO[i]=1 using registers indicated in Table 7-7.
   - Arm platform software triggers the channel by writing to the HSTART register to set HE[i]=1, thereby setting the above equation true.
2. To start a channel triggered by DMA request event.
   - Initially, configure HO[i]=1, DO[i]=1, and EO[i]=0 using registers indicated in Table 7-7.
   - The DMA request is asserted to trigger the channel by setting EP[i]=1, which makes the above equation true.

7.2.2.2.4 Context Switching

On execution of a done or yield(ge) instruction, the current channel may be changed either because it has finished (which necessarily happens when the done instruction is executed), or it was preempted by a higher priority channel (which is possible but not systematic when the yield(ge) is executed).

Upon a channel change the SDMA goes through a context switch procedure.

When the current channel yields or ends, the context for that channel is saved into the context RAM locations for that channel. When the next channel starts running, its context is first restored from RAM.

Since context RAM is not yet initialized by reset, there will be no context restore at the beginning of the first channel (bootload channel) run after reset. It is expected that the bootload channel will be used to initialize the context for all other channels. When the bootload channel finishes running or yields, SDMA will enter its SAVE state and save that channel's context into RAM. Then, if the bootload channel is called again later, the context will be restored from RAM when the channel starts again.

The context structure for each channel is defined in Context Switching-Programming and Table 7-13. There will be one context area reserved for each channel. When a channel ends or yields, the SDMA core registers are automatically saved into the context RAM and later restored from the context RAM when the channel is next run. The total RAM

space reserved for 32-channel contexts is either 3K or 4K depending on whether the SMSZ bit is set in the CHN0ADDR register, which enables an additional 8 words of scratch RAM for each context.

7.2.2.2.4.1 Context Switch Modes

The exact procedure to save the context of the old channel, and to restore the context of the new channel depends on the context switch mode selected by the Arm platform in the CONFIG control register.

The following are the context switch modes:

- By default, the "dynamic" context switch is set. This mode provides the most efficient context switch for an average of eight cycles to stop the current channel, save its context, restore the next channel context, and resume its execution. It consists of saving modified registers of the current channel in the background (for example, during the channel execution)-which leaves very few registers to save when the switch is decided-resuming execution of the next channel as soon as possible (for example, when the minimal set of registers is restored), and continuing the restore phase during this execution.
- In "dynamic with no loop" mode, the same principle is followed except the modified registers are only saved in the background when the loop flag is not set. This mode offers almost the same effectiveness as the previous one, but it prevents the system from accessing the RAM during loops to save power. This is the recommended mode for an efficient context-switch when the loop bodies are short.
- In "dynamic power" mode, no background saving is performed, which reduces power consumption to the minimum. The modified registers are only saved when the context switch starts. The restore phase is the same as before. This is the mode that achieves the optimal power consumption at the cost of a slower context-switch.
- In a "static" context switch, all the registers are saved when a context switch is decided, and all the registers are restored before starting the execution of the new channel. This mode enables a predictable behavior of the context switch since all the registers are restored prior to the channel start and all registers are saved after the channel termination.

NOTE

Static context mode should be used for the first channel called after reset to ensure that the all context RAM for that channel is initialized during the context SAVE phase when the channel is done or yields. Subsequent calls to the same channel or different channels may use any of the dynamic context modes. This will ensure that all context locations for the bootload

channel are initialized, and prevent undefined values in context RAM from being loaded during the context restore if the channel is re-started later.

7.2.2.2.4.2 Context Switch Procedure

The Program Control Unit goes into the *save* state, the current context is spilled into memory, and the next channel context is restored according to the context-switch mode that was selected by the Arm platform.

The context switch procedure is as follows:

1. Load the current context's spill base address.
2. Spill the modified registers of the current channel to memory according to the selected context switch mode while the channel is running.

   On a done or yield(ge) that causes the channel preemption, the PCU goes into the *save* state. In *static* mode, all the registers are saved; whereas, in either *dynamic* mode, the registers that were modified but not yet saved are then saved, and the PCU registers and flags are finally saved.

3. Put the SDMA core into *sleep* and wait for new channels to be serviced. This step is skipped if there are pending channels when the current channel is saved.

   As soon as there is at least one pending channel, the PCU goes into its *restore* state to restore the context of the channel that was elected by the scheduler.

   Once a channel is elected, it remains the current channel until its script requests a rescheduling operation with a done or yield(ge) instruction. That means the current channel cannot be modified by the Arm platform, even if it is no more runnable or if its priority is modified.

   The Arm platform can however force a reschedule by writing the corresponding bit in the CONFIG register, which has the same effect as if the script had executed a done instruction. That feature should only be used to stop the SDMA in emergency cases.

4. Load the context base-address of the new channel.

   In "static" mode, all the registers are restored. In either "dynamic" modes, only the PCU registers are restored.

   The new channel is running. In "static" mode, no more activity regarding context restoring or saving is performed. In either "dynamic" modes, the registers are restored in the background every time an access to the context RAM is possible, and

priority is given to restoring the registers that are required by the next instruction to be executed. When a register has not been restored and the next instruction needs it, this instruction gets stalled until the register was restored.

In "dynamic" and "dynamic with no loop" modes, background saving of dirty registers is performed every time an access to the context RAM is possible and allowed by the context switch mode.

NOTE

The contents of a channel context space in the context RAM depends on the selected context switch mode. In "dynamic" and "dynamic with no loop" modes, the contents of the context RAM tend to match the contents of the SDMA registers (except for the PCU registers and flags that are never saved in the background). In "dynamic power" and "static" modes, the contents of the context RAM remain unchanged until the channel terminates with a done or gets preempted.

7.2.2.2.4.3 Context Map in Memory

Refer to Context Switching-Programming.

7.2.2.3 Functional Units

The functional units are small systems that are used by the SDMA core to handle data transfers between the core and a bus domain external to the SDMA.

The SDMA core is able to control and exchange data with these systems by sending instructions and reading or writing data from/to the functional units' registers via the FUBUS. This is done with the ldf and stf instructions.

The following sections provide introductions to the available functional units. Functional Units Programming Model provides descriptions the functional units' behaviors.

7.2.2.3.1 Burst DMA Unit

The burst DMA unit enables the SDMA core to perform data transfers to and from the Arm platform memory.

It is optimized for accessing SDRAM-like devices. It does not provide control to assign a privilege level to the DMA access. The burst DMA unit provides the SDMA with means to do the following:

- Perform up to 8-beat read and write bursts to the Arm platform memory, which optimizes throughput when accessing SDRAM-type devices because of an internal, 36-byte FIFO
- Access the Arm platform memory at once or twice the SDMA core frequency
- Copy data from one Arm platform memory location to another Arm platform memory location at the Arm platform bus speed, which provides a very high throughput
- Control the method for addressing the Arm platform memory (automatic increment of addresses or frozen addresses-the former aimed at accessing RAM-like memory and the latter aimed at accessing single-address FIFOs)
- Enable or disable automatic prefetch when reading data from the Arm platform memory. When the prefetch mode is selected, the burst DMA automatically triggers external bursts to fill its FIFO without waiting for the SDMA core to request the corresponding data, greatly improving throughput.
- Rely on the DMA to automatically flush its FIFO content when there is enough data to generate an 8-beat burst to the Arm platform memory. Or, it forces a flush when a data transfer must terminate.
- In the former case, the SDMA core may only be stalled when it tries writing data and there is not enough room left in the FIFO. In the latter case, the core is stalled until the data is effectively written to the Arm platform memory.

  In automatic flush mode, the core receives an acknowledge that does not reflect the actual error status when the data is effectively written into the Arm platform memory. This error status is retrieved by a later access to the burst DMA. Terminating a write data transfer with a forced flush command guarantees that any bus error to the Arm platform memory is caught.

- Handle address alignment issues between the Arm platform memory map and the SDMA core data. This enables the core to read or write 32-bit data from the burst DMA, whereas the corresponding Arm platform address is not 32-bit aligned. This drastically improves the SDMA scripts' efficiency since the same loop that transfers 32 bits at a time can be used regardless of the start and end addresses in the Arm platform memory space.

This unit structure and registers are described in Burst DMA Structure and Burst DMA Registers.

7.2.2.3.1.1 Burst DMA Structure

The burst DMA is essentially made up of a 36-byte FIFO, address registers, and a controlling state-machine. The 36-byte FIFO enables eight-word buffering with address alignment, and the state-machine manages clock adaptation when required.

The burst DMA is depicted in the figure below.

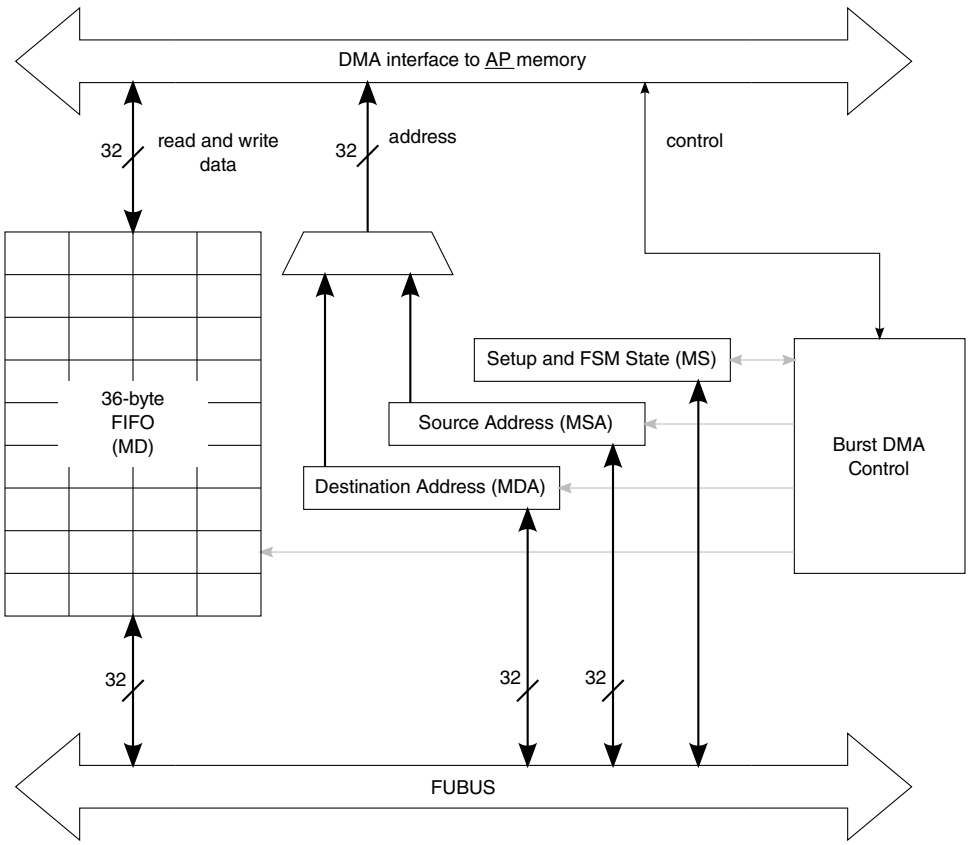


Figure 7-9. Burst DMA Structure

7.2.2.3.1.2 Burst DMA Registers

There are four registers, as follows, that may be accessed from the SDMA core:

- MSA (Memory Source Address) - Holds the source byte address in the Arm platform memory map for reading data from this location. This register is automatically modified every time the core reads new data from the FIFO.
- MDA (Memory Destination Address) - Holds the destination byte address in the Arm platform memory map for writing data to this location. This register is automatically modified every time the core writes new data into the FIFO.
- MD (Memory Data) - Labels the 36-byte FIFO access point: Reading a byte, halfword, or word from MD respectively retrieves the first 1, 2, or 4 bytes of the

FIFO (for example, the bytes that were stored first by the DMA state-machine when transferring data from the Arm platform memory).

- When the FIFO does not hold as many bytes as required by the SDMA core, the core is stalled until the missing bytes are read from the Arm platform memory. In the case of prefetch mode, the DMA controller decides when it should start a burst to Arm platform memory in order to reduce the risk to not have the required data for the future accesses of the core. When there is no prefetching, a burst is triggered when the required data is not available in the FIFO.

  Writing a byte, halfword, or word to MD stores 1, 2, or 4 bytes, respectively, at the end of the FIFO (for example, these bytes are transmitted to the Arm platform memory after all the other bytes that were previously stored in the FIFO). When the FIFO does not have enough room left to hold the written data, the SDMA core is stalled until a sufficient amount of FIFO contents are flushed out to the Arm platform memory. Flushing is decided by the DMA controller when there are enough bytes in the FIFO to perform the largest allowed burst to Arm platform memory (the exact size depends on the burst start address and the AHB 1 Kbyte boundary rule). However, the SDMA core has the ability to force the flushing operation at any time, for example, when at the end of the data transfer, prior to channel closure.

- MS (Memory Setup) - Contains the state of the burst DMA control, the two flags that define whether each address register is incremented after every access to the external memory, and another flag that is set when a bus error occurred.

7.2.2.3.1.3 Burst DMA Data Transfers

Three typical usages have been identified that involve the burst DMA: the data transfer startpoint, the endpoint, or both.

Every case requires a different procedure, as listed in the following sections:

7.2.2.3.1.3.1 Data Retrieval from the Arm platform Memory

The following steps retrieve data from Arm platform memory using the burst DMA unit:

- Set up the MS flags to reflect the mode for the source address (incremented or frozen according to the type of accessed device: memory or peripheral FIFO), then initialize the source address register itself (MSA).
- Read data from the FIFO using the *ldf MD* instruction as many times as needed. If an error occurred during the fetch from Arm platform memory, the DMA control tags the error status on the data and the SDMA core SF flag is set when reading this data from the FIFO.

7.2.2.3.1.3.2 Storing Data Into the Arm platform Memory

The following steps store data from Arm platform memory using the burst DMA unit:

- Set up the MS flags to reflect the mode for the destination address (incremented or frozen according to the type of accessed device: memory or peripheral FIFO), then initialize the destination address register itself (MDA).
- Store data into the FIFO using the *stf MD* instruction as many times as needed.
- When the transfer is finished and if the DMA worked in automatic flush mode, force the flush of the FIFO. This instruction is stalled until all the FIFO data is effectively sent to the Arm platform memory and the error status of the transfer is available in the DF flag.

7.2.2.3.1.3.3 Transferring Data Between Two Arm platform Memory Locations-Burst DMA Unit

The following steps copy data between two Arm platform memory locations using the burst DMA unit:

- Set up the MS flags to reflect the modes for the source and destination addresses (all the combinations are possible), then initialize the source address register (MSA) and the destination address register (MDA). Both addresses must be word-aligned.
- Use as many *stf MD* instructions with the *COPY* flag as needed. Every instruction triggers a burst read of a given number of words from the source address (this number is provided to the burst DMA via the SDMA core general purpose register, which is referenced in the *stf* instruction). Once all the data is loaded into the FIFO, the DMA empties it with a write burst of the same count to the destination address. The DMA acknowledges prior to instruction completion, which frees the SDMA core for other tasks at no delay cost.
- Once the transfer is done, there should be a final access to the burst DMA to check the error status.

7.2.2.3.2 Peripheral DMA Unit

The peripheral DMA unit is the second functional unit that connects the SDMA to the Arm platform memory.

Unlike the burst DMA, it does not support burst transfers and is optimized for accessing peripherals. It does not provide control to assign a privilege level to the DMA access. Its feature list comprises the following:

- Access to the Arm platform peripherals or memory at once or twice the SDMA core frequency

- Data copy from one Arm platform memory location to another Arm platform memory location at memory bus speed, improving throughput
- Control of the method for addressing the Arm platform memory (automatic increment or decrement of addresses or frozen addresses, the first ones aimed at accessing RAM-like memory and the last one aimed at accessing single-address FIFOs)
- Selectable automatic prefetch when reading data from the Arm platform memory. In prefetch mode, the peripheral DMA automatically fetches another data-without waiting for the SDMA core to request it-when its data register is empty, which improves the throughput
- Selectable automatic flush. In this mode, the SDMA core may only be stalled when it tries writing data and the previous write operation is not finished yet; whereas, in forced flush mode, the core is stalled until the data is effectively written to the Arm platform memory.
- In automatic flush mode, the core receives an acknowledge that does not reflect the actual error status when the data is effectively written into the Arm platform memory or the peripheral. This error status is retrieved by a later access to the peripheral DMA. Terminating a write data transfer with a forced flush command guarantees that any bus error to the Arm platform memory has been caught.

This unit structure and registers are described in Peripheral DMA Structure and Peripheral DMA Registers.

7.2.2.3.2.1 Peripheral DMA Structure

The peripheral DMA is made up of a 32-bit data register, two address registers, and a controlling state-machine. The state-machine manages clock adaptation, when required.

It is shown in the following figure.

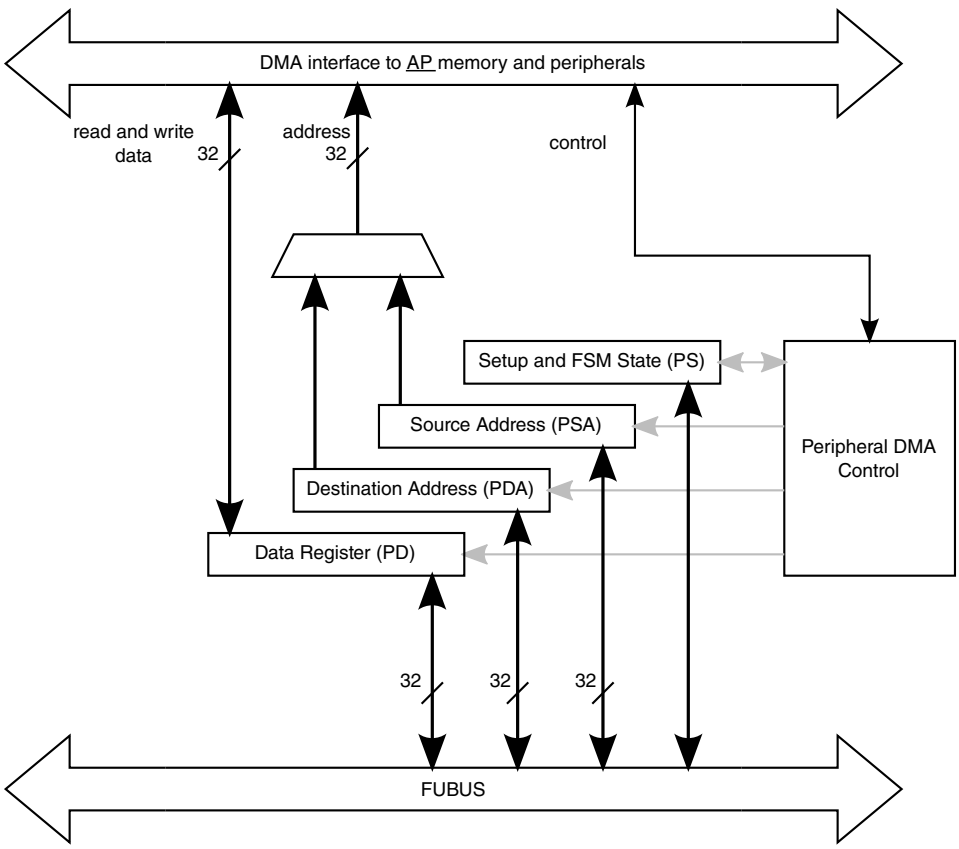


Figure 7-10. Peripheral DMA structure

7.2.2.3.2.2 Peripheral DMA Registers

According to Figure 7-10, the peripheral DMA has four registers that may be read or written by the SDMA core:

- *PD (Peripheral Data)* is the DMA 32-bit data register.
- *PSA (Peripheral Source Address)* holds the source byte address in the Arm platform memory map for reading data from this location. This register is automatically modified every time the core reads a new data from PD.

- *PDA (Peripheral Destination Address)* holds the destination byte address in the Arm platform memory map for writing data to this location. This register is automatically modified every time the core writes a new data into PD.
- *PS (Peripheral Setup)* contains the state of the peripheral DMA control, two configuration fields that define the way address registers are modified after every data access, two additional configuration fields that define the data size to access the source and destination devices, and another field that contains the latest transfer error status.

7.2.2.3.2.3 Peripheral DMA Data Transfers

There are three typical usages that involve the peripheral DMA, whether it is the data transfer start-point, endpoint, or both.

Every case requires a different procedure, as described in Data Retrieval from the Arm platform Memory or Peripheral, Storing Data into the Arm platform Memory or Peripheral, and Transferring Data Between Two Arm platform Memory Locations-Peripheral DMA Unit.

7.2.2.3.2.3.1 Data Retrieval from the Arm platform Memory or Peripheral

The following steps retrieve data from Arm platform memory using the peripheral DMA unit:

- Set up the PS fields to reflect the mode and data size for the source (incremented, decremented, or frozen address register; 8-bit, 16-bit, or 32-bit data transfers), then initialize the source address register itself (PSA) with an address that is aligned to the programmed data size.
- Read data from PD using the ldf PD instruction as many times as needed. If an error occurs during the fetch from the Arm platform memory or peripheral, the DMA control tags the error status on the data and the SDMA core SF flag is set when reading this data from PD.

7.2.2.3.2.3.2 Storing Data into the Arm platform Memory or Peripheral

The following steps store data to Arm platform memory using the peripheral DMA unit:

- Set up the PS fields to reflect the mode and data size for the destination (incremented, decremented, or frozen address register; 8-bit, 16-bit, or 32-bit data transfers), then initialize the destination address register itself (PDA) with an address that is aligned to the programmed data size.

- Store data into PD using the *stf PD* instruction as many times as needed.
- When the transfer is finished and if the peripheral DMA worked in automatic flush mode, force the flush of PD. This instruction is stalled until PD contents are effectively sent to the Arm platform memory or peripheral, and the error status of the transfer is available in the DF flag.

7.2.2.3.2.3.3 Transferring Data Between Two Arm platform Memory Locations-Peripheral DMA Unit

The following steps copy data between two Arm platform memory locations using the peripheral DMA unit:

- Set up the PS fields to reflect the modes and data size for the source and destination addresses (all the combinations of addressing modes are possible, but both data sizes must be identical), then initialize the source address register (PSA) and the destination address register (PDA). Both addresses must be aligned with the programmed data size.
- Use as many *stf PD* instructions with the *COPY* flag as needed. Every instruction triggers a single read from the source address; a single write of the received data immediately follows. The DMA acknowledges prior to instruction completion, which frees the SDMA core for other tasks at no delay cost.
- Once the transfer is done, there should be a final access to the peripheral DMA to check the error status.

7.2.2.4 SDMA Security Support

The SDMA provides support to SDMA software to block unauthorized updates to the scripts in RAM.

SDMA supports the following Security modes:

- Open Mode: has full control to load scripts and context into SDMA RAM. This is the default mode.
- Locked Mode: The Arm platform loads scripts and channel contexts at startup when it is still executing known safe software. When finished, it locks the SDMA to prevent further updates to RAM and selected registers. More details described in Locked Mode.

7.2.2.4.1 Locked Mode

The LOCK bit in the SDMA_LOCK register provides support for SDMA scripts to freeze RAM contents after the initial bootload routine to prevent future unauthorized updates to SDMA RAM.

After initial RAM contents are uploaded, Arm platform software can set the LOCK bit to secure the RAM contents to prevent future updates by an unauthorized. After the LOCK bit is written with a '1', the SDMA is "locked" until reset.

The LOCK bit can be read in the SDMA's internal memory map in the LOCK register (see Section SDMA LOCK (SDMAARM_SDMA_LOCK)). SDMA scripts which load information into RAM can check the value of the LOCK bit to determine if an upload to RAM is allowed. If not allowed, the script can refuse to allow the request to copy data into the RAM to continue. The exact use of the LOCK bit in SDMA scripts for security control will be described in SDMA software documentation (see SDMA Scripts).

While SDMA is locked, attempts to write to the SDMA_LOCK, CHN0ADR, ILLINSTADDR, and ONCE_ENB registers will be ignored. All registers remain readable. Writes to other registers are still allowed.

Once the SDMA is locked, the LOCK bit can only be cleared by a reset. A hardware reset will always clear the LOCK bit. A software reset initiated by writing to the RESET register will only clear the LOCK bit if the SRESET_LOCK_CLR bit in the SDMA_LOCK register is set. Since SDMA_LOCK register cannot be updated if SDMA is locked, the SRESET_LOCK_CLR bit must be configured before setting the LOCK bit. The SREST_LOCK_CLR bit will also be cleared by resets that clear the LOCK bit.

The SDMA RISC core uses the ILLINST and CHN0ADDR registers as pointers to determine where to jump to after an illegal instruction or upon boot after a reset. The LOCK bit prevents updates to these registers to protect against unauthorized changes to these pointers.

While SDMA is locked, the ONCE_ENB register cannot be written to prevent the OnCE under Arm platform control from being used to gain access to SDMA internal memory. If Arm platform control of the OnCE is enabled before setting the LOCK bit, the Arm platform can use the ONCE for debug purpose after LOCK is set.

7.2.2.5 OnCE and PCU Debug States

The SDMA has two different debug modes in which the OnCE performs debug instructions.

Refer to Figure 7-4 for an example of the PCU states in debug. The following are the two debug states:

- When a channel is running (that is, when CCR and CCPRI are different from 0, which can be read in the PSW register), SDMA can execute a SoftBkpt instruction from the channel script or receive a debug request. When either happens, the SDMA enters its "Classical" *Debug* state, which is described in OnCE and Real-Time Debug.
- When a channel is not running, the SDMA can be in *Sleep* state or in *Sleep after Reset* state. If a debug request is sent to the core, it enters its *Debug in Sleep* state. This debug mode works similarly to the "Classical" *Debug* state, except it returns to the original state (*Sleep* or *Sleep after Reset*) when the debug mode is left via the exec_core instruction of the OnCE. From this *Debug in Sleep* state, the SDMA can execute a program whereas no channel is running. If a new debug request is sent to the core or if a SoftBkpt is executed, it comes back to this *Debug in Sleep* state.

The OnCE is provided with several instructions that can be executed when the core is in either debug state. The following table summarizes the behavior of these OnCE debug instructions. There exists other secondary OnCE instructions that are described in OnCE and Real-Time Debug.

Table 7-9. SDMA in Debug Mode

Instruction	Debug	Debug in Sleep
exec_once	exec_once <instruction> SDMA executes the <instruction> and returns to the *Debug* state. The Program Counter (PC) is not incremented. This command must not be used with an instruction that modifies the PC value.	exec_once <instruction> SDMA executes the <instruction> and returns to the *Debug in Sleep* state. The Program Counter (PC) is not incremented. This command must not be used with an instruction that modifies the PC value.
run_core	run_core <instruction> SDMA executes the <instruction>, leaves the *Debug* state and continues executing the channel script from the position where it stopped. This command must not be used with an instruction that modifies the PC value.	run_core <instruction> SDMA executes the <instruction> and returns to its *Sleep* or *Sleep after Reset* initial state. This command must not be used with an instruction that modifies the PC value.
exec_core	exec_core <instruction> It is similar to run_core except it requires an instruction that changes the PC value (jump, branch...): the SDMA jumps to the new PC value, leaves the *Debug* state and starts executing instructions from this new PC value.	exec_core <instruction> If the previous state was *Sleep after Reset*, the SDMA returns to this state, and Chn0Addr value overrides the PC value. Otherwise, the SDMA jumps to the new PC value and starts executing instructions from this new PC.

NOTE

The feature exec_core in *Debug in Sleep* after *Sleep after Reset* was added for the Channel boot (channel 0) to allow the debugger to return to *Sleep after Reset* state with a new PC

value. The SDMA will be ready to boot at the Chn0Addr address.

7.2.2.6 SDMA Clocks and Low Power Modes

The SDMA receives several root clocks from the SoC clock controller block and performs adaptive clock gating to optimize its power consumption. From a user standpoint, clock gating and power mode selection are fully automatized inside the SDMA.

Root clock control is available from the SoC clock controller block.

There are numerous clock sources that are used in the SDMA. They belong to one of two possible clock domains listed in the following table, and have frequency constraints within each domain. Clocks are considered asynchronous between domains.

Within the Arm platform/SDMA clock domain, all clocks must come from the same DPLL. The Arm platform DMA interfaces (peripheral DMA and burst DMA) receive their clock from the Arm platform DMA clock source whose frequency can be once or twice the frequency of the SDMA core clock. The DMA interfaces are designed to work at the Arm platform DMA frequency, but the SDMA core is physically limited to a maximum 104 MHz frequency. Since this is lower than the maximum Arm platform DMA frequency, the SDMA core clock is tied to the Arm platform peripheral clock frequency.

The Arm platform Peripheral Bus Clock source must be an exact sub-frequency of the SDMA Core clock source (any integer value greater or equal to 1).

Table 7-10. Clocking Scheme

Clock Domain	Source Clock	Comments
Arm platform	SDMA core (SDMA main core)	Source clock for the core and all its operations; this clock is thus used by most of the SDMA sub-blocks.
	Arm platform DMA	DMA interface for the peripheral DMA and the burst DMA. It is balanced with the main clock source, and its frequency is either once or twice the main clock frequency.
	Arm platform peripheral	Connection to the Arm platform peripheral bus. It is a sub-frequency of the main clock frequency.
JTAG	TCK	Clock for JTAG access, limited to maximum of 1/8 of the SDMA core clock frequency.

The JTAG clock is sampled by the SDMA main clock to determine its rising edge. This simplifies design and clock management, but it also adds a ratio constraint between those two clocks. It is guaranteed the JTAG interface works properly when the frequency of TCK is lower than $1/8^{th}$ of the frequency of the SDMA main clock (which is about 8 MHz when the SDMA core clock frequency is 66 MHz).

7.2.2.6.1 Clock Gating and Low Power Modes

The SDMA automatically performs power saving without requiring user involvement. It implements two levels of automatic clock gating.

7.2.2.6.1.1 Coarse Clock Gating

Every sub-block clock comes from one of the five available sources, and is gated with the sub-block specific enabling condition.

The following table displays the sub-block clocks and their source. It also indicates the relationships that may exist between different sub-blocks clock enables.

Table 7-11. Sub-blocks Clocks

Sub-block	Source Clocks	Enabling Condition and Comments	Related Enabling Conditions
Core	SDMA Main Core	The core sub-block clock is running when the core is not in one of its sleep states (Sleep or Sleep after Reset) or there is a pending channel. Typically, the core sub-block clock is stopped once all the channels are processed and the core enters its sleep state. A new pending channel awakes the core sub-block clock.	None
Memories	SDMA Main Core	The clock activation only occurs during a core access.	Disabled when Core sub-block clock is disabled or no memory access in progress
Scheduler	SDMA Main Core	Its clock only runs when scheduling is needed: for example, when there are pending channels, upon reception of a DMA request, and anytime the Arm platform modifies the channel running conditions.	None
Arm platform Control	SDMA Main Core & Arm platform peripheral	The Arm platform peripheral clock is solely used to determine the frequency ratio with the SDMA main clock. The control registers' clock is based on *SDMA main clock*; it is active when the Arm platform or the SDMA modifies the contents of one of these registers.	None
Burst DMA	SDMA Main Core & Arm platform DMA	The burst DMA has two clocks: The first clock is derived from the SDMA main core clock and drives registers that are connected to the FUBUS. The second clock is derived from the Arm platform DMA clock and drives registers that are connected to the Arm platform DMA bus outside the SDMA. Both clocks are enabled	Disabled when Core sub-block clock is disabled

Table continues on the next page...

Table 7-11. Sub-blocks Clocks (continued)

Sub-block	Source Clocks	Enabling Condition and Comments	Related Enabling Conditions
		during active phases of data transfers (for example, these clocks are turned off when the burst DMA is not used by the running channel script).	
Peripheral DMA	SDMA Main Core & Arm platform DMA	The peripheral DMA has two clocks: The first clock is derived from SDMA main clock and drives registers that are connected to the FUBUS. The second clock is derived from the Arm platform DMA clock and drives registers that are connected to the Arm platform DMA bus outside the SDMA. Both clocks are enabled during active phases of data transfers (for example, these clocks are turned off when the peripheral DMA is not used by the running channel script).	Disabled when Core sub-block clock is disabled
OnCE	SDMA Main Core	The OnCE clock is derived from main source clock. It is disabled by default. In order to use the OnCE, its clock must be explicitly turned on, either by enabling the OnCE access from the Arm platform peripheral bus (register ONCE_ENB), or by driving the clk_gating_off input pin high. This is a SDMA input whose driver depends on the SoC implementation (typically a JTAG controller). The OnCE also receives the TCK input, which is the JTAG clock. It does not use it as a functional clock; the TCK input is sampled instead. Refer to Synchronization Implementation.	When enabled, all other clocks are systematically on (clock gating is off)

7.2.2.6.1.2 Refined Clock Gating

The SDMA implements a second level of clock gating on a register-per-register basis.

Unlike the first level that covers all the SDMA flip-flops, except the synchronizers (only five flip-flops are always running), the second level is only available for eligible registers, which amounts to about 90% of the SDMA flip-flops.

These gated registers are only clocked when the hardware logic detects a new data loading. This additional gating further reduces dynamic power consumption.

7.2.2.6.1.3 Low Power Modes and User Control

Power savings are automatically managed by the SDMA hardware without any user involvement; however, one can distinguish three different power modes: SLEEP, RUN, and DEBUG.

The following table describes these modes, and shows how to switch from one mode to another.

Table 7-12. Power Modes

Power Mode	Sub-blocks							Comments
	Core	Memories	Scheduler	Arm platform Control	Burst DMA	Peripheral DMA	OnCE	
SLEEP	off[1]	off	wait[2]	wait	off	off	off	Set when the PCU state is either *Sleep* or *Sleep after Reset* and the SDMA is not in DEBUG mode. This is the default mode after reset.
RUN	on[3]	wait	wait	wait	wait	wait	off	Set for the other PCU states that are reachable out of debug: *Program*, *Data*, *Change of Flow*, *Error in Loop*, *Debug*, *Functional Unit*, *Save,* or *Restore*.
DEBUG	on	on	on	on	on	on	on	Set regardless of the PCU state when clock gating is turned off to use the OnCE features (either *clk_gating_off* pin high or ONCE_ENB[0] set).

1. *off*: no clock
2. *wait*: only clocked when accessed or stimulated
3. *on*: clock is always running

It is possible to control the SDMA power mode. The procedures to force the SDMA into either mode are described in SLEEP Mode.

7.2.2.6.1.3.1 SLEEP Mode

This is the default mode after reset; therefore, resetting the SDMA forces this mode.

However, the common procedure is as follows:

- Ensure the *clk_gating_off* pin is low and ONCE_ENB[0] is cleared.
- Disable all channels (via the STOP_STAT control register, and the HO, DO, EO if necessary).
- Wait for the active channels to complete or force a reschedule via the reschedule bit in the RESET register.
- The SDMA is in SLEEP mode making it possible to completely shut off its clock from the chip level clock controller using the procedure described in Stop Mode Response.

7.2.2.6.1.3.2 RUN Mode

This is the default mode when a channel is running:

- Ensure the *clk_gating_off* pin is low and ONCE_ENB[0] is cleared.
- Activate at least one channel (via the HSTART control registers, a DMA request, and/or the HO, DO, EO register bits).

7.2.2.6.1.3.3 DEBUG Mode

The DEBUG mode must be set when one needs to use the debugging facilities of the SDMA.

- Ensure the SDMA clocks are running from the CCM.
- Set the *clk_gating_off* pin high or use the SDMA to set ONCE_ENB[0].

7.2.2.6.1.4 Stop Mode Response

The SDMA receives a stop request from the chip level clock controller. This request may be asserted when the chip enters the stop low power mode.

If the SDMA is running when the request is received, then the SDMA will complete all pending channels before returning to the SLEEP state. The SDMA sends an acknowledgement to the clock controller when the SLEEP state is entered indicating that the SDMAs clocks can be turned off.

7.2.2.6.2 Reset

After reset (either received from the reset block or a software reset required by the Arm platform), the SDMA is in IDLE mode. It will start its boot code located at address 0 once a channel is activated.

Activating a channel can be done by the Arm platform after programming a positive priority and setting the channel bit in the EVTPEND register.

There will not be a context RESTORE for the first channel (bootload channel) called after a reset because the context data in RAM has not been initialized. Static context mode should be used for the first channel called after reset to ensure that the all context RAM for that channel is initialized. Subsequent calls to the same channel or different channels may use any of the dynamic context modes

7.2.2.7 Software Interface

Appendix A fully describes the SDMA Application Programming Interface (API).

7.2.2.8 Initialization Information

This section discusses the following:

- Hardware Reset
- Channel Script Execution
- Initialization and Script Execution Setup Sequence

7.2.2.8.1 Hardware Reset

After reset, the program RAM, context RAM, data RAM, and RAM containing the channel enable registers (CHNENBLn) have unpredictable contents.

The active register set is assigned to channel 0 and the PC is initialized to all zeros. However, since the channel enable register is all zeros, there are no active channels and the SDMA is halted waiting for the boot channel to start.

The Arm platform will have to setup the SDMA in order to boot it. The CONFIG register must be initialized to determine the DMA/core clock ratio (1 or 2). Channel Enable Registers must also be initialized.

To start up the SDMA, the Arm platform first creates some channel control blocks (CCB) and buffer descriptors (BD) in Arm platform memory for the boot channel (channel 0) and then initializes the channel 0 pointer register (SDMA_MC0PTR) to the address of the first control block. Data Structures for Boot Code and Channel Scripts provides an overview of the data structure for the CCB and BD's. The SDMA_HSTART, SDMA_HOSTOVR and SDMA_EVTOVR registers are then configured according to Runnable Channels Evaluation to allow channel 0 to run.

Upon being enabled, the SDMA begins executing the script located at the address indicated by the Channel 0 Boot Address register (SDMA_CHN0ADDR) in the program memory. The reset value of SDMA_CHN0ADDR points to the default bootload script in ROM. This ROM script will read the channel 0 pointer register (SDMA_MC0PTR) to determine the location of the Channel Control Block (SDMA_CCB) in Arm platform memory. The script will then begin fetching by DMA the first channel control block which contains a pointer to the location channel 0 Buffer Descriptor chain which is also fetched via DMA. If the buffer descriptor contains a valid command, the script interprets the command in each buffer descriptor and proceeds to implement the command and move on to the next buffer descriptor control block. The buffer descriptor commands for channel zero are typically set up to load SDMA's program RAM, Data RAM, and initial values for the channel contexts. Some channel scripts expect particular parameters to be passed

There are two ways to make the SDMA boot on a user-defined script. The OnCE (either via its JTAG interface or its Arm platform Control interface) can be used to download any code in the SDMA RAM and force the SDMA to boot on that code. Also, the SDMA_CHN0ADDR register in the Arm platform programming model can be modified to point to user code in RAM which would need to either have been loaded via the ONCE or default bootload routine (ex before a S/W reset).

7.2.2.8.2 Channel Script Execution

The execution of an SDMA script depends on both the instructions that make up the script, the data context upon which it operates, and commands or parameters allowed to the buffer. All these items must be initialized before the script is allowed to execute.

Each of the 32 channels has a separate context, but may share scripts and locations in data RAM.

Channels are initialized by the Arm platform by using channel 0 to download any required scripts and data values and the channels initial context. The context contains all the initial values of the SDMA core registers. This includes the Program Counter (PC) which is set to the start of the desired script in SDMA program memory.

The Arm platform selects which trigger conditions that must occur for the channel to start by configuring the SDMA_CHNENBL, SDMA_HOSTOVR and SDMA_EVTOVR registers. The trigger events include Arm platform setting HE (SDMA_HSTART) or a hardware DMA request asserts an event input to SDMA. The channel can become active according to its priority compared with other runnable channels when the selected trigger(s) cause the condition described in Runnable Channels Evaluation to evaluate as true.

The specific parameters to be passed to each script in the buffer descriptor or context are documented in the software documentation for each script. Please refer to SDMA Scripts for complete script documentation. Buffer Descriptor Format provides an overview of the buffer descriptor format.

7.2.2.8.3 Initialization and Script Execution Setup Sequence

To summarize, the following steps are minimally required to setup SDMA and run channel scripts.

- Perform Hardware Reset. The program RAM, context RAM, data RAM and SDMA_CHNENBLn registers have unpredictable contents after this reset.
- Initialize SDMA_CHNENBLn registers to map DMA request events to desired channels.

- Configure SDMA_CHNPRIn registers to select priority for runnable channels. A non-zero priority is required for the channel to run.
- Configure the SDMA_CONFIG register to select DMA to SDMA core clock ratio .
- Set up channel control blocks and buffer descriptors in Arm platform to specify the loading of SDMA program RAM and channel contexts for each SDMA channel to be used. Reference Data Structures for Boot Code and Channel Scripts.
- Configure SDMA_MC0PTR register with base address of Arm platform Channel Control Block base address.
- Initialize SDMA_CHNENBLn registers to map DMA request events to associated channel. Reference Mapping DMA Requests to Pending Channels.
- Configure SDMA_CHNPRIn registers to set priority for each channel to be run.
- For each channel to be run, configure SDMA_HOSTOVR (HO) and SDMA_EVTOVR (EO) registers to select which events (hardware and/or software trigger events) must occur for the channel to be runnable. Reference Runnable Channels Evaluation.
- Set bit 0 of the SDMA_HSTART register to set HE[0] and allow Channel 0 to run (assumes EO[0] andDO[0] were both set in previous step). This will cause SDMA to load the program RAM and channel contexts configured previously.
- Wait for Channel 0 to finish running. This is indicated by HI[0]=1 in the SDMA_SDMA_INTR register, or by optional interrupt to the Arm platform.
- Set the LOCK bit in the SDMA_SDMA_LOCK register to prevent un-authorized uploads of data to SDMA RAM.
- Additional channel scripts can now be run by enabling the selected software or hardware trigger event according to Runnable Channels Evaluation.

7.2.2.9 SDMA Programming Model

This section describes the programming model for the SDMA RISC engine, including its processor, memory, and internal control registers.

All addresses are related to the internal SDMA memory map, which is completely different from the Arm platform memory maps. The Arm platform processor has no access to any hardware resource described, except when those resources are described in Arm Platform Memory Map and Control Register Summary. .

7.2.2.9.1 State and Registers Per Channel

The SDMA can be seen as a set of 32 identical devices that are able to perform one data transfer channel each. Only one channel can work at a time, but every channel state is available at any time.

This chapter lists the components of every channel state.

7.2.2.9.2 General Purpose Registers

Each channel has eight general purpose registers of 32 bits for use by scripts. General register 0 has a dedicated function for the loop instruction, but otherwise can be used for any purpose.

7.2.2.9.3 Functional Unit State

Each channel context has some state that is part of the functional units.

The specific allocation of this state is part of the functional unit definition that is described in Burst DMA Unit Programming, Peripheral DMA Unit Programming .

This state must be saved/restored on context switches.

7.2.2.9.3.1 Program Counter Register (PC)

The PC is 14 bits. Since instructions are 16 bits in width and all memory in the SDMA is 32 bits in width, the low order bit of the PC selects which half of the 32-bit word contains the current instruction.

A low order bit of zero selects the most significant half of the word.[1]

7.2.2.9.3.2 Flags

Each channel has the following four flags:

- The T bit reflects the status of some arithmetic and test instructions. It is set when the result of an addition or a subtraction is zero and cleared otherwise. It is also the copy of the tested bits. Finally, it can also be set when the loop counter (GReg0) reaches zero. When the last instruction of the hardware loop is an operation that can modify the T flag, its effect on T is discarded and replaced by the GReg0 status.
- Two additional bits, SF and DF, are used to indicate error conditions resulting from loading data sources and storing to destinations, respectively. Access errors set these bits, and successful transactions clear them. They can also be cleared by specific instructions (CLRF and loop). The source fault (SF) is updated by the loads LD and LDF; the destination fault (DF) is updated by the stores ST and STF.

1. For example, big-Endian.

- Access errors are caused by several conditions including writing to the ROM, writing to a read-only memory mapped register, accessing an unmapped address, or any transfer error received by a peripheral when it is accessed.

  The SF and DF flags have a major impact on the behavior of the hardware loop: If SF or DF is set when starting a hardware loop and it is not masked by the loop instruction, the loop body will not be executed. Inside the loop body, if a load or store sets the corresponding SF or DF flag, the loop exits immediately. Testing the status of the T flag at the end of the loop (as well as testing both SF and DF) tells if the loop exited abnormally as any anticipated exit prevents GReg0 from reaching the zero value and thus setting the T flag. This is also valid if the fault occurs at the last instruction of the last loop.

- The last flag is the loop mode flag, LM, which is composed of two bits. The most significant bit indicates when the processor is currently operating in loop mode. It is set by the loop instruction and is cleared after execution of the last instruction of the last loop. The least significant bit is set when the program counter points to the last instruction of a loop on the last path. It is used for a channel that is restored with this configuration to know that the next program counter is EPC. As with the dynamic context switch Greg0, which indicates when the program must get out of the loop, it can be restored only on the last instruction of the loop. This, however, is too late to fetch the next instruction after the loop.

7.2.2.9.3.3 Return Program Counter (RPC)

The RPC is 14 bits. It is set by the jump to the subroutine instructions and used by the return from the subroutine instructions.

Instructions are available to transfer its contents to and from a general register.

7.2.2.9.3.4 Loop Mode Start Program Counter (SPC)

The SPC is 14 bits. It is set by the loop instruction to the location immediately following it.

7.2.2.9.3.5 Loop Mode End Program Counter (EPC)

The EPC is 14 bits. It is set by the loop instruction to the location of the next instruction after the loop.

7.2.2.9.4 Context Switching-Programming

Each channel has a separate context consisting of the eight general purpose registers and additional registers representing the state of the functional units.

The active registers and functional units contain the context of the active channel. The context of inactive channels is stored in SDMA RAM, which is part of the SDMA address space.

In a function of the selected context switching mode (Context Switching), modified registers by the program can be saved in the channel RAM space while the program is going on. In every cycle, a write access to the RAM is possible.

On a done or yield(ge) instruction, SDMA goes into "real" context switching. In one of the dynamic modes, modified registers not previously saved, as well as the PC-Loop registers, are stored into the context area of the channel that will be closed. The new PC-Loop registers are loaded from the context area of the new channel. All other registers are restored while the program is executed, giving priority to registers used by the decoded instruction. Therefore, in the best case, only the PC and Loop registers should be saved and restored during this context-switching phase, which only requires five SDMA cycles.

In static mode, the context switch stores all registers in the old channel RAM space, and restores all registers from the new channel RAM space. It requires 26 SDMA cycles.

The address of the context memory for channel *i* is *CONTEXT_BASE* + 24**i* or *CONTEXT_BASE* + *32*i w*here CONTEXT_BASE equals 0x0800. The table below presents the layout of a channel context in memory:

Table 7-13. Layout of a Channel Context in Memory for SDMA

OFFSET	31	30	29-16	15	14	13-0
0	SF	-	RPC	T	-	PC
1	LM		EPC	DF	-	SPC
2	GR0					
3	GR1					
4	GR2					
5	GR3					
6	GR4					
7	GR5					
8	GR6					
9	GR7					
10	MDA (burst DMA)					
11	MSA (burst DMA)					
12	MS (burst DMA)					
13	MD (burst DMA)					

Table continues on the next page...

Table 7-13. Layout of a Channel Context in Memory for SDMA (continued)

14	PDA (peripheral DMA)
15	PSA (peripheral DMA)
16	PS (peripheral DMA)
17	PD (peripheral DMA)
18	
19	
20	Reserved[1]
21	Reserved[1]
22	Reserved[1]
23	Reserved[1]
24	Scratch RAM (optional)
25	Scratch RAM (optional)
26	Scratch RAM (optional)
27	Scratch RAM (optional)
28	Scratch RAM (optional)
29	Scratch RAM (optional)
30	Scratch RAM (optional)
31	Scratch RAM (optional)

7.2.2.9.5 Address Space

The SDMA has four internal buses which are listed here.

- The Instruction bus reads instructions from the memory. Its address map is described in Instruction Memory Map.
- The Data bus (DMBUS) accesses the same memories as those visible on the Instruction bus, some memory-mapped registers (scheduler status and OnCE registers), and up to 14 peripherals. Its address map is described in Data Memory Map.
- The Functional Units bus (FUBUS) accesses the , Burst DMA, Peripheral DMA . The addressing mechanism is further detailed in Functional Units Programming Model.
- The Context Switch bus reads/writes registers into context-switch RAM space. It is a 64-bit bus dedicated for accessing this RAM space for updating the context of the running channel. While the program is going on, this bus has the lowest priority compared to the Instruction and Data buses, except for restoring a register needed for the decoded instruction to be executed. On the save part of a context switch (when the PCU is in its slave state), this is the only one used. On the restore part, the Instruction bus has the priority to read the next instruction at the restored PC and

otherwise the Context Switch bus is used. It is not possible to control the actual data transfers that occur on this bus.

7.2.2.9.5.1 Instruction Memory Map

The instruction memory map is based on a 14-bit address bus and a 16-bit data (instruction) bus.

Instructions are fetched from either program ROM or program RAM. An SDMA script is able to change the contents of the program RAM, which is also visible from the data bus.

The first two instruction locations (at 0 and 1) are special. Location 0 is where the PC is set on reset. Location 1 is where the PC is set upon the execution of an illegal instruction. It is expected that both of these locations will contain a jmp to handle routines.

Table 7-14. SDMA Instruction Memory Space

Device	SDMA Address (Hex)	Base Address Label	Block Name	WS	Description
ROM	0x0000 ↓ 0x07FF	SDMA_IBUS_ROM_ADDR	-	0	4 Kbyte internal ROM with boot code and standard routines.
RAM	0x1000 ↓ 0x1FFF	SDMA_IBUS_RAM_ADDR	-	0	8 Kbyte internal RAM with channels context and user data/routines.

7.2.2.9.5.2 Data Memory Map

All of the data accessible to SDMA scripts make up the data memory space of the SDMA.

This address space has several components:

- ROM (also visible on the Instruction bus)
- RAM (also visible on the Instruction bus)
- Shared Peripherals Registers
- SDMA Internal Registers (scheduler, OnCE, and registers that are also accessible by the Arm platform)

SDMA scripts can read and write to the context RAM, data RAM, shared peripheral registers, and internal registers.

The address range is 16 bits and the data width is 32 bits. When accessing peripheral registers (USB and so on), the data width may be different. The exact address map for the peripherals depends on the project (as presented in each respective chapter).

Data access is performed with *ld* and *st* instructions that take the address from a general purpose register in the core (GReg*n*). The mapping between the general purpose register contents and the address bus is given in the following table:

Table 7-15. GReg*n* to DMBUS Address Mapping

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
sz	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
address															

Grayed bits are simply discarded but they must be cleared to ensure forward-script compatibility.

- sz (bit 31) indicates the peripheral data width: 0 is used for a 32-bit peripheral and 1 is used for a 16-bit peripheral.
- address (bits 15 down to 0) is the address of the accessed resource (internal memory, internal register, or shared peripheral).

Table 7-16. SDMA Data Memory Space

Device	SDMA Address (Hex)	Size	Description
ROM	0x0000 → 0x03FF	4 Kbyte	4 Kbyte internal ROM with boot code and standard routines
Reserved	0x0400 → 0x07FF	4 Kbyte	4 Kbyte Reserved
RAM	0x0800 → 0x0FFF	8 Kbyte	8 Kbyte internal RAM with channels contexts and user data/routines
per1	0x1000 → 0x1FFF	16 Kbyte	*peripheral 1* memory space (4 Kbyte peripheral's address space)
per2	0x2000 → 0x2FFF	16 Kbyte	*peripheral 2* memory space (4 Kbyte peripheral's address space)
per3	0x3000 → 0x3FFF	16 Kbyte	*peripheral 3* memory space (4 Kbyte peripheral's address space)
per4	0x4000 → 0x4FFF	16 Kbyte	*peripheral 4* memory space (4 Kbyte peripheral's address space)
per5	0x5000 → 0x5FFF	16 Kbyte	*peripheral 5* memory space (4 Kbyte peripheral's address space)
per6	0x6000 → 0x6FFF	16 Kbyte	*peripheral 6* memory space (4 Kbyte peripheral's address space)
Registers	0x7000 → 0x7FFF	16 Kbyte	Memory mapped registers
per7	0x8000 → 0x8FFF	16 Kbyte	*peripheral 7* memory space (4 Kbyte peripheral's address space)
per8	0x9000 → 0x9FFF	16 Kbyte	*peripheral 8* memory space (4 Kbyte peripheral's address space)
per9	0xA000 → 0xAFFF	16 Kbyte	*peripheral 9* memory space (4 Kbyte peripheral's address space)
per10	0xB000 → 0xBFFF	16 Kbyte	*peripheral 10* memory space (4 Kbyte peripheral's address space)
per11	0xC000 → 0xCFFF	16 Kbyte	*peripheral 11* memory space (4 Kbyte peripheral's address space)
per12	0xD000 → 0xDFFF	16 Kbyte	*peripheral 12* memory space (4 Kbyte peripheral's address space)
per13	0xE000 → 0xEFFF	16 Kbyte	*peripheral 13* memory space (4 Kbyte peripheral's address space)
per14	0xF000 → 0xFFFF	16 Kbyte	*peripheral 14* memory space (4 Kbyte peripheral's address space)

7.2.2.10 SDMA Initialization

Appendix A describes the setup of the SDMA . This section provides a quick description of several initialization procedures.

NOTE

There may be differences with the actual implementation in the API.

7.2.2.10.1 Hardware Reset-SDMA

After reset, the RAM that holds contexts, data, scripts, and the DMA request-channels matrix has unpredictable content.

The core registers are all reset to 0, including the PC; the PCU state is *Sleep after Reset*. No channel can be activated because all of the priorities are also reset to 0.

7.2.2.10.2 Standard Boot Sequence

The following is the standard boot sequence:

1. Initialize the CONFIG register-detailed in Configuration Register (SDMAARM_CONFIG)-to determine the Arm platform DMA/core clock ratio (1 or 2)
2. Initialize the DMA request-channels matrix (seeChannel Enable RAM (SDMAARM_CHNENBL*n*)).
3. Program the channel control registers-Channel Event Override (SDMAARM_EVTOVR), Channel BP Override (SDMAARM_DSPOVR), Channel BP Override (SDMA_HOSTOVR), and Channel Event Pending (SDMAARM_EVTPEND)-according to the channel allocation.
4. Perform any necessary setup as required by the standard boot script in ROM (this is described in Appendix A).
5. Trigger channel 0 with the Channel Start (SDMAARM_HSTART) register, which starts the execution of the ROM script starting at address 0. This boot downloads channel scripts and contexts in RAM.

7.2.2.10.3 User-Defined Boot Sequence

The following is a user-defined boot sequence:

1. Initialize the Configuration Register (SDMAARM_CONFIG)Channel Enable RAM (SDMAARM_CHNENBL*n*), Channel Event Override (SDMAARM_EVTOVR), Channel BP Override (SDMAARM_DSPOVR), Channel Arm platform Override (SDMAARM_HOSTOVR), and Channel Event Pending (SDMAARM_EVTPEND).

2. Use the OnCE (either via its JTAG interface or its Arm platform control registers) to download any code in the SDMA RAM. Accessing the Memory describes how to write data to the RAM via the OnCE.
3. Use the OnCE instructions to make the PC default value point to the new boot script start address, or rely on the ROM startup script, which first jumps to the address in Channel 0 Boot Address (SDMAARM_CHN0ADDR). (This register default address points to the standard boot script.)

7.2.2.10.4 Script Loading and Context Initialization

The execution of an SDMA script depends on both the instructions that make up the script and the data context upon which it operates. Both must be initialized before the script is allowed to execute.

Each of the 32 channels has a separate data context, but may share scripts and locations in the data RAM.

The Arm platform manages the space in program RAM and data RAM. It also manages the assignment of SDMA channels to the device drivers that need them. Channels are initialized by the Arm platform via the channel 0 boot script. The boot channel downloads any required scripts with their data and the channels' initial contexts. Every context contains all the initial values of the registers, including the PC. Then the Arm platform can enable any channel that becomes active and begins fetching and executing instructions from its script.

7.2.2.11 Instruction Description

The following sections introduce the instruction of the SDMA.

Instruction set details are available in Instruction Set.

7.2.2.11.1 Scheduling Instructions

The following are scheduling instructions:

- done-The instruction causes certain scheduling or interrupt bits to be set or cleared, which may cause a change in the schedule-ability of the running channel. Then the instruction causes the SDMA to evaluate the current scheduling priorities and to choose the highest priority ready channel. If this channel is not the current channel, a context switch will take place. If there are no runnable channels, the SDMA will enter the stopped mode. The done 5 has a special usage reserved for debug, as explained in Debug Instructions.

- yield-These instructions are special cases of the done instruction. They do not modify the scheduling bits, but allow the highest pending channel (if it exists) to preempt the current channel if the pending channel priority is strictly greater than the current channel priority.
- yieldge-These instructions are special cases of the done instruction. They do not modify the scheduling bits, but allow the highest pending channel (if it exists) to preempt the current channel if the pending channel priority is strictly greater or equal to the current channel priority.
- notify-The notify instruction affects the scheduling bits, but does not cause rescheduling.

7.2.2.11.2 Conditional Branch Instructions

The conditional branch instructions of an 8-bit displacement, which is sign-extended and added to the current PC (which points to the next instruction) if the condition is satisfied.

Otherwise, control passes to the next sequential instruction.

- BF-Branch if False. The branch is taken if the T bit in the processor status is zero (false).
- BT-Branch if True. The branch is taken if the T bit in the processor status is one (true).
- BSF-Branch if Source Fault. The branch is taken if the SF bit in the processor status is one.
- BDF-Branch if Destination Fault. The branch is taken if the DF bit in the processor status is one.

7.2.2.11.3 Unconditional Jump Instructions

There are two varieties of unconditional control transfers: an absolute transfer and a through-register transfer.

Absolute transfers have a 14-bit address field that replaces the current PC.

- JMP-Jump. Causes the processor to jump to an absolute address encoded in the instruction itself.
- JSR-Jump to Subroutine. Causes the processor to jump to a subroutine, the address of which is encoded in the instruction itself.

- JMPR-Jump through Register. Causes the processor to jump to an absolute address contained in a General register. This instruction is meant to be used when more than one level of subroutines are required.
- JSRR-Jump to Subroutine through Register. Causes the processor to jump to a subroutine, the address of which is contained in a General register. This instruction is meant to be used when more than one level of subroutines are required.

7.2.2.11.4 Subroutine Return Instructions

The following are subroutine return instructions:

- RET-Return from Subroutine. The RET restores the contents of RPC to PC.
- LDRPC-Load from RPC to Register. THe LDRPC instruction is meant to be used when more than one level of subroutines are required. It stores the contents of RPC in any General register.

7.2.2.11.5 Loop Instruction

The following is a loop instruction:

LOOP-Enters Loop Mode. Before entering loop mode, the loop instruction can optionally clear the fault flags (SF and/or DF) based on a 2-bit field in the instruction. This feature is linked to the fact that setting SF or DF in loop mode will cause an immediate exit of the loop.

7.2.2.11.6 Miscellaneous Instructions

The following are miscellaneous instructions:

- CLRF-Clear Fault Flags. This instruction clears any combination of SF and DF.
- MOV r,s-This moves data from GReg[s] to GReg[r].
- LDI r,immediate-This loads GReg[r] with a zero-extended immediate value.

7.2.2.11.7 Logic Instructions

The following are logic instructions:

- XORr,s-This performs an exclusive or between GReg[r] and GReg[s], and stores the result in GReg[r].
- XORIr,immediate-This performs an exclusive or between GReg[r] and a zero-extended immediate value, and stores the result in GReg[r].
- ORr,s-This performs an or between GReg[r] and GReg[s], and stores the result in GReg[r].

- ORIr,immediate-This performs an or between GReg[r] and a zero-extended immediate value and, stores the result in GReg[r].
- ANDNr,s-This performs an and between GReg[r] and the negated GReg[s], and stores the result in GReg[r].
- ANDNIr,immediate-This performs an and between GReg[r] and the negated zero-extended immediate value, and stores the result in GReg[r].
- ANDr,s-This performs an and between GReg[r] and GReg[s], and stores the result in GReg[r].
- ANDIr,immediate-This performs an and between GReg[r] and a zero-extended immediate value, and stores the result in GReg[r].

7.2.2.11.8 Arithmetic Instructions

Arithmetic instructions modify the T bit in the processor status according to the result of the operation. The T bit is set if the result is zero, otherwise it is cleared.

- ADD r,s-This performs the addition of GReg[r] and GReg[s], and stores the result in GReg[r].
- ADDI r,immediate-This performs the addition of GReg[r] and a zero-extended immediate value, and stores the result in GReg[r].
- SUB r,s-This performs the subtraction of GReg[s] from GReg[r], and stores the result in GReg[r].
- SUBIr,immediate-This performs the subtraction of a zero-extended immediate value from GReg[r], and stores the result in GReg[r].

7.2.2.11.9 Compare Instructions

Compare instructions modify the T bit in the processor status according to the result of the operation. The T bit is set if the comparison is true, otherwise it is cleared.

NOTE

Only one version of the immediate form is implemented. Non-equality comparisons to immediate values will require two instructions.

- CMPEQ r,s-This sets T when registers GReg[r] and GReg[s] are equal.
- CMPEQIr,immediate-This sets T when register GReg[r] and the zero-extended immediate value are equal.
- CMPLTr,s-This sets T when register GReg[r] is less than and not equal to GReg[s]. The comparison is signed.
- CMPHS r,s-This sets T when register GReg[r] is greater than or equal to GReg[s]. The comparison is signed.

7.2.2.11.10 Test Instructions

Test instructions modify the T bit in the processor status according to the result of the operation. The T bit is set if any bit in the result is one, otherwise it is cleared.

- TSTr,s-This performs an and between GReg[r] and GReg[s], and sets T if the result is not zero.
- TSTIr,immediate-This performs an and between GReg[r] and a zero-extended immediate value, and sets T if the result is not zero.

7.2.2.11.11 Byte Permutation Instructions

These instructions shuffle the bytes in a register. For the purpose of describing these instructions, have the bytes in a register be numbered from the most significant as b_3, b_2, b_1, b_0.

- RORBr-The rotate right byte. The result is b_0, b_3, b_2, b_1.
- REVBr-The reverse bytes in word. The result is b_0, b_1, b_2, b_3.
- REVBLOr-The reverse, two low-order bytes. The result is b_3, b_2, b_0, b_1.

7.2.2.11.12 Bit Shift Instructions

The following are bit shift instructions:

- ROR1r-The rotate right 1 bit. This instruction does a circular right shift of 1 bit.
- LSR1r-The logical shift right 1 bit. This instruction shifts all bits to the right by 1. The high order bit is replaced by a 0.
- ASR1r-The arithmetic shift right 1 bit. This instruction shifts all bits to the right by 1. The high order bit is replaced by itself.
- LSL1r-The logical shift left 1 bit. This instruction shifts all bits to the left by 1. The low order bit is replaced by zero.

7.2.2.11.13 Bit Manipulation Instructions

- BCLRIr,n-The bit clear is immediate; clears bit number *i* in register *r*.
- BSETIr,n-The bit set is immediate; sets bit number *i* in register *r*.
- BTSTIr,n-The bit test is immediate; tests bit number *i* in register *r* (T becomes equal to the selected register bit).

7.2.2.11.14 SDMA Memory Access Instructions

All memory accesses are 32 bits.

Any memory location that is implemented with less than 32 bits (for example, peripheral registers) causes unimplemented bits to be read as 0s.

All memory accesses will cause either the SF or DF flags in the processor status to be set if they cause a fault.

What constitutes a fault, especially when accessing peripheral registers, is a property of the memory location.

- LDr,(b,d)-The load instruction creates an address by adding the displacement field (d) to the contents of the base register (b). The SDMA location at the resulting address is read and placed in the destination register (r).
- STr,(b,d)-The store instruction creates an address in the same manner as the load instruction. The register (r) is stored in the SDMA location at the resulting address.

7.2.2.11.15 Functional Unit Instructions

The functional unit instructions have an 8-bit field that is placed on the functional unit bus.

Some of these bits are used to select which functional unit should be involved in the transfer. The remaining bits are decoded by the selected functional unit so their specific use depends on the functional unit. See Functional Units Programming Model.

There are two functional unit instructions, as follows:

- LDFr,fub-The 8-bit field is placed on the functional unit bus and a read is issued to the selected functional unit. As a result of this instruction, the SF may be set in the processor status.
- STFr,fub-The 8-bit field is placed on the functional unit bus and a write is issued to the selected functional unit. As a result of this instruction, the DF may be set in the processor status.

7.2.2.11.16 Illegal Instructions

All instruction encodings that are illegal cause the following actions:

- The current PC (which points to one beyond the offending instruction) is put in the EPC register.
- The loop mode bit is cleared.
- The PC is set to the value stored in the Illegal Instruction Trap Address (SDMAARM_ILLINSTADDR) register (the default value is 0x0001).

ILLEGAL-Although any instruction other than those indicated in the SDMA specification will trigger the illegal instruction mechanism, the ILLEGAL instruction code is preferred as it will always be kept as *illegal* in the possible future versions of the SDMA core.

7.2.2.11.17 Debug Instructions

The following are debug instructions:

- SOFTBKPT-The software breakpoint instruction causes the core to stop and enter debug mode. The core can then be accessed and started by the OnCE debug block only.
- done 5-This instruction is used for debugging, as it copies the contents of the PCU registers and flags to the context memory. Information on this instruction is described in Saving the Context.
- CpShReg-This instruction copies the context memory into the PCU registers and flags. Modifying the corresponding memory location before executing this instruction enables you to have the channel continue from a new instruction address. This instruction is described in Restoring the Context.

7.2.2.12 Functional Units Programming Model

The functional unit instructions cause an 8-bit code, found in the low eight bits of the instruction, to be asserted on the functional unit control bus.

Some of these bits are used to select one of several functional units. Functional units which can be selected include SDMA registers such as MSA and MSD which are not mapped in the SDMA memory map, and are accessible only through the functional unit bus. These Functional Unit Registers are listed in the following table. In order to establish a programming convention, assume the selection bits are some number of the most significant bits of the 8-bit code. Furthermore, some number of the least significant bits is decoded by a given functional unit to establish the type of operation to perform.

Table 7-17. Functional Unit Registers

Functional Unit	Register	Register Name	Section/Page
Burst DMA Unit Programming	SDMSA	Memory Source Address Register	Memory Source Address Register (MSA)
	MDA	Memory Destination Address Register	Memory Destination Address Register (MDA)
	MD	Memory Data Buffer Register	Memory Data Buffer Register (MD)

Table continues on the next page...

Table 7-17. Functional Unit Registers (continued)

Functional Unit	Register	Register Name	Section/Page
			(Write) Burst DMA Write (stf) (Read) Burst DMA Read (ldf)
	MS	Memory State Register	State Register (MS)
Peripheral DMA Unit Programming	PSA	Peripheral Source Address Register	Peripheral Source Address Register (PSA)
	PDA	Peripheral Destination Address Register	Peripheral Destination Address Register (PDA)
	PD	Peripheral Data Buffer Register	Peripheral Data Register (PD) (Write) Peripheral DMA Write (stf)-Write Mode (Read) Peripheral DMA Read (ldf)-Read Mode
	PS	Peripheral State Register	Peripheral State Register (PS)

More information regarding the functional units can be found in Peripheral DMA Unit, and Burst DMA Unit.

7.2.2.12.1 Burst DMA Unit Programming

The DMA instructions control the DMA state machine and may cause a DMA cycle on the associated memory bus.

There are four registers associated with the burst DMA unit: a Memory Source Address register (MSA), a Memory Destination Address register (MDA), a Memory Data buffer (MD), and a state register (MS). The burst DMA has two different uses:

- A data transfer between External Memory Interface and SDMA general register
- A data transfer in copy mode where blocks of data are transferred from the source address to the destination address

7.2.2.12.1.1 Memory Source Address Register (MSA)

The source address register contains the pointer into EXTMC memory associated with the next read data transfer. It has byte granularity.

Reading the register with the ldf instruction has no side effects, and gives the address value in the EXTMC memory of the next data that is read by the SDMA during an ldf MD instruction.

Writing the source address register has two side effects: If the prefetch bit is set, a DMA read cycle (8-word read access) is issued with the new address. Any data still located in the buffer is lost. If there is valid write data in the buffer, it is necessary to force the DMA to completely flush it out before modifying MSA to guarantee all the data is effectively written to memory.

The MSA register has two modes of programming:

- Frozen-In frozen mode, the MSA register is not modified after DMA accesses.
- Incremented (default mode)-In incremental mode, MSA is incremented by the number of bytes transferred during read cycles.

7.2.2.12.1.2 Memory Destination Address Register (MDA)

The destination address register contains the pointer into EXTMC memory associated with the next write data transfer. It has byte granularity.

Reading the MDA register with the ldf instruction has no side effects. It gives the address value in the EXTMC memory where the next SDMA data (stf r,MD instruction) is stored when MD FIFO is flushed.

Writing the destination address register has one side effect. Any data still located in the buffer is lost. If there is valid write data in the buffer, it is necessary to force the DMA to completely flush it out before modifying MDA to guarantee all the data is effectively written to memory.

The MDA register has two modes of programming:

- Frozen-In frozen mode, the MDA register is not modified after DMA accesses.
- Incremented (default mode)-The MDA register is incremented by the number of bytes transferred during write cycles.

7.2.2.12.1.3 Memory Data Buffer Register (MD)

The data buffer register consists of a bank of 36 bytes that behave like FIFO.

This FIFO stores the eight words received when a read burst is triggered by the DMA (DMA is in read mode).

The MD register is in write mode after a writing in MDA or after an stf MD instruction.

In that case, a burst write access is automatically triggered when there are more than eight words in MD. For bandwidth optimization, any transfers between DMA and the EXTMC controller are based on burst accesses.

An ldf r,MD|SIZE instruction that reads the data buffer may cause a DMA cycle, as follows:

- If there are less bytes in the FIFO than the size parameter of the instruction. For instance, if only two bytes are available in MD and a 4-byte read is requested, a burst read access is executed to complete the two bytes.
- If the prefetch bit is set, and after reading there is enough space in the FIFO to store a full burst, a burst read access is triggered.

An stf r,MD|SIZE instruction that writes to the data buffer may cause a DMA cycle if the number of written bytes in MD is higher than 32 (eight words) or if the flush bit is set.

When DMA is used for data transfer between SDMA and EXTMC (reading or writing), no immediate error is possible because the block manages a data misalignment issue; therefore, it is allowed to read/write a word to/from a half-word address. However, the addresses (source or destination) must belong to the EXTMC memory mapping. The only potential error, in this mode, would be the error sent back by the EXTMC controller when an access to a super-user page is detected. The whole transfer on the DMA associated bus will be considered successful when there are no errors seen on the bus during the transfer. In copy mode, an immediate error could be returned to SDMA as described in Burst DMA Unit Error Management.

7.2.2.12.1.4 State Register (MS)

The state register contains the DMA state-machine value. It can be accessed in case of an error received during a transfer. MS is also accessed to set-up the conditional yielding feature.

The initialization value of this register is 0 and it consists of the following:

Table 7-18. SDMA_MS Structure

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	spriv	stype	0	0	dpriv	dtype
W																
R	0	0	0	0	y	d	e		0	0	n					
W																

Table 7-19. SDMA_MS Field Descriptions

Field	Description
31-22	Reserved

Table continues on the next page...

Table 7-19. SDMA_MS Field Descriptions (continued)

Field	Description
21 spriv	The spriv value is ignored for this device. 0 = valid value 1 = Reserved
20 stype	Source Mode. Indicates if MSA has to be incremented (or not) during accesses. 0 Frozen-MSA is not modified. 1 Incremented-MSA is incremented by the number of transferred bytes during read access.
19-18	Reserved
17 dpriv	The dpriv value is ignored for this device. 0 = valid value 1 = Reserved
16 dtype	Destination Mode. Indicates if MDA has to be incremented (or not) during accesses. 0 Frozen-MDA is not modified. 1 Incremented-MDA is incremented by the number of transferred bytes during write access.
15-12	Reserved
11 y	Conditional Yielding selector. When selected, theyield/yieldge instructions will not switch channels if the Burst DMA is in Write Mode, and it has less than four bytes in its FIFO. This is aimed at reducing the number of inefficient FIFO flushes due to context switches. 0 Always yields 1 Yields conditionally (when there are less than four bytes in the FIFO in write mode)
10 d	Access Direction or DMA Mode. DMA is in write mode when data was written into MD by stf MD instructions, or if a previous DMA cycle on the external bus was a write access. Writing MDA or MSA changes the DMA mode to the respective value. DMA is in read mode when a previous DMA cycle was a read access, and DMA stays in read mode when data is read by SDMA with an ldf MD instruction. Reading MDA or MSA does not change the DMA mode. 0 Read Mode 1 Write Mode
9-8 e	Error. Indicates if the previous access was acknowledged with a bus error. 00 No error was received. 01 *reserved* 10 Error mode 11 error read burst
7-6	Reserved
5-0 n	Number of bytes in the MD FIFO.

7.2.2.12.1.5 Burst DMA Write (stf)

When received from a stf instruction, the function code bits are interpreted as follows, depending on the addressed register:

Table 7-20. STF Code Bits

Register	7	6	5	4	3	2	1	0
MSA	s		p	freeze	r			spriv
MDA								dpriv
MD			f	cpy			sz	
MS								

Table 7-21. STF Code Bit Field Descriptions

Field	Description
7-6 s	Functional Unit selector 00 for Burst DMA
5 p (MSA)	Prefetch Flag 0 No prefetch 1 Prefetch required from new MSA
5 f (MD)	Forced Flush Flag 0 Automatic flush 1 FIFO contents are flushed (including the new written data).
4 freeze (MSA/MDA)	Address Freeze Mode 0 Address is normally incremented. 1 Address is frozen.
4 cpy (MD)	Copy Mode selection 0 Write Mode 1 Copy Mode
3-2 r	Register selection 00 MSA 01 MDA 10 MD 11 MS
1-0 sz (MD/MS)	Transfer Size 00 size 0 (no data stored in the FIFO) 01 byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)
0 spriv (MSA)	The spriv value is ignored for this device. 0 = valid value

Table continues on the next page...

Table 7-21. STF Code Bit Field Descriptions (continued)

Field	Description
	1 = Reserved
0 dpriv (MDA)	The dpriv value is ignored for this device. 0 = valid value 1 = Reserved

The possible write instructions are listed in the table below (unused bits should always be cleared).

Table 7-22. Burst DMA STF Instruction List

Binary	Assembly	Comments
00_0_0_00_00	stf r,MSA	Writes content of the SDMA general register (r) to the source address register. MSA is in incremented mode.
00_0_1_00_00	stf r,MSA|FR	Writes content of the SDMA general register (r) to the source address register. MSA is in frozen mode.
00_1_0_00_00	stf r,MSA|PF	Writes content of the SDMA general register (r) to the source address register, and starts a read burst access. MSA is in incremented mode.
00_1_1_00_00	stf r,MSA|PF|FR	Writes content of the SDMA general register (r) to the source address register, and starts a read burst access.
00_0_0_01_00	stf r,MDA	Writes content of the SDMA general register (r) to the destination address register. MDA is in incremented mode.
00_0_1_01_00	stf r,MDA|FR	Writes content of the SDMA general register (r) to the destination address register. MDA is in frozen mode.
00_1_0_10_00	stf r,MD|SZ0|FL	No data transfers between the SDMA and MD, but all valid written data of the MD is flushed to the memory. An acknowledge or error is sent back to the SDMA core on transfer completion.
00_0_0_10_01	stf r,MD|SZ8	8-bit (byte) transfer to write buffer MD
00_1_0_10_01	stf r,MD|SZ8|FL	8-bit (byte) transfer to write buffer MD and flush after transfer. All valid written data of the MD is flushed to memory.
00_0_0_10_10	stf r,MD|SZ16	16-bit (half-word) transfer to write buffer MD
00_1_0_10_10	stf r,MD|SZ16|FL	16-bit (half-word) transfer to write buffer MD and flush after transfer. All valid written data of the MD is flushed to memory.
00_0_0_10_11	stf r,MD|SZ32	32-bit (word) transfer to write buffer MD
00_1_0_10_11	stf r,MD|SZ32|FL	32-bit (word) transfer to write buffer MD and flush after transfer. All valid written data of MD is flushed to memory.
00_0_1_10_00	stf r,MD|CPY	No data transfer between SDMA and MD but starts a copy transfer whose length is given by the 4 LSB of *r* register. (Maximum burst length is eight words.)
00_0_0_11_11	stf r,MS	32-bit (word) transfer to status register MS
00_0_0_11_00	stf r,MS|SZ0	Clears the error flag (if set). Other MS bits are unchanged; this instruction is also known as clref MS.

NOTE

When a flush bit is set, the SDMA flushes the FIFO including the newly written data. An acknowledge is sent to the core before the flush completes (except if size 0 is used). The goal of this flush bit is to force a flush, but it is recommended to use it only when needed (for example, when finishing a row of pixels during 2D data transfers). Indeed, if this bit is omitted and if there are more than 32 bytes in the FIFO, a burst write access is automatically triggered.

Since all the stf r,MD instructions (including the copy mode) acknowledge the SDMA core before the store is effective (except if size 0 is used), it is recommended to perform an ldf from MS before terminating a channel in order to check the final error status. (The ldf from MS will stall the core until all the data was flushed out and the transfer status is known.)

After every stf MD instruction, the MDA is incremented by the number of bytes that are written in MD, except when it is programmed in frozen mode.

7.2.2.12.1.6 Burst DMA Read (ldf)

When received from an ldf instruction, the function code bits are interpreted as follows, depending on the addressed register:

Table 7-23. LDF Code Bits

Register	7	6	5	4	3	2	1	0
MSA	s				r			
MDA								
MD			p				sz	
MS								

Table 7-24. LDF Code Bit Field Descriptions

Field	Description
7-6 s	Functional Unit selector 00 for Burst DMA
5 p (MD)	Prefetch Flag 0 no prefetch 1 automatic prefetch

Table continues on the next page...

Table 7-24. LDF Code Bit Field Descriptions (continued)

Field	Description
3-2 r	Register selection 00 MSA 01 MDA 10 MD 11 MS
1-0 sz (MD)	Transfer Size 00 reserved 01 byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)

The table below lists the possible write instructions (unused bits should always be cleared).

Table 7-25. Burst DMA LDF Instruction List

Binary	Assembly	Comments
00_0_0_00_00	ldf r,MSA	Copies the source address register value into an SDMA general register. It gives the memory address of the next data that will be read with an ldf MD instruction.
00_0_0_01_00	ldf r,MDA	Copies the destination address register value into an SDMA general register. It gives the memory address where the next incoming data will be flushed.
00_0_0_10_01	ldf r,MD|SZ8	8-bit (byte) read
00_1_0_10_01	ldf r,MD|SZ8|PF	8-bit (byte) read. If after this reading and the MD FIFO is empty, a burst read access at the MSA address is triggered.
00_0_0_10_10	ldf r,MD|SZ16	16-bit (half-word) read
00_1_0_10_10	ldf r,MD|SZ16|PF	16-bit (half-word) read. If after this reading, and the MD FIFO is empty, a burst read access at the MSA address is triggered.
00_0_0_10_11	ldf r,MD|SZ32	32-bit (word) read
00_1_0_10_11	ldf r,MD|SZ32|PF	32-bit (word) read. If after this reading and the MD FIFO is empty, a burst read access at the MSA address is triggered.
00_0_0_11_00	ldf r,MS	Copy the status register value into an SDMA general register.

NOTE

Read data is 0-extended before writing in the SDMA general registers. When reading the MD register, the DMA takes data from the FIFO if it is available. If part or whole data is not in the FIFO, an external burst read access is performed to provide the missing data. The SDMA is stalled as long as the required read data is not complete.

After every reading, MSA is incremented by the number of read bytes from MD FIFO, except when MSA is programmed in frozen mode.

7.2.2.12.1.7 Prefetch/Flush and Auto-Flush Management-Burst DMA Unit

The prefetch and auto-flush management enables the SDMA RISC machine to go on while a DMA access is performed.

When the RISC core requires a prefetch (p = 1) to the Burst DMA, it will receive an immediate transfer acknowledge before the DMA has finished the external access. This enables the RISC core to do other things like accessing another DMA machine.

The basic principle in prefetch mode is for the DMA to anticipate data reads from the SDMA RISC engine by fetching external bursts of data as soon as there is enough space in the DMA FIFO to store it. If ever the RISC engine required data that is not available in the FIFO, the read acknowledge is delayed until the data is available, but it does not have to wait until the burst completes.

The auto-flush basic principle is similar: An automatic flush is triggered every time there are eight words to be written in the FIFO. If the FIFO is full and the RISC engine requires another write, it is stalled until the burst has started and enough space was freed in the FIFO to store that new data. This means the SDMA RISC engine does not have to wait for the completion of a burst to receive its acknowledge and continue its processing.

In particular, an auto-flush is executed when DMA is in write mode and if the following is true:

- If the FIFO is empty and the first write is to a word-aligned address of any size (ex: the 2 LSB of MDA[1:0]= 0x0), the auto-flush is triggered immediately after the write of the 32'nd byte.
- If the FIFO is empty, and if MDA is an odd byte address (1, 3, 5, 7,...) and an stf MD|SZ8 is executed, the byte is flushed to memory. Once MDA increments to a word aligned address, the auto-flush will be triggered every 32 bytes.
- If the FIFO is empty, and if MDA is a half-word address (2, 6, 0xA,...) and an stf MD|SZ16 is executed, the two bytes of the incoming data are flushed to memory. Once MDA increments to a word aligned address, the auto-flush will be triggered every 32 bytes.
- If the FIFO is empty, and if MDA is not a word-aligned address (ex 1, 2, 3, 5, 6, 7, 9,...), and an stf MD|SZ32 is executed, the first 1 to 3 bytes will be flushed up to the next word aligned address. Afterwards, an auto-flush will be triggered each time the FIFO receives 32-bytes.

- Therefore, if an stf MD|SZ32 is executed with MDA equal to 0x1 and with an empty MD FIFO, the bytes located at addresses 1, 2, and 3 are flushed, and the byte located at address 4 remains in MD FIFO. This solves the misalignment issue. Additionally, the next write instructions (stf) complete the FIFO until it contains eight words; then a burst write is executed by the DMA to empty the FIFO. Protocol on the external bus does not support bursts of different data types (byte, half-word, or word).

  For example, consider the case where data is written using a byte access, stf MD| SZ8. The value of MDA during the very first byte write determines when the auto-flush will occur as follows:

- If MDA=0x0, the flush occurs following the write of byte 32
- If MDA=0x1, the flush occurs following the write of byte 1, byte 3 and byte 35.
- If MDA=0x2, the flush occurs following the write of byte 2 and byte 34.
- If MDA=0x3, the flush occurs following the write of byte 1 and byte 33.
- If MDA=0x4, the flush occurs following the write of byte 32

The flush command forces the DMA to flush all MD valid bytes to the EXTMC controller. An acknowledge is sent immediately to the SDMA, and any potential error is reported on a future access. It is thus essential to conclude a transfer with a last read from MS, which will stall the core until all data was flushed out and returned to the transfer status (acknowledge or error).

NOTE

During this kind of auto-flush (which occurs only at the beginning of a misaligned write transfer) no acknowledge is sent back to the SDMA, which is stalled until a flush is completed.

7.2.2.12.1.8 Data Alignment and Endianness-Burst DMA Unit

7.2.2.12.1.8.1 Burst DMA in Read Mode

For every read access to MD, the data returned to the SDMA core and the new FIFO state depends on the MSA status and the access size.

The FIFO is considered as a stack of 36 bytes: Data is fetched externally on a 32-bit bus, but the valid bytes only are stored in the FIFO and left-aligned (for a transfer of consecutive words, it is only the first word that may be truncated). The following table shows the FIFO byte alignment strategy and the corresponding MSA, the returned data, and the new FIFO state for any access size of an internal read from MD.

Table 7-26. FIFO Read Configuration

Before read		Internal read access size	Read data	After read	
MSA[1:0]	FIFO state			MSA[1:0]	FIFO state
00	x0 x1 x2 x3 y0 y1 y2 y3 z0 z1 z2 z3 and so on...	sz8	00 00 00 x0	01	x1 x2 x3 y0 y1 y2 y3 z0
		sz16	00 00 x0 x1	10	x2 x3 y0 y1 y2 y3 z0 z1
		sz32	x0 x1 x2 x3	00	y0 y1 y2 y3 z0 z1 z2 z3
01	x1 x2 x3 y0 y1 y2 y3 z0 z1 z2 z3 t0 and so on...	sz8	00 00 00 x1	10	x2 x3 y0 y1 y2 y3 z0 z1
		sz16	00 00 x1 x2	11	x3 y0 y1 y2 y3 z0 z1 z2
		sz32	x1 x2 x3 y0	01	y1 y2 y3 z0 z1 z2 z3 t0
10	x2 x3 y0 y1 y2 y3 z0 z1 z2 z3 t0 t1 and so on...	sz8	00 00 00 x2	11	x3 y0 y1 y2 y3 z0 z1 z2
		sz16	00 00 x2 x3	00	y0 y1 y2 y3 z0 z1 z2 z3
		sz32	x2 x3 y0 y1	10	y2 y3 z0 z1 z2 z3 t0 t1
11	x3 y0 y1 y2 y3 z0 z1 z2 z3 t0 t1 t2 and so on...	sz8	00 00 00 x3	00	y0 y1 y2 y3 z0 z1 z2 z3
		sz16	00 00 x3 y0	01	y1 y2 y3 z0 z1 z2 z3 t0
		sz32	x3 y0 y1 y2	11	y3 z0 z1 z2 z3 t0 t1 t2

7.2.2.12.1.8.2 Burst DMA in Write Mode

For every write access to the MD, the new FIFO state depends on the MDA status and the access size.

The FIFO is considered as a stack of 36 bytes: Data is stored in the FIFO according to the internal access size and the former MDA value. The following table shows the FIFO byte alignment strategy corresponding to MDA, as well as the new FIFO state for any access size of an internal write to MD.

Table 7-27. FIFO Write Configuration

Before write		Internal write access size	Written data	After write	
MDA[1:0]	FIFO state			MDA[1:0]	FIFO state
00	tt uu vv ww ?? ?? ?? ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	01	tt uu vv ww x0 ?? ?? ?? ?? ?? ?? ??
		sz16	?? ?? x0 x1	10	tt uu vv ww x0 x1 ?? ?? ?? ?? ?? ??
		sz32	x0 x1 x2 x3	00	tt uu vv ww x0 x1 x2 x3 ?? ?? ?? ??
01	tt uu vv ww xx ?? ?? ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	10	tt uu vv ww xx x0 ?? ?? ?? ?? ?? ??
		sz16	?? ?? x0 x1	11	tt uu vv ww xx x0 x1 ?? ?? ?? ?? ??
		sz32	x0 x1 x2 x3	01	tt uu vv ww xx x0 x1 x2 x3 ?? ?? ??
10	tt uu vv ww xx yy ?? ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	11	tt uu vv ww xx yy x0 ?? ?? ?? ?? ??
		sz16	?? ?? x0 x1	00	tt uu vv ww xx yy x0 x1 ?? ?? ?? ??
		sz32	x0 x1 x2 x3	10	tt uu vv ww xx yy x0 x1 x2 x3 ?? ??
11	tt uu vv ww xx yy zz ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	00	tt uu vv ww xx yy zz x0 ?? ?? ?? ??
		sz16	?? ?? x0 x1	01	tt uu vv ww xx yy zz x0

Table continues on the next page...

Table 7-27. FIFO Write Configuration (continued)

Before write		Internal write access size	Written data	After write	
MDA[1:0]	FIFO state			MDA[1:0]	FIFO state
					x1 ?? ?? ??
		sz32	x0 x1 x2 x3	11	tt uu vv ww xx yy zz x0 x1 x2 x3 ??

NOTE

If the FIFO mode changes from a write to a read mode, all remaining written bytes in MD are lost but no error is returned. Typically, this happens if an ldf MD is executed after stf MD instructions. Before a mode change, it is recommended to force the flush of a potential remaining byte by a stfMD|SZ0|FL instruction. In the same way, if a FIFO mode changes from a read to a write mode, all prefetched data present in the FIFO is lost and no error is returned.

7.2.2.12.1.8.3 Endianness-Burst DMA Unit

Big and Little Endian are supported by the Burst DMA, but data is always stored in MD in Big Endian.

Byte manipulation is performed when data is exchanged with an Burst controller (for example, during read or write burst accesses).

7.2.2.12.1.9 Burst DMA Unit Copy Mode

A mechanism is available to perform fast Arm-to-Arm transfers.

Data does not flow through the SDMA core: It is kept in the DMA FIFO. This mechanism is selected when writing MD with a special option in the instruction code (copy flag).

It is possible to transfer up to eight words in one SDMA instruction (this does not mean in one cycle). In this mode, every time an stf MD|CPY is executed, a read burst is executed and directly followed by a write burst transfer. Burst transfers are limited to eight words. The size of the transfer (in words)-given by the SDMA general register (4 LSB)-is also limited to eight. The following SDMA code shows how 100 bytes could be copied from the MSA address to the MDA address. This is sample code only.

Burst DMA copy mode example

```
          ldi r0,@src
          stf r0,MSA                          // Source address setup
          ldi r1,@dst
          stf r1,MSA                          // Destination address setup
          ldi r0,0x64                         // data transfer counter
          ldi r1,0x8
MAIN_XFER:
          cmphs r0,r1                         // Is r0 >= 0x8
          bf LAST_XFER                      // If not, jump to last transfer label
          stf r1,MD|CPY                        // Copy 8 words from MSA to MDA address.
          subi r0,0x8                         // Decrement counter
          jmp MAIN_XFER                     // return to main transfer loop
LAST_XFER:
          stf r0,MD|CPY
```

The main transfer loop is executed 12 times; then r0 equals 4 and the last transfer loop is run.

In this mode, an acknowledge is transmitted to the core as soon as the read burst can start; thus, a first copy instruction returns an immediate acknowledge and subsequent copy instructions will be acknowledged as soon as the previous copy has finished.

7.2.2.12.1.10 Burst DMA Unit Error Management

Another point to consider is the management of errors.

Because the DMA immediately sends an acknowledge to the RISC core (except for the stf MS|SZ0|FLS instruction), it assumes no error will occur. If an error occurs, it is flagged (transfer error acknowledge) for the following DMA access.

This should not be a problem if the DMA is used properly. The MD accesses are meant to stall the SDMA as little as possible to optimize throughput and hide calculation time. Therefore, final access to MS should be performed before closing a channel. This access waits until any pending operation is finished in the burst DMA and gather any remaining error.

In copy mode, an error could be immediately returned to the SDMA on execution of the ldf copy or stf copy instruction. It happens when MSA or MDA are not word addresses (for example, 0[4]). This is because copy mode must only be used for transferring a large packet of aligned data.

When an error is received during a *read* transfer to the external bus, which may occur during the burst accesses, the MD FIFO contains the valid beats of the burst, and the error flag of MS is set to 2'b11 (error read burst). It is possible to read MS ("n" field) to know how much valid data remains in MD and when MD is empty (after ldf instructions). The next read MD instruction sets the MS error flag to 2'b10 (error mode), and an error is sent back to the SDMA core. In error mode, it is possible to read MSA, which gives the address of the error data. Any attempt to read or write MD, or to modify MDA or MSA in error mode, gives rise to an error; therefore, an error flag must be reset by clearing MS at the end of the SDMA code section responsible for error management.

In "error read burst" mode, writing MDA, MSA, or MD, or starting a copy transfer by a stf MD|COPY instruction will cancel the error mode. The following table shows when an immediate error is sent back according to the executed instruction.

Table 7-28. Possibilities in ERROR READ BURST Mode

DMA Instruction	Immediate Error	Comments
stf rn, MD stf rn, MSA (|U |PF) stf rn, MDA stf rn,MD|COPY	NO	Error mode is reset. MSA, MDA, or MD are updated and a DMA cycle may start. For the stf MD|COPY, a copy loop is executed.
stf rn, MS	NO	MS is updated.
ldf rn, MS ldf rn, MSA ldf rn, MDA	NO	MS, MSA, and MDA could be read in ERROR READ mode without any side effects (for example, no DMA cycle is triggered).
ldf rn, MD	YES/NO	Immediate error if there is no more data available for read in the FIFO.

When an error is received during a *write* transfer, the error is reported to the next DMA access. In this case, an error is sent to the SDMA core and the DMA goes to its error mode. Reading MS gives the number of bytes that remain in MD; reading MDA gives the address of the error data. Any attempt to read or write MD, or to modify MDA or MSA in error mode, give rise to an error; therefore, an error flag must be reset by clearing MS at the end of the SDMA code section responsible for error management.

Table 7-29. Possibilities in ERROR Mode

DMA Instruction	Immediate Error	Comments
stf rn, MD stf rn, MSA stf rn, MDA	Yes	Any attempt to modify MD, MSA, MDA will raise an immediate error and burst DMA remains in error mode. When address registers are write-accessed, an error is returned.
stf rn, MS	No	This is the only way to exit error mode. MS[9:8] must be reset by an stf MS|SZ0 instruction.
ldf rn, MS ldf rn, MSA ldf rn, MDA	No	MS, MSA, and MDA could be read in error mode without any side effects (for example, no DMA cycle is triggered).
ldf rn, MD	Yes	Whatever the DMA direction (read or write), an ldf rn triggers an immediate error.

7.2.2.12.1.11 Conditional Yielding-Burst DMA Unit

The standard SDMA transfer is based upon a hardware loop that has the following structure:

Hardware Loop

```
loop
load Rn,source            // can be ldf or ld
<computation>            // can be done through functional units
store Rn,dest              // can be st or stf
done 0               // yield
```

This structure needs to be kept independent of the functional units' particularities regarding the context switch. However, there can be variations in the context switch's efficiency, which can depend on the number of data received up to that point, and on the data itself.

The DMA, with its 8-word burst capability, has a preferable context switch period when its address register is 8-word aligned: It is the only moment that occurs once every eight loops when the succession of bursts is not broken by the context switch. When this is not the case, a context switch requires the storing (or loading) of less than eight words, which requires separate accesses and is far less efficient. The rest of the 8-word packet is stored (or loaded) after the context restore, and this is done as separate accesses.

The proposed solution is a conditional yielding, which occurs only when the DMA is in an optimum state. It does not require any modification to the scripts. The condition is decided at the DMA level.

The DMA can be programmed in two modes-conditional or always-true-for every channel, which provides complete flexibility. By default, the DMA is not in conditional mode.

The DMA condition is computed from the FIFO fill level and the various modes, as follows:

- When copy mode is selected, regardless of the transfer direction ('read' or 'write'), the condition is always true.
- In read mode, the condition is always true.
- In write mode, the condition is true when there are four bytes or less in the FIFO; it is false when there are more than four bytes. The 4-byte limit comes from the possibility of saving those bytes as MD with absolutely no impact on the bus accesses.

The aim at conditional yielding is to avoid splitting bus accesses (especially bursts).

7.2.2.12.2 Peripheral DMA Unit Programming

The peripheral DMA unit is connected to the Multi-Layer DMA Crossbar Switch of the Arm platform.

Its goal is to perform data transfers between any blocks connected to the DMA bus of this platform. These blocks are either peripherals or memories. The peripheral DMA could be seen as the Arm platform DMA controller.

The DMA performs data transfers in three modes:

- Read mode, where data is read from peripherals or from memory connected to the Arm platform and copied in a SDMA general register.
- Write mode, where data of a general register has to be written in a peripheral or a memory.
- Copy mode, where data is read from a peripheral (or memory) at a source address (PSA) and automatically written to a peripheral (or memory) at a destination address (PDA).

In copy mode, no SDMA general register is involved as transferred data only goes through the data register of the DMA.

The peripheral DMA has three addressing modes: frozen, incremented, and decremented, as follows:

- Frozen mode-When source or destination addresses are frozen, their value is not modified after a transfer. This mode is typically used for addressing peripheral FIFOs located at a fixed address.
- Incremented mode-When source or destination addresses are in incremented mode, after every transfer they are incremented by the number of bytes transferred.
- Decremented mode-In decremented mode, addresses are decremented by the number of bytes transferred.

The peripheral DMA registers are as follows:

- Two, 32-bit address registers (PSA and PDA) that respectively contain the source address for a read access and the destination address for a write access
- A 32-bit status register (PS) that contains information on the peripheral DMA configuration, such as the number of valid bytes in the data register, the error flag, the source and destination address mode, and so on.
- A 32-bit data register (PD) that stores data involved in a data transfer

7.2.2.12.2.1 Peripheral Source Address Register (PSA)

The source address register contains a pointer to a source peripheral or a memory associated with the next read data transfer. It has byte granularity.

It is based on the following:

- A 32-bit register (PSA) to store the address value
- A 2-bit register (stype) to store the source address mode (frozen, incremented, or decremented)
- A 2-bit register (ssize) to store the source target data path size (byte, half-word, or word)

Reading the register with the ldf instruction has no side effects and gives the address value of the next data that will be read by the SDMA during an ldf MD instruction. Writing the source address register may have side effects. If there is valid write data in the data register and the source address is changed, the write data is discarded. If the prefetch bit is set, a DMA read cycle is issued with the new address.

When PSA is to be written, you must specify the source target address mode, providing its size (byte, half-word, or word). This enables omission of the size field in all ldf MD instructions. When DMA performs a read cycle, its size is given by the value of the PSA source size register (ssize). If source is a memory in incremented mode, first programmed in word mode (stf PSA|SZ32|I), and if an SDMA script needs to read bytes from this memory, the size of the source target must be updated before executing new accesses. The source address mode and its size are given by labels added to the stf PSA instruction as described in the write section. The ssize and stype registers are part of the DMA status register (PS).

Writing to PSA may issue an immediate error if the source size is not compatible with the value to be written into the PSA register. For instance, writing a 2 in PSA and specifying that it is memory-accessed in word mode creates an immediate error.

7.2.2.12.2.2 Peripheral Destination Address Register (PDA)

The destination address register contains a pointer to a source peripheral or a memory associated with the next write data transfer. It has byte granularity.

It is based on the following:

- A 32-bit register (PDA) to store the address value
- A 2-bit register (dtype) to store the destination address mode (frozen, incremented, or decremented)
- A 2-bit register (dsize) to store the destination target data path size (byte, half-word, or word)

Reading the register with the ldf instruction has no side effects, and gives the address value of the next data that will be written by SDMA during an stfMD instruction. Writing the destination register has no side effect. Similar to the PSA register, the destination address mode and source are specified in the stf PDA instruction and may also generate an error in case of incorrect programming.

7.2.2.12.2.3 Peripheral Data Register (PD)

The data register of the peripheral DMA is a 32-bit register. When the destination address is correctly set up, any writing to PD will automatically flush the new input data.

The number of SDMA bytes that will be transferred is given by the PDA size register. Unlike other SDMA DMAs, PD is not a FIFO: It is not used to accumulate bytes that from the SDMA and must be packed before being sent to external memories. In read mode, and if the source address is correctly set up, an ldf instruction will empty PD. If a prefetch is required along with the instruction, the DMA will initiate a new read transfer.

Reading PD in prefetch mode only stalls the SDMA when the prefetched data is not yet available. Writing PD only stalls the SDMA if the previous write operation was not completed. As soon as the previous operation is over, the acknowledge is sent back to the SDMA RISC engine.

An error flag-part of PS-is set when an external access fails. The error is thus reported to the next SDMA instruction that involves the peripheral DMA.

7.2.2.12.2.4 Peripheral State Register (PS)

The state register contains the DMA state-machine value. It can be accessed in case of an error received during a transfer.

Although all PS fields can be written by an stf instruction, it is recommended to access only the error bit (to reset it). Modifying other PS fields will provide an un-guaranteed DMA behavior.

The initialization value of PS is 0, and it consists of the following structure:

Table 7-30. PS Structure

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	ssize		stype		dsize		dtype	
W																
R	0	0	0	0	0	d	e		0	0	0	0	0	n		
W																

Table 7-31. PS Field Descriptions

Field	Description
31-24	Reserved
23-22 ssize	Source Target Size. Determines the size of the read transfers on the external bus. It should match the accessed device characteristics. 00 *reserved* 01 Byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)
21-20 stype	Source address Mode. Determines whether PSA is incremented, decremented, or kept unmodified after every read from the external bus. 00 Frozen Mode 01 Incremented Mode 10 Decremented Mode 11 *reserved*
19-18 dsize	Destination Target Size. Determines the size of the write transfers on the external bus. It should match the accessed device characteristics. 00 *reserved* 01 Byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)
17-16 dtype	Destination address Mode. Determines whether PDA is incremented, decremented, or kept unmodified after every write on the external bus. 00 Frozen Mode 01 Incremented Mode 10 Decremented Mode 11 *reserved*
15-11	Reserved
10 d	Direction Flag or DMA Mode. DMA is in write mode when data was written into PD by stf PD instructions, or if a previous DMA cycle on the external bus was a write access. Writing PDA or PSA does not change the DMA mode. DMA is in read mode when a previous DMA cycle was a read access, and DMA stays in read mode when data is read by the SDMA with an ldf PD instruction. Reading PDA or PSA does not change the DMA mode. 0 Read Mode 1 Write Mode
9-8 e	Error. Indicates if the previous access was acknowledged with a bus error. 00 No error was received. 01 *reserved* 10 Error mode 11 Error read
7-3	Reserved

Table continues on the next page...

Table 7-31. PS Field Descriptions (continued)

Field	Description
2-0 n	number of bytes in PD

NOTE

dtype, dsize, stype, and ssize are updated when PSA and PDA are written.

7.2.2.12.2.5 Peripheral DMA Write (stf)-Write Mode

When written by an stf instruction, the function code bits are interpreted as follows:

Table 7-32. STF Code Bits

Register	7	6	5	4	3	2	1	0
PSA	s		p	ar	am		sz	
PDA								
PD			pdsel					
PS			pssel					

Table 7-33. STF Code Bits Field Descriptions

Field	Description
7-6 s	Functional Unit selector 11 for Peripheral DMA
5 p (PSA)	Prefetch Flag 0 no prefetch 1 automatic prefetch
4 ar (PSA/PDA)	Address Register Selector 0 PSA 1 PDA
3-2 am (PSA/PDA)	Address Mode. Determines how PSA or PDA is modified after every read or write access to the PD. 00 Frozen-Address registers are not modified after the transfer. 01 Incremented-Address registers are incremented by the number of transferred bytes. 10 Decremented-Address registers are decremented by the number of transferred bytes. 11 Updated-PSA and PDA are not modified. Either address mode is not modified, but the width of the data path is updated by the sz field.
1-0 sz	Transfer Size 00 *reserved*

Table continues on the next page...

Table 7-33. STF Code Bits Field Descriptions (continued)

Field	Description
	01 byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)
5-0 pdsel	PD access selector 001000 is the only valid option
5-0 pssel	PS access selector 111111 writes to PS 001100 only clears the error flag in PS

Due to the large number of possible stf instructions, the following table provides only a short list of all the possible write instructions:

Table 7-34. Peripheral DMA STF Instruction List

Binary	Assembly	Comments
11_00_00_01 11_00_00_10 11_00_00_11	stf Rn, PSA|SZ8 |F stf Rn, PSA|SZ16|F stf Rn, PSA|SZ32|F	• Source is a byte, half-word, or word target at the Rn address. Any further PD read instructions will trigger a byte, half-word, or word access to the source. • Source address is frozen.
11_10_00_01 11_10_00_10 11_10_00_11	stf Rn,PSA|SZ8 |F|PF stf Rn,PSA |SZ16|F|PF stf Rn,PSA |SZ32|F|PF	• Source is a byte, half-word, or word target at the Rn address. Any further PD read instructions will trigger a byte, half-word, or word access to the source. • 1, 2, or 4 bytes are *fetched* from the peripheral source. • Source address is frozen.
11_00_01_01 11_00_01_10 11_00_01_11	stf Rn, PSA|SZ8 |I stf Rn, PSA|SZ16|I stf Rn, PSA| SZ32|I	• Source is a byte, half-word, or word target at the Rn address. Any further PD read instructions will trigger a byte, half-word, or word access to the source. • Source address is in incremented mode: PSA = PSA + 1,2 or 4 after read PD.
11_10_01_01 11_10_01_10 11_10_01_11	stf Rn, PSA|SZ8 |I|PF stf Rn, PSA|SZ16|I|PF stf Rn, PSA|SZ32|I|PF	• Source is a byte, half-word, or word target at the Rn address. Any further PD read instructions will trigger a byte, half-word, or word access to the source. • Source address is in incremented mode: PSA = PSA + 1, 2, or 4 after read PD. • 1, 2, or 4 bytes are *fetched* from the peripheral source.
11_00_10_01 11_00_10_10 11_00_10_11	stf Rn, PSA|SZ8 |D stf Rn, PSA|SZ16|D stf Rn, PSA|SZ32|D	• Source is a byte, half-word, or word target at the Rn address. Any further PD read instructions will trigger a byte, half-word, or word access to the source. • Source address is in incremented mode: PSA = PSA-1,2, or 4 after read PD.
11_10_10_01 11_10_10_10 11_10_10_11	stf Rn, PSA|SZ8 |D|PF stf Rn, PSA|SZ16|D|PF stf Rn, PSA|SZ32|D|PF	• Source is a byte, half-word, or word target at the Rn address. Any further PD read instructions will trigger a byte, half-word, or word access to the source. • Source address is in incremented mode: PSA = PSA-1,2, or 4 after read PD. • 1, 2, or 4 bytes are *fetched* from the peripheral source.

Table continues on the next page...

Table 7-34. Peripheral DMA STF Instruction List (continued)

Binary	Assembly	Comments
11_00_11_01 11_00_11_10 11_00_11_11	stf Rn, PSA|SZ8 |U stf Rn, PSA|SZ16 |U stf Rn, PSA|SZ32 |U	• *Update* source pointer to memory, which becomes a pointer to a memory accessed in byte, half-word, or word. • PSA value is not modified by Rn. • Bytes present in PD are lost.
11_10_11_01 11_10_11_10 11_10_11_11	stf Rn, PSA|SZ8 |PF|U stf Rn, PSA|SZ16 |PF|U stf Rn, PSA|SZ32 |PF|U	• *Update* source pointer, which becomes a pointer to a target accessed in byte, half-word, or word. • PSA value is not modified by Rn. • Bytes present in PD are lost. • 1, 2, or 4 bytes are *fetched* from the memory source.
11_01_00_01 11_01_00_10 11_01_00_11	stf Rn, PDA|SZ8 |F stf Rn, PDA|SZ16|F stf Rn, PDA|SZ32|F	• Destination is a byte, half-word, or word target at the Rn address, and any further PD write instructions will trigger byte, half-word, or word access to the destination. • Destination address is frozen.
11_01_01_01 11_01_01_10 11_01_01_11	stf Rn, PDA|SZ8 |I stf Rn, PDA|SZ16|I stf Rn, PDA|SZ32|I	• Destination is a byte, half-word, or word target at the Rn address, and any further PD write instructions will trigger byte, half-word, or word access to the destination. • Destination address is in incremented mode: PDA = PDA + 1, 2, or 4 after write PD.
11_01_10_01 11_01_10_10 11_01_10_11	stf Rn, PDA|SZ8 |D stf Rn, PDA|SZ16|D stf Rn, PDA|SZ32|D	• Destination is a byte, half-word, or word target at the Rn address, and any further PD write instructions will trigger byte, half-word, or word access to the destination. • Destination address is in incremented mode: PDA = PDA-1, 2, or 4 after write PD.
11_01_11_01 11_01_11_10 11_01_11_11	stf Rn, PDA|SZ8 |U stf Rn, PDA|SZ16 |U stf Rn, PDA|SZ32 |U	• *Update* destination pointer to memory, which becomes a pointer to a memory accessed in byte, half-word, or word. • PDA value is not modified by Rn • bytes present in PD are lost
11_00_10_00	stf Rn, PD	• Write "dsize" bytes of Rn in PD and automatically flush to destination target
11_11_11_11	stf Rn, PS	• Write status register
11_00_11_00	stf Rn,clrefPS	• Clear error flag if set

NOTE

When writing PD, size information is not important: It is embedded in the dsize field of PDA register. If dsize is 1, 2, or 4, then one, two, or four bytes from Rn is written to the PD register, and automatically flushed out to the destination target.

7.2.2.12.2.6 Peripheral DMA Read (ldf)-Read Mode

When received from an ldf instruction, the function code bits are interpreted as follows.

Table 7-35. LDF Code Bits

Register	7	6	5	4	3	2	1	0
PSA	s			ar	a			
PDA								
PD			p	cpy				
PS			pssel					

Table 7-36. LDF Code Bits Descriptions

Field	Description
7-6 s	Functional Unit selector 11 for Peripheral DMA
5 p (PD)	Prefetch Flag 0 no prefetch 1 automatic prefetch
4 ar (PSA/PDA)	Address Register Selector 0 PSA 1 PDA
4 cpy (PD)	Copy Mode 0 standard access 1 copy mode access
3 a	Register Set selection 0 PSA or PDA 1 PD or PS
5-0 pssel	PS access selector 111111 is the only valid option to read PS

Table 7-37. Peripheral DMA LDF Instruction List

Binary	Assembly	Comments
11_0_0_0_000	ldf Rn, PSA	Reads 32-bit of PSA value
11_0_1_0_000	ldf Rn, PDA	Reads 32-bit of PDA value
11_0_0_1_000	ldf Rn, PD	Reads programmed source size bytes of PD (0-extended)
11_1_0_1_000	ldf Rn, PD|PF	Reads programmed source size bytes of PD (0-extended), and starts a prefetch at PSA address.
11_0_1_1_000	ldf Rn, PD|COPY	Starts a copy transfer from the source target at the PSA address to the destination target at the PDA address. No data transmits through Rn, but Rn contents are lost (Rn is loaded with PD temporary contents that are *not* the copied data).
11_111111	ldf Rn, PS	Reads 32-bit of PS value

NOTE

When reading PD, size information is not important: It is embedded in the ssize field of the PSA register. If ssize is 1, 2, or 4, the one, two, or four bytes is transferred from PD to Rn. Read data is 0-extended.

7.2.2.12.2.7 Peripheral DMA Unit Copy Mode

Like burst DMA, the peripheral DMA unit has a copy mode that is used when data transfers do not involve SDMA general registers.

Data is read from the source target at a PSA address, stored in PD, and then automatically flushed to the destination target at the PDA address. Copy mode is only available for transfers that involve two targets of the same data path width.

Since copy mode is invoked with an ldf instruction, the *loaded* general purpose register loses its previous contents. (However, the new contents are unpredictable as they depend on temporary values that are seen on the external DMA bus.)

7.2.2.12.2.8 Error Management

Peripheral DMA generates two kinds of errors: the immediate error that sanctioned incorrect register programming; and the error triggered by the previous access and stored in the error flag of PS until a DMA instruction is executed.

7.2.2.12.2.8.1 Immediate Errors

The following table lists all incorrect DMA register setups.

Table 7-38. Immediate Errors with Peripheral DMA

Rn[1:0] values	DMA instruction	Comments
0x01 0x11	stf Rn, PSA|SZ16|F stf Rn, PSA|SZ16|I stf Rn, PDA|SZ16|F stf Rn, PDA|SZ16|I	If PSA points to a half-word peripheral or to a half-word address in memory, its value must be 0 modulo 2.
0x01 0x10 0x11	stf Rn, PSA|SZ32|F stf Rn, PSA|SZ32|I stf Rn, PDA|SZ32|F stf Rn, PDA|SZ32|I	If PSA points to a word peripheral or to a word address in memory, its value must be 0 modulo 4.
PSA[1:0]-PDA[1:0]	DMA instruction	Comments

Table continues on the next page...

Table 7-38. Immediate Errors with Peripheral DMA (continued)

Rn[1:0] values	DMA instruction	Comments
0x01 0x10 0x11	stf Rn, PSA|SZ32|U stf Rn, PDA|SZ32|U	When PDA or PSA is updated and becomes a pointer to a word address in memory, its content must be 0 modulo 4.
0x01 0x11	stf Rn, PSA|SZ16|U stf Rn, PDA|SZ16|U	When PDA or PSA is updated and becomes a pointer to a half-word address in memory, its content must be 0 modulo 2.
Read/Write PD instruction	Comments	
stf Rn,PD ldf Rn,PD	If PDA size (dsize) has never been set up before an stf PD instruction (dsize=0) If PSA size (ssize) has never been set up before an ldf PD instruction (ssize=0)	
ldf Rn,PD|CPY	Copy mode is possible only between two targets whose data path width is identical. It is P8↔P8, P16↔P16, or P32↔P32 regardless of the way the address registers are incremented.	

7.2.2.12.2.8.2 Data Transfer Errors

When PSA and PDA are correctly set up, the only error that may arise for an ldf PD or stf PD instruction would be the error of the previous DMA cycle.

Error handling is driven by a single consideration: When an error occurred during a data read on the DMA interface, this error should appear as a transfer error to the core when the core attempts to retrieve the data that was not successfully read from the accessed device (memory or peripheral).

When an error occurred during a write access to the DMA interface, the data is still available in PD and should not be destroyed by subsequent core accesses: The core must be warned about the error issue.

There are three error handling mechanisms for each case: Read Error (First Phase), Write Error and Read Error (Second Phase), and Copy Mode Errors handling.

7.2.2.12.2.8.3 Read Error (First Phase)

If an error occurred during a prefetch command, the peripheral DMA enters its ERROR READ mode (PS[9:8]=11). In this mode, the error is reported on the next ldf PD instruction and writing PSA, PDA, or PD will cancel the error flag.

The block returns no error mode and instructions are normally executed (a DMA cycle may be triggered). Similarly, initiating a copy transfer will reset the error flag and start a copy transfer. The following table details which instructions can be executed in this mode.

Table 7-39. Possibilities in ERROR READ Mode

DMA Instruction	Immediate Error	Comments
stf rn, PD stf rn, PSA (IU IPF) stf rn, PDA ldf rn,PD|COPY	NO	Error mode is reset, PSA or PDA are updated, or a write cycle is started. For the ldf PD|COPY, a copy loop is executed.
stf rn, PS	NO	PS is updated.
ldf rn, PS ldf rn, PSA ldf rn, PDA	NO	PS, PSA, and PDA could be read in ERROR READ mode without any side effects (for example, no DMA cycle is triggered).
ldf rn, PD	YES	Error of the previous read access is reported here and the peripheral DMA enters its ERROR mode.

7.2.2.12.2.8.4 Write Error and Read Error (Second Phase)

The peripheral DMA enters its ERROR mode (PS[9:8]=10) when the previous DMA write cycle failed, or, as explained in Read Error (First Phase), when an ldf PD is executed while the block is in ERROR READ mode. When a DMA cycle failed, address registers (PSA, PDA) are not modified and continue to point to the problematic address. In ERROR mode, stf instructions may raise an immediate error, and ldf instructions will not (as detailed in the table below).

Table 7-40. Possibilities in ERROR Mode

DMA Instruction	Immediate Error	Comments
stf rn, PD stf rn, PSA stf rn, PDA	YES	Any attempt to modify PD, PSA, or PDA will raise an immediate error, and the peripheral DMA stays in ERROR mode. When address registers are write accessed, an error is returned.
stf rn, PS	NO	This is the only way to exit the ERROR mode. PS[3] must be reset by an stf PS instruction.
ldf rn, PS ldf rn, PSA ldf rn, PDA	NO	PS, PSA, and PDA could be read in ERROR mode without any side effects (for example, no DMA cycle is triggered).
ldf rn, PD	YES	Whatever the DMA direction (read or write), an ldf rn, PD instruction will show an immediate error.

7.2.2.12.2.8.5 Copy Mode Errors

Because copy mode is a write access that follows a read access, there are two possible cases of bus error.

When the read access incurs a bus error, the peripheral DMA behaves exactly as described in Read Error (First Phase) and Write Error and Read Error (Second Phase) : It enters its ERROR READ mode, and so on.

When the error occurred during the write access of the copy transfer, the DMA enables the core to retrieve the data that was read because it is assumed the read from the peripheral removed the data from its source device. Therefore, the data to be flushed is still in PD. Any subsequent access to PD triggers an error to the core, which should execute its error handling procedure.

Once the ERROR mode is left (after writing to PS), it is possible for the core to retrieve the data in PD with an ldf instruction or try to flush PD contents once again (for example, when the error was due to a full FIFO and the script waited for the FIFO to be emptied) with another ldf instruction in copy mode. This latter instruction detects that there is valid data in PD, tries to flush it, and thus skips the read phase of the copy instruction. This is a different behavior from the usual stf PD instruction that overwrites PD with the selected General Purpose register contents. The same mechanism can be used any time PD holds data that is not written because of a bus error on the DMA interface; when the data was written via a copy instruction, or via the usual stf PD instruction.

7.2.2.12.2.8.6 Error Check Example

The following code illustrates an example checking for both immediate and data transfer errors on a store to the PD register. The first bdf instruction checks for an immediate error, but if a data transfer error occurred it is reported until the next instruction to access the Peripheral DMA. A second check of the error flags is done after the ldf PS instruction. The value of PS here can be ignored. The act of reading any register in Peripheral DMA while it is in an error mode that returns the error to the core to set either the SF or DF flag. Any error returned on an ldf command sets the SF flag and any error returned on an stf instruction sets the DF flag. This can create a situation as shown in the example where a bus error during a DMA write which would normally be considered as a destination fault is reported as a source fault because the error was reported to the SDMA core during an ldf instruction.

Peripheral DMA Error Check

```
    clrf    0            // Clear SF and DF flags
 stf    R4, PD        // Write data to memory
bdf    error_routine    // Check for immediate error from write to PD.
ldf    r3, PS        // Read PS (PS value in R3 can be ignored)
    bsf    error_routine    // Check for bus error from "stf R4,PD"
           // SF is set because it is a ldf instruction, even though
           // the original error was a destination fault
```

7.2.2.12.2.9 Peripheral DMA Unit Prefetch/Flush Management

There is no flush bit because every time data is stored in PD by a stf PD instruction-assuming PDA is correctly programmed-it is automatically flushed to the destination.

An acknowledge is returned in the cycle of the DMA instruction, and the SDMA is only stalled by an instruction that addresses the peripheral DMA when the previous DMA access is not over.

7.2.2.12.3 OnCE and Real-Time Debug

The On-Chip Emulation block (OnCE) is the debug interface to the SDMA.

It supports the access to all core internal devices (registers, memory, and so on), and provides a set of mechanisms that control the core. The OnCE is accessed by JTAG ports at the chip's board level, or by the host via its peripheral bus.

To reduce the size of the hardware material involved, all tasks supported by the OnCE are performed on the SDMA core. The architecture of the SDMA OnCE is relatively simple and very flexible.

The commands supported by the SDMA OnCE are listed in the following sections.

7.2.2.12.3.1 Memory and Register Access

A set of mechanisms is provided to access SDMA memory and register locations. Both reading and writing are allowed. The access is supported if the processor is in debug mode.

Those registers can also be accessed through the Arm platform Control interface when the OnCE is controlled by the Arm platform, as described in the "Using BP" section.

7.2.2.12.3.2 Hardware Breakpoints

An event detection unit is implemented to support memory breakpoints. The unit watches the data exchanged between the SDMA memory bus and the core.

A debug request is sent to the core when matching conditions occur. The unit supports mixed conditions based on address range, access type, and data value. Event detection unit configuration registers are memory mapped in the SDMA space (see Arm platform Channel 0 Pointer (SDMAARM_MC0PTR)): You can modify them through a regular memory access or the Arm platform control interface.

7.2.2.12.3.3 Watchpoints

One output pin is provided to monitor matching trigger conditions that are defined in the event detection unit.

7.2.2.12.3.4 Software Breakpoints

The SDMA instruction set contains a software breakpoint. Upon executing a software breakpoint instruction, the core suspends normal execution and enters debug mode.

No hardware step execution mode is implemented in the OnCE, but this feature may be implemented at the software level with this instruction.

7.2.2.12.3.5 Core Control

Commands are provided to monitor and control processor activity. You can halt the core, rerun the core from another address location, and get processor status.

Any hardware breakpoint on the instruction bus is not supported, but this feature may be implemented by inserting a software breakpoints program.

7.2.2.13 The OnCE Controller

The OnCE controller receives commands from the Arm platform or from the JTAG controller. Each command is interpreted before being sent to the core.

7.2.2.13.1 OnCE Commands

A small set of commands supports the communication between the OnCE and the external world.

This command set enables you to perform any of the following tasks: control processor activity, save core context, and execute an SDMA instruction from the OnCE. Combined together, these tasks perform more complex commands.

A full OnCE command contains a 4-bit instruction (the OnCE command opcode) and a variable length data field (the OnCE data). During command execution, the OnCE data is transferred in a OnCE internal register before being exchanged with the SDMA. Some data values are also exported. This mechanism creates a link between the processor and the external world. Nine commands are defined: The following table presents their formats.

Table 7-41. OnCE Command Opcode Values

Instruction Opcode	Name	Action	Register	Data Field Size	Mode
0000	rstatus	Reads the OnCE status register	STATUS	16-bit	normal/debug
0001	dmov	Updates general register GReg1	GREG1	32-bit	debug
0010	exec_once	Runs the instruction from the SDMA instruction register	INSTRUCTION	16-bit	debug
0011	run_core	Returns to normal execution	BYPASS	1-bit	debug
0100	exec_core	Returns to normal execution via a jump instruction that specifies the new address	INSTRUCTION	16-bit	debug
0101	debug_rqst	Stops the core after execution of current instruction	BYPASS	1-bit	normal
0110	rbuffer	Reads the real time buffer	RTB	32-bit	normal/debug
0111-1110	reserved	Reserved	BYPASS	1-bit	normal/debug
1111	bypass	Bypasses TARM platform controller	BYPASS	1-bit	normal/debug

Each instruction corresponds to a specific action performed on the OnCE. The nature of the associated data field is clearly identified. The dmov command is followed by a 32-bit data value (which is a data value for the SDMA); the exec_once and the exec_core commands are followed by a 16-bit data value (which is an instruction for the SDMA); the rstatus command is followed by a 16-bit control value (which is the content of the OnCE status register); the rbuffer command is followed by a 32-bit data value. The debug_rqst and the run_core commands are followed by a single bit data field (this is a bypass value). Finally, the bypass instruction enables the SDMA JTAG TAP controller to be daisy-chained with another JTAG TAP controller. This is a JTAG-only feature. The set of commands is simple, but enables you to perform any possible task on the SDMA during a debug process.

7.2.2.13.2 Sending Commands to the OnCE Controller

The JTAG access is the standard access to the OnCE, but sometimes the JTAG is not available to fix some bugs (if the chip is in production for instance), an additional access is then required. Therefore, one Arm platform access to the OnCE is provided.

7.2.2.13.2.1 Using the JTAG Interface

A serial access is performed through the five JTAG pins TCK, TRST, TMS, TDI, and TDO. A Test Access Port controller is provided to decode the TMS control signal.

It produces shift-enable signals (shift_ir and shift_dr), and updates enable signals (update_ir and update_dr). It is fully compliant with the IEEE 1149.1 testability (JTAG) standard.

During the shift_ir state, the command opcode is shifted into the OnCE controller (for example, the signal from the TDI pin is shifted into the command register and the TDO pin receives the signal shifted out). After transferring the four bits of the command, an update_ir signal is asserted and the command is decoded. The target data register is now clearly identified and the corresponding control signal is produced, as follows: bypass enable signal (bp_en), instruction enable signal (inst_en), data enable (data_en), and status enable signal (stat_en).

During the shift_dr state, the TDI signal is shifted into one of the following target registers: bypass register (1 bit), SDMA instruction register (16 bits), SDMA data register (32 bits), or OnCE status register (16 bits). The TDO pin is connected to the output of the selected register to receive the signals shifted out.

The JTAG access is disabled when the Arm platform access is enabled.

7.2.2.13.2.2 Using the Arm platform

The Arm platform access to the OnCE is not the standard access, but it is required if the JTAG is not available.

For example, if the SDMA ROM is out of use on a chip in production, and the Arm platform needs to download new code and restart the SDMA, the OnCE can easily perform this operation. This type of debug operation justifies the use of an Arm platform access to the OnCE.

To drive the OnCE, the Arm platform uses some registers contained in the Arm platform Control block of the SDMA. These registers are accessed through the Arm platform peripheral bus. Most of these registers are connected to another register in the OnCE controller. Thus, accessing one of these registers is equivalent to accessing the associated register in the OnCE controller.

The set of registers in the Arm platform Control block is listed below:

- ONCE_ENB register (1 bit, read/write)-This 1-bit register enables the Arm platform access to the OnCE. When this bit is set, the signals from the JTAG are ignored. When it is cleared, all writing operations to the following registers through the Host Control interface are ignored. This register is reset on a JTAG reset.
- ONCE_CMD register (4 bits, read/write)-This 4-bit register receives the command opcode. It is connected to the command register in the controller. A write access to this register causes the associated command to be executed on the OnCE. For example, after writing "0001" in this register, a dmov command is executed.

NOTE

On the Arm platform side, the rstatus and bypass commands are not supported. This register is reset on a JTAG reset.

- ONCE_DATA register (32 bits, read/write)-This 32-bit register is connected to the SDMA data register. This register is used when executing a dmov or rbuffer command.

NOTE

Before requesting a dmov command, the 32-bit data to transfer must be written in the ONCE_DATA register. At the end of the execution, the register is updated with GReg1 former value. This register is reset on a JTAG reset.

- ONCE_INSTR register (16 bits, read/write)-This 16-bit register is connected to the SDMA instruction register. This register is used when executing an exec_core or an exec_once command.

NOTE

Before requesting an exec_core or an exec_once command, the appropriate instruction must be written in the ONCE_INSTR register. This register is reset on a JTAG reset.

- ONCE_STAT register (16 bits, read only)-A read access to the ONCE_STAT register returns the content of the OnCE status register (OSTAT). This register is read only.
- The bypass register is not useful when the Arm platform controls the OnCE, therefore no register is defined in the Arm platform Control block to access the bypass register.

7.2.2.13.2.3 Conflicts Between the JTAG and the Arm platform Accesses

When Arm platform access to the SDMA OnCE is enabled (that is, when the bit in the ONCE_ENB register is set), the JTAG access is disabled. This guarantees that the block is not accessed at the same time on both sides.

It is possible to check whether the JTAG access to the SDMA OnCE is enabled from the JTAG port. When the JTAG access is disabled, the SDMA TDO always returns 1. The check requires the following steps:

- Execute a dmov command from debug mode (with neither 0xffffffff nor 0x0 as dmov value: 0x5a5a5a5a is good).

- Execute another dmov command (the value here is not important).
- The returned value from the latter dmov command should be the original one if the JTAG access is enabled; if it is 0xffffffff instead of the original input value, this means the JTAG access is disabled.

7.2.2.13.3 Executing a Command from the OnCE

All the commands defined in OnCE Commands can be accessed through the JTAG. The Arm platform can access all these commands except the rstatus command.

On the Arm platform side, the OnCE status is directly accessed by reading the ONCE_STAT register.

7.2.2.13.3.1 Nature of the Commands

Two types of commands may be distinguished. First, there are two commands that do not interact with the core: rstatus and rbuffer. Those commands may be requested at any time: They do not depend on the core status.

NOTE

Each of these commands exports a data value or a status value from the SDMA.

There are also commands that interact with the core: dmov, run_core, exec_core, exec_once, and debug_rqst. These commands are core status dependent, as follows:

- During user mode only the debug_rqst is taken into account.
- During debug mode, all these commands are taken into account except the debug_rqst. For example, an exec_once command requested while not in debug mode has no effect.

7.2.2.13.3.2 Execution Request

The SDMA starts executing a task in debug mode when requested by the OnCE controller. The execution starting time depends on the type of access used to communicate with the OnCE.

If the JTAG is used, the request is send after decoding the update_dr state in the TAP controller. Therefore, always cross this state when sending a command through the JTAG. If the OnCE is driven from the Arm platform side, the request is sent after detecting a write access to the ONCE_CMD register. All the registers involved in this operation must be loaded first.

The following is an example of an exec_core command execution from the Arm platform side: After writing '010' in the ONCE_CMD register, the OnCE controller asks the SDMA to execute the instruction contained in the ONCE_INSTR register. The instruction involved should be available in the ONCE_INSTR register before the beginning of the execution.

7.2.2.13.3.3 Command Execution

The following list shows the commands and details how each command is executed:

- rstatus command execution-The rstatus command exports the content of the OnCE status register (OSR). If the JTAG is used, the status information is captured in the OnCE status register during the capture_dr state, and shifted out after 16 TCK clock cycles in the shift_dr state. The rstatus command is not supported on the Arm platform side, but a status register is provided instead. The rstatus may be performed in both debug and user modes.
- dmov command execution-The dmov command accesses SDMA internal registers. Executing a dmov instruction exchanges the 32-bit data values between the SDMA data register and the general register GReg[1].
- If the JTAG is used, the content of GReg1 is captured in the SDMA data register during the capture_dr state, then it is shifted out after 32 TCK clock cycles in the shift_dr state. During the update_dr state, GReg1 is updated with the new, shifted-in 32-bit data value. If the OnCE is driven from the Arm platform side, the data values contained in GReg1 and the SDMA data register are exchanged after detecting a write access to the ONCE_CMD register. The ONCE_DATA register must therefore be loaded first.

- exec_once command execution-The exec_once command executes the instruction loaded in the SDMA instruction register. The command may only be requested from debug mode. The SDMA returns to debug mode at the end of the execution.
- Change of flow instructions as well as instructions that may cause a context switch are not supported: The comprehensive list comprises done/yield/yiedge (except done 5), BF, BT, BSF, BDF, JMP, JSR, JMPR, JSRR, RET, and LOOP, as well as all the illegal instructions.

  No other command should be requested before the SDMA returns to debug mode. The SDMA status (for example, whether it is in debug mode or not) can be detected by polling with the rstatus OnCE command, monitoring the debug_mode pin, or checking the OnCE Status Register (SDMAARM_ONCE_STAT) register via the Arm platform control interface.

NOTE

Most of the instructions are single-cycle, which omits the step of polling the status. Loads and stores to DMA units are typical instructions that might require this polling.

If the JTAG is used, the 16-bit instruction is shifted in the SDMA instruction register after 16 TCK clock cycles in the shift_dr state. A request is sent to the core when the update_dr state is decoded in the TAP controller. If the OnCE is driven from the Arm platform side, the request is sent to the SDMA when detecting a write access to the ONCE_CMD register. The ONCE_INSTR register must be therefore be loaded first.

- run_core command execution-The run_core command leaves debug mode and resume normal program execution. The next instruction executed is the last instruction decoded before entering debug mode. Be sure to restore core context before re-running the core. This procedure is detailed in Restoring the Context.
- If the JTAG is used, a 1-bit bypass value is shifted in the bypass register in the shift_dr state. The SDMA is rerun when the update_dr state is decoded in the TAP controller. If the OnCE is driven from the Arm platform side, the core is rerun when detecting a write access to the ONCE_CMD register.

- exec_core command execution-The exec_core command resumes program execution from any address. The 16-bit instruction provided with the exec_core overwrites the last instruction decoded before entering debug mode. This command is designed to support change of flow instructions, so that a program execution can be restarted from any address. After executing an exec_core command, the SDMA leaves debug mode. The exec_core command is usually used with a jmp instruction.
- If the JTAG is used, the 16-bit branch instruction is shifted in the SDMA instruction register after 16 TCK clock cycles in the shift_dr state. The SDMA is rerun when the update_dr state is decoded in the TAP controller. If the OnCE is driven from the Arm platform side, the SDMA reruns when detecting a write access to the ONCE_CMD register. The ONCE_INSTR register must therefore be loaded first. For example, to restart the SDMA from the program address 0x100, the instruction loaded should be a jump to address 0x100 instruction.

- debug_rqst command execution-The debug_rqst command puts the SDMA in debug mode. If the JTAG is used, a 1-bit bypass value is shifted in the bypass register during the shift_dr state. A debug request is sent to the SDMA when the update_dr state is decoded in the TAP controller. If the OnCE is driven from the Arm platform side, the debug request is sent when detecting a write access to the ONCE_CMD register. When the SDMA is already in debug mode, this command is simply ignored.
- rbuffer command execution-The rbuffer command exports the content of the real time buffer (RTB). If the JTAG is used, the content of the real time buffer (RTB) is

captured in the SDMA data register during the capture_dr state. The register is completely shifted out after maintaining the shift_dr state during 32 TCK clock cycles. If the OnCE is driven from the Arm platform side, the content of the RTB is captured in the ONCE_DATA register after detecting a write access to the ONCE_CMD register.

- bypass command execution-This command is only available from the JTAG interface. It enables daisy-chaining of the SDMA JTAG TAP controller with other JTAG TAP controllers. This command does not change the SDMA state and can be executed in any mode (run, debug, or sleep). It selects the bypass register of the TAP controller.

7.2.2.13.4 Registers Descriptions

See SDMACORE, and SDMAARM, for detailed information on each register.

7.2.2.13.4.1 Event Cell Counter Register (ECOUNT)

The event cell counter register is a 16-bit register that contains the number of times minus one that an event detection occurs before generating a debug request.

This register should be written before attempting to use the event detection counter during an event detection process. The event cell counter register is cleared on a JTAG reset.

7.2.2.13.4.2 Event Cell Address Registers (EAA or EAB)

The event cell contains two address registers-the event cell address register (a), called EAA, and the event cell address register (b), called EAB. Every address register is a 16-bit register that stores a user-defined address value. This value computes one of the following address conditions: addra_cond or addrb_cond. Every address register is cleared on a JTAG reset.

7.2.2.13.4.3 Event Cell Address Mask Register (EAM)

The event cell address mask register is a 16-bit register that contains a user-defined address mask value. This mask is applied to the address value latched from the memory address bus before comparing addresses.

NOTE

There is a common address mask value for the two address comparators. If bit *i* of this register is set, then bit *i* of the

address value latched from the memory bus does not influence the result of the address comparison. The event cell address mask register is cleared on a JTAG reset.

7.2.2.13.4.4 Event Cell Data Register (ED)

The event cell data register is a 32-bit register that contains a user-defined data value. This data value is an input for the data comparator, which generates the data_cond condition.

The event cell data register is cleared on a JTAG reset.

7.2.2.13.4.5 Event Cell Data Mask Register (EDM)

The event cell data mask register is a 32-bit register that contains a user-defined data mask value. This mask is applied to the data value latched from the memory bus before comparing data.

Setting bit *i* of the event cell data mask register means that bit *i* of the data value latched from the address bus does not influence the result of the data comparison. The event cell data mask register is cleared on a JTAG reset.

7.2.2.13.4.6 Real Time Buffer Register (RTB)

The real Time Buffer register is a 32-bit register that stores and retrieves run-time information without putting the SDMA in debug mode.

Refer to Real Time Buffer for more details.

7.2.2.13.4.7 Event Control Register (ECTL)

The event cell control register is a 16-bit register that defines cell event occurrence conditions.

The event cell control register is cleared on a JTAG reset. See also OnCE Event Detection Unit for more details.

7.2.2.13.4.8 Trace Buffer (TB)

The Trace Buffer register retrieves the information in the Trace Buffer.

See Trace Buffer for more details.

7.2.2.13.4.9 OnCE Status Register (OSTAT)

The OnCE status register is a 16-bit register that contains processor and event detection unit status. The OSTAT is a read-only register.

Refer to OnCE Status Register (SDMAARM_ONCE_STAT) for detailed description of the individual fields in the OSTAT register.

The following figure shows the OSTAT structure.

Table 7-42. OnCE Status Register (OnCE)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PST[3:0]				RCV	EDR	ODR	SWB	MST					ECDR[2:0]		

Where PST[3:0] is the SDMA core state, RCV is set when the real-time buffer (RTB) is modified. EDR, ODR, and SWB are set, respectively, when the SDMA has entered debug mode because of an external debug request, a OnCE debug_rqst command, or a software breakpoint. MST is set when the OnCE is controlled from the Arm platform control interface, and when ECDR is a three-flag set that shows the event cell condition(s) that put the core in debug mode. The OSTAT never provides more than one reason for entering debug mode.

There are two ways of accessing OSTAT content, as follows:

1. Send an rstatus command to the OnCE controller through the JTAG, or read the ONCE_STAT register through the Arm platform access. Executing the rstatus command through the JTAG can be performed in both user and debug modes.
2. Perform an SDMA read access to the location in the SDMA core memory map (OSTAT register) debug mode using the exec_once command. With this method of access, the SDMA state reflected by the PST (processor status bit) is always DATA.

The register may also be accessed by a running application.

7.2.2.13.5 JTAG Interface Requirements

Because the signals received from the JTAG (running on TCK) are transferred to the OnCE controller (running on the SDMA clock), a synchronization mechanism is required.

7.2.2.13.5.1 TCK Speed Limitation

In the JTAG top-level layer, the TDO signal is always captured on a TCK falling edge. To guarantee a stable TDO signal from the SDMA during this operation, a falling edge detection is performed on TCK.

Before being latched in the *I* flip-flop (see Figure 7-11) on TCK falling edge, the TDO signal must be stable at the input of the flip-flop. This condition is verified if the TCK period is superior to the following delay:

worst-case edge detection delay + negative-edge signal propagation delay + JTAG top-level logic propagation delay

The frequency relationship, TCK < CLK/8, limitation guarantees that all operations are performed as expected.

7.2.2.13.5.2 Synchronization Implementation

The figure found here shows the synchronization mechanism.

Flip-flops tck0, tck1, and tck2 perform falling- and rising-edge detections on TCK. They generate the posedge_detected and negedge_detected nets that are used to sample the TDI and TMS inputs into the respective tdi and tms flip-flops, and update the tdo flip-flop to yield the TDO output. In the design, the only signal that might go metastable is the output of the tck0 flip-flop. This signal is captured in the tck1 flip-flop and no logical operation is performed on it to minimize a metastability propagation risk.

The TDI and TMS flip-flops also cannot go metastable: The propagation time of the rising-edge detection signal through tck0, tck1, and tck2 guarantees that the TDI and TMS inputs are stable when captured in the TDI and TMS flip-flops.

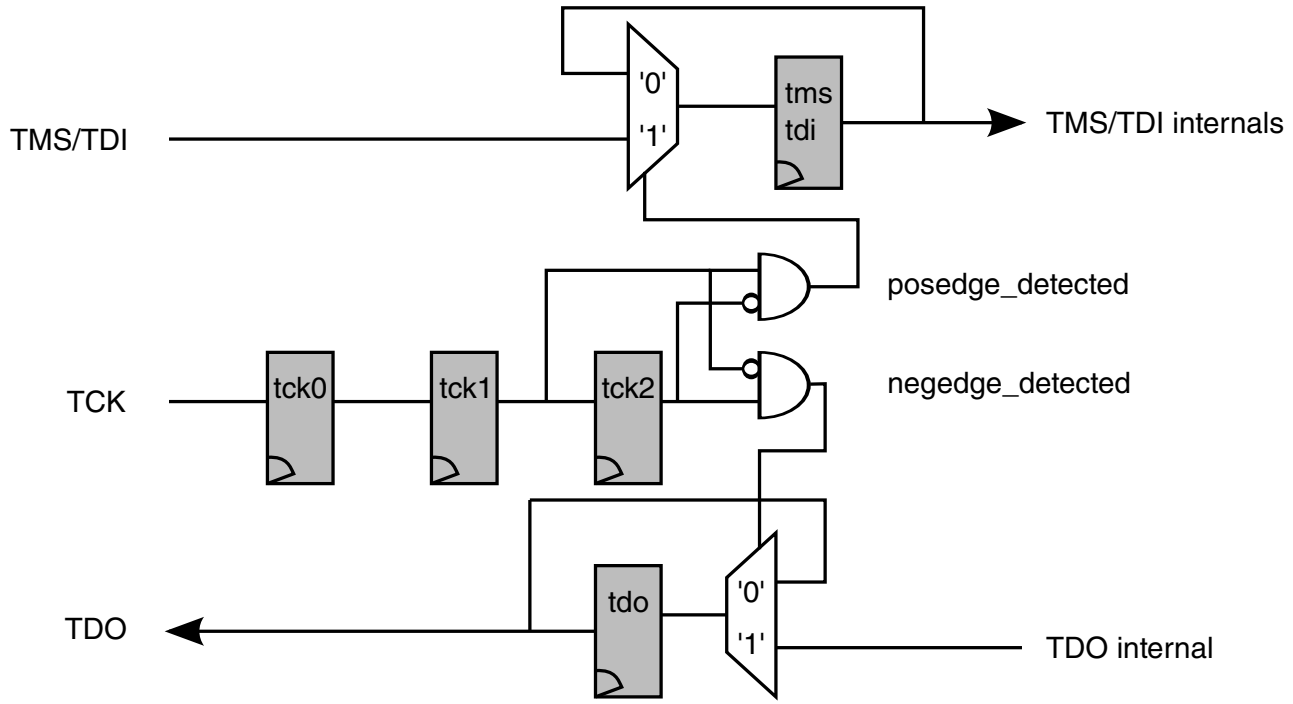


Figure 7-11. OnCE Synchronization Layer

The following figure shows synchronization timings. It takes three CLK clock cycles to synchronize TDI on the SDMA clock.

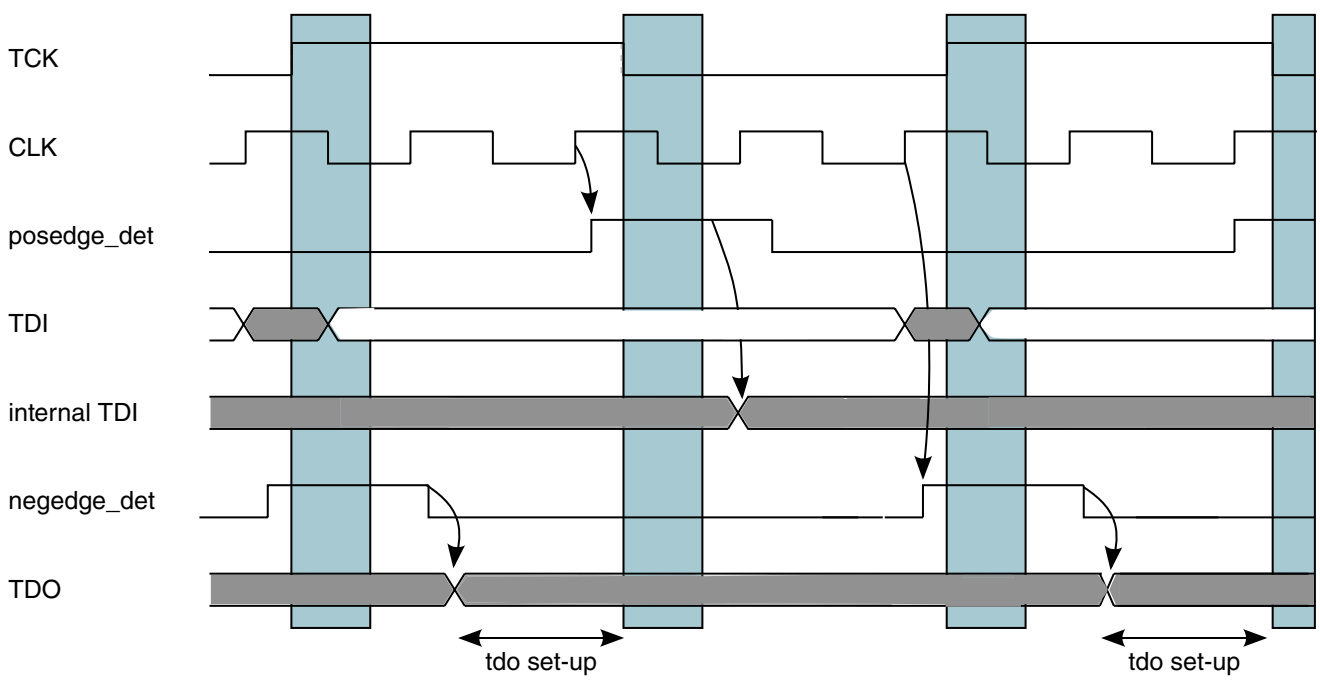


Figure 7-12. Synchronization Timings

7.2.2.13.5.3 JTAG Controller Start-Up Recommended Procedure

To ensure correct TAP controller initialization, it is recommended to use the following procedure:

1. Assert JTAG reset TRSTB (for example, set low).
2. Set TMS low.
3. Wait for 1 TCK clock.
4. Release JTAG reset TRSTB (for example, set high).
5. Wait for a minimum of five TCK cycles.

7.2.2.14 Using the OnCE

This section provides the elements necessary to run the OnCE during a debug process.

In addition to the basic set of commandsdescribed in OnCE Commands, more complex commands can be built to meet users' requirements.

7.2.2.14.1 Activating Clocks in Debug Mode

For power consumption issues, some clocks in the SDMA are disabled when not needed.

This is the case for instances when the SDMA is in sleep mode. Clock gating management depends on the interface used to control the OnCE.

- For the JTAG access, the SDMA clock gating must be turned off via the clk_gating_off input.
- For the Arm platform access, the SDMA clock gating is automatically turned off when the Arm platform access is enabled (see OnCE Enable (SDMAARM_ONCE_ENB)).

7.2.2.14.2 Getting the Current Status

Most of the commands the OnCE supports have an impact on the status of the SDMA.

It is not permissible to request the execution of an instruction on the SDMA from the OnCE while the SDMA is not in debug mode. Such a violation may cause unpredictable behavior, and it might be necessary to reset the SDMA.

Therefore, the value of the PST bits provided in the OnCE status register should always be checked before sending any request to the SDMA.

7.2.2.14.3 Methods of Entering Debug Mode

A debug request may be asserted at any time, but it is not always taken into account immediately. Debug mode cannot be entered in the middle of an instruction, or during the save or restore states of a context switch.

The request is ignored when the core is already in debug mode. Refer to Figure 7-4, which shows all possible transitions to the debug state, as there are several ways to enter debug mode.

7.2.2.14.3.1 External Debug Request During Reset

To enter debug mode after exiting reset, the external debug line has to be maintained high. This line is handled by the JTAG top-level block.

NOTE

The SDMA detects the debug requests only if the SDMA clock is running (see Activating Clocks in Debug Mode). The debug request line should be not be maintained high when the SDMA is in debug mode.

NOTE

The debug_rqst command (from the OnCE command set) is not supported during system reset.

7.2.2.14.3.2 Debug Request During Normal Activity

During normal activity, the SDMA enters debug mode when the following is true:

1. If the debug request line from the JTAG top-level is asserted, or
2. If the OnCE controller receives a debug_rqst command.

   The debug_rqst command can be sent by the JTAG access or by an access on the Arm platform side (if the Arm platform access is enabled).

7.2.2.14.3.3 Software Breakpoint Instruction

The SDMA enters debug mode at the end of the execution of a software breakpoint instruction. This instruction must be inserted in program flow executed by the core.

7.2.2.14.3.4 Event Detection Unit Matching Condition

If the event detection is enabled, a debug request is sent to the core after detecting a matching condition on the SDMA memory bus.

See OnCE Event Detection Unit for more details.

7.2.2.14.4 Executing Instructions in Debug Mode

The OnCE supports a mechanism to execute instructions in debug mode. If the SDMA is in debug mode, then the exec_once command can be used to execute an SDMA instruction from the OnCE controller. The SDMA returns to debug mode at the end of each execution.

Some instructions are not supported by the exec_once command: done/yield/yiedge (except done 5), BF, BT, BSF, BDF, JMP, JSR, JMPR, JSRR, RET, and LOOP, as well as all the illegal instructions are not supported.

NOTE

While instructions are executed in debug mode from the OnCE, the program counter of the SDMA is not incremented.

7.2.2.14.5 Command Sequences Examples

This section provides examples of command sequences that run the SDMA in debug mode. These sequences are available for both the Arm platform and JTAG accesses.

The following presents the syntax used in this section. The data field provided with each command is put in parenthesis with the command name. A '-' is used if the data field provided is a *don't care* value.

```
my_command(data_field);                 // executing my_command with a data field
my_command(-);                        // executing my_command with a don't care data field
```

The value returned by the command (if there is one) is referred by an assignment. In case the value returned by the command is not used, the assignment is omitted. For an Arm platform access, the value returned (it is always a data value) is obtained by reading back into the SDMA data register.

```
data_out = my_command(data_in); // returning a data value
```

To clarify the syntax, the instructions' opcodes are referred to by their names. In practice, use the corresponding 16-bit encoding.

7.2.2.14.5.1 Getting the SDMA Status

NOTE

Before executing any command that affects the SDMA (like dmov or exec_once), check that the SDMA is in debug mode.

Use the following snippet:

```
rstatus();        // read SDMA status until the SDMA is in debug mode
...
rstatus();
```

If the SDMA is not in debug mode, then a debug request must be generated. In this case, the SDMA enters debug mode at the end of the execution of the current instruction. Use this snippet:

```
debug_rqst(-);   // debug request
```

In the following sections, it is assumed that the SDMA was successfully put into debug mode.

7.2.2.14.5.2 Saving the Context

The first debug task is to save the SDMA context, which is the content of the eight general-purpose registers, the loop and PC-related registers, and the flags.

Use the general register GReg[1] as an intermediate register to export the entire context of the SDMA.

The following example shows how to save GReg[0], GReg[1], GReg[2] and GReg[3]. The sequence of commands used to export additional general registers is very similar to this.

Save GReg[0], GReg[1], GReg[2], and GReg[3]

```
GReg1_data = dmov(-);                          // the value exported is the content of
GReg[1]
exec_once("mov GReg1,GReg0");                      // puts the content of GReg[0] into
GReg[1]
GReg0_data = dmov(-);                          // the value exported is the content of
GReg[0]
exec_once("mov GReg1, GReg2");                     // puts the content of GReg[2] into
GReg[1]
GReg2_data = dmov(-);                          // the value exported is the content of
GReg[2]
exec_once("mov GReg1, GReg3");                     // puts the content of GReg[3] into
GReg[1]
GReg3_data = dmov(-);                          // the value exported is the content of
GReg[3]
```

Get the value of the internal flags (SF, DF, T, and LM), of the loop related registers (EPC and SPC), and of the PC-related registers (PC and RPC). Use a done 5, which is the formatting instruction dedicated to the debug. This instruction formats the flags and the

values contained in the registers. It also writes the resulting values into the channel context memory. It should not be used when entering debug from the IDLE state (for example, with no active channel script running on the SDMA), because it will update a channel context that may belong to any channel.

```
exec_once("done 5");                    // formatting the value of flags and registers
```

At this point, the channel context should be up-to-date in memory, and debug operations should now be possible. However, the context can be exported with the following instructions:

Exporting the Context

```
dmov(ctx_base_addr);                                  // loading GReg[1] with the channel
context base address
exec_once("ld GReg0,(GReg1,0)");                          // get RPC-PC into GReg0
exec_once("ld GReg1, (GReg1,1)");                         // get SPC-EPC into GReg1
Loop_data = dmov(-);                                // read back the value of Loop registers
exec_once("mov GReg1, GReg0");                          // puts the PC info into GReg1
PC_data = dmov(-);                                  // reads back the content of the PC registers
```

After this sequence of operations, the entire SDMA context is exported via the OnCE.

7.2.2.14.5.3 Restoring the Context

At this point in the operation, restore the context of the SDMA. It can be different from the original context located in memory, and the content previously saved into the debugging application via the OnCE.

The example found hereshows how it is possible to modify the current channel context.

Modifying the Current Channel Context

```
dmov(Loop_data);                                    // put Loop former value into GReg[1]
exec_once("mov GReg0, GReg1");                          // copy to GReg[0]
dmov(PC_data);                                      // put PC former value into GReg[1]
exec_once("mov GReg2, GReg1");                          // copy to GReg[2]
dmov(ctx_base_addr);                                  // put channel context base address into
GReg[1]
exec_once("st GReg0, (GReg1,1)");                         // restore Loop context
exec_once("st GReg2, (GReg1,0)");                         // restore PC context
```

Once the context in memory is the desired context (with or without applying the previous instruction sequence), it can be restored to the *real* PC and loop registers in the SDMA core:

```
exec_once("cpShReg");                    // restore flags and PC & loop related registers
```

After this command, the SDMA core PC, RPC, SPC, EPC registers, as well as the flags contain the same data as what is stored in the context RAM for the current channel.

The following example shows how to restore the context of general registers GReg[0], GReg[1], GReg[2] and GReg[3].

Restoring the General Register Context

```
dmov(GReg3_data);                    // put GReg[3] restore value in GReg[1]
exec_once("mov GReg3, GReg1"); // restore GReg[3]
dmov(GReg2_data);                    // put GReg[2] restore value in GReg[1]
exec_once("mov GReg2, GReg1"); // restore GReg[2]
dmov(GReg0_data);                    // put GReg[0] restore value in GReg[1]
exec_once("mov GReg0, GReg1"); // restore GReg[0]
dmov(GReg1_data);                    // restore GReg[1]
```

At this point, it is possible to restart the normal program execution.

NOTE

Every SDMA core general register value can be modified by a mov instruction, which makes modification of these registers easy during debug. Unfortunately, there is no such instruction as a mov to directly modify the contents of either PCU register or flag (PC, RPC, SPC, EPC, T, LM, SF, or DF). The cpShReg instruction is meant to provide a means for changing these register contents via the context memory.

7.2.2.14.5.4 Accessing the Memory

In the example shown here, it is assumed that the SDMA context is entirely saved. If true, it is permissible to modify the general purpose registers during debugging activity.

To perform a memory read access, the target address is stored via the OnCE in GReg[1], then the load instruction is executed on the SDMA (the data loaded from the memory overwrites the address contained in GReg[1]), and then the result value is read back via the OnCE.

```
macro READ:            dmov(target_addr);                                  // put the target
address in GReg[1]
                    exec_once("ld GReg1,(GReg1,0)");                        // execute the
load instruction
                    res_data = dmov(-);                                 // exports the result
data value
```

For a memory write access, the target address is written in GReg[0], and the value to store is written in GReg[1]. Then the store instruction is executed on the SDMA.

```
macro WRITE:             dmov(target_addr);                                  // puts the
target address in GReg[1]
                    exec_once("mov GReg0,GReg1");                          // puts the target
address in GReg[0]
                    dmov(target_data);                                  // puts the target
data in GReg[1]
                    exec_once("st GReg1,(GReg0,0)");                            // performs the
store operation
```

This sequence is shown as an example; however, many other sequences are possible.

NOTE

This sequence of commands can also be applied to memory-mapped registers.

7.2.2.14.5.5 Resuming Program Execution

Before resuming program execution, it is assumed that the SDMA context is properly restored. There are two ways to restart the SDMA.

Start by executing the last instruction fetched before entering debug mode, as follows.

```
run_core(-);                          // resume execution from where we stopped before
```

If necessary, restart the execution from a different address. In this case, use the exec_core command. The data field provided with this command must be the encoding of a jump instruction.

```
exec_core("jmp start_addr"); // rerun the SDMA from another address
```

In these two examples, the SDMA exits debug mode and keeps executing the code fetched from the memory.

7.2.2.14.5.6 Single Stepping in RAM

To execute a program step-by-step from the RAM, insert software breakpoints in the program flow at appropriate places so that the SDMA only executes one instruction before returning to debug mode.

First, read the next instruction to execute in the RAM. Then, depending on the value of this instruction, compute the address where a software breakpoint instruction should be inserted. The instruction at the corresponding address must be saved, and, the software breakpoint instruction is inserted. After restarting the SDMA, there is only one instruction executed before meeting the software breakpoint.

The following example shows the macro functions READ and WRITE, which correspond to the sequence of commands (described above) used to access the memory.

NOTE

The data read from the memory are 32-bit values, while the instructions are 16-bit values only. This is why it is best to only use addresses divided by two when accessing the memory.

READ and WRITE Macro Functions

```
next_instr = READ(run_addr/2);                        // read the next instruction to execute
// the tool now has to compute the address where the breakpoint
// instruction should be inserted, this address is the "bkpt_addr"
```

```
instr_save = READ(bkpt_addr/2);                          // save the instruction before
overwriting
STORE("bkpt instruction",bkpt_addr/2);                        // store the bkpt instruction
in memory
exec_core("jmp run_addr");                              // rerun the SDMA
rstatus(-);                                          // wait for the SDMA to enter debug mode
...
rstatus(-);
STORE(instr_save,bpkt_addr/2);                           // restore the instruction
overwritten
```

In case of branched conditional instructions, a breakpoint instruction should be written at the two possible target addresses.

7.2.2.14.5.7 Single Stepping in ROM

No single-step mechanism is supported in ROM. The program code can be loaded in the RAM, where the single-step mechanism can be executed.

7.2.2.14.6 OnCE Event Detection Unit

The event detection unit watches signals from the data memory bus (DMBUS), which the SDMA core uses to access its RAM, ROM, and memory mapped registers.

A debug request is sent to the OnCE controller when user-defined conditions on address and/or data values are true.

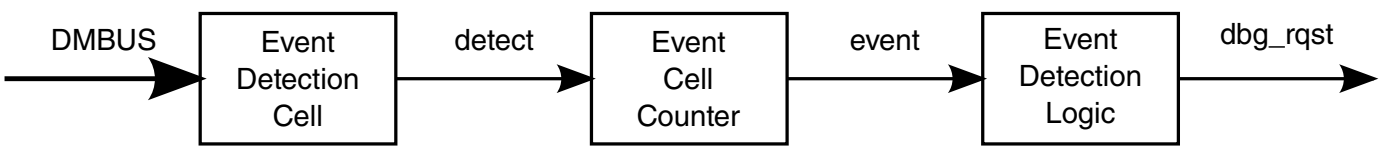


Figure 7-13. Event Detection Unit

A counter, provided with the detection cell, is decreased after an event detection. A debug request is sent to the core only when the counter reaches the value of 0. It is possible to disable the use of the counter if a debug request has to be generated after each event detection.

The event cell is the basic block that supports hardware breakpoints on an address value and/or data values coming from the SDMA memory bus. The trigger condition that generates the debug request is a mixed condition based on those values.

The following figure shows the event cell architecture. The event cell contains the address (stored in the memory address register) and the data (stored in the memory data register) used during the last memory access. There are some user-defined reference

values located in memory mapped registers-the event cell addresses, the event cell address mask, the event cell data, and the event cell data mask. These registers are accessed by standard load/store instructions just like regular memory locations.

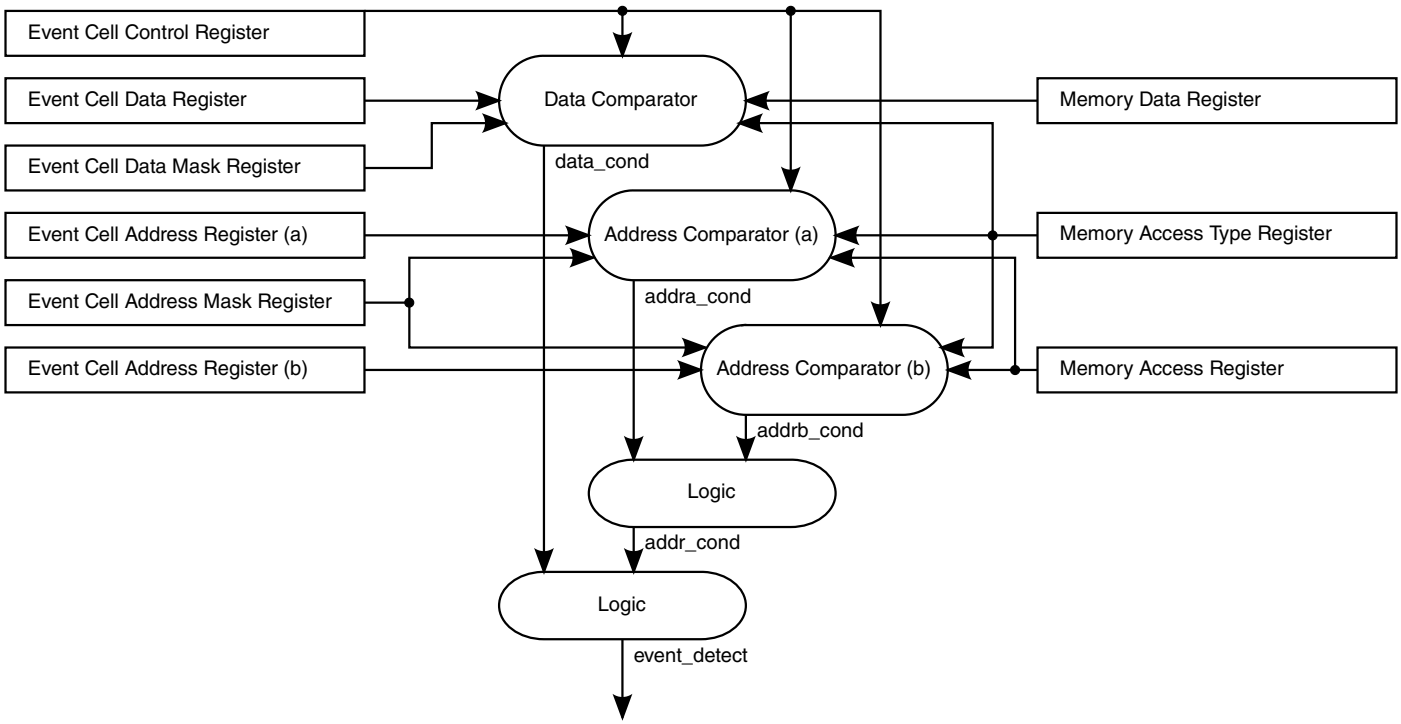


Figure 7-14. Event Cell Architecture

To define a memory breakpoint, three conditions are taken into account: The first two conditions are comparisons of the current memory address with user-defined reference addresses (these conditions are called addressA and addressB). The third condition consists of a comparison between the data received on the DMBUS and a user-defined reference data (this condition is called data). An intermediate address condition is set to express a dependency between addressA and addressB conditions.

7.2.2.14.7 Clock Gating and Reset

This section details how to use the clocks and handle the reset signals.

7.2.2.14.7.1 Clocks

Because the SDMA uses clock gating to save power, it is necessary to disable the clock gating and force the clocks to be enabled when using the OnCE.

When the OnCE is accessed through its JTAG interface, clock gating must be disabled outside the SDMA via a dedicated SDMA input port clk_gating_off. The reason why detection is not performed automatically by the SDMA internal hardware is that it would cost power to monitor activity on the JTAG interface.

When the OnCE is accessed through the Arm platform Control interface, clock gating is automatically turned off. This is done when bit 0 of the ONCE_ENB register (see OnCE Enable (SDMAARM_ONCE_ENB)) is set. A write access to this register is possible even when the OnCE clock is not running. If the Arm platform access is used, the bit in the ONCE_ENB register must be set before any attempt to access any other OnCE register.

7.2.2.14.7.2 Resets

The OnCE reset is different from the SDMA main reset.

Normally, activating the SDMA reset while keeping the OnCE reset inactive (when possible) enables you to reset the core without having to reprogram the OnCE.

7.2.2.14.8 Real Time Features

To rebuild the skeleton of a program execution, it is necessary to store the addresses of the program instructions where jumps are taken: A trace buffer is therefore provided. A real time buffer has also been added to receive data values written during a program execution.

The content of this register may be exported through JTAG ports without stopping the core.

7.2.2.14.8.1 Trace Buffer

The Trace Buffer is a 32-stage buffer that contains appropriate information to identify the 32 last changes of flow detected during a program execution.

The following figure shows an overview of the Trace Buffer.

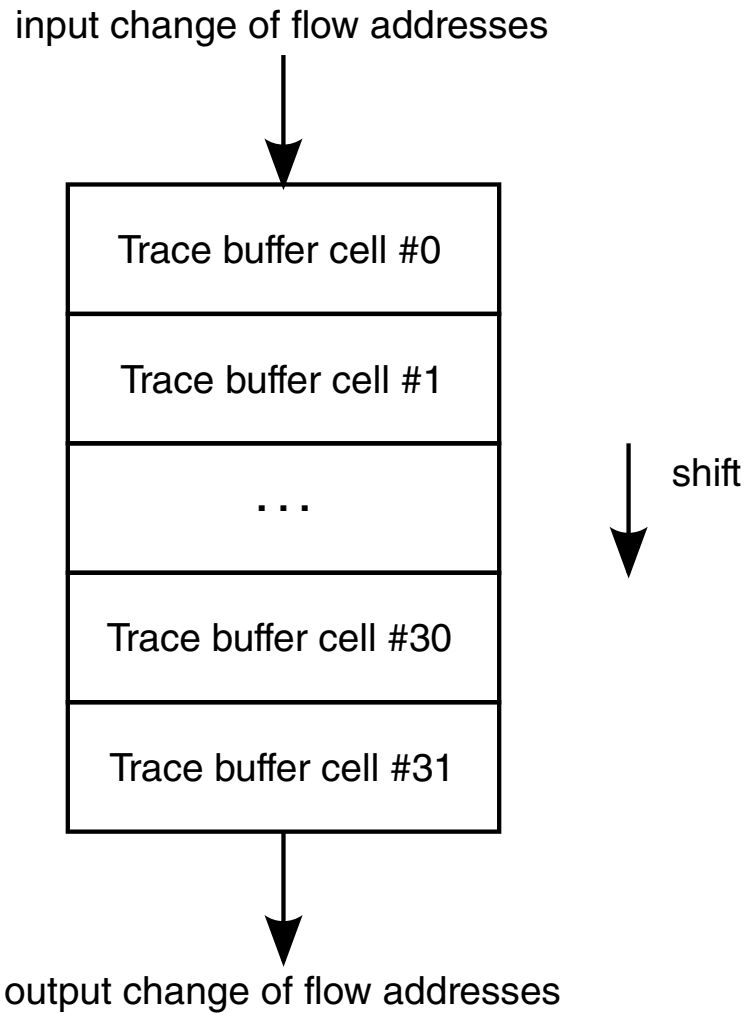


Figure 7-15. Trace Buffer

Each cell of the trace buffer contains two reference addresses and a flag. The flag is set when the addresses stored in the cell correspond to a valid change of flow; otherwise, the flag is cleared. The three most significant bits are unused.

After every change of flow detection, the address of current instruction and the address of the target instruction are stored at the top of the Trace Buffer (cell #0). The flag in the cell is set to indicate that a valid change of flow was detected. Former cell values are shifted one level down. The Trace Buffer contains the 32 last changes of flow. All the flags are reset on a software or a hardware reset, and after each transition from debug mode to user mode.

A memory mapped register of SDMA core, the Trace Buffer register (TB), is provided to read the content of the Trace Buffer. This operation should be done in debug mode. Performing a read access to the Trace Buffer register returns the content of the bottom of the Trace Buffer (cell #31). After every read access, the trace buffer is shifted one level down, and the flag at the top of the trace buffer is cleared.

A typical OnCE command sequence that retrieves the oldest change-of-flow information is a follows:

```
exec_once("mov r1, TB");                              // stores the oldest change-of-flow in
GReg1
dmov(-);                                          // retrieves GReg1 contents
```

This sequence requires the SDMA to be put in debug mode.

7.2.2.14.8.2 Real Time Buffer

The Real Time Buffer register (RTB) is a memory mapped register that can be accessed as a regular memory location by the SDMA core during program execution. This register is located in the OnCE.

Executing ar rbuffer command (see The OnCE Controller for further details) exports the content of this register through JTAG ports.

When a write access is performed at the memory location corresponding to the RTB, the receive flag (for example, the RCV bit) is set in the OnCE Status Register (OSR). This flag is cleared at the end of the execution of a rbuffer command.

NOTE

Every write access to the RTB memory location updates the RTB register even if the RCV flag is set. The RTB is cleared on a JTAG reset.

7.2.2.14.8.3 Emulation Pin

The debug_matched_event emulation pin reflects the matching condition status detected by the Event Detection Unit.

Since it can be necessary to detect conditions without triggering debug requests, it is possible to disable the generation of debug requests by the Event Detection Unit and still have the matching condition available on the emulation pin. This can be done by clearing the EN flag in the ECTL register.

7.2.2.14.8.4 Real-Time Debug Outputs

The table found here shows the debug signals that are available at the SDMA boundaries. Their availability at chip boundaries depends on the project.

Table 7-43. Real-Time Debug Output Pins

Pin	Description
debug_core_state[3:0]	The core_state bits reflect the state of the SDMA core. • The "Program" state is the usual instruction execution cycle. • The "Data" state is inserted when there are wait-states during a load or a store on the data bus (ld or st). • The "Change of Flow" state is the second cycle of any instruction that breaks the sequence of instructions (jumps and channel switching instructions). • The "Change of Flow in Loop" state is used when an error causes a hardware loop exit. • The "Debug" state means the SDMA is in debug mode. • The "Functional Unit" state is inserted when there are wait-states during a load or a store on the functional units bus (ldf or stf). • In "Sleep" modes, no script is running (this is the core idle state); the "after Reset" is slightly different because no context restoring phase will happen when a channel is triggered: The script located at address 0 is executed (boot operation). • The "in Sleep" states are the same as above except they do not have any corresponding channel: they are used when entering debug mode after reset; the reason is that it is necessary to return to the "Sleep after Reset" state when leaving debug mode. 0 Program 1 Data 2 Change of Flow 3 Change of Flow in Loop 4 Debug 5 Functional Unit 6 Sleep 7 Context Switch Saving Channel 8 Program in Sleep 9 Data in Sleep 10 Change of Flow in Sleep 11 Change of Flow in Loop in Sleep 12 Debug in Sleep 13 Functional Unit in Sleep 14 Sleep after Reset 15 Context Switch Restoring Channel
debug_yield	Pulse that is active when a yield (done 0) or a yieldge (done 1) instruction is executed. 0 - 1 yield/yieldge executed
debug_core_run	Active when the SDMA core is executing instructions. 0 Debug or sleep mode

Table continues on the next page...

Table 7-43. Real-Time Debug Output Pins (continued)

Pin	Description
	1 Run mode
debug_event_channel_sel	Indicates if debug_event_channel displays current channel or last received event 0- debug_event_channel[5:0] gives the number of the current channel 1- debug_event_channel[5:0] gives the number of the last received event
debug_event_channel[5:0]	Gives the number of any DMA request as soon as it is received or the number of the current channel. The value of debug_event_channel_sel indicates if debug_event_channel displays the current channel or last received event. The signal debug_event_channel_sel must be observed to determine what information is provided on debug_event_chanel at any given time.
debug_pc[13:0]	Program Counter value; it has a meaning when the core is in run mode.
debug_mode	Set when the core is in debug. 0 - 1 Core is in debug
debug_bus_error	Set when an error was received during a load or a store (ld, st, ldf, or stf instruction) and registered in SF or DF flag. 0 No error during last load/store 1 Error during last load/store
debug_bus_device[4:0]	Indicates the device or functional unit that is accessed by the current instruction. The debug_bus_device output is always valid when in sleep mode, debug mode, or executing any instruction that does not access the functional units or the memory mapped devices, "no access" is output. 0 No access 1 MSA 2 MDA 3 MD 4 MS 5 PSA 6 PDA 7 PD 8 PS 9 RESERVED 10 RESERVED 11 RESERVED 12 RESERVED 13 CA 14 CS 15 Reserved 16 Memory (RAM or ROM) 17 Memory mapped register

Table continues on the next page...

Table 7-43. Real-Time Debug Output Pins (continued)

Pin	Description
	18 Peripheral #1 19 Peripheral #2 20 Peripheral #3 21 Peripheral #4 22 Peripheral #5 23 Peripheral #6 24 Peripheral #7 25 Peripheral #8 26 Peripheral #9 27 Peripheral #10 28 Peripheral #11 29 Peripheral #12 30 Peripheral #13 31 Peripheral #14
debug_bus_rwb	Indicates the direction of the access given by debug_bus_device 0 Write access (st or stf) 1 Read access (ld or ldf)
debug_matched_dmbus	Pulse indicating the OnCE event detection unit has detected a match on the data bus during an access to memory (RAM or ROM), a memory mapped register or a peripheral that is hooked to the SDMA. 0 - 1 data bus match detected
debug_rtbuffer_write	Pulse indicating when the real-time buffer is written by the core. 0 - 1 RTB was modified
debug_evt_chn_lines[7:0]	Eight lines that generate short pulses when DMA requests are received or channels are (re)started. Every line is controlled through two parameters defined in registers Cross-Trigger Events Configuration Register 1 (SDMAARM_XTRIG_CONF1) (as described in SDMAARM). The following two parameters are available for every line: • CNF-Indicates what is monitored on the line: 0 for a channel start, 1 for a DMA request reception • NUM[5:0]-Gives the number of the DMA request or channel to monitor

The matched_event emulation pin reflects the matching condition status detected by the Event Detection Unit. Because it can be necessary to detect conditions without triggering debug requests, it is possible to disable the generation of debug requests by the Event Detection Unit and still have the matching condition available on the emulation pin. This can be done by clearing the EN flag in the ECTL register.

All real-time debug outputs are disabled by default (for example, they are stuck to 0) to avoid power consumption when they are not used. They are enabled when bit 11 (RTDOBS) of the Configuration Register (SDMAARM_CONFIG) is set. Signals provided to the system JTAG controller for SDMA debug mode status will also be enabled when the *clk_gating_off* input is asserted.

7.2.3 Instruction Set

7.2.3.1 Instruction Encoding

This section presents a short summary of the instruction codes. All context switch instructions are listed for information only; they cannot function properly out of the context switch routine.

```
x...x - don't care

rrr - destination/source general register

sss - additional source general register

bbb - general register used as address base register

ddddd - address displacement

nnnnn - bit number
uuuuuuuu - function unit command bits

pppppppp - branch displacement (signed)

iiiiiiii - 8-bit immediate

jjj - control bit to clear

ff - flag to clear
00000jjj00000000        - done (done,yield,wait)
00000jjj00000001        - notify
00000xxx00000010        - reserved
00000xxx00000011        - reserved
00000xxx00000100        - reserved
0000000000000101     - softBkpt
0000000100000101     - reserved
0000001000000101     - reserved
0000001100000101     - reserved
0000010000000101     - reserved
0000010100000101     - reserved
0000011000000101     - reserved
0000011100000101     - reserved
0000000000000110     - ret
0000000100000110     - reserved
0000001000000110     - reserved
0000001100000110     - reserved
0000010000000110     - reserved
0000010100000110     - reserved
0000011000000110     - reserved
0000011100000110     - reserved
```

```
000000ff00000111       - clrf ff
0000010000000111     - reserved
0000010100000111     - reserved
0000011000000111     - reserved
0000011100000111     - illegal
00000rrr00001000        - jmpr r
00000rrr00001001        - jsrr
00000rrr00001010        - ldrpc r
00000rrr00001011        - reserved
00000rrr000011xx          - reserved
00000rrr00010000        - revb
00000rrr00010001        - revblo
00000rrr00010010        - rorb
00000rrr00010011        - reserved
00000rrr00010100        - ror1
00000rrr00010101        - lsr1
00000rrr00010110        - asr1
00000rrr00010111        - lsl1
00000rrr001nnnnn        - bclri r,n
00000rrr010nnnnn        - bseti r,n
00000rrr011nnnnn        - btsti r,n
00000xxx10000xxx           - reserved
00000rrr10001sss           - mov
00000rrr10010sss           - xor
00000rrr10011sss           - add
00000rrr10100sss           - sub
00000rrr10101sss           - or
00000rrr10110sss           - andn
00000rrr10111sss           - and
00000rrr11000sss           - tst
00000rrr11001sss           - cmpeq
00000rrr11010sss           - cmplt
00000rrr11011sss           - cmphs
0000011011100000     - reserved
0000011011100001     - reserved
0000011011100010     - cpShReg
0000011011100011     - reserved
0000011011100100     - reserved
0000011011100101     - reserved
0000011011100110     - reserved
0000011011100111     - reserved
00000xxx11101xxx           - reserved
00000xxx11110xxx           - reserved
00000xxx11111xxx           - reserved
00001rrriiiiiiii                - ldi r,i
00010rrriiiiiiii                - xori r,i
00011rrriiiiiiii                - addi r,i
00100rrriiiiiiii                - subi r,i
00101rrriiiiiiii                - ori r,i
00110rrriiiiiiii                - andni r,i
00111rrriiiiiiii                - andi r,i
01000rrriiiiiiii                - tsti r,i
01001rrriiiiiiii                - cmpeqi r,i
01010rrrddddbbb         - ld r,(d,b)
01011rrrddddbbb         - st r,u
01100rrruuuuuuuu         - ldf r,u
01101rrruuuuuuuu         - stf r,u
011100xxxxxxxxxx               - reserved
011101xxxxxxxxxx               - reserved
011110ffnnnnnnnn       - Loop ff flags are reset
01111100pppppppp     - bf pc=pc+signed(pppppppp)+1
01111101pppppppp     - bt pc=pc+signed(pppppppp)+1
01111110pppppppp     - bsf pc=pc+signed(pppppppp)+1
01111111pppppppp     - bdf pc=pc+signed(pppppppp)+1
10aaaaaaaaaaaaaa     - jmp absolute
11aaaaaaaaaaaaaa     - jsr absolute
```

7.2.3.2 SDMA Instruction Set

This section describes all the useful instructions from the SDMA set.

Table 7-44. SDMA Instruction List

Instruction	Description	Page
ADD	Addition	ADD (Addition)
ADDI	Add with Immediate Value	ADDI (Add with Immediate Value)
AND	Logical AND	AND (Logical AND)
ANDI	Logical AND with Immediate Value	ANDI (Logical AND with Immediate Value)
ANDN	Logical AND NOT	ANDN (Logical AND NOT)
ANDNI	Logical AND with Negated Immediate Value	ANDNI (Logical AND with Negated Immediate Value)
ASR1	Arithmetic Shift Right by 1 Bit	ASR1 (Arithmetic Shift Right by 1 Bit)
BCLRI	Bit Clear Immediate	BCLRI1 (Bit Clear Immediate)
BDF	Conditional Branch if Destination Fault	BDF (Conditional Branch if Destination Fault)
BF	Conditional Branch if False	Functional Units Programming Model
BSETI	Bit Set Immediate	BSETI (Bit Set Immediate)
BSF	Conditional Branch if Source Fault	BSF (Conditional Branch if Source Fault)
BT	Conditional Branch if True	BT (Conditional Branch if True)
BTSTI	Bit Test immediate	BTSTI (Bit Test immediate)
CLRF	Clear Arm platform flags	CLRF (Clear Arm platform flags)
CMPEQ	Compare for Equal	CMPEQ (Compare for Equal)
CMPEQI	Compare with Immediate for Equal	CMPEQI (Compare with Immediate for Equal)
CMPHS	Compare for Higher or Same	CMPHS (Compare for Higher or Same)
CMPLT	Compare for Less Than	CMPLT (Compare for Less Than)
cpShReg	Update Context of PCU Registers and Flags	cpShReg (Update Context of PCU Registers and Flag)
DONE	DONE, Yield	DONE (DONE, Yield)
ILLEGAL	ILLEGAL Instruction	ILLEGAL (ILLEGAL Instruction)
JMP	Unconditional Jump Immediate	JMP (Unconditional Jump Immediate)
JMPR	Unconditional Jump	JMPR (Unconditional Jump)
JSR	Unconditional Jump to Subroutine Immediate	JSR (Unconditional Jump to Subroutine Immediate)
JSRR	Unconditional Jump to Subroutine	JSRR (Unconditional Jump to Subroutine)
LD	Load Register	LD (Load Register)
LDF	Load Register from Functional Unit	LDF (Load Register from Functional Unit)
LDI	Load Register with Immediate Value	LDI (Load Register with Immediate Value)
LDRPC	Load from RPC to Register	LDRPC (Load from RPC to Register)

Table continues on the next page...

Table 7-44. SDMA Instruction List (continued)

Instruction	Description	Page
LOOP	Hardware Loop	LOOP (Hardware Loop)
LSL1	Logical Shift Left by 1 Bit	LSL1 (Logical Shift Left by 1 Bit)
LSR1	Logical Shift Right by 1 Bit	LSR1 (Logical Shift Right by 1 Bit)
MOV	Logical Move	MOV (Logical Move)
NOTIFY	Notify to Arm platform	NOTIFY (Notify to Arm platform)
OR	Logical OR	OR (Logical OR)
ORI	Logical OR with Immediate Value	ORI (Logical OR with Immediate Value)
RET	Return from Subroutine	RET (Return from Subroutine)
REVB	Reverse Byte Order	REVB (Reverse Byte Order)
REVBLO	Reverse Low Order Bytes	Reverse Low Order Bytes(REVBLO)
ROR1	Rotate Right by 1 Bit	ROR1 (Rotate Right by 1 Bit)
RORB	Rotate Right by 1 Byte	RORB (Rotate Right by 1 Byte)
SOFTBKPT	Software Breakpoint	SOFTBKPT (Software Breakpoint)
ST	Store Register	ST (Store Register)
STF	Store Register in Functional Unit	STF (Store Register in Functional Unit)
SUB	Subtract	SUB (Subtract)
SUBI	Subtract with Immediate	SUBI (Subtract with Immediate)
TST	Test with Zero	TST (Test with Zero)
TSTI	Test Immediate	TSTI (Test Immediate)
XOR	Logical Exclusive OR	XOR (Logical Exclusive OR)
XORI	Exclusive OR with Immediate	XORI (Exclusive OR with Immediate)

7.2.3.2.1 ADD (Addition)

Operation:

```
GReg[r] ← GReg[s] + GReg[r]

T ← (GReg[r] == 0)
```

Assembler:

```
Syntax: add r,s

Example: add 0,3
```

ADD GReg[3] and GReg[0] and store the result in GReg[0]

CPU Flags: T

Cycles: 1

Description: Performs the ADDition of the source general register *s* and the destination general register *r*, and stores the result in the destination general register *r*. The T flag is set if the result of the operation is 0. It is cleared if the result is not 0.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	0	1	1	s	s	s

Instruction Fields:

rrr / sss - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.2 ADDI (Add with Immediate Value)

Operation:

```
GReg[r] ← GReg[r] + immediate

T ← (GReg[r] == 0)
```

Assembler:

```
Syntax: addi r,immediate

Example: add 6,112
```

ADD GReg[6] and decimal value 112 and store the result in GReg[6]

CPU Flags: T

Cycles: 1

Description: Adds a 0-extended immediate value to a general register; stores the result in the general register. The flag T is set when the result of the operation is 0; otherwise, it is cleared. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

iiiiiiii - immediate value:

```
00000000 - 0

00000001 - 1

 ...

11111110 - 254

11111111 - 255
```

7.2.3.2.3 AND (Logical AND)

Operation:

```
GReg[r] ← GReg[s] & GReg[r]
```

Assembler:

```
Syntax: and r,s

Example: and 1,2
```

AND GReg[1] and GReg[2] and store the result in GReg[1]

CPU Flags: Unaffected

Cycles: 1

Description: Performs the AND of the source general register *s* and the destination general register *r*, and stores the result in the destination general register *r*.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	1	1	1	s	s	s

Instruction Fields:

rrr / sss - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.4 ANDI (Logical AND with Immediate Value)

Operation:

```
GReg[r] ← GReg[r] & immediate
```

Assembler:

```
Syntax: andi r,immediate

Example: andi 7,45
```

AND GReg[7] and decimal value 45 and store the result in GReg[7]

CPU Flags: unaffected

Cycles: 1

Description: Performs an AND between a 0-extended immediate value and a general register; stores the result in the general register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	1	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

iiiiiiii - immediate value:

```
00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255
```

7.2.3.2.5 ANDN (Logical AND NOT)

Operation:

```
GReg[r] ← ~GReg[s] & GReg[r]
```

Assembler:

```
Syntax:andn r,s

Example: andn 3,4
```

AND GReg[3] and NOT GReg[4] (bit inverted) and store the result in GReg[3]

CPU Flags: Unaffected

Cycles: 1

Description: Performs the AND of the negation of the source general register *s* and the destination general register *r*, and stores the result in the destination general register *r*.

Instruction Format:

Table 7-45. Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	1	1	0	s	s	s

Instruction Fields:

rrr /sss - destination register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.6 ANDNI (Logical AND with Negated Immediate Value)

Operation:

```
GReg[r] ← GReg[r] & ~immediate
```

Assembler:

```
Syntax: andni r,immediate

Example: andni 0,2
```

AND GReg[0] and decimal value -3 (inverted 32-bit value 2) and store the result in GReg[0]

CPU Flags: unaffected

Cycles: 1

Description: Performs an AND between the negation of a 0-extended 8-bit immediate value and a general register; stores the result in the general register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	0	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

iiiiiiii - immediate value:

```
00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255
```

7.2.3.2.7 ASR1 (Arithmetic Shift Right by 1 Bit)

Operation:

```
GReg[r]:{b31,b30,...,b1,b0} ← GReg[r]:{b31,b31,b30,...,b1}
```

Assembler:

```
Syntax: asr1 r

Example: asr1 3
```

divide by 2 the signed value of GReg[3] and store the result in GReg[3]

CPU Flags: Unaffected

Cycles: 1

Description: Shift the bits of any general register to the right and keep the same sign: The left bit (bit 31) is kept untouched.

Instruction Format:

Table 7-46. Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	1	1	0

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.8 BCLRI1 (Bit Clear Immediate)

Operation:

```
GReg[r]:{b31,...,b(i+1),0,b(i-1),...,b0} ← GReg[r]:{b31,...,b(i+1),b(i),b(i-1),...,b0}
```

Assembler:

```
Syntax: bclri r,i

Example: bclri 1,12
```

clear bit 12 in GReg[1]

CPU Flags: Unaffected

Cycles: 1

Description: Clear the bit of register r specified by the 5-bit immediate field

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	1	i	i	i	i	i

rrr - register field:

```
000 - GReg[0]
```

```
001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

iiiii - immediate value:

```
00000 - 0

00001 - 1

...

11110 - 30

11111 - 31
```

7.2.3.2.9 BDF (Conditional Branch if Destination Fault)

Operation:

```
if (DF == 1) PC ← PC + 1 + displacement else PC ← PC + 1
```

Assembler:

```
Syntax:bdf label

Example: bdf LLL
```

Jump to LLL if DF is set, or go to the next instruction if DF is cleared; the displacement value is calculated by the assembler.

CPU Flags: Unaffected

Cycles: 2 when the branch is done, 1 otherwise

Description: If flag DF is set, jump to the new address that is calculated by adding the sign-extended 8-bit displacement to the next PC address. If flag DF is cleared, no jump is performed: The next instruction is located at the next PC address.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	1	1	p	p	p	p	p	p	p	p

Instruction Fields:

pppppppp - signed displacement field:

```
00000000 - 0

00000001 - 1

...

01111110 - 126

01111111 - 127

10000000 - (-128)

10000001 - (-127)

...

11111110 - (-2)

11111111 - (-1)
```

7.2.3.2.10 BF (Conditional Branch if False)

Operation:

```
if (T == 0)

PC ← PC + 1 + displacement

else

PC ← PC + 1
```

Assembler:

```
Syntax: bf label

Example: bf LLL
```

Jump to LLL if T is cleared, or go to the next instruction if T is set. The displacement value is calculated by the assembler.

CPU Flags: Unaffected

Cycles: 2 when the branch is done, 1 otherwise

Description: Conditional branch: If flag T is cleared, jump to the new address that is calculated by adding the sign-extended 8-bit displacement to the next PC address. If flag T is set, no jump is performed: The next instruction is located at the next PC address.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	0	0	p	p	p	p	p	p	p	p

Instruction Fields:

ppppppppp - signed displacement field:

```
00000000 - 0

00000001 - 1

...

01111110 - 126

01111111 - 127

10000000 - (-128)

10000001 - (-127)

...

11111110 - (-2)

11111111 - (-1)
```

7.2.3.2.11 BSETI (Bit Set Immediate)

Operation:

```
GReg[r]:{b31,...,b(i+1),1,b(i-1),...,b0} ← GReg[r]:{b31,...,b(i+1),b(i),b(i-1),...,b0}
```

Assembler:

```
Syntax: bseti r,i

Example: bseti 6,5
```

Set bit 5 in GReg[6]

CPU Flags: Unaffected

Cycles: 1

Description: Sets bit number i in the selected General Register.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	1	0	i	i	i	i	i

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]
```

```
010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

iiiii - bit number field:

```
00000 - 0

00001 - 1

...

11110 - 30

11111 - 31
```

7.2.3.2.12 BSF (Conditional Branch if Source Fault)

Operation:

```
if (SF == 1) PC ← PC + 1 + displacement else PC ← PC + 1
```

Assembler:

```
Syntax: bsf label

Example: bsf LLL
```

Jump to LLL if SF is set, or go to the next instruction if SF is cleared. The displacement value is calculated by the assembler.

CPU Flags: Unaffected

Cycles: 2 when the branch is done, 1 otherwise

Description: Conditional branch: If flag SF is set, jump to the new address that is calculated by adding the sign-extended 8-bit displacement to the next PC address. If flag SF is cleared, no jump is performed: The next instruction is located at the next PC address.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	1	0	p	p	p	p	p	p	p	p

Instruction Fields:

pppppppp - signed displacement field:

```
00000000 - 0

00000001 - 1

...

01111110 - 126

01111111 - 127

10000000 - (-128)

10000001 - (-127)

...

11111110 - (-2)

11111111 - (-1)
```

7.2.3.2.13 BT (Conditional Branch if True)

Operation

```
if (T == 1)

PC ← PC + 1 + displacement

else

PC ← PC + 1
```

Assembler

```
Syntax: bt label

bt LLL
```

Jump to LLL if T is set, or go to the next instruction if T is cleared. The displacement value is calculated by the assembler.

CPU Flags: Unaffected

Cycles: 2 when the branch is done, 1 otherwise

Description: Conditional branch: If flag T is set, jump to the new address that is calculated by adding the sign-extended 8-bit displacement to the next PC address. If flag T is cleared, no jump is performed: The next instruction is located at the next PC address.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	0	1	p	p	p	p	p	p	p	p

pppppppp - signed displacement field:

```
00000000 - 0

00000001 - 1

...

01111110- 126

01111111 - 127

10000000 - (-128)

10000001 - (-127)

...

11111110 -  (-2)

11111111 - (-1)
```

7.2.3.2.14 BTSTI (Bit Test immediate)

Operation:

```
T ← GReg[r]:b(i)
```

Assembler:

```
Syntax: btsti r,i

Example: btsti 2,29
```

Test bit 29 in GReg[2] and copy its value in flag T

CPU flags: T

Cycles: 1

Description: T is loaded with the value of bit number i from the selected general register.

Instruction Format:

Table 7-47. Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	1	1	i	i	i	i	i

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]
```

```
010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

iiiii - bit number field:

```
0000 - 0

0001 - 1

...

11110 - 30

11111 - 31
```

7.2.3.2.15 CLRF (Clear Arm platform flags)

Operation:

```
 if (ff%2 == 0)

SF ← 0

if (ff/2 == 0)

DF ← 0
```

Assembler:

```
Syntax: clrf ff

Example: clrf 2
```

Clear flag SF and keep flag DF unchanged

CPU Flags: SF, DF

Cycles: 1

Description: Clears a selection of the Arm platform fault flags: SF, DF, both SF and DF or none can be cleared.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	f	f	0	0	0	0	0	1	1	1

Instruction Fields:

ff - flags field:

```
00 - clear SF and clear DF

01 - clear DF

10 - clear

SF 11 - no clear
```

7.2.3.2.16 CMPEQ (Compare for Equal)

Operation:

```
T ← (GReg[s] == GReg[r])
```

Assembler:

```
Syntax: cmpeq r,s

Example: cmpeq 7,5
```

Compare GReg[7] and GReg[5] and set flag T if they are equal

CPU flags: T

Cycles: 1

Description: Subtracts the destination general register *r* from the source general register *s*, and sets T if the result is 0, clears T if the result is not 0.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	1	0	0	1	s	s	s

Instruction Fields:

rrr / sss - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.17 CMPEQI (Compare with Immediate for Equal)

Operation:

```
T ← (GReg[r] == immediate)
```

Assembler:

```
Syntax: cmpeqi r,immediate

Example: cmpeqi 2,13
```

Compare GReg[2] and decimal value 13 and set flag T if they are equal

CPU Flags: T

Cycles: 1

Description: Subtracts the 0-extended 8-bit immediate value from the general register, and sets T if the result is 0, clears T if the result is not 0. The immediate value is the low-order byte of the instruction.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

rrr - destination register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

iiiiiiii - immediate value:

```
00000000 - 0

00000001 - 1

...

11111110 - 254
```

```
11111111 - 255
```

7.2.3.2.18 CMPHS (Compare for Higher or Same)

Operation:

```
T ← (GReg[r] ≥ GReg[s])
```

Assembler:

```
Syntax: cmphs r,s

Example: cmphs 0,1
```

Compare GReg[0] and GReg[1] and set flag T if GReg[0] is higher than or equal to GReg[1]

CPU Flags: T

Cycles: 1

Description: Compares the destination general register *r* and the source general register *s*, and sets T if the destination general register *r* is higher than or equal to the source general register *s*, clears T otherwise. The comparison is unsigned.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	1	0	1	1	s	s	s

Instruction Fields:

rrr / sss - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.19 CMPLT (Compare for Less Than)

Operation:

```
T ← (GReg[r] < GReg[s])
```

Assembler:

```
Syntax: cmplt r,s

Example: cmplt 7,4
```

Compare GReg[7] and GReg[4] and set flag T if GReg[7] is lower than GReg[4]

CPU Flags: T

Cycles: 1

Description: Compares the destination general register *r* and the source general register *s*, and sets T if the destination general register *r* is lower than the source general register *s*, clears T otherwise. The comparison is signed.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	1	0	1	0	s	s	s

rrr / sss - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.20 cpShReg (Update Context of PCU Registers and Flag)

Assembler:

```
Syntax: cpShReg
```

CPU Flags: none

Cycles: 1

Description: SF, RPC, T, PC,LM, EPC, DF, and SPC registers are updated according to the value of their corresponding bits in the context memory. This instruction must only be used in debug mode via the OnCE. It reverses the done *5* operation.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	0	1	1	1	0	0	0	1	0

7.2.3.2.21 DONE (DONE, Yield)

Operation:

```
if (jjj&6 == 2) HE[CCR] ← 0

if (jjj == 3) HI[CCR] ← 1

if (jjj == 4) EP[CCR] ← 0


if ((jjj == 0) && (NCP > CCP)) CCR ← NCR

else if ((jjj == 1) && (NCP >= CCP))

CCR ← NCR

else

CCR ← NCR
```

(CCR stands for Current Channel Register; NCR stands for Next Channel Register)

Assembler:

```
Syntax: done jjj

Example: done 3
```

Clear HE bit for the current channel, send an interrupt to the Arm platform for the current channel and reschedule.

CPU Flags: Unaffected

Cycles: Variable if a context switch is done, 1 otherwise

Description: Clears one of the channel enabling bits (HE or EP for the corresponding channel number) if required. Sends an interrupt to the corresponding Arm platform by setting the appropriate flag, if required (HI for the corresponding channel number). Reschedules according to the mode and the NCP (Next Channel Priority) and CCP (Current Channel Priority) values. According to the scheduling decision, the NCR (Next Channel Register) is copied to the CCR (Current Channel Register) and channel contexts are switched. If several channels with the same highest priority are pending, they are ordered by their number from 31 down to 0. The higher number is selected (for example, channel 26 is selected if channels 3, 12, 14, and 26 with the same highest priority are pending). If no flag is modified, the reschedule can allow the replacement of the current

channel by another channel with a priority strictly greater than the current channel priority (yield). Or, it can allow the replacement of the current channel by another channel with a priority greater than or equal to the current channel priority (yieldge). In the latter case, the selected channel will always be the first one with the same priority, starting from channel number 31 down to channel 0 (the current channel does not belong to the set of selectable channels).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	j	j	j	0	0	0	0	0	0	0	0

jjj - Channel Flags field:

000 - No channel flags affected: Reschedule only if the next channel priority is greater than current channel priority (yield)

001 - No channel flags affected: Reschedule only if the next channel priority is greater than or equal to the current channel priority (yieldge)

010 - Clear HE for the current channel and reschedule 011 - Clear HE, set HI for the current channel and reschedule 100 - Clear EP for the current channel and reschedule

101 - Reserved for debug to copy relevant registers into context memory

110 - RESERVED

111 - RESERVED

For the scheduling rules, refer to Scheduler Functional Description. Every possible done instruction is further described as follows:

- done 0/yield is executed by a channel script when it accepts preemption by a higher priority channel;
- done 1/yieldge is executed by a channel script when it accepts preemption by a higher priority channel and it also accepts a roll-up with other channels that have the same priority;
- done 2 is executed by a channel script that was triggered by a Arm platform start via the Channel Start (SDMAARM_HSTART) register, when its task is completed and it requires termination;
- done 3 is executed by a channel script that was triggered by a Arm platform start via the Channel Start (SDMAARM_HSTART) register, when its task is completed, it requires termination and it needs to trigger an interrupt to the Arm platform upon closure;

- done 4 is executed by a channel script that was triggered by a DMA request, when its task is completed and it requires termination;
- done 5 is used in debug mode only; it copies the PCU registers and flags to the context memory of the current channel;

7.2.3.2.22 ILLEGAL (ILLEGAL Instruction)

Operation:

```
PC ← 0001
```

Assembler:

```
Syntax: illegal
```

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the Illegal instruction routine located at address 0001. All unauthorized instructions result in an Illegal instruction behavior; however, the ILLEGAL instruction must be used to guarantee software compatibility with future versions of the SDMA.

Instruction Format

Table 7-48. Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

7.2.3.2.23 JMP (Unconditional Jump Immediate)

Operation:

```
PC ← absolute_address
```

Assembler:

```
Syntax: jmp label

Example: jmp LLL
```

The assembler translates the label to the exact address

CPU Flags:Unaffected

Cycles: 2

Description: Jumps to the absolute address contained the lower 14 bits of the instruction (the PC is a 14-bit register).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	a	a	a	a	a	a	a	a	a	a	a	a	a	a

aaaaaaaaaaaaaa - address field:

```
00000000000000 - 0

00000000000001 - 1

...

11111111111110 - 16382

11111111111111 - 16383
```

7.2.3.2.24 JMPR (Unconditional Jump)

Operation:

```
PC ← GReg[r]
```

Assembler:

```
Syntax: jmpr r

Example: jmpr 0
```

Jump to address stored in GReg[0]

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the absolute address contained in a General Register.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	0	1	0	0	0

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]
```

```
010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.25 JSR (Unconditional Jump to Subroutine Immediate)

Operation:

```
RPC ← PC + 1

PC ← absolute_address
```

Assembler:

```
Syntax: jsr r

Example:jsr LLL
```

Jumps to subroutine starting at LLL; the assembler translates the label to exact address

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the subroutine located at the absolute address contained the lower 14 bits of the instruction (the PC is a 14-bit register).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	a	a	a	a	a	a	a	a	a	a	a	a	a	a

aaaaaaaaaaaaaa - address field:

```
00000000000000 - 0

00000000000001 - 1

...

11111111111110 - 16382

11111111111111 - 16383
```

7.2.3.2.26 JSRR (Unconditional Jump to Subroutine)

Operation:

```
RPC ← PC + 1

PC ← GReg[r]
```

Assembler:

```
Syntax: jsrr r

Example:jsrr 5
```

Jumps to subroutine located at address stored in GReg[5]

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the subroutine at address contained in a General Register

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	0	1	0	0	1

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.27 LD (Load Register)

Operation:

```
GReg[r] ← [GReg[b] + displacement]

if (transfer_error)

SF ← 1

else

SF ← 0
```

Assembler:

```
Syntax: ld r,(b,displacement)

Example: ld 1,(2,23)
```

Loads data into GReg[1]; the data is located at address obtained by adding decimal value 23 to GReg[2]

CPU Flags: SF

Cycles: 2+n where n is 0 for ROM, RAM or memory mapped registers, and n is the number of wait-states of the peripheral for a peripheral access

Description: Adds a 5-bit 0-extended displacement to a base address in General Register b; the result is the address of the data to fetch on the DM bus. The data received from the bus is stored in the destination General Register r. If an error occurs during the transfer, the flag SF is set, else it is cleared.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	0	r	r	r	d	d	d	d	d	b	b	b

rrr / bbb - register field:

```
 000 - GReg[0]

 001 - GReg[1]

 ...

 111 - GReg[7]
```

ddddd - displacement value:

```
00000 - 0

00001 - 1

...

11111 - 31
```

7.2.3.2.28 LDF (Load Register from Functional Unit)

Operation:

```
GReg[r] ← [fu_address]

if (transfer_error)

SF ← 1
```

```
else

SF ←0
```

fu_address is an 8-bit field and depends on addressed functional unit

Assembler:

```
Syntax: ldf r,fu_address

Example: ldf 0,13
```

Loads data coming from the Burst DMA register MD into GReg[0]; it is a 32-bit access with no prefetch

CPU Flags: SF

Cycles: 1+n where n is the number of wait-states that may be inserted by the functional unit

Description: Sends an 8-bit address on the Functional Unit Bus (FU bus) and stores the data received from the bus in the destination General Register r. If an error occurs during the transfer, the flag SF is set, else it is cleared.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	0	r	r	r	f	f	f	f	f	f	f	f

See the following sections for more details of the LDF instruction usage with each functional unit:

- Burst DMA Read (ldf) for Burst DMA
- Peripheral DMA Read (ldf)-Read Mode for Peripheral DMA

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

ffffffff - functional unit source register and action (unspecified values are reserved):

```
00000000 - MSA

00000100 - MDA

00001001 - MD byte

00001010 - MD halfword

00001011 - MD word

00001100 - MS

00101001 - MD byte - prefetch

00101010 - MD halfword - prefetch

00101011 - MD word - prefetch

01000000 - DSA



11000000 - PSA

11001000 - PD

11010000 - PDA

11011000 - PD in copy mode (rrr contents are lost)

11101000 - PD - prefetch next data

11111111 - PS
```

7.2.3.2.29 LDI (Load Register with Immediate Value)

Operation:

```
GReg[r] ← immediate
```

Assembler:

```
Syntax: ldi r,immediate

Example: ldi 6,1
```

loads decimal value 1 into GReg[6]

CPU Flags: Unaffected

Cycles: 1

Description: Stores a 0-extended immediate value in a General Register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

iiiiiiii - immediate value:

```
00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255
```

7.2.3.2.30 LDRPC (Load from RPC to Register)

Operation:

```
GReg[r] ← RPC
```

Assembler:

```
Syntax: ldrpc r

Example: ldrpc 3
```

copies RPC to GReg[3]

CPU Flags: Unaffected

Cycles: 1

Description: Stores the contents of the RPC in a General Register. That instruction may be used to have more than one level of subroutines.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	0	1	0	1	0

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.31 LOOP (Hardware Loop)

Operation:

```
if (ff%2 == 0)
   SF ← 0
if (ff/2 == 0)
  DF ← 0
if ((GReg[0] == 0) || (SF == 1) || (DF == 1))
   PC ← PC + loop_size + 1
else
   {
      SPC ← PC + 1
      EPC ← PC + loop_size + 1
      LM ← 1
      PC ← PC + 1
   }
```

during every instruction execution in the loop:

```
if ((SF == 1) || (DF == 1))
 {
```

```
    LM ← 0
    PC ← EPC
  }
else if ((PC + 1) == EPC)
 {
    GReg[0] ← GReg[0] - 1
    if (GReg[0] == 0)
      {
         LM ← 0
         PC ← EPC
       }
     else
         PC ← SPC
   }
else
  PC ← nextPC(instruction)
```

after the execution of the last instruction of the loop body:

```
if (GReg[0] == 0)
   T ← 1
else
   T ← 0
```

Assembler:

```
Syntax: loop n{,ff}
Example: loop 3,1
```

Executes GReg[0] times the instructions comprised between PC+1 and PC+3 (included); ff=1 clears the DF flag before starting the loop. When omitted, the ff field is set to 0 (clearing both SF and DF).

CPU Flags: LM[1:0], T

Cycles: 2 when the loop count (GReg[0]) is 0 or SF or DF is set at loop start, 1+1 when the loop starts but exits abnormally (SF or DF set inside the loop which adds 1 cycle to the offending load or store to jump to EPC), 1 when the loop is executed normally

Description: The loop instruction executes a sequence of instructions several times. The number of times is given by the contents of GReg[0], the loop counter. SDMA will jump to the first instruction after the end of the loop if the value in GReg[0] is 0. Otherwise the SDMA enters loop mode. It sets the most significant bit of the LM flag that will only be reset once the last instruction of the last loop is executed. The instructions in the loop are executed GReg[0] times. The management of fault flags (SF and DF) is as follows. When entering the hardware loop, SF and DF can be cleared according to the ff field of the instruction. After that operation, if any flag is still set the loop will not be executed. The SDMA will jump to the first instruction after the end of the loop without entering loop mode. During the execution of the loop, if any fault flag is set by a LD, LDF, ST, or STF instruction, the SDMA will immediately exit loop mode and jump to the first instruction after the end of the loop. In that case, GReg0 is not decremented for that last piece of the loop body execution (even if the SF or DF flag is set at the last instruction of the loop body). The T flag reflects the state of GReg[0] after the end of the loop, which is an indicator of the complete execution of the loop. If the loop exited because of an error (SF or DF set), GReg[0] will not be 0 at the end of the loop, hence T will be cleared. If the loop executes without fault, GReg[0] will be 0 at the end of the loop, hence T will be set. The boundary case when a source or destination fault occurs at the last instruction of the last loop is considered as an anticipated exit of the loop, which causes the T flag to be cleared. If the last instruction executed before leaving the hardware loop also tries to modify the T flag, the flag is updated according to the value of GReg[0], NOT according to the result of the last executed instruction.

Limitations:

1. 1. Jump instructions (JMP, JMPR, JSR, JSRR, BF, BT, BSF, BDF) are not allowed inside the hardware loop.
2. 2. GReg[0] cannot be written to inside the hardware loop (it can be read).
3. 3. The empty loop (0 instruction in the body) is forbidden.
4. 4. If GReg[0] == 0 at the start of the loop, which causes a jump to EPC, the T flag is not updated.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	0	f	f	n	n	n	n	n	n	n	n

Instruction Fields:

```
ff - flags field:

00 - clear SF and clear DF

01 - clear DF

10 - clear SF
```

```
11 - no clear
```

nnnnnnnn - loop size

```
00000000 - empty loop: forbidden value

00000001 - 1 instruction in the loop

00000010 - 2 instructions in the loop

...

11111111 - 255 instructions in the loop
```

7.2.3.2.32 LSL1 (Logical Shift Left by 1 Bit)

Operation:

```
GReg[r]:{b30,...,b1,b0,0} ← GReg[r]:{b31,b30,...,b1,b0}
```

Assembler:

```
Syntax: lsl1 r

Example: lsl1 2
```

multiplies by 2 the value in GReg[2]

CPU Flags: Unaffected

Cycles: 1

Description: Shift the bits of any General Register to the left. The right bit (bit 0) is set to 0. No overflow is detected by the hardware.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	1	1	1

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]
```

```
111 - GReg[7]
```

7.2.3.2.33 LSR1 (Logical Shift Right by 1 Bit)

Operation:

```
GReg[r]:{0,b31,b30,...,b1} ← GReg[r]:{b31,b30,...,b1,b0}
```

Assembler:

```
Syntax: lsr1 r

Example: lsr1 4
```

divides by 2 the unsigned value contained in GReg[4]

CPU Flags: Unaffected

Cycles: 1

Description: Shift the bits of any General Register to the right. The left bit (bit 31) is set to 0.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	1	0	1

Instruction Fields:

rrr - destination register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.34 MOV (Logical Move)

Operation:

```
GReg[r] ← GReg[s]
```

Assembler:

```
Syntax: mov r,s

Example: mov 4,0
```

copies GReg[0] to GReg[4]

CPU Flags: Unaffected

Cycles: 1

Description: Move the contents of the source General Register s to the destination General Register r.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	0	0	1	s	s	s

Instruction Fields:

rrr / sss - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.35 NOTIFY (Notify to Arm platform)

Operation:

```
 if (jjj & 4 == 0)

 {

   if (jjj&2 == 2)

      HE[CCR] ← 0

   if (jjj&1== 1)

       HI[CCR] ← 1

  }

else if (jjj == 4)
```

```
   EP[CCR] ← 0
else
```

(CCR stands for Current Channel Register)

Assembler:

```
Syntax: notify jjj
Example: notify 3
```

clears the HE bit for the current channel and sends an interrupt to the Host for the current channel

CPU Flags: Unaffected

Cycles: 1

Description: Clears one of the channel enabling bits (HE or EP for the corresponding channel number) if required, sends an interrupt to the corresponding Arm platform by setting the appropriate flag if required (HI for the corresponding channel number).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	j	j	j	0	0	0	0	0	0	0	1

jjj - Channel Flags field:

```
000 - unused
001 - set HI for the current channel
010 - clear HE for the current channel
011 - clear HE, set HI for the current channel
100 - clear EP for the current channel
101 - RESERVED
110 - RESERVED
111 - RESERVED
```

7.2.3.2.36 OR (Logical OR)

Operation:

```
GReg[r] ← GReg[s] | GReg[r]
```

Assembler:

```
Syntax: or r,s
```

```
Example: or 3,6
```

ORs GReg[3] and GReg[6] and stores the result in GReg[3]

CPU Flags: Unaffected

Cycles: 1

Description: Performs the OR of the source General Register s and the destination General Register r, and stores the result in the destination General Register r.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	1	0	1	s	s	s

Instruction Fields:

rrr / sss - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.37 ORI (Logical OR with Immediate Value)

Operation:

```
GReg[r] ← GReg[r] | immediate
```

Assembler:

```
Syntax: ori r,immediate

Example: ori 1,56
```

ORs GReg[1] and the decimal value 56 and stores the result in GReg[1]

CPU Flags: unaffected

Cycles: 1

Description: Performs an OR between a 0-extended 8-bit immediate value and a General Register; stores the result in the General Register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	1	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

iiiiiiii - immediate value:

```
00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255
```

7.2.3.2.38 RET (Return from Subroutine)

Operation:

```
PC ← RPC
```

Assembler:

```
Syntax: ret
```

CPU Flags: Unaffected

Cycles: 2

Description: Return from subroutine.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

7.2.3.2.39 REVB (Reverse Byte Order)

Operation:

```
GReg[r]:{B3,B2,B1,B0} ← GReg[r]:{B0,B1,B2,B3}
```

Assembler:

```
Syntax: revb r

Example:  revb 5
```

reverses bytes order in GReg[5]

CPU Flags: Unaffected

Cycles: 1

Description: Reverse the byte order of any General Register.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	0	0	0

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.40 Reverse Low Order Bytes(REVBLO)

Operation:

```
GReg[r]:{B3,B2,B0,B1} ← GReg[r]:{B3,B2,B1,B0}
```

Assembler:

```
Syntax: revblo r

Example: revblo 0
```

reverses low order bytes in GReg[0]

CPU Flags: Unaffected

Cycles: 1

Description: Reverse both low order bytes of any General Register.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	0	0	1

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.41 ROR1 (Rotate Right by 1 Bit)

Operation:

```
GReg[r]:{b0,b31,b30,...,b1} ← GReg[r]:{b31,b30,...,b1,b0}
```

Assembler:

```
Syntax: ror1 r

Example: ror1 3
```

rotates bits to the right in GReg[3]

CPU Flags: Unaffected

Cycles: 1

Description: Rotate the bits of any General Register to the right.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	1	0	0

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.42 RORB (Rotate Right by 1 Byte)

Operation:

```
GReg[r]:{B0,B3,B2,B1} ← GReg[r]:{B3,B2,B1,B0}
```

Assembler:

```
Syntax: rorb r

Example: rorb 2
```

rotates bytes to the right in GReg[2]

CPU Flags: Unaffected

Cycles: 1

Description: Rotate the bytes of any General Register to the right.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	0	1	0

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.43 SOFTBKPT (Software Breakpoint)

Operation:

Stops the current script and enters debug mode

Assembler:

```
softbkpt
```

CPU Flags: Unaffected

Description: When the core executes this instruction, it has the same effect as receiving a debug request from the OnCE or via the external debug request input: the script execution halts, the PCU enters its debug state and waits for the OnCE commands that are described in OnCE and Real-Time Debug.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

7.2.3.2.44 ST (Store Register)

Operation:

```
[GReg[b] + displacement] ← GReg[r]
```

```
if (transfer_error)

 DF ← 1

 else

DF ← 0
```

Assembler:

```
Syntax: st r,(b,displacement)

Example: st 7,(0,9)
```

stores the value from GReg[7] into memory at address obtained by adding decimal value 9 to GReg[0]

CPU Flags: DF

Cycles: 2+n where n is 0 for ROM, RAM or memory mapped registers, and n is the number of wait-states of the peripheral for a peripheral access

Description: Adds a 5-bit 0-extended displacement to a base address in General Register b; the result is the address of the data to store on the DM bus. The data sent on the bus comes from the source General Register r. If an error occurs during the transfer, the flag DF is set, else it is cleared.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	1	r	r	r	d	d	d	d	d	b	b	b

Instruction Fields:

rrr / bbb - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

ddddd - displacement value:

```
00000 - 0
```

```
00001 - 1

...

11111 - 31
```

7.2.3.2.45 STF (Store Register in Functional Unit)

Operation:

```
[fu_address] ← GReg[r] 0

if (transfer_error) 0

DF ← 1 0

else 0

DF ← 0
```

fu_address is an 8-bit field

Assembler:

```
Syntax: stf r,fu_address

Example: stf 3,0x2B
```

stores the 32-bit contents of GReg[3] to the Burst DMA register MD; waits until the flush to external memory is completed

CPU Flags: DF

Cycles: 1+n where n is the number of wait-states that may be inserted by the functional unit

Description: Sends an 8-bit address on the Functional Unit Bus (FU bus) and sends the contents of the source General Register r on the bus. If an error occurs during the transfer, the flag DF is set, else it is cleared.

Table 7-49. Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	1	r	r	r	f	f	f	f	f	f	f	f

See the following sections for more details of the STF instruction usage with each functional unit:

- Burst DMA Write (stf) for Burst DMA
- Peripheral DMA Write (stf)-Write Mode for Peripheral DMA

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

ffffffff - functional unit destination register and action (unspecified values are reserved):

00000000 - MSA in incremented mode

00000100 - MDA in incremented mode

00001001 - MD byte

00001010 - MD halfword

00001011 - MD word

00001100 - clear MS error flag

00001111 - MS

00010000 - MSA in frozen mode

00010100 - MDA in frozen mode

00011000 - MD in copy mode - number of words in rrr

00100000 - MSA in incremented mode - start prefetch

00101000 - MD no data - flush

00101001 - MD byte - flush

00101010 - MD halfword - flush

00101011 - MD word - flush

00110000 - MSA in frozen mode - start prefetch

11000001 - PSA in frozen mode - 8-bit data width

11000010 - PSA in frozen mode - 16-bit data width

11000011 - PSA in frozen mode - 32-bit data width

11000101 - PSA in incremented mode - 8-bit data width

11000110 - PSA in incremented mode - 16-bit data width

11000111 - PSA in incremented mode - 32-bit data width

11001000 - PD

11001001 - PSA in decremented mode - 8-bit data width

11001010 - PSA in decremented mode - 16-bit data width

11001011 - PSA in decremented mode - 32-bit data width

11001100 - clear PS error flag

11001101 - PSA data width becomes 8-bit

11001110 - PSA data width becomes 16-bit

11001111 - PSA data width becomes 32-bit

11010001 - PDA in frozen mode - 8-bit data width

11010010 - PDA in frozen mode - 16-bit data width

11010011 - PDA in frozen mode - 32-bit data width

11010101 - PDA in incremented mode - 8-bit data width

11010110 - PDA in incremented mode - 16-bit data width

11010111 - PDA in incremented mode - 32-bit data width

11011001 - PDA in decremented mode - 8-bit data width

11011010 - PDA in decremented mode - 16-bit data width

11011011 - PDA in decremented mode - 32-bit data width

11011101 - PDA data width becomes 8-bit

11011110 - PDA data width becomes 16-bit

11011111 - PDA data width becomes 32-bit

11100001 - PSA in frozen mode - 8-bit data width - prefetch data

11100010 - PSA in frozen mode - 16-bit data width - prefetch data

11100011 - PSA in frozen mode - 32-bit data width - prefetch data

11100101 - PSA in incremented mode - 8-bit data width - prefetch data

11100110 - PSA in incremented mode - 16-bit data width - prefetch data

11100111 - PSA in incremented mode - 32-bit data width - prefetch data

11101001 - PSA in decremented mode - 8-bit data width - prefetch data

11101010 - PSA in decremented mode - 16-bit data width - prefetch data

11101011 - PSA in decremented mode - 32-bit data width - prefetch data

11101101 - PSA data width becomes 8-bit - prefetch data

11101110 - PSA data width becomes 16-bit - prefetch data

11101111 - PSA data width becomes 32-bit - prefetch data

11111111- PS

7.2.3.2.46 SUB (Subtract)

Operation:

```
GReg[r] ← GReg[r] - GReg[s]

T ← (GReg[r] == 0)
```

Assembler:

```
Syntax: sub r,s

Example: sub 4,7
```

SUBtracts GReg[7] from GReg[4] and stores the result in GReg[4]

CPU Flags: T

Cycles: 1

Description: Subtracts the source General Register s from the destination General Register r, and stores the result in the destination General Register r. The T flag is set if the result of the operation is 0; it is cleared if the result is not 0.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	1	0	0	s	s	s

Instruction Fields:

rrr / sss - register fields:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.47 SUBI (Subtract with Immediate)

Operation:

```
GReg[r] ← GReg[r] - immediate

T ← (GReg[r] == 0)
```

Assembler:

```
Syntax: sub r,immediate

Example: sub 1,255
```

SUBtracts decimal value 255 from GReg[1] and stores the result in GReg[1]

CPU Flags: T

Cycles: 1

Description: Subtracts a 0-extended 8-bit immediate value from a General Register; stores the result in the General Register. The flag T is set when the result of the operation is 0; otherwise, it is cleared. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	0	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

iiiiiiii - immediate value:

```
00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255
```

7.2.3.2.48 TST (Test with Zero)

Operation:

```
T ← ((GReg[s] & GReg[r]) != 0)
```

Assembler:

```
Syntax: tst r,s

Example: tst 2,3
```

ANDs GReg[2] and GReg[3] and sets T if the result is non-null

CPU Flags: T

Cycles: 1

Description: Performs the AND of the source General Register s and the destination General Register r, and sets T if the result is not 0, clears T if the result is 0.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	1	0	0	0	s	s	s

Instruction Fields:

rrr / sss - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.49 TSTI (Test Immediate)

Operation:

```
T ← ((GReg[r] & immediate) != 0)
```

Assembler:

```
Syntax: tsti r,immediate

Example: tsti 5,13
```

ANDs GReg[5] and decimal value 13 and sets T if the result is non-null

CPU Flags: T

Cycles: 1

Description: Performs the AND of a 0-extended 8-bit immediate value and the destination General Register r, and sets T if the result is not 0, clears T if the result is 0. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

rrr - destination register field:

```
000 - GReg[0]

001 - GReg[1]
```

```
010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

iiiiiiii - immediate value:

```
00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255
```

7.2.3.2.50 XOR (Logical Exclusive OR)

Operation:

```
GReg[r] ← GReg[s] ^ GReg[r]
```

Assembler:

```
Syntax: xor r,s

Example: xor 0,3
```

XORs GReg[0] and GReg[3] and stores the result in GReg[0]

CPU Flags: Unaffected

Cycles: 1

Description: Performs the eXclusive OR of the source General Register s and the destination General Register r, and stores the result in the destination General Register r.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	0	1	0	s	s	s

Instruction Fields:

rrr / sss - register field:

```
000 - GReg[0]

001 - GReg[1]
```

```
010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]
```

7.2.3.2.51 XORI (Exclusive OR with Immediate)

Operation:

```
GReg[r] ← GReg[r] ^ immediate
```

Assembler:

```
Syntax: xori r,immediate

Example: xor 7,5
```

XORs GReg[5] and decimal value 5 and stores the result in GReg[7]

CPU Flags: Unaffected

Cycles: 1

Description: Performs an eXclusive OR between a 0-extended 8-bit immediate value and a General Register; stores the result in the General Register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

rrr - register field:

```
000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]
```

```
111 - GReg[7]
```

iiiiiiii - immediate value:

```
00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255
```

7.2.3.2.52 YIELD, YIELDGE (DONE, Yield)

By default, unsupported assembler syntax. Can be aliased to the corresponding done instructions (yield = done 0; yieldge = done 1). Refer to the done instruction description DONE (DONE, Yield) .

7.2.4 Software Restrictions

7.2.4.1 Unsupported Burst DMA Access Sequence

The SDMA does not support triggering a pre-fetch followed by a flush of the Burst DMA without reading or writing any data. If the flush occurs while the background pre-fetch DMA operation is still in progress, it could result in un-defined behavior.

An example of the sequence which could result in undefined results is shown in the following example:

Instruction sequence not supported

```
stf r1, MSA|PF          ; Update source address, triggers data pre-fetch in the
                        ; background
mov R0,R0               ; Execute multiple assembly instructions, none of which
                        ; read
mov R0,R0               ; or write data to/from MD
stf MD|SZ0|FL           ; Flush FIFO without writing data. If the pre-fetch is still
                        ; in progress when this instruction is executed, there
                        ; could be undefined operation
```

A work-around to avoid any undesirable results is to first read MD to ensure the pre-fetch is complete before the flush is attempted.

Work-Around to previous example

```
stf r1, MSA|PF          ; Update source address, triggers data pre-fetch.
```

```
mov R0,R0              ; Execute multiple assembly instructions, none of which
                       ; read
mov R0,R0              ; or write data to/from MD
ldf r2, MD             ; dummy read of MD to ensure pre-fetch is complete
                       ; before the next instruction
stf MD|SZ0|FL          ; Flush FIFO without writing data
```

7.2.5 Application Notes

7.2.5.1 Data Structures for Boot Code and Channel Scripts

SDMA boot code downloads the different channel contexts and the scripts that will be executed on SDMA channels during the application.

The boot code is run after reset when channel 0 is started by the Arm platform. The boot code is also known as channel 0 script.

The boot code is based on the Channel Control Block (CCB) and Buffer Descriptor (BD) mechanisms that are data structures located into the Arm platform memory space. With these data structures, it is possible to instruct SDMA to download scripts and contexts but also to dump a context or a script to a destination data buffer. Channel scripts also use the CCB and BD data structures to pass instructions and/or pointers to data to be copied.

The format, processing, and field definition of the CCB and BD are defined and performed entirely by the software script rather than the SDMA hardware. An overview of the format and structure is provided here, but for complete details refer to the SDMA software documentation (see SDMA Scripts).

The CCB and BD data structures are accessed by SDMA using DMA and processed by the SDMA scripts. The ROM contains common sub-routines for processing these data structures which may be called by the bootload and channel scripts.

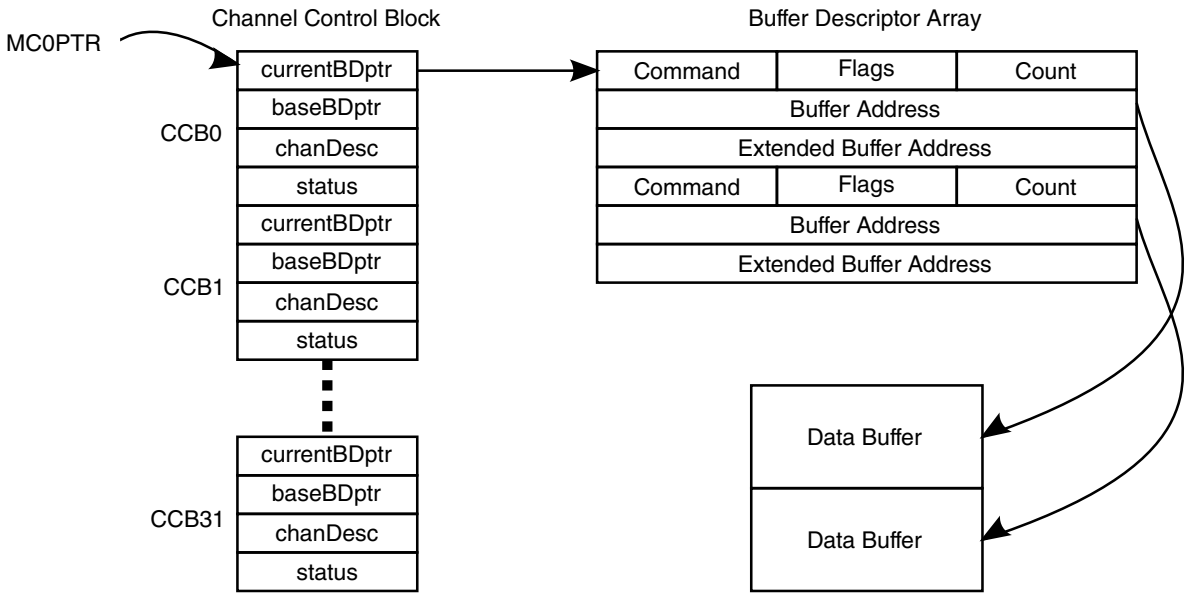


Figure 7-16. Data Structures Layout

The previous figure shows an example how these data structures are linked to pass command and pointers to data buffers. The SDMA's MC0PTR register holds the base address of the Channel 0 Control Block (CCB0). The Channel 0 control block holds a pointer to the array of buffer descriptors. The buffer descriptors are used to tell the channel 0 (boot channel) what to do as described Buffer Descriptor Format.

7.2.5.1.1 Buffer Descriptor Format

Buffer descriptors are three longs (32-bit words) in size as, shown in the figure found here.

A buffer descriptor describes the properties of the data buffer it points to. The buffer descriptors can be used for linear or circular data buffers in the Arm platform processor memory. The CCB contains a pointer to the base BD as well as the current BD.

Table 7-50. Buffer Descriptor

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Command								-	-	L	R	I	C	W	D	Count															
Buffer Address																															
Extended Buffer Address																															

Table 7-51. Buffer Descriptor Field Descriptions

Field	Description
31-24 Command	Command. The command field is used to differentiate operations performed within a script when the script accesses this particular buffer descriptor. The use of this field can be defined by the script. The command values defined for the bootload script are defined in Buffer Descriptor Commands for Bootload scripts. Refer to the individual script definition in script library documents in SDMA Scripts for command field definitions for other scripts.
23	Reserved
22	Reserved
21 L	Last Buffer Descriptor: This bit is set in SDMA IPC scripts to indicate to the receiving Core that the transfer has ended. Whenever the source finishes transferring the count it wanted to transfer, it sets LAST_BIT in the destination BD, to let the destination know that transfer is over.This bit also tells the destination software that when it processes the destination BDs, they need not process any BD after the BD with the LAST_BIT set.For example, when the DSP prepares a single buffer descriptor with count equals to 25 and Arm platform prepares a single buffer descriptor with count equals 100. When 25 bytes have been transferred from DSP to Arm platform, the DSP buffer descriptor is normally closed while the Arm platform buffer descriptor will have the L bit set and the byte count updated to 25.
20 R	erroR. Indicates an error occurred on the channel's buffer descriptor requested command. Some scripts may overwrite the command field with an error code indicating the source of the error. 0 No Error 1 Error
19 I	Interrupt. When SDMA has finished to process data transfer attached to this buffer descriptor, send an interrupt to the Arm platform. 0 No Interrupt 1 Interrupt the processor when BD is complete
18 C	Continuous. This buffer is allowed to receive multiple transmit buffers or is allowed to transmit to multiple receive buffers.The Continuous bit is decoded at the end of the processing of a BD to determine if the SDMA script must open a new BD to potentially continue the data transfer. 0 No further buffer descriptors 1 SDMA should move to the next Buffer descriptor after this one
17 W	Wrap. Indicates if this buffer descriptor is the last one for the channel control block. When encountering this bit set, the SDMA scripts updates the CurrentBD pointer to point to the first Buffer Descriptor of the array. This bit is set if the Arm platform wants to organize the array of BD in a circular way (like a ring). When all BD have been processed and if Wrap bit and CONtinuous bit are set in the last BD, the SDMA script will wrap around and it will try to re-open the first BD. 0 No Error 1 Wrap to first buffer descriptor after this one is processed.
16 D	D - "Done": bit 16: indicates the "ownership" of the buffer descriptor. When D=0 the host owns the buffer descriptor; when D=1 SDMA owns the buffer descriptor. In the case of the channel 0, D=1 indicates the SDMA has not yet processed this buffer, D=0 indicates the SDMA has processed this buffer. 0 Arm platform owns the buffer. 1 SDMA owns the buffer
15-0 Count	Count. the count field (bit 15-0) indicates the size of the data to be transmitted, the size of the data buffer pointed to by the buffer descriptor. The SDMA memory structure is different for program memory (16-bits shorts/half-words) and data memory (32-bits long). For channel 0 buffer descriptors, Count is expressed in 16-bit half-words when PM is addressed and in 32-bit words when DM is addressed. Count is typically expressed in bytes for other channel scripts, but the unit is dependant on the script.
31-0	Buffer address. Address pointer to the data buffer.
31-0	Extended buffer address. Additional pointer or other information required by some scripts.

The buffer descriptors form an array of programmable size. If the last buffer descriptor is marked by the Wrap flag-bit W=1, the array of buffer descriptor is treated as a ring with some logically continuous portion owned by the Arm platform with D=0, and the remainder owned by the SDMA with D=1. The count field of the buffer descriptor indicates how much data has been transmitted.

If Arm platform has prepared 3 buffers to be filled by the SDMA script, it has also prepared 3 BD, one for each buffer. The *Cont* and *Wrap* bits are used to organize the buffers in a circular way. For example, *CONT*inous bit is set to 1 in the 2 first BDs and *Wrap* is set in the 3rd BD. The SDMA script opens and processes BD#1. Since *CONT*inous bit is set for this BD, the SDMA will open the second BD and it will process it. Each time a BD is processed, its *Done* bit is reset by the SDMA. After the 3rd BD, if *CONT*inous is not set but if *Wrap* is set, the SDMA script stops here and the next time the channel will be triggered, the script will open the BD pointed by the currentBDptr pointer of the CCB and it will correspond to the first buffer descriptor.

If the *CONT*inous bit and *Wrap* bits are both set in the 3rd BD, the script will close it and it will try to open the first BD. An error may occur at this point if the BD#1 has already been processed and its *Done* bit is 0. The SDMA script cannot process a BD with a *Done* bit to 0. It means the BD is not ready to be processed. To avoid this situation, the *CONT*inous bit should not be set for the last BD if *Wrap* is set, and the Interrupt flag must set for the last BD. It will warn the owner of the BD that all the BDs have been processed and it has to re-set to 1 the *Done* bit of all the BD's if it desires the SDMA to fill them again. Basically, if the Arm platform expects the SDMA to fill up the buffers in a circular fashion, then it's the responsibility of the Arm platform to set the *Done* bit of a buffer descriptor at an appropriate time.

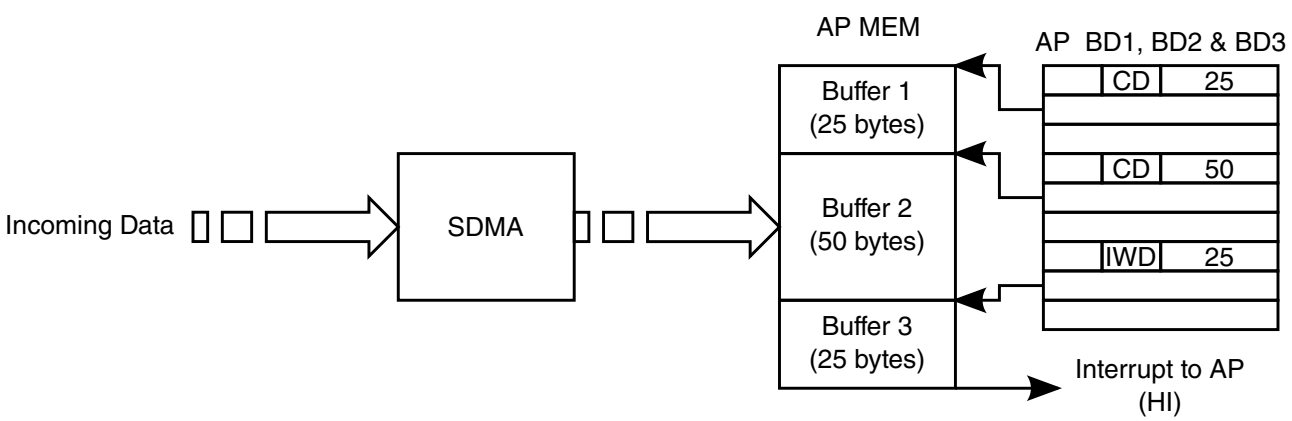


Figure 7-17. Buffer Descriptor Flow

The previous figure shows an example buffer descriptor flow. When the incoming data is stored and fills the first buffer of 25 bytes, the SDMA script opens the second BD because the CONTinuous bit was set. Then next incoming data is put in the second buffer. After receiving 50 bytes, the second buffer descriptor is also closed. The Done bit is reset and the third BD is opened. After receiving another 25 bytes, the third buffer is full and an interrupt is sent to the Arm platform because the Interrupt flag is set in the 3rd BD. The CONTinuous flag is not present the transfer is over. The next time the script will be triggered, the BD to be opened will be the first buffer descriptor since the Wrap flag was set in the 3rd BD. It is the Arm platform responsibility to set the Done bit of all the BD if it wants to use the same buffers.

7.2.5.1.2 Buffer Descriptor Commands for Bootload scripts

The command field of the buffer descriptor is defined separately for each script.

The following table lists the buffer descriptor commands defined for the channel 0 bootload script.

Table 7-52. Channel Zero Buffer Descriptor Commands

Command Field (binary)	Command	Description	Buffer Address	Extended Buffer Address
0000_0001 (0x01)	C0_SET_DM	Load SDMA data memory (RAM) from Arm platform memory buffer	Arm platform memory source address	SDMA memory destination address
0000_0010 (0x02)	C0_GET_DM	Copy SDMA data memory (RAM) to Arm platform memory buffer	Arm platform memory destination address	SDMA memory source address
0000_0100 (0x04)	C0_SET_PM	Load SDMA program memory (RAM) from Arm platform memory buffer	Arm platform memory source address	SDMA memory destination address
0000_0110 (0x06)	C0_GET_PM	Copy SDMA program memory (RAM) to Arm platform memory buffer	Arm platform memory destination address	SDMA memory source address
cccc_c111 (0x07 | CHN)	C0_SETCTX	Load Context for channel cccc into SDMA RAM from Arm platform memory buffer	Arm Platform memory source address	-
cccc_c011 (0x03 | CHN)	C0_GETCTXT	Copy Context for channel ccccc from SDMA RAM to Arm platform memory buffer	Arm platform memory destination address	-

The Channel 0 bootload commands are summarized as follows:

- C0_SET_[PM-DM]: load the buffer descriptor data in the SDMA local memory at the address pointed to by the "extended buffer address" field. The SDMA RAM can be seen as a Program Memory (PM, 16-bit address) or Data Memory (DM 32-bit address). When C0_SET_PM is used, the count field is expressed in "shorts" (16-bit half words), this command can be used to download scripts. When C0_SET_DM is

used, the count field is expressed in "long" (32-bit words), this command can be used to download channel contexts to the context channel area in RAM.

- C0_GET_[PM-DM]: write to the buffer descriptor's data buffer the content of the SDMA local memory from the address pointed to by the "extended buffer address" field for the length defined by the count in the buffer descriptor. C0_GET_PM is used to dump some part of the Program Memory (may be used to dump context of a channel), therefore count is expressed in "shorts"; while C0_GET_DM is used to dump to the buffer descriptor's data buffer, so the count field is in "longs."
- C0_SETCTX: load a context into the SDMA context page area. The handling script decodes the channel number from the 5 MSB of the command field of the buffer descriptor. Using the channel number the script computes the offset of the context data pointer for the channel relative to the context page base to use as the destination address in SDMA memory. Then the C0_SET_DM command explained above is invoked to load SDMA RAM from memory. The counter indicates the size in words of the context structure.
- Command value: (in binary) cccc c111, where ccccc is the channel number (5 bits). For instance, 0x0F means set context for channel 1, 0xFF means set context for channel 31.

- C0_GETCTX: write to the buffer descriptor's data buffer the content of the SDMA context page area. The handling script decodes the channel number from the 5 MSB of the command field of the buffer descriptor. Using this channel number, the script computes the offset of the context data pointer for the channel relative to the context page base to use as the source address for the copy. Then the C0_GET_DM command explained above is invoked to copy the context to memory. The counter indicates the size in words of the context structure.
- Command value: (in binary): cccc c011, where ccccc is the channel number (5 bits). For instance, 0x03 means get context of channel 1, 0xFB means get context of channel 31.

NOTE

To download channel context, C0_SETDM and C0_SETCTXT command can be used but the second one is easier because the channel number is embedded into the command field, whereas with the C0_SETDM, the pointer to the channel context area must be written into the extended buffer address field of the buffer descriptor.

7.2.5.1.3 Example of Buffer Descriptors for Channel 0.

Figure 7-19 illustrates the buffer descriptors that must be set in Arm platform memory space, before execution of boot code, to download contexts and scripts of channels 1, 4, and 10. After boot code execution, SDMA memory will be populated with the different contexts and scripts as presented in the following figure.

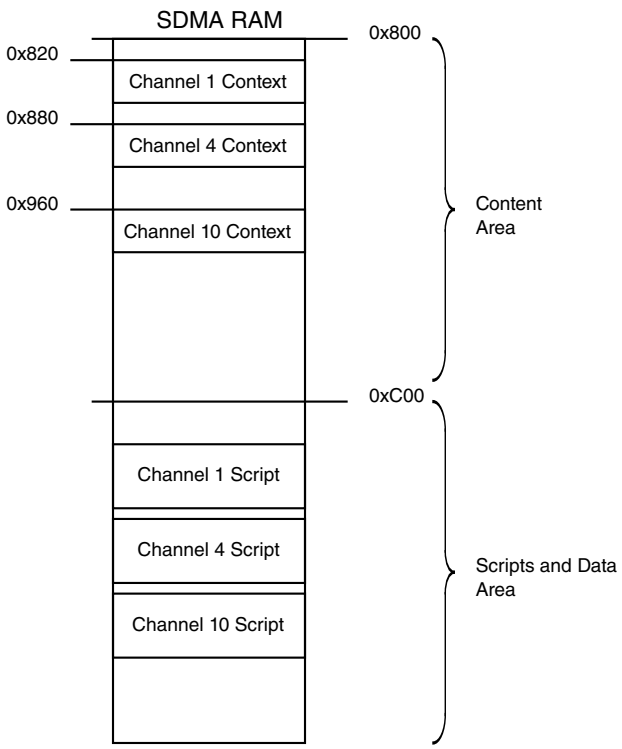
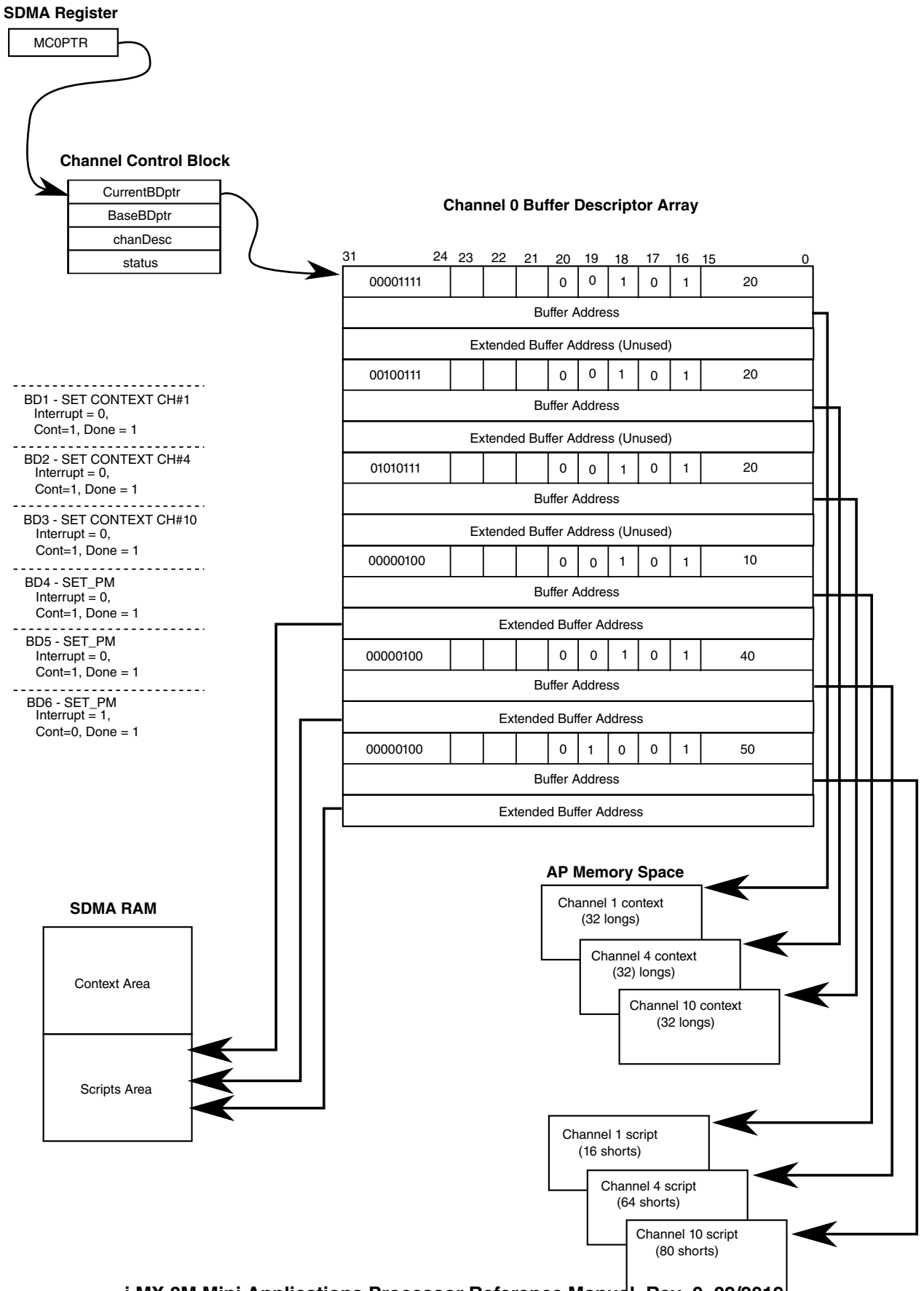


Figure 7-18. Example of SDMA RAM After Boot Session


SDMA Register
MC0PTR
Channel Control Block
CurrentBDptr
BaseBDptr
chanDesc
status
Channel 0 Buffer Descriptor Array
31 24 23 22 21 20 19 18 17 16 15 0
00001111 0 0 1 0 1 20
Buffer Address
Extended Buffer Address (Unused)
00100111 0 0 1 0 1 20
Buffer Address
Extended Buffer Address (Unused)
01010111 0 0 1 0 1 20
Buffer Address
Extended Buffer Address (Unused)
00000100 0 0 1 0 1 10
Buffer Address
Extended Buffer Address
00000100 0 0 1 0 1 40
Buffer Address
Extended Buffer Address
00000100 0 1 0 0 1 50
Buffer Address
Extended Buffer Address
BD1 - SET CONTEXT CH#1
Interrupt = 0,
Cont=1, Done = 1
BD2 - SET CONTEXT CH#4
Interrupt = 0,
Cont=1, Done = 1
BD3 - SET CONTEXT CH#10
Interrupt = 0,
Cont=1, Done = 1
BD4 - SET_PM
Interrupt = 0,
Cont=1, Done = 1
BD5 - SET_PM
Interrupt = 0,
Cont=1, Done = 1
BD6 - SET_PM
Interrupt = 1,
Cont=0, Done = 1
AP Memory Space
Channel 1 context
(32 longs)
Channel 4 context
(32) longs)
Channel 10 context
(32 longs)
SDMA RAM
Context Area
Scripts Area
Channel 1 script
(16 shorts)
Channel 4 script
(64 shorts)
Channel 10 script
(80 shorts)

7.2.5.1.4 Channel Context

There are 32 channel context memory structures pointed to by the local save area pointer. These channel context memory structures are fixed.

The script in the SDMA computes the memory offset for a given channel based on the structure length and channel number. Figure below shows the structure of the channel context as it is saved in the SDMA local memory (RAM).

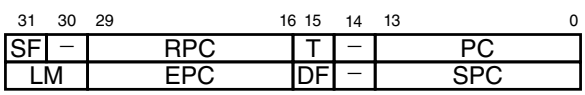
A channel context consists in 24 words, one per register. A total of 32 words are reserved for every channel. The additional 8 words are called scratch ram and they are dedicated to each channel. This memory area is commonly used for stack management.

The structure is divided in 4 areas:

- Channel status registers
- General purpose registers
- Functional units state registers reflecting the state of the Arm platform DMAs (Burst and Peripheral DMA).
- Scratch RAM

The details of the channel context status registers are described in the following figure.

The PC field of the first long register must point to the SDMA RAM address where the script that will be executed on the channel is located and this value equals the one stored in the extended buffer address of the buffer descriptor with C0_SETPM command.



SF: Source fault while loading data
RPC: Return program counter
T: Test bit: status of arithmetic and test instructions
PC: Program counter
LM: Loop mode
EPC: Loop end program counter
DF: Destination fault while storing data
SPC: Loop Start program counter

Figure 7-20. SDMA State Registers (ShPC, ShLoop)

7.2.5.2 Typical Data Transfer Supported by SDMA DMA Units

This section presents a library of SDMA scripts that perform data transfers through the peripheral DMA and the burst DMA units.

The Arm platform memory and peripherals are devices that either the peripheral DMA or the burst DMA can access. The scripts are given for a peripheral DMA whose address registers are programmed in incremented mode when internal memory is involved. See the following table for the summary.

Table 7-53. Typical Data Transfers Summary

Data Transfer	Peripheral DMA	Burst DMA	Comments
Arm platform External Memory ↔ Arm platform External Memory		3	Copy mode Script example, see Burst DMA Unit Copy Mode and External Memory to External Memory.
Arm platform Peripheral ↔ Arm platform Peripheral	3		Copy mode if same data path width Script example, see Peripheral to Peripheral Transfer.
Arm platform External Memory ↔ Arm platform Peripheral	3	3	Data transit through SDMA Script example, see Transfer Between Peripheral and External Memory.
Arm platform External Memory ↔ Arm platform Internal Memory		3	Copy mode Script example, see Transfer Between External Memory and Internal Memory.
Arm platform Internal Memory ↔ Arm platform Internal Memory		3	Copy mode Script example, see Internal Memory to Internal Memory.
Arm platform Internal memory ↔ Arm platform Peripheral	3		Data transit through SDMA Script example, see Transfer Between Peripheral and Internal Memory.

NOTE

These scripts are provided as examples of how to use DMA blocks to perform required data transfers: They are not "official" programs.

7.2.5.2.1 External Memory to External Memory

This section describes the SDMA script that performs data moves in external memory.

For this particular data transfer, only the burst DMA is used. It is programmed in copy mode, so no data transmits through an SDMA general register.

The SDMA core only monitors data transfer status. It is assumed source and destination address values are already present in two SDMA general registers (r1 and r2). For this example, it is also assumed that a 32-bit word-to-move for source-to-destination address is present in r0 and equals 64.

Data Moves in External Memory

```
1         stf r1,MSA                          // Source address setup

2         stf r2,MDA                          // Destination address setup

3         ldi r0,0x64                         // 64 words must be transferred from MSA to
MDA

4         ldi r1,0x8

MAIN_XFER:

5         cmphs r0,r1                         // Is r0 >= 0x8

6         bf LAST_XFER                      // If not, jump to last transfer label

7         stf r1,MD|CPY                        // Copy 8 words from MSA to MDA address.

8         subi r0,0x8                         // Decrement counter

9         jmp MAIN_XFER                     // return to main transfer loop

LAST_XFER:

10        stf r0,MD|CPY                        // perform last transfer
```

All instructions are performed in one cycle (jumps excepted). Instruction 7 triggers a copy transfer: A read burst access of 8-word starts, data is staged in MD and then a write burst of 8 words is executed. Instruction 8, 9, 5, and 6 are executed while the burst access is in progress. If this access is not complete when instruction 7 is executed a second time, SDMA stalls on this instruction as long as the previous copy transfer is not over. In this case, the instruction is no longer a one-cycle instruction.

During the main loop (MAIN_XFER), r1 always equals 8, so burst lengths are 8 words. On the last ldf |CPY instruction (10), r1 equals the reminder of r0 divided by 8; therefore, the length of bursts triggered in copy mode equal r1 value, which is between 1 and 7.

7.2.5.2.2 Peripheral to Peripheral Transfer

For this data transfer, only the peripheral DMA is used.

It is programmed in copy mode, so no data will transmit through the SDMA general register used in the ldf instruction, but the contents of the general register are lost. The SDMA core only monitors the transfer.

7.2.5.2.2.1 Source and Destination Target Have the Same Data Path Width

When the source and destination target have the same data path width, the following is true:

- Source target is a *half-word* (16-bit) peripheral located at address 0x1002.
- Destination is a *half-word* (16-bit) peripheral located at address 0x2006.

It is assumed the address values are already present in two SDMA general registers (r1, r2). The script for a transfer of 10 half-word is as follows:

Same Data Path Width for Source and Destination

```
//SETUP SECTION
1         stf r1, PSA|SZ16|F                    //r1=0x1002 Source address register setup
2         stf r2, PDA|SZ16|F                    //r2=0x2006 Destination address register
setup
3         bdf ERROR_ADDR_SETUP
4         ldi r0,0xa                           //loop counter is 10
//MAIN LOOP TRANFER
copy_loop:

5         loop 2,0
6           ldf r7,PD|CPY                      //Reads 1 half-word from src and writes to
dest.
7           yield
8         bdf ERROR_DURING_XFER
ERROR_ADDR_SETUP:
                    //correction of PSA/PDA setup and jumps to main loop transfer
ERROR_DURING_XFER:
//flag error is set,
//PS can be read to know if error occurs during read or write access.
```

If a data transfer must occur between two word peripherals, only the setup section should be updated. The transfer itself is always performed by the hardware loop instruction.

All instructions are executed in one cycle (change of flow excepted). On instruction 6, a single read access is triggered, read data is staged in PD, and a write-to-destination is executed. When the transfers are in progress, the SDMA can execute he next instructions in parallel. If instruction 6, which performs the copy transfer, is executed while the previous access is not over, SDMA is stalled and instruction ldf is a multi-cycle instruction.

7.2.5.2.2.2 Source and Destination Target Have a Different Data Path Width

When the source and destination target have a different data path width, copy mode cannot be used, and any attempt to initiate a copy transfer immediately raises an error, which is stored in the SF flag.

The following example shows the SDMA code that could transfer 10 words from a *word* (32-bit) peripheral to a *half-word* peripheral whose addresses are preliminary and stored in r1 and r2.

Different Data Path Width for Source and Destination

```
//SETUP SECTION
1         stf r1, PSA|SZ32|F|PF                    //r1=0x1000 and prefetch data
2         stf r2, PDA|SZ16|F                      //r2=0x2006
3         bdf ERROR_ADDR_SETUP
4         ldi r0,0xa                             //loop counter is 10
//MAIN LOOP TRANFER
main_loop_xfer_16_16:
5         loop 6,0
6           ldf r7,PD                           //copy 32-bit of PD in r7
7           stf r7,PD                            //store 16 LSB of r7 in PD and a flush is
executed
8           rorb r7
9           rorb r7                            //16 MSB --> 16 LSB
10          stf r7,PD                            //store 16 LSB of r6 in PD and a flush.
11          yield
```

On instruction 1, when the source address register is programmed and a data prefetch is required, a read access is executed. In parallel, the SDMA executes instructions 2 to 5. On instruction 6, the SDMA tries to read data that was fetched by instruction 1. If data is ready, the ldf will be a one cycle instruction; otherwise, the SDMA is stalled as long as the read access is not finished. Then, the 16 LSB of the read data is stored in PD and automatically flushed to the destination peripheral. In parallel, the SDMA executes the rotation instructions (8, 9), and stores the 16 MSB of the read data into PD. If a previous write access is finished, instruction 10 will be a one-cycle instruction.

The main loop transfer may appear inefficient, but due to wait states imposed to the peripheral DMA each time an external access is performed, a software pipeline is in place. During the time needed to flush PD, the SDMA executes the move and rotation operations. SDMA executes instructions in parallel with DMA accesses.

7.2.5.2.3 Transfer Between Peripheral and External Memory

7.2.5.2.3.1 Peripheral to External Memory Transfer

A transfer from a peripheral to the external memory controller involves the peripheral DMA and the burst DMA.

The code for transferring 100 word from word peripheral to the external memory would be as follows:

Peripheral to External Memory Transfer

```
//SETUP SECTION source and destination addresses are already in r1 and r2
1         stf r1, PSA|SZ16|F|PF                    //r1=0x1000 and prefetch 32-bit data
2         stf r2, MDA                            //r2=0x2000, setup burst DMA destination
address
3         bdf ERROR_ADDR_SETUP
4         ldi r0,0x64                            //loop counter is 100
5
          //MAIN LOOP TRANFER
6         loop 3,0
7           ldf r1,PD|PF                         // read 32 bits of PD and initiate a new read
```

```
access.
8           stf r1,MD|32                    // store 32 bits of r1 in the MD fifo.
9           yield
10        ldf r1,PD                       // last word data is read
11        stf r1,MD|32|FL                   // to flush all remaining bytes of MD
```

On instruction 1, the source address register of the peripheral DMA is programmed and data is fetched. This data is stored in PD and the SDMA reads PD during instruction 7, which is a one-cycle instruction that is read-access finished. On the same instruction (7), a data prefetch is required and a read access to the source peripheral is executed. In parallel, the SDMA stored the previous read data into the data register of MD. When MD (which is an eight-word FIFO) is full, a burst write access is executed to empty the FIFO. As long as the next SDMA instructions do not access the burst DMA, they will be one-cycle instructions. The following figures show how the peripheral DMA and burst DMA work in parallel.

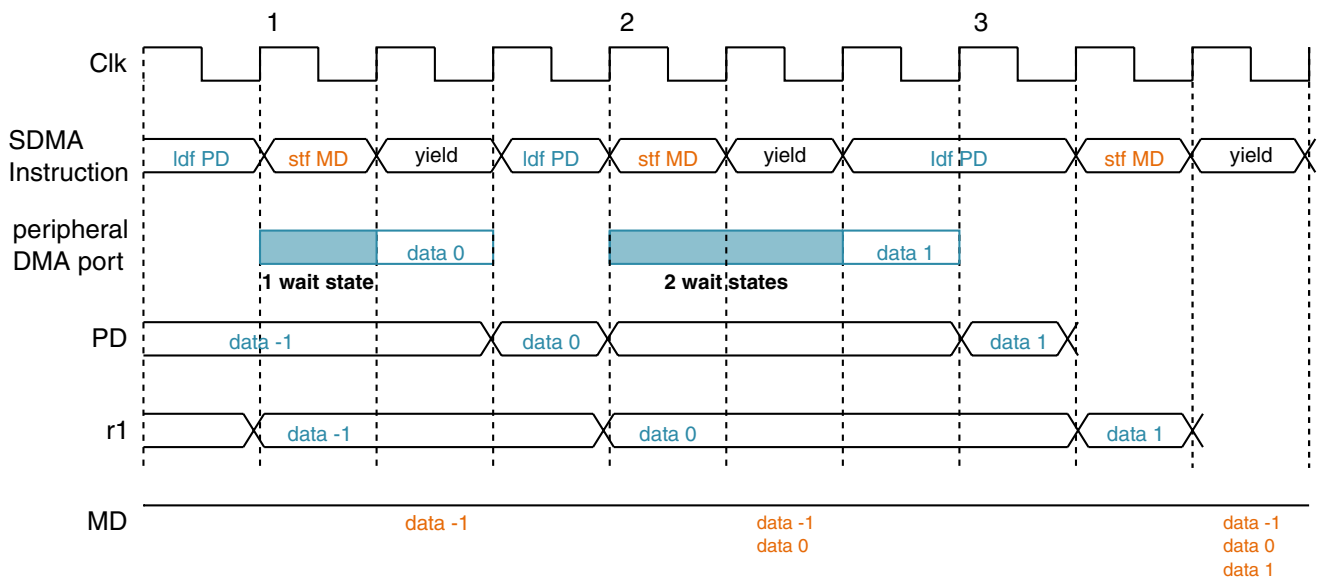


Figure 7-21. Peripheral to External Memory Example (1)

As seen in the figure above, the read access triggered by the ldf PD instruction is symbolized by the blue bar when in progress. After wait states, the read data (data 0, data 1) is stored in PD on the clk rising edge. On edge 2, data 0 is available in PD so it can be transferred to the SDMA general register r1, and then stored in MD FIFO. On edge 3, data 1 is not in PD; therefore, SDMA is stalled on the ldf instruction, which lasts two cycles. The figure below shows an example of when MD FIFO is full with data.

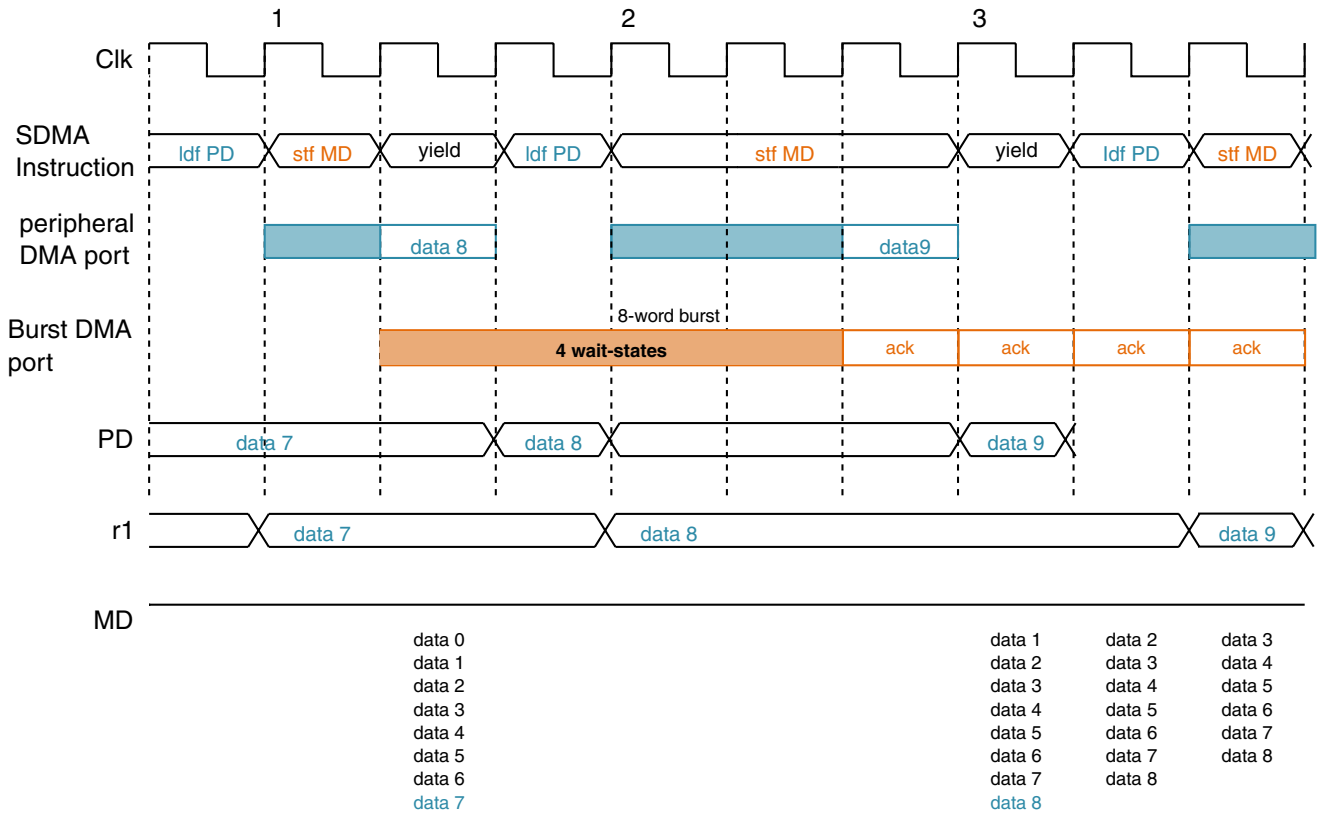


Figure 7-22. Peripheral to External Memory Example (2)

In the previous figure, the write bar means the burst DMA is performing a write burst access. The latency to have the first write acknowledge is four cycles. SDMA is stalled on instruction stf because no acknowledge was received, MD FIFO is full, and there is no empty slot to store data 9. When an acknowledge is sampled by the burst DMA, FIFO is shifted and data 8 is written. As long as there is at least one empty slot in MD FIFO, the stf MD instruction lasts one cycle.

7.2.5.2.3.2 External Memory to Peripheral Transfer

A transfer from the external memory to a peripheral involves the peripheral DMA and the burst DMA.

The code for transferring 100 word from external memory to a word peripheral would be as follows:

External Memory to Peripheral Transfer

```
//SETUP SECTION source and destination addresses are already in r1 and r2
1         stf r1, MSA|PF             //r1=0x1000 and starts a 8-word read burst
2         stf r2, PDA|SZ32|P          //r2=0x2010, setup peripheral DMA destination address
3         bdf ERROR_ADDR_SETUP
4         ldi r0,0x64               //loop counter is 100
//MAIN LOOP TRANFER
```

```
6          loop 3,0
7            ldf r1,MD|32|PF          // read 32 bits of MD and initiate a new read access
                                   // if MD is empty after this reading.
8            stf r1,PD              // store 32 bits of r1 in the PD.
9            yield
           10 ldf r1,MD|32            // last word data is read
11         stf r1,PD                 // last write access
```

On instruction 1, a read burst of 8 words begins. Read data is staged into MD. On instruction 7 (and if data is available in MD), 32 bits are copied into r1. Then instruction 8 writes them into PD and an automatic flush is executed. The SDMA core, peripheral DMA, and burst DMA can work in parallel as long as no SDMA instruction tries to start a new write access on the peripheral DMA while the previous access is still in progress, or as long as there is data in MD when the SDMA tries to read it.

7.2.5.2.4 Transfer Between External Memory and Internal Memory

Since the internal memory (Arm platform RAM) is accessed via the peripheral DMA and the external memory is accessed via the burst DMA, the SDMA scripts that are described in Transfer Between Peripheral and External Memory can be reused. The exception is that the peripheral DMA address registers (PSA or PDA, depending on the script) should be programmed in incremented mode rather than frozen mode.

7.2.5.2.4.1 Internal Memory to Internal Memory

The internal memory can only be accessed via the peripheral DMA, so the script described in Peripheral to Peripheral Transfer can be reused with a different programming of the peripheral DMA address registers.

7.2.5.2.4.2 Transfer Between Peripheral and Internal Memory

For this transfer, the peripheral DMA is also used in copy mode.

The SDMA script is very similar to the one described in Peripheral to Peripheral Transfer, except for the peripheral DMA address registers programming.

7.2.6 Arm Platform Memory Map and Control Register Definitions

The Arm platform controls the SDMA by means of several interface registers. Those registers are described in the current section.

All registers are clocked with the SDMA clock (which means the Arm platform must ensure that the SDMA clock is running when it wants to access any register).

SDMAARM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302B_0000	Arm platform Channel 0 Pointer (SDMAARM3_MC0PTR)	32	R/W	0000_0000h	7.2.6.1/ 1182
302B_0004	Channel Interrupts (SDMAARM3_INTR)	32	w1c	0000_0000h	7.2.6.2/ 1182
302B_0008	Channel Stop/Channel Status (SDMAARM3_STOP_STAT)	32	w1c	0000_0000h	7.2.6.3/ 1182
302B_000C	Channel Start (SDMAARM3_HSTART)	32	R/W	0000_0000h	7.2.6.4/ 1183
302B_0010	Channel Event Override (SDMAARM3_EVTOVR)	32	R/W	0000_0000h	7.2.6.5/ 1183
302B_0014	Channel BP Override (SDMAARM3_DSPOVR)	32	R/W	FFFF_FFFFh	7.2.6.6/ 1184
302B_0018	Channel Arm platform Override (SDMAARM3_HOSTOVR)	32	R/W	0000_0000h	7.2.6.7/ 1184
302B_001C	Channel Event Pending (SDMAARM3_EVTPEND)	32	w1c	0000_0000h	7.2.6.8/ 1184
302B_0024	Reset Register (SDMAARM3_RESET)	32	R	0000_0000h	7.2.6.9/ 1185
302B_0028	DMA Request Error Register (SDMAARM3_EVTERR)	32	R	0000_0000h	7.2.6.10/ 1186
302B_002C	Channel Arm platform Interrupt Mask (SDMAARM3_INTRMASK)	32	R/W	0000_0000h	7.2.6.11/ 1186
302B_0030	Schedule Status (SDMAARM3_PSW)	32	R	0000_0000h	7.2.6.12/ 1187
302B_0034	DMA Request Error Register (SDMAARM3_EVTERRDBG)	32	R	0000_0000h	7.2.6.13/ 1187
302B_0038	Configuration Register (SDMAARM3_CONFIG)	32	R/W	0000_0003h	7.2.6.14/ 1188
302B_003C	SDMA LOCK (SDMAARM3_SDMA_LOCK)	32	R/W	0000_0000h	7.2.6.15/ 1189
302B_0040	OnCE Enable (SDMAARM3_ONCE_ENB)	32	R/W	0000_0000h	7.2.6.16/ 1190

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302B_0044	OnCE Data Register (SDMAARM3_ONCE_DATA)	32	R/W	0000_0000h	7.2.6.17/ 1190
302B_0048	OnCE Instruction Register (SDMAARM3_ONCE_INSTR)	32	R/W	0000_0000h	7.2.6.18/ 1191
302B_004C	OnCE Status Register (SDMAARM3_ONCE_STAT)	32	R	0000_E000h	7.2.6.19/ 1191
302B_0050	OnCE Command Register (SDMAARM3_ONCE_CMD)	32	R/W	0000_0000h	7.2.6.20/ 1193
302B_0058	Illegal Instruction Trap Address (SDMAARM3_ILLINSTADDR)	32	R/W	0000_0001h	7.2.6.21/ 1193
302B_005C	Channel 0 Boot Address (SDMAARM3_CHN0ADDR)	32	R/W	0000_0050h	7.2.6.22/ 1194
302B_0060	DMA Requests (SDMAARM3_EVT_MIRROR)	32	R	0000_0000h	7.2.6.23/ 1195
302B_0064	DMA Requests 2 (SDMAARM3_EVT_MIRROR2)	32	R	0000_0000h	7.2.6.24/ 1195
302B_0070	Cross-Trigger Events Configuration Register 1 (SDMAARM3_XTRIG_CONF1)	32	R/W	0000_0000h	7.2.6.25/ 1196
302B_0074	Cross-Trigger Events Configuration Register 2 (SDMAARM3_XTRIG_CONF2)	32	R/W	0000_0000h	7.2.6.26/ 1197
302B_0100	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI0)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0104	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI1)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0108	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI2)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_010C	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI3)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0110	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI4)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0114	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI5)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0118	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI6)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_011C	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI7)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0120	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI8)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0124	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI9)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0128	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI10)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_012C	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI11)	32	R/W	0000_0000h	7.2.6.27/ 1198

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302B_0130	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI12)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0134	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI13)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0138	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI14)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_013C	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI15)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0140	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI16)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0144	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI17)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0148	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI18)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_014C	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI19)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0150	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI20)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0154	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI21)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0158	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI22)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_015C	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI23)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0160	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI24)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0164	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI25)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0168	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI26)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_016C	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI27)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0170	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI28)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0174	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI29)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0178	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI30)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_017C	Channel Priority Registers (SDMAARM3_SDMA_CHNPRI31)	32	R/W	0000_0000h	7.2.6.27/ 1198
302B_0200	Channel Enable RAM (SDMAARM3_CHNENBL0)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0204	Channel Enable RAM (SDMAARM3_CHNENBL1)	32	R/W	0000_0000h	7.2.6.28/ 1199

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302B_0208	Channel Enable RAM (SDMAARM3_CHNENBL2)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_020C	Channel Enable RAM (SDMAARM3_CHNENBL3)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0210	Channel Enable RAM (SDMAARM3_CHNENBL4)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0214	Channel Enable RAM (SDMAARM3_CHNENBL5)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0218	Channel Enable RAM (SDMAARM3_CHNENBL6)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_021C	Channel Enable RAM (SDMAARM3_CHNENBL7)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0220	Channel Enable RAM (SDMAARM3_CHNENBL8)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0224	Channel Enable RAM (SDMAARM3_CHNENBL9)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0228	Channel Enable RAM (SDMAARM3_CHNENBL10)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_022C	Channel Enable RAM (SDMAARM3_CHNENBL11)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0230	Channel Enable RAM (SDMAARM3_CHNENBL12)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0234	Channel Enable RAM (SDMAARM3_CHNENBL13)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0238	Channel Enable RAM (SDMAARM3_CHNENBL14)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_023C	Channel Enable RAM (SDMAARM3_CHNENBL15)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0240	Channel Enable RAM (SDMAARM3_CHNENBL16)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0244	Channel Enable RAM (SDMAARM3_CHNENBL17)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0248	Channel Enable RAM (SDMAARM3_CHNENBL18)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_024C	Channel Enable RAM (SDMAARM3_CHNENBL19)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0250	Channel Enable RAM (SDMAARM3_CHNENBL20)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0254	Channel Enable RAM (SDMAARM3_CHNENBL21)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0258	Channel Enable RAM (SDMAARM3_CHNENBL22)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_025C	Channel Enable RAM (SDMAARM3_CHNENBL23)	32	R/W	0000_0000h	7.2.6.28/ 1199

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302B_0260	Channel Enable RAM (SDMAARM3_CHNENBL24)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0264	Channel Enable RAM (SDMAARM3_CHNENBL25)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0268	Channel Enable RAM (SDMAARM3_CHNENBL26)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_026C	Channel Enable RAM (SDMAARM3_CHNENBL27)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0270	Channel Enable RAM (SDMAARM3_CHNENBL28)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0274	Channel Enable RAM (SDMAARM3_CHNENBL29)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0278	Channel Enable RAM (SDMAARM3_CHNENBL30)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_027C	Channel Enable RAM (SDMAARM3_CHNENBL31)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0280	Channel Enable RAM (SDMAARM3_CHNENBL32)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0284	Channel Enable RAM (SDMAARM3_CHNENBL33)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0288	Channel Enable RAM (SDMAARM3_CHNENBL34)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_028C	Channel Enable RAM (SDMAARM3_CHNENBL35)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0290	Channel Enable RAM (SDMAARM3_CHNENBL36)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0294	Channel Enable RAM (SDMAARM3_CHNENBL37)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_0298	Channel Enable RAM (SDMAARM3_CHNENBL38)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_029C	Channel Enable RAM (SDMAARM3_CHNENBL39)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_02A0	Channel Enable RAM (SDMAARM3_CHNENBL40)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_02A4	Channel Enable RAM (SDMAARM3_CHNENBL41)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_02A8	Channel Enable RAM (SDMAARM3_CHNENBL42)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_02AC	Channel Enable RAM (SDMAARM3_CHNENBL43)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_02B0	Channel Enable RAM (SDMAARM3_CHNENBL44)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_02B4	Channel Enable RAM (SDMAARM3_CHNENBL45)	32	R/W	0000_0000h	7.2.6.28/ 1199

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302B_02B8	Channel Enable RAM (SDMAARM3_CHNENBL46)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_02BC	Channel Enable RAM (SDMAARM3_CHNENBL47)	32	R/W	0000_0000h	7.2.6.28/ 1199
302B_1000	SDMA DONE0 Configuration (SDMAARM3_DONE0_CONFIG)	32	R/W	1F1F_1F1Fh	7.2.6.29/ 1199
302B_1004	SDMA DONE1 Configuration (SDMAARM3_DONE1_CONFIG)	32	R/W	1F1F_1F1Fh	7.2.6.30/ 1201
302C_0000	Arm platform Channel 0 Pointer (SDMAARM2_MC0PTR)	32	R/W	0000_0000h	7.2.6.1/ 1182
302C_0004	Channel Interrupts (SDMAARM2_INTR)	32	w1c	0000_0000h	7.2.6.2/ 1182
302C_0008	Channel Stop/Channel Status (SDMAARM2_STOP_STAT)	32	w1c	0000_0000h	7.2.6.3/ 1182
302C_000C	Channel Start (SDMAARM2_HSTART)	32	R/W	0000_0000h	7.2.6.4/ 1183
302C_0010	Channel Event Override (SDMAARM2_EVTOVR)	32	R/W	0000_0000h	7.2.6.5/ 1183
302C_0014	Channel BP Override (SDMAARM2_DSPOVR)	32	R/W	FFFF_FFFFh	7.2.6.6/ 1184
302C_0018	Channel Arm platform Override (SDMAARM2_HOSTOVR)	32	R/W	0000_0000h	7.2.6.7/ 1184
302C_001C	Channel Event Pending (SDMAARM2_EVTPEND)	32	w1c	0000_0000h	7.2.6.8/ 1184
302C_0024	Reset Register (SDMAARM2_RESET)	32	R	0000_0000h	7.2.6.9/ 1185
302C_0028	DMA Request Error Register (SDMAARM2_EVTERR)	32	R	0000_0000h	7.2.6.10/ 1186
302C_002C	Channel Arm platform Interrupt Mask (SDMAARM2_INTRMASK)	32	R/W	0000_0000h	7.2.6.11/ 1186
302C_0030	Schedule Status (SDMAARM2_PSW)	32	R	0000_0000h	7.2.6.12/ 1187
302C_0034	DMA Request Error Register (SDMAARM2_EVTERRDBG)	32	R	0000_0000h	7.2.6.13/ 1187
302C_0038	Configuration Register (SDMAARM2_CONFIG)	32	R/W	0000_0003h	7.2.6.14/ 1188
302C_003C	SDMA LOCK (SDMAARM2_SDMA_LOCK)	32	R/W	0000_0000h	7.2.6.15/ 1189
302C_0040	OnCE Enable (SDMAARM2_ONCE_ENB)	32	R/W	0000_0000h	7.2.6.16/ 1190
302C_0044	OnCE Data Register (SDMAARM2_ONCE_DATA)	32	R/W	0000_0000h	7.2.6.17/ 1190
302C_0048	OnCE Instruction Register (SDMAARM2_ONCE_INSTR)	32	R/W	0000_0000h	7.2.6.18/ 1191

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302C_004C	OnCE Status Register (SDMAARM2_ONCE_STAT)	32	R	0000_E000h	7.2.6.19/ 1191
302C_0050	OnCE Command Register (SDMAARM2_ONCE_CMD)	32	R/W	0000_0000h	7.2.6.20/ 1193
302C_0058	Illegal Instruction Trap Address (SDMAARM2_ILLINSTADDR)	32	R/W	0000_0001h	7.2.6.21/ 1193
302C_005C	Channel 0 Boot Address (SDMAARM2_CHN0ADDR)	32	R/W	0000_0050h	7.2.6.22/ 1194
302C_0060	DMA Requests (SDMAARM2_EVT_MIRROR)	32	R	0000_0000h	7.2.6.23/ 1195
302C_0064	DMA Requests 2 (SDMAARM2_EVT_MIRROR2)	32	R	0000_0000h	7.2.6.24/ 1195
302C_0070	Cross-Trigger Events Configuration Register 1 (SDMAARM2_XTRIG_CONF1)	32	R/W	0000_0000h	7.2.6.25/ 1196
302C_0074	Cross-Trigger Events Configuration Register 2 (SDMAARM2_XTRIG_CONF2)	32	R/W	0000_0000h	7.2.6.26/ 1197
302C_0100	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI0)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0104	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI1)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0108	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI2)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_010C	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI3)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0110	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI4)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0114	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI5)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0118	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI6)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_011C	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI7)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0120	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI8)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0124	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI9)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0128	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI10)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_012C	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI11)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0130	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI12)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0134	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI13)	32	R/W	0000_0000h	7.2.6.27/ 1198

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302C_0138	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI14)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_013C	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI15)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0140	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI16)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0144	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI17)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0148	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI18)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_014C	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI19)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0150	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI20)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0154	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI21)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0158	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI22)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_015C	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI23)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0160	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI24)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0164	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI25)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0168	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI26)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_016C	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI27)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0170	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI28)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0174	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI29)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0178	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI30)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_017C	Channel Priority Registers (SDMAARM2_SDMA_CHNPRI31)	32	R/W	0000_0000h	7.2.6.27/ 1198
302C_0200	Channel Enable RAM (SDMAARM2_CHNENBL0)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0204	Channel Enable RAM (SDMAARM2_CHNENBL1)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0208	Channel Enable RAM (SDMAARM2_CHNENBL2)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_020C	Channel Enable RAM (SDMAARM2_CHNENBL3)	32	R/W	0000_0000h	7.2.6.28/ 1199

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302C_0210	Channel Enable RAM (SDMAARM2_CHNENBL4)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0214	Channel Enable RAM (SDMAARM2_CHNENBL5)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0218	Channel Enable RAM (SDMAARM2_CHNENBL6)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_021C	Channel Enable RAM (SDMAARM2_CHNENBL7)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0220	Channel Enable RAM (SDMAARM2_CHNENBL8)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0224	Channel Enable RAM (SDMAARM2_CHNENBL9)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0228	Channel Enable RAM (SDMAARM2_CHNENBL10)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_022C	Channel Enable RAM (SDMAARM2_CHNENBL11)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0230	Channel Enable RAM (SDMAARM2_CHNENBL12)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0234	Channel Enable RAM (SDMAARM2_CHNENBL13)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0238	Channel Enable RAM (SDMAARM2_CHNENBL14)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_023C	Channel Enable RAM (SDMAARM2_CHNENBL15)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0240	Channel Enable RAM (SDMAARM2_CHNENBL16)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0244	Channel Enable RAM (SDMAARM2_CHNENBL17)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0248	Channel Enable RAM (SDMAARM2_CHNENBL18)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_024C	Channel Enable RAM (SDMAARM2_CHNENBL19)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0250	Channel Enable RAM (SDMAARM2_CHNENBL20)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0254	Channel Enable RAM (SDMAARM2_CHNENBL21)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0258	Channel Enable RAM (SDMAARM2_CHNENBL22)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_025C	Channel Enable RAM (SDMAARM2_CHNENBL23)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0260	Channel Enable RAM (SDMAARM2_CHNENBL24)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0264	Channel Enable RAM (SDMAARM2_CHNENBL25)	32	R/W	0000_0000h	7.2.6.28/ 1199

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302C_0268	Channel Enable RAM (SDMAARM2_CHNENBL26)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_026C	Channel Enable RAM (SDMAARM2_CHNENBL27)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0270	Channel Enable RAM (SDMAARM2_CHNENBL28)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0274	Channel Enable RAM (SDMAARM2_CHNENBL29)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0278	Channel Enable RAM (SDMAARM2_CHNENBL30)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_027C	Channel Enable RAM (SDMAARM2_CHNENBL31)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0280	Channel Enable RAM (SDMAARM2_CHNENBL32)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0284	Channel Enable RAM (SDMAARM2_CHNENBL33)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0288	Channel Enable RAM (SDMAARM2_CHNENBL34)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_028C	Channel Enable RAM (SDMAARM2_CHNENBL35)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0290	Channel Enable RAM (SDMAARM2_CHNENBL36)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0294	Channel Enable RAM (SDMAARM2_CHNENBL37)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_0298	Channel Enable RAM (SDMAARM2_CHNENBL38)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_029C	Channel Enable RAM (SDMAARM2_CHNENBL39)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_02A0	Channel Enable RAM (SDMAARM2_CHNENBL40)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_02A4	Channel Enable RAM (SDMAARM2_CHNENBL41)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_02A8	Channel Enable RAM (SDMAARM2_CHNENBL42)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_02AC	Channel Enable RAM (SDMAARM2_CHNENBL43)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_02B0	Channel Enable RAM (SDMAARM2_CHNENBL44)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_02B4	Channel Enable RAM (SDMAARM2_CHNENBL45)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_02B8	Channel Enable RAM (SDMAARM2_CHNENBL46)	32	R/W	0000_0000h	7.2.6.28/ 1199
302C_02BC	Channel Enable RAM (SDMAARM2_CHNENBL47)	32	R/W	0000_0000h	7.2.6.28/ 1199

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302C_1000	SDMA DONE0 Configuration (SDMAARM2_DONE0_CONFIG)	32	R/W	1F1F_1F1Fh	7.2.6.29/ 1199
302C_1004	SDMA DONE1 Configuration (SDMAARM2_DONE1_CONFIG)	32	R/W	1F1F_1F1Fh	7.2.6.30/ 1201
30BD_0000	Arm platform Channel 0 Pointer (SDMAARM1_MC0PTR)	32	R/W	0000_0000h	7.2.6.1/ 1182
30BD_0004	Channel Interrupts (SDMAARM1_INTR)	32	w1c	0000_0000h	7.2.6.2/ 1182
30BD_0008	Channel Stop/Channel Status (SDMAARM1_STOP_STAT)	32	w1c	0000_0000h	7.2.6.3/ 1182
30BD_000C	Channel Start (SDMAARM1_HSTART)	32	R/W	0000_0000h	7.2.6.4/ 1183
30BD_0010	Channel Event Override (SDMAARM1_EVTOVR)	32	R/W	0000_0000h	7.2.6.5/ 1183
30BD_0014	Channel BP Override (SDMAARM1_DSPOVR)	32	R/W	FFFF_FFFFh	7.2.6.6/ 1184
30BD_0018	Channel Arm platform Override (SDMAARM1_HOSTOVR)	32	R/W	0000_0000h	7.2.6.7/ 1184
30BD_001C	Channel Event Pending (SDMAARM1_EVTPEND)	32	w1c	0000_0000h	7.2.6.8/ 1184
30BD_0024	Reset Register (SDMAARM1_RESET)	32	R	0000_0000h	7.2.6.9/ 1185
30BD_0028	DMA Request Error Register (SDMAARM1_EVTERR)	32	R	0000_0000h	7.2.6.10/ 1186
30BD_002C	Channel Arm platform Interrupt Mask (SDMAARM1_INTRMASK)	32	R/W	0000_0000h	7.2.6.11/ 1186
30BD_0030	Schedule Status (SDMAARM1_PSW)	32	R	0000_0000h	7.2.6.12/ 1187
30BD_0034	DMA Request Error Register (SDMAARM1_EVTERRDBG)	32	R	0000_0000h	7.2.6.13/ 1187
30BD_0038	Configuration Register (SDMAARM1_CONFIG)	32	R/W	0000_0003h	7.2.6.14/ 1188
30BD_003C	SDMA LOCK (SDMAARM1_SDMA_LOCK)	32	R/W	0000_0000h	7.2.6.15/ 1189
30BD_0040	OnCE Enable (SDMAARM1_ONCE_ENB)	32	R/W	0000_0000h	7.2.6.16/ 1190
30BD_0044	OnCE Data Register (SDMAARM1_ONCE_DATA)	32	R/W	0000_0000h	7.2.6.17/ 1190
30BD_0048	OnCE Instruction Register (SDMAARM1_ONCE_INSTR)	32	R/W	0000_0000h	7.2.6.18/ 1191
30BD_004C	OnCE Status Register (SDMAARM1_ONCE_STAT)	32	R	0000_E000h	7.2.6.19/ 1191
30BD_0050	OnCE Command Register (SDMAARM1_ONCE_CMD)	32	R/W	0000_0000h	7.2.6.20/ 1193

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BD_0058	Illegal Instruction Trap Address (SDMAARM1_ILLINSTADDR)	32	R/W	0000_0001h	7.2.6.21/ 1193
30BD_005C	Channel 0 Boot Address (SDMAARM1_CHN0ADDR)	32	R/W	0000_0050h	7.2.6.22/ 1194
30BD_0060	DMA Requests (SDMAARM1_EVT_MIRROR)	32	R	0000_0000h	7.2.6.23/ 1195
30BD_0064	DMA Requests 2 (SDMAARM1_EVT_MIRROR2)	32	R	0000_0000h	7.2.6.24/ 1195
30BD_0070	Cross-Trigger Events Configuration Register 1 (SDMAARM1_XTRIG_CONF1)	32	R/W	0000_0000h	7.2.6.25/ 1196
30BD_0074	Cross-Trigger Events Configuration Register 2 (SDMAARM1_XTRIG_CONF2)	32	R/W	0000_0000h	7.2.6.26/ 1197
30BD_0100	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI0)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0104	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI1)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0108	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI2)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_010C	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI3)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0110	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI4)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0114	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI5)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0118	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI6)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_011C	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI7)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0120	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI8)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0124	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI9)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0128	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI10)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_012C	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI11)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0130	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI12)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0134	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI13)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0138	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI14)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_013C	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI15)	32	R/W	0000_0000h	7.2.6.27/ 1198

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BD_0140	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI16)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0144	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI17)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0148	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI18)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_014C	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI19)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0150	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI20)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0154	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI21)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0158	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI22)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_015C	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI23)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0160	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI24)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0164	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI25)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0168	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI26)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_016C	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI27)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0170	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI28)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0174	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI29)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0178	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI30)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_017C	Channel Priority Registers (SDMAARM1_SDMA_CHNPRI31)	32	R/W	0000_0000h	7.2.6.27/ 1198
30BD_0200	Channel Enable RAM (SDMAARM1_CHNENBL0)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0204	Channel Enable RAM (SDMAARM1_CHNENBL1)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0208	Channel Enable RAM (SDMAARM1_CHNENBL2)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_020C	Channel Enable RAM (SDMAARM1_CHNENBL3)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0210	Channel Enable RAM (SDMAARM1_CHNENBL4)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0214	Channel Enable RAM (SDMAARM1_CHNENBL5)	32	R/W	0000_0000h	7.2.6.28/ 1199

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BD_0218	Channel Enable RAM (SDMAARM1_CHNENBL6)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_021C	Channel Enable RAM (SDMAARM1_CHNENBL7)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0220	Channel Enable RAM (SDMAARM1_CHNENBL8)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0224	Channel Enable RAM (SDMAARM1_CHNENBL9)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0228	Channel Enable RAM (SDMAARM1_CHNENBL10)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_022C	Channel Enable RAM (SDMAARM1_CHNENBL11)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0230	Channel Enable RAM (SDMAARM1_CHNENBL12)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0234	Channel Enable RAM (SDMAARM1_CHNENBL13)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0238	Channel Enable RAM (SDMAARM1_CHNENBL14)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_023C	Channel Enable RAM (SDMAARM1_CHNENBL15)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0240	Channel Enable RAM (SDMAARM1_CHNENBL16)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0244	Channel Enable RAM (SDMAARM1_CHNENBL17)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0248	Channel Enable RAM (SDMAARM1_CHNENBL18)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_024C	Channel Enable RAM (SDMAARM1_CHNENBL19)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0250	Channel Enable RAM (SDMAARM1_CHNENBL20)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0254	Channel Enable RAM (SDMAARM1_CHNENBL21)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0258	Channel Enable RAM (SDMAARM1_CHNENBL22)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_025C	Channel Enable RAM (SDMAARM1_CHNENBL23)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0260	Channel Enable RAM (SDMAARM1_CHNENBL24)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0264	Channel Enable RAM (SDMAARM1_CHNENBL25)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0268	Channel Enable RAM (SDMAARM1_CHNENBL26)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_026C	Channel Enable RAM (SDMAARM1_CHNENBL27)	32	R/W	0000_0000h	7.2.6.28/ 1199

Table continues on the next page...

SDMAARM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BD_0270	Channel Enable RAM (SDMAARM1_CHNENBL28)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0274	Channel Enable RAM (SDMAARM1_CHNENBL29)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0278	Channel Enable RAM (SDMAARM1_CHNENBL30)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_027C	Channel Enable RAM (SDMAARM1_CHNENBL31)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0280	Channel Enable RAM (SDMAARM1_CHNENBL32)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0284	Channel Enable RAM (SDMAARM1_CHNENBL33)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0288	Channel Enable RAM (SDMAARM1_CHNENBL34)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_028C	Channel Enable RAM (SDMAARM1_CHNENBL35)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0290	Channel Enable RAM (SDMAARM1_CHNENBL36)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0294	Channel Enable RAM (SDMAARM1_CHNENBL37)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_0298	Channel Enable RAM (SDMAARM1_CHNENBL38)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_029C	Channel Enable RAM (SDMAARM1_CHNENBL39)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_02A0	Channel Enable RAM (SDMAARM1_CHNENBL40)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_02A4	Channel Enable RAM (SDMAARM1_CHNENBL41)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_02A8	Channel Enable RAM (SDMAARM1_CHNENBL42)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_02AC	Channel Enable RAM (SDMAARM1_CHNENBL43)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_02B0	Channel Enable RAM (SDMAARM1_CHNENBL44)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_02B4	Channel Enable RAM (SDMAARM1_CHNENBL45)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_02B8	Channel Enable RAM (SDMAARM1_CHNENBL46)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_02BC	Channel Enable RAM (SDMAARM1_CHNENBL47)	32	R/W	0000_0000h	7.2.6.28/ 1199
30BD_1000	SDMA DONE0 Configuration (SDMAARM1_DONE0_CONFIG)	32	R/W	1F1F_1F1Fh	7.2.6.29/ 1199
30BD_1004	SDMA DONE1 Configuration (SDMAARM1_DONE1_CONFIG)	32	R/W	1F1F_1F1Fh	7.2.6.30/ 1201

7.2.6.1 Arm platform Channel 0 Pointer (SDMAARMx_MC0PTR)

Address: Base address + 0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MC0PTR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_MC0PTR field descriptions

Field	Description
MC0PTR	**Channel 0 Pointer** contains the 32-bit address, in Arm platform memory, of channel 0 control block (the boot channel). Appendix A fully describes the SDMA Application Programming Interface (API). The Arm platform has a read/write access and the SDMA has a read-only access.

7.2.6.2 Channel Interrupts (SDMAARMx_INTR)

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HI[31:0]																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_INTR field descriptions

Field	Description
HI[31:0]	The Arm platform Interrupts register contains the 32 HI[i] bits. If any bit is set, it will cause an interrupt to the Arm platform. This register is a "write-ones" register to the Arm platform. When the Arm platform sets a bit in this register the corresponding HI[i] bit is cleared. The interrupt service routine should clear individual channel bits when their interrupts are serviced, failure to do so will cause continuous interrupts. The SDMA is responsible for setting the HI[i] bit corresponding to the current channel when the corresponding `done` instruction is executed.

7.2.6.3 Channel Stop/Channel Status (SDMAARMx_STOP_STAT)

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HE																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_STOP_STAT field descriptions

Field	Description
HE	This 32-bit register gives access to the Arm platform Enable bits. There is one bit for every channel. This register is a "write-ones" register to the Arm platform. When the Arm platform writes 1 in bit i of this register, it clears the HE[i] and HSTART[i] bits. Reading this register yields the current state of the HE[i] bits.

7.2.6.4 Channel Start (SDMAARMx_HSTART)

Address: Base address + Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HSTART_HE																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_HSTART field descriptions

Field	Description
HSTART_HE	The HSTART_HE registers are 32 bits wide with one bit for every channel. When a bit is written to 1, it enables the corresponding channel. Two physical registers are accessed with that address (HSTART and HE), which enables the Arm platform to trigger a channel a second time before the first trigger is processed. • This register is a "write-ones" register to the Arm platform. Neither HSTART[i] bit can be set while the corresponding HE[i] bit is cleared. • When the Arm platform tries to set the HSTART[i] bit by writing a one (if the corresponding HE[i] bit is clear), the bit in the HSTART[i] register will remain cleared and the HE[i] bit will be set. • If the corresponding HE[i] bit was already set, the HSTART[i] bit will be set. The next time the SDMA channel *i* attempts to clear the HE[i] bit by means of a `done` instruction, the bit in the HSTART[i] register will be cleared and the HE[i] bit will take the old value of the HSTART[i] bit. • Reading this register yields the current state of the HSTART[i] bits. This mechanism enables the Arm platform to pipeline two HSTART commands per channel.

7.2.6.5 Channel Event Override (SDMAARMx_EVTOVR)

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EO																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_EVTOVR field descriptions

Field	Description
EO	The Channel Event Override register contains the 32 EO[i] bits. A bit set in this register causes the SDMA to ignore DMA requests when scheduling the corresponding channel.

7.2.6.6 Channel BP Override (SDMAARMx_DSPOVR)

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DO																															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

SDMAARMx_DSPOVR field descriptions

Field	Description
DO	This register is reserved. All DO bits should be set to the reset value of 1. A setting of 0 will prevent SDMA channels from starting according to the condition described in Runnable Channels Evaluation. 0 - Reserved 1 - Reset value.

7.2.6.7 Channel Arm platform Override (SDMAARMx_HOSTOVR)

Address: Base address + 18h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HO																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_HOSTOVR field descriptions

Field	Description
HO	The Channel Arm platform Override register contains the 32 HO[i] bits. A bit set in this register causes the SDMA to ignore the Arm platform enable bit (HE) when scheduling the corresponding channel.

7.2.6.8 Channel Event Pending (SDMAARMx_EVTPEND)

Address: Base address + 1Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EP																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_EVTPEND field descriptions

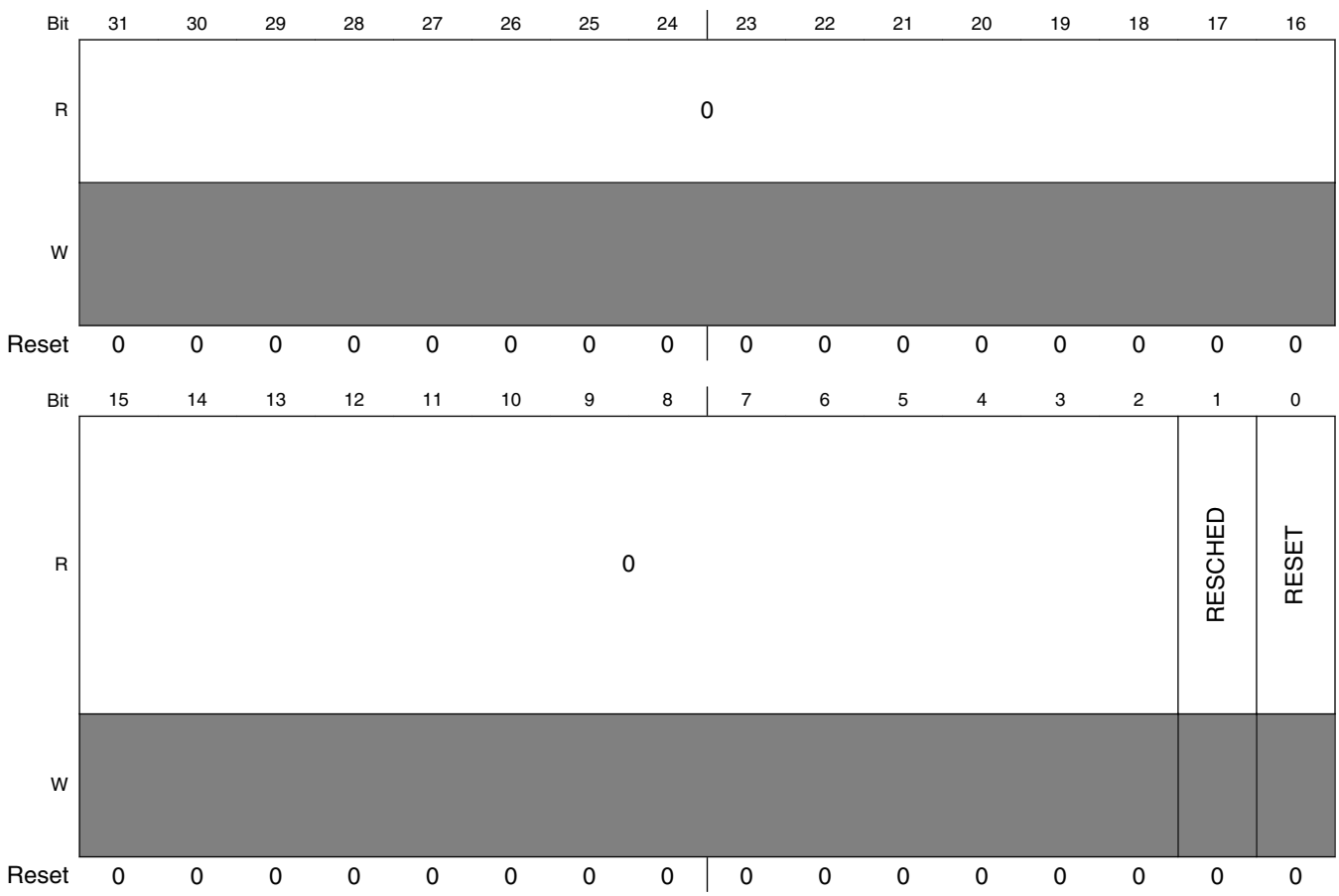
Field	Description
EP	The Channel Event Pending register contains the 32 EP[i] bits. Reading this register enables the Arm platform to determine what channels are pending after the reception of a DMA request.

SDMAARMx_EVTPEND field descriptions (continued)

Field	Description
	• Setting a bit in this register causes the SDMA to reevaluate scheduling as if a DMA request mapped on this channel had occurred. This is useful for starting up channels, so that initialization is done before awaiting the first request. The scheduler can also set bits in the EVTPEND register according to the received DMA requests. • The EP[i] bit may be cleared by the `done` instruction when running the channel *i* script. This a "write-ones" mechanism: Writing a '0' does not clear the corresponding bit.

7.2.6.9 Reset Register (SDMAARMx_RESET)

Address: Base address + 24h offset



SDMAARMx_RESET field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1 RESCHED	When set, this bit forces the SDMA to reschedule as if a script had executed a `done` instruction. This enables the Arm platform to recover from a runaway script on a channel by clearing its HE[i] bit via the STOP register, and then forcing a reschedule via the RESCHED bit. The RESCHED bit is cleared when the context switch starts.

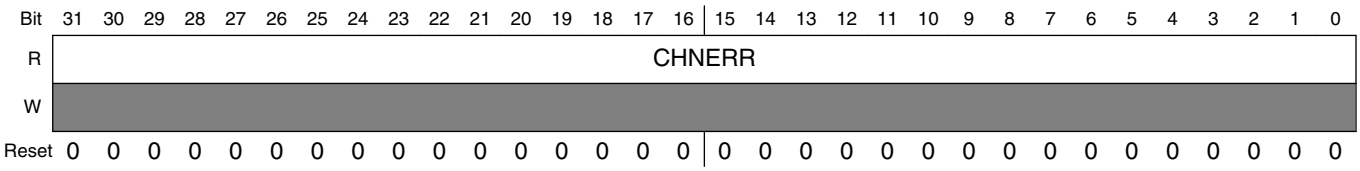
Table continues on the next page...

SDMAARMx_RESET field descriptions (continued)

Field	Description
0 RESET	When set, this bit causes the SDMA to be held in a software reset. The internal reset signal is held low 16 cycles; the RESET bit is automatically cleared when the internal reset signal rises.

7.2.6.10 DMA Request Error Register (SDMAARMx_EVTERR)

Address: Base address + 28h offset

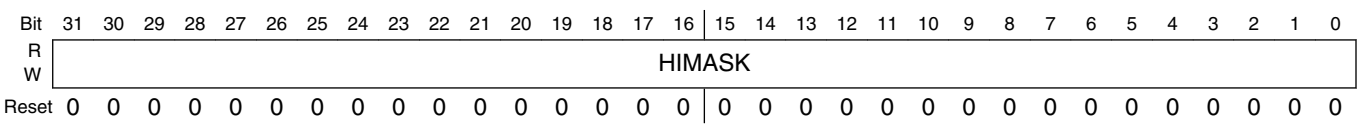


SDMAARMx_EVTERR field descriptions

Field	Description
CHNERR	This register is used by the SDMA to warn the Arm platform when an incoming DMA request was detected and it triggers a channel that is already pending or being serviced. This probably means there is an overflow of data for that channel. • An interrupt is sent to the Arm platform if the corresponding channel bit is set in the INTRMASK register. • This is a "write-ones" register for the scheduler. It is only able to set the flags. The flags are cleared when the register is read by the Arm platform or during SDMA reset. • The CHNERR[i] bit is set when a DMA request that triggers channel *i* is received through the corresponding input pins and the EP[i] bit is already set; the EVTERR[i] bit is unaffected if the Arm platform tries to set the EP[i] bit, whereas, that EP[i] bit is already set.

7.2.6.11 Channel Arm platform Interrupt Mask (SDMAARMx_INTRMASK)

Address: Base address + 2Ch offset



SDMAARMx_INTRMASK field descriptions

Field	Description
HIMASK	The Interrupt Mask Register contains 32 interrupt generation mask bits. If bit HIMASK[i] is set, the HI[i] bit is set and an interrupt is sent to the Arm platform when a DMA request error is detected on channel *i* (for example, EVTERR[i] is set).

7.2.6.12 Schedule Status (SDMAARMx_PSW)

Address: Base address + 30h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																NCP[2:0]			NCR[4:0]					CCP[2:0]				CCR[4:0]			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_PSW field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–13 NCP[2:0]	The Next Channel Priority gives the next pending channel priority. When the priority is 0, it means there is no pending channel and the NCR value has no meaning. 0 No running channel 1 Active channel priority
12–8 NCR[4:0]	The Next Channel Register indicates the number of the next scheduled pending channel with the highest priority.
7–4 CCP[2:0]	The Current Channel Priority indicates the priority of the current active channel. When the priority is 0, no channel is running: The SDMA is idle and the CCR value has no meaning. In the case that the SDMA has finished running the channel and has entered sleep state, CCP will indicate the priority of previous running channel. 0 No running channel 1 Active channel priority
CCR[4:0]	The Current Channel Register indicates the number of the channel that is being executed by the SDMA. SDMA. In the case that the SDMA has finished running the channel and has entered sleep state, CCR will indicate the previous running channel.

7.2.6.13 DMA Request Error Register (SDMAARMx_EVTERRDBG)

Address: Base address + 34h offset

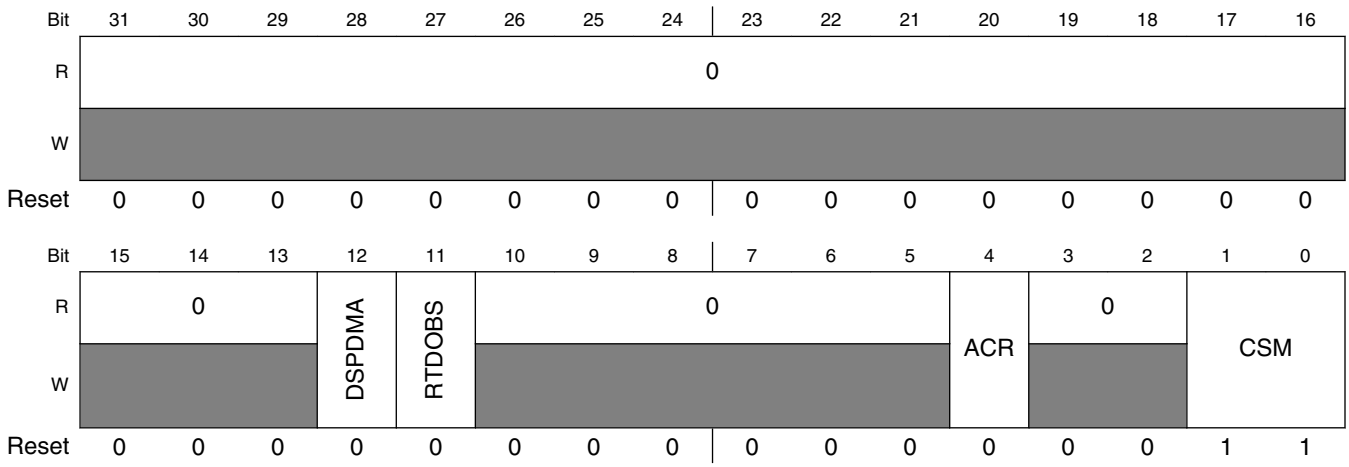
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHNERR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_EVTERRDBG field descriptions

Field	Description
CHNERR	This register is the same as EVTERR, except reading it does not clear its contents. This address is meant to be used in debug mode. The Arm platform OnCE may check this register value without modifying it.

7.2.6.14 Configuration Register (SDMAARMx_CONFIG)

Address: Base address + 38h offset



SDMAARMx_CONFIG field descriptions

Field	Description
31–13 Reserved	This read-only field is reserved and always has the value 0.
12 DSPDMA	This bit's function is reserved and should be configured as zero. 0 - Reset Value 1 - Reserved
11 RTDOBS	Indicates if Real-Time Debug pins are used: They do not toggle by default in order to reduce power consumption. 0 RTD pins disabled 1 RTD pins enabled
10–5 Reserved	This read-only field is reserved and always has the value 0.
4 ACR	Arm platform DMA / SDMA Core Clock Ratio. Selects the clock ratio between Arm platform DMA interfaces (burst DMA and peripheral DMA) and the internal SDMA core clock. The frequency selection is determined separately by the chip clock controller. This bit has to match the configuration of the chip clock controller that generates the clocks used in the SDMA. 0 Arm platform DMA interface frequency equals twice core frequency 1 Arm platform DMA interface frequency equals core frequency
3–2 Reserved	This read-only field is reserved and always has the value 0.
CSM	Selects the Context Switch Mode. The Arm platform has a read/write access. The SDMA cannot modify that register. The value at reset is 3, which selects the dynamic context switch by default. That register can be modified at anytime but the new context switch configuration will only be taken into account at the start of the next restore phase. NOTE: The first call to SDMA's channel 0 Bootload script after reset should use static context switch mode to ensure the context RAM for channel 0 is initialized in the channel SAVE Phase. After Channel 0 is run once, then any of the dynamic context modes can be used.

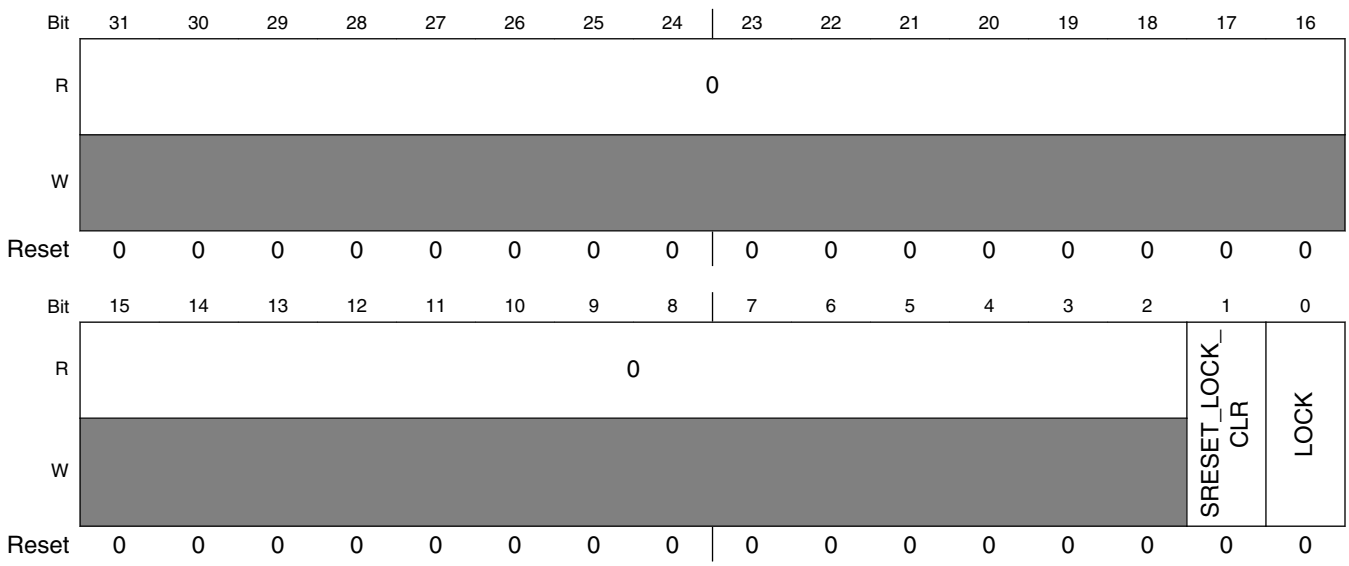
Table continues on the next page...

SDMAARMx_CONFIG field descriptions (continued)

Field	Description
	0 static 1 dynamic low power 2 dynamic with no loop 3 dynamic

7.2.6.15 SDMA LOCK (SDMAARMx_SDMA_LOCK)

Address: Base address + 3Ch offset



SDMAARMx_SDMA_LOCK field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1 SRESET_LOCK_CLR	The SRESET_LOCK_CLR bit determine if the LOCK bit is cleared on a software reset triggered by writing to the RESET register. This bit cannot be changed if LOCK=1. SREST_LOCK_CLR is cleared by conditions that clear the LOCK bit. 0 Software Reset does not clear the LOCK bit. 1 Software Reset clears the LOCK bit.
0 LOCK	The LOCK bit is used to restrict access to update SDMA script memory through ROM channel zero scripts and through the OnCE interface under Arm platform control. The LOCK bit is set: • The SDMA_LOCK, ONCE_ENB,CH0ADDR, and ILLINSTADDR registers cannot be written. These registers can be read, but writes are ignored. • SDMA software executing out of ROM or RAM may check the LOCK bit in the LOCK register Lock Status Register (SDMACORE_SDMA_LOCK) to determine if certain operations are allowed, such as up-loading new scripts.

Table continues on the next page...

SDMAARMx_SDMA_LOCK field descriptions (continued)

Field	Description
	Once the LOCK bit is set to 1, only a reset can clear it. The LOCK bit is cleared by a hardware reset. LOCK is cleared by a software reset only if SRESET_LOCK_CLR is set. 0 LOCK disengaged. 1 LOCK enabled.

7.2.6.16 OnCE Enable (SDMAARMx_ONCE_ENB)

Address: Base address + 40h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															ENB
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_ONCE_ENB field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 ENB	The OnCE Enable register selects the OnCE control source: When cleared (0), the OnCE registers are accessed through the JTAG interface; when set (1), the OnCE registers may be accessed by the Arm platform through the addresses described, as follows. • After reset, the OnCE registers are accessed through the JTAG interface. • Writing a 1 to ENB enables the Arm platform to access the ONCE_* as any other SDMA control register. • When cleared (0), all the ONCE_xxx registers cannot be written. The value of ENB cannot be changed if the LOCK bit in the SDMA_LOCK register is set.

7.2.6.17 OnCE Data Register (SDMAARMx_ONCE_DATA)

Address: Base address + 44h offset

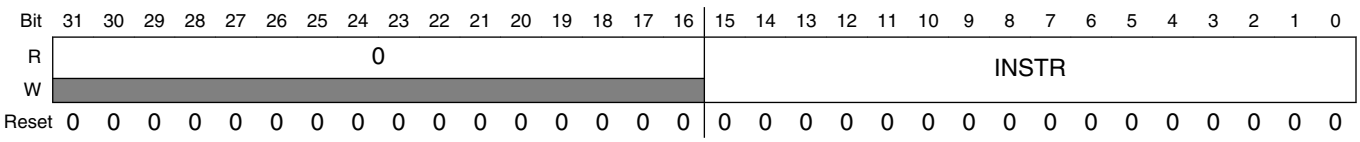
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_ONCE_DATA field descriptions

Field	Description
DATA	Data register of the OnCE JTAG controller. Refer to OnCE and Real-Time Debug for information on this register.

7.2.6.18 OnCE Instruction Register (SDMAARMx_ONCE_INSTR)

Address: Base address + 48h offset

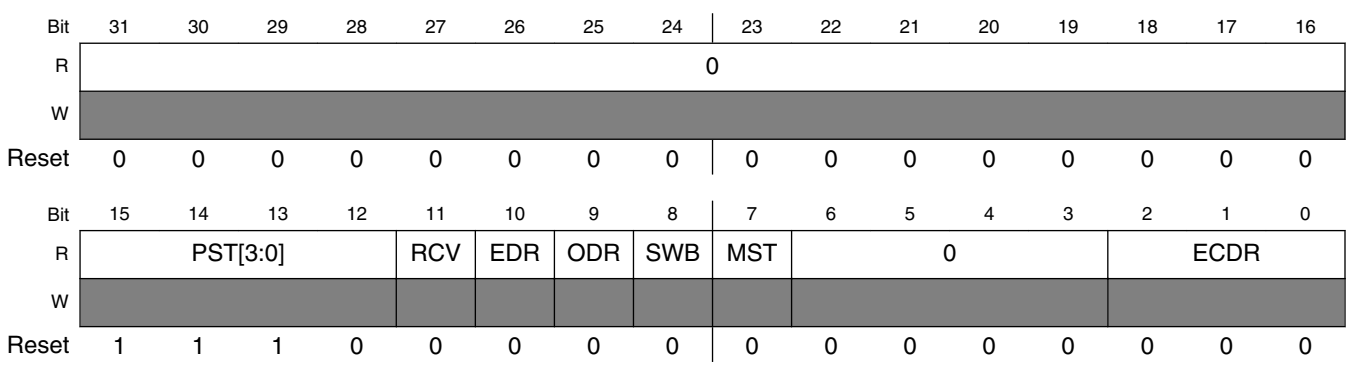


SDMAARMx_ONCE_INSTR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
INSTR	Instruction register of the OnCE JTAG controller. Refer to OnCE and Real-Time Debug for information on this register.

7.2.6.19 OnCE Status Register (SDMAARMx_ONCE_STAT)

Address: Base address + 4Ch offset



SDMAARMx_ONCE_STAT field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–12 PST[3:0]	The Processor Status bits reflect the state of the SDMA RISC engine. Its states are as follows: • The "Program" state is the usual instruction execution cycle. • The "Data" state is inserted when there are wait-states during a load or a store on the data bus (ld or st).

Table continues on the next page...

SDMAARMx_ONCE_STAT field descriptions (continued)

Field	Description
	• The "Change of Flow" state is the second cycle of any instruction that breaks the sequence of instructions (jumps and channel switching instructions). • The "Change of Flow in Loop" state is used when an error causes a hardware loop exit. • The "Debug" state means the SDMA is in debug mode. • The "Functional Unit" state is inserted when there are wait-states during a load or a store on the functional units bus (ldf or stf). • In "Sleep" modes, no script is running (this is the RISC engine idle state). The "after Reset" is slightly different because no context restoring phase will happen when a channel is triggered: The script located at address 0 will be executed (boot operation). • The "in Sleep" states are the same as above except they do not have any corresponding channel: They are used when entering debug mode after reset. The reason is that it is necessary to return to the "Sleep after Reset" state when leaving debug mode. 0 Program 1 Data 2 Change of Flow 3 Change of Flow in Loop 4 Debug 5 Functional Unit 6 Sleep 7 Save 8 Program in Sleep 9 Data in Sleep 10 Change of Flow in Sleep 11 Change Flow in Loop in Sleep 12 Debug in Sleep 13 Functional Unit in Sleep 14 Sleep after Reset 15 Restore
11 RCV	After each write access to the real time buffer (RTB), the RCV bit is set. This bit is cleared after execution of an `rbuffer` command and on a JTAG reset.
10 EDR	This flag is raised when the SDMA has entered debug mode after an external debug request.
9 ODR	This flag is raised when the SDMA has entered debug mode after a OnCE debug request.
8 SWB	This flag is raised when the SDMA has entered debug mode after a software breakpoint.
7 MST	This flag is raised when the OnCE is controlled from the Arm platform peripheral interface. 0 The JTAG interface controls the OnCE. 1 The Arm platform peripheral interface controls the OnCE.
6–3 Reserved	This read-only field is reserved and always has the value 0.
ECDR	Event Cell Debug Request. If the debug request comes from the event cell, the reason for entering debug mode is given by the EDR bits. If all three bits of the EDR are reset, then it did not generate any debug request. If the cell did generate a debug request, then at least one of the EDR bits is set (the meaning of the encoding is given below). The encoding of the EDR bits is useful to find out more precisely why the debug request was generated. A debug request from an event cell is generated for a specific combination of the addra_cond, addrb_cond, and data_cond conditions. The value of those fields is given by the EDR bits.

Table continues on the next page...

SDMAARMx_ONCE_STAT field descriptions (continued)

Field	Description
	0 1 matched addra_cond 1 1 matched addrb_cond 2 1 matched data_cond

7.2.6.20 OnCE Command Register (SDMAARMx_ONCE_CMD)

Address: Base address + 50h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																												CMD			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_ONCE_CMD field descriptions

Field	Description
31–4 Reserved	This read-only field is reserved and always has the value 0.
CMD	Writing to this register will cause the OnCE to execute the command that is written. When needed, the ONCE_DATA and ONCE_INSTR registers should be loaded with the correct value before writing the command to that register. For a list of the OnCE commands and their usage, see OnCE and Real-Time Debug. **NOTE:** 7-15 *reserved* 0 rstatus 1 dmov 2 exec_once 3 run_core 4 exec_core 5 debug_rqst 6 rbuffer

7.2.6.21 Illegal Instruction Trap Address (SDMAARMx_ILLINSTADDR)

Address: Base address + 58h offset

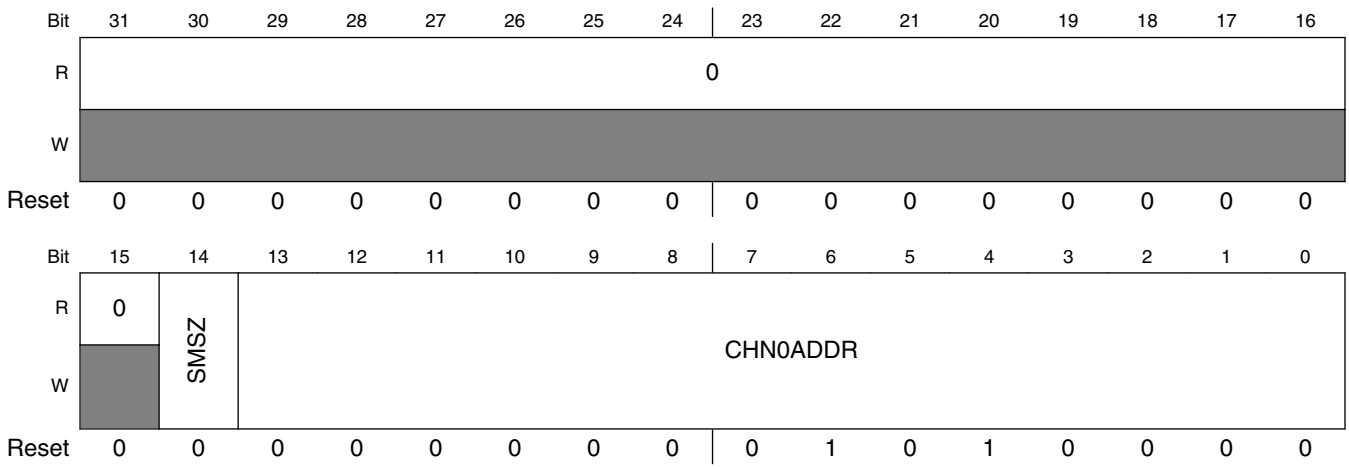
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																		ILLINSTADDR													
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

SDMAARMx_ILLINSTADDR field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
ILLINSTADDR	The Illegal Instruction Trap Address is the address where the SDMA jumps when an illegal instruction is executed. It is 0x0001 after reset. The value of ILLINSTADDR cannot be changed if the LOCK bit in the SDMA_LOCK register is set.

7.2.6.22 Channel 0 Boot Address (SDMAARMx_CHN0ADDR)

Address: Base address + 5Ch offset



SDMAARMx_CHN0ADDR field descriptions

Field	Description
31–15 Reserved	This read-only field is reserved and always has the value 0.
14 SMSZ	The bit 14 (Scratch Memory Size) determines if scratch memory must be available after every channel context. After reset, it is equal to 0, which defines a RAM space of 24 words for each channel. All of this area stores the channel context. By setting this bit, 32 words are reserved for every channel context, which gives eight additional words that can be used by the channel script to store any type of data. Those words are never erased by the context switching mechanism. The value of SMSZ cannot be changed if the LOCK bit in the SDMA_LOCK register is set. 0 24 words per context 1 32 words per context
CHN0ADDR	This 14-bit register is used by the boot code of the SDMA. After reset, it points to the standard boot routine in ROM (channel 0 routine). By changing this address, you can perform a boot sequence with your own routine. The very first instructions of the boot code fetch the contents of this register (it is also mapped in the SDMA memory space) and jump to the given address. The reset value is 0x0050 (decimal 80). The value of CHN0ADDR cannot be changed if the LOCK bit in the SDMA_LOCK register is set.

7.2.6.23 DMA Requests (SDMAARMx_EVT_MIRROR)

Address: Base address + 60h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EVENTS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_EVT_MIRROR field descriptions

Field	Description
EVENTS	This register reflects the DMA requests received by the SDMA for events 31-0. The Arm platform and the SDMA have a read-only access. There is one bit associated with each of 32 DMA request events. This information may be useful during debug of the blocks that generate the DMA requests. The EVT_MIRROR register is cleared following read access. 0 DMA request event not pending 1 DMA request event pending

7.2.6.24 DMA Requests 2 (SDMAARMx_EVT_MIRROR2)

Address: Base address + 64h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																EVENTS[47:32]															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_EVT_MIRROR2 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
EVENTS[47:32]	This register reflects the DMA requests received by the SDMA for events 47-32. The Arm platform and the SDMA have a read-only access. There is one bit associated with each of DMA request events. This information may be useful during debug of the blocks that generate the DMA requests. The EVT_MIRROR2 register is cleared following read access. 0 - DMA request event not pending 1- DMA request event pending

7.2.6.25 Cross-Trigger Events Configuration Register 1 (SDMAARMx_XTRIG_CONF1)

Address: Base address + 70h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	CNF3	NUM3[5:0]						0	CNF2	NUM2[5:0]					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CNF1	NUM1[5:0]						0	CNF0	NUM0[5:0]					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_XTRIG_CONF1 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30 CNF3	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by the reception of a DMA request or by the starting of a channel script execution. 0 channel 1 DMA request
29–24 NUM3[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.
23 Reserved	This read-only field is reserved and always has the value 0.
22 CNF2	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution. 0 channel 1 DMA request
21–16 NUM2[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.
15 Reserved	This read-only field is reserved and always has the value 0.
14 CNF1	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution.

Table continues on the next page...

SDMAARMx_XTRIG_CONF1 field descriptions (continued)

Field	Description
	0 channel 1 DMA request
13–8 NUM1[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.
7 Reserved	This read-only field is reserved and always has the value 0.
6 CNF0	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution. 0 channel 1 DMA request
NUM0[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.

7.2.6.26 Cross-Trigger Events Configuration Register 2 (SDMAARMx_XTRIG_CONF2)

Address: Base address + 74h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	CNF7	NUM7[5:0]						0	CNF6	NUM6[5:0]					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CNF5	NUM5[5:0]						0	CNF4	NUM4[5:0]					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_XTRIG_CONF2 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30 CNF7	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution. 0 channel 1 DMA request

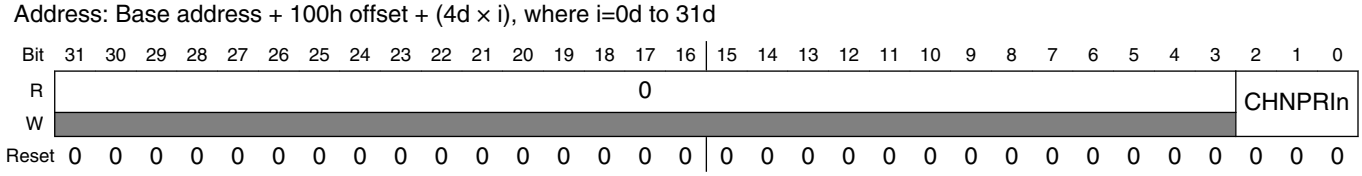
Table continues on the next page...

SDMAARMx_XTRIG_CONF2 field descriptions (continued)

Field	Description
29–24 NUM7[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.
23 Reserved	This read-only field is reserved and always has the value 0.
22 CNF6	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution. 0 channel 1 DMA request
21–16 NUM6[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.
15 Reserved	This read-only field is reserved and always has the value 0.
14 CNF5	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution 0 channel 1 DMA request
13–8 NUM5[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.
7 Reserved	This read-only field is reserved and always has the value 0.
6 CNF4	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution. 0 channel 1 DMA request
NUM4[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.

7.2.6.27 Channel Priority Registers (SDMAARMx_SDMA_CHNPRI*n*)

Address: Base address + 100h offset + (4d × i), where i=0d to 31d



SDMAARMx_SDMA_CHNPRI*n* field descriptions

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
CHNPRIn	This contains the priority of channel number *n*. Useful values are between 1 and 7; 0 is reserved by the SDMA hardware to determine when there is no pending channel. Reset value is 0, which prevents the channels from starting.

7.2.6.28 Channel Enable RAM (SDMAARMx_CHNENBL*n*)

Address: Base address + 200h offset + (4d × i), where i=0d to 47d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ENBLn																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMAARMx_CHNENBL*n* field descriptions

Field	Description
ENBLn	This 32-bit value selects the channels that are triggered by the DMA request number *n*. If ENBLn[i] is set to 1, bit EP[i] will be set when the DMA request *n* is received. These 48 32-bit registers are physically located in a RAM, with no known reset value. It is thus essential for the Arm platform to program them before any DMA request is triggered to the SDMA, otherwise an unpredictable combination of channels may be started.

7.2.6.29 SDMA DONE0 Configuration (SDMAARMx_DONE0_CONFIG)

Address: Base address + 1000h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DONE_SEL3	SW_DONE_DIS3	0	CH_SEL3					DONE_SEL2	SW_DONE_DIS2	0	CH_SEL2				
W																
Reset	0	0	0	1	1	1	1	1	0	0	0	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DONE_SEL1	SW_DONE_DIS1	0	CH_SEL1					DONE_SEL0	SW_DONE_DIS0	0	CH_SEL0				
W																
Reset	0	0	0	1	1	1	1	1	0	0	0	1	1	1	1	1

SDMAARMx_DONE0_CONFIG field descriptions

Field	Description
31 DONE_SEL3	Select Done from SW or HW for channel 3 0 HW 1 SW
30 SW_DONE_DIS3	Disable SW Done for channel 3 0 Enable 1 Disable
29 Reserved	This read-only field is reserved and always has the value 0.
28–24 CH_SEL3	Select event for channel 3 when Done is selected from HW. HW Done will be asserted when the negative edge of selected event's Event Pending (EP) is detected. 00000 - Event 0 00001 - Event 1 ... 11111 - Event 31
23 DONE_SEL2	Select Done from SW or HW for channel 2 0 HW 1 SW
22 SW_DONE_DIS2	Disable SW Done for channel 2 0 Enable 1 Disable
21 Reserved	This read-only field is reserved and always has the value 0.
20–16 CH_SEL2	Select event for channel 2 when Done is selected from HW. HW Done will be asserted when the negative edge of selected event's Event Pending (EP) is detected. 00000 - Event 0 00001 - Event 1 ... 11111 - Event 31
15 DONE_SEL1	Select Done from SW or HW for channel 1 0 HW 1 SW
14 SW_DONE_DIS1	Disable SW Done for channel 1 0 Enable 1 Disable
13 Reserved	This read-only field is reserved and always has the value 0.
12–8 CH_SEL1	Select event for channel 1 when Done is selected from HW. HW Done will be asserted when the negative edge of selected event's Event Pending (EP) is detected.

Table continues on the next page...

SDMAARMx_DONE0_CONFIG field descriptions (continued)

Field	Description
	00000 - Event 0 00001 - Event 1 ... 11111 - Event 31
7 DONE_SEL0	Select Done from SW or HW for channel 0 0 HW 1 SW
6 SW_DONE_DIS0	Disable SW Done for channel 0 0 Enable 1 Disable
5 Reserved	This read-only field is reserved and always has the value 0.
CH_SEL0	Select event for channel 0 when Done is selected from HW. HW Done will be asserted when the negative edge of selected event's Event Pending (EP) is detected. 00000 - Event 0 00001 - Event 1 ... 11111 - Event 31

7.2.6.30 SDMA DONE1 Configuration (SDMAARMx_DONE1_CONFIG)

Address: Base address + 1004h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DONE_SEL7	SW_DONE_DIS7	0	CH_SEL7					DONE_SEL6	SW_DONE_DIS6	0	CH_SEL6				
W																
Reset	0	0	0	1	1	1	1	1	0	0	0	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DONE_SEL5	SW_DONE_DIS5	0	CH_SEL5					DONE_SEL4	SW_DONE_DIS4	0	CH_SEL4				
W																
Reset	0	0	0	1	1	1	1	1	0	0	0	1	1	1	1	1

SDMAARMx_DONE1_CONFIG field descriptions

Field	Description
31 DONE_SEL7	Select Done from SW or HW for channel 7 0 HW 1 SW
30 SW_DONE_DIS7	Disable SW Done for channel 7 0 Enable 1 Disable
29 Reserved	This read-only field is reserved and always has the value 0.
28–24 CH_SEL7	Select event for channel 7 when Done is selected from HW. HW Done will be asserted when the negative edge of selected event's Event Pending (EP) is detected. 00000 - Event 0 00001 - Event 1 ... 11111 - Event 31
23 DONE_SEL6	Select Done from SW or HW for channel 6 0 HW 1 SW
22 SW_DONE_DIS6	Disable SW Done for channel 6 0 Enable 1 Disable
21 Reserved	This read-only field is reserved and always has the value 0.
20–16 CH_SEL6	Select event for channel 6 when Done is selected from HW. HW Done will be asserted when the negative edge of selected event's Event Pending (EP) is detected. 00000 - Event 0 00001 - Event 1 ... 11111 - Event 31
15 DONE_SEL5	Select Done from SW or HW for channel 5 0 HW 1 SW
14 SW_DONE_DIS5	Disable SW Done for channel 5 0 Enable 1 Disable
13 Reserved	This read-only field is reserved and always has the value 0.
12–8 CH_SEL5	Select event for channel 5 when Done is selected from HW. HW Done will be asserted when the negative edge of selected event's Event Pending (EP) is detected.

Table continues on the next page...

SDMAARMx_DONE1_CONFIG field descriptions (continued)

Field	Description
	00000 - Event 0 00001 - Event 1 ... 11111 - Event 31
7 DONE_SEL4	Select Done from SW or HW for channel 4 0 HW 1 SW
6 SW_DONE_DIS4	Disable SW Done for channel 4 0 Enable 1 Disable
5 Reserved	This read-only field is reserved and always has the value 0.
CH_SEL4	Select event for channel 4 when Done is selected from HW. HW Done will be asserted when the negative edge of selected event's Event Pending (EP) is detected. 00000 - Event 0 00001 - Event 1 ... 11111 - Event 31

7.2.7 BP Memory Map and Control Register Definitions

The following section describes SDMA control registers available to the BP.

NOTE

These registers are physically implemented in all platforms, but are not accessible when the SDMA BP control port is not connected. Reset values are calculated to allow the system to work when those registers cannot be accessed.

All registers are clocked with the SDMA clock (which means the SDMA clock must be running when the BP wants to access any register).

SDMABP memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302B_0000	Channel 0 Pointer (SDMABP3_DC0PTR)	32	R/W	0000_0000h	7.2.7.1/ 1205
302B_0004	Channel Interrupts (SDMABP3_INTR)	32	w1c	0000_0000h	7.2.7.2/ 1205
302B_0008	Channel Stop/Channel Status (SDMABP3_STOP_STAT)	32	R/W	0000_0000h	7.2.7.3/ 1205
302B_000C	Channel Start (SDMABP3_DSTART)	32	R	0000_0000h	7.2.7.4/ 1206
302B_0028	DMA Request Error Register (SDMABP3_EVTERR)	32	R	0000_0000h	7.2.7.5/ 1206
302B_002C	Channel DSP Interrupt Mask (SDMABP3_INTRMASK)	32	R/W	0000_0000h	7.2.7.6/ 1207
302B_0034	DMA Request Error Register (SDMABP3_EVTERRDBG)	32	R	0000_0000h	7.2.7.7/ 1207
302C_0000	Channel 0 Pointer (SDMABP2_DC0PTR)	32	R/W	0000_0000h	7.2.7.1/ 1205
302C_0004	Channel Interrupts (SDMABP2_INTR)	32	w1c	0000_0000h	7.2.7.2/ 1205
302C_0008	Channel Stop/Channel Status (SDMABP2_STOP_STAT)	32	R/W	0000_0000h	7.2.7.3/ 1205
302C_000C	Channel Start (SDMABP2_DSTART)	32	R	0000_0000h	7.2.7.4/ 1206
302C_0028	DMA Request Error Register (SDMABP2_EVTERR)	32	R	0000_0000h	7.2.7.5/ 1206
302C_002C	Channel DSP Interrupt Mask (SDMABP2_INTRMASK)	32	R/W	0000_0000h	7.2.7.6/ 1207
302C_0034	DMA Request Error Register (SDMABP2_EVTERRDBG)	32	R	0000_0000h	7.2.7.7/ 1207
30BD_0000	Channel 0 Pointer (SDMABP1_DC0PTR)	32	R/W	0000_0000h	7.2.7.1/ 1205
30BD_0004	Channel Interrupts (SDMABP1_INTR)	32	w1c	0000_0000h	7.2.7.2/ 1205
30BD_0008	Channel Stop/Channel Status (SDMABP1_STOP_STAT)	32	R/W	0000_0000h	7.2.7.3/ 1205
30BD_000C	Channel Start (SDMABP1_DSTART)	32	R	0000_0000h	7.2.7.4/ 1206
30BD_0028	DMA Request Error Register (SDMABP1_EVTERR)	32	R	0000_0000h	7.2.7.5/ 1206
30BD_002C	Channel DSP Interrupt Mask (SDMABP1_INTRMASK)	32	R/W	0000_0000h	7.2.7.6/ 1207
30BD_0034	DMA Request Error Register (SDMABP1_EVTERRDBG)	32	R	0000_0000h	7.2.7.7/ 1207

7.2.7.1 Channel 0 Pointer (SDMABPx_DC0PTR)

Address: Base address + 0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DC0PTR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMABPx_DC0PTR field descriptions

Field	Description
DC0PTR	**Channel 0 Pointer** contains the 32-bit address, in BP memory, of the array of channel control blocks starting with the one for channel 0 (the control channel). This register should be initialized by the BP before it enables a channel (for example, channel 0). See the API document SDMA Scripts User Manual for the use of this register. The BP has a read/write access and the SDMA has a read-only access.

7.2.7.2 Channel Interrupts (SDMABPx_INTR)

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DI																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMABPx_INTR field descriptions

Field	Description
DI	The BP Interrupts register contains the 32 DI[i] bits. If any bit is set, it will cause an interrupt to the BP. • This register is a "write-ones" register to the BP. When the BP sets a bit in this register, the corresponding DI[i] bit is cleared. • The interrupt service routine should clear individual channel bits when their interrupts are serviced; failure to do so will cause continuous interrupts. • The SDMA is responsible for setting the DI[i] bit corresponding to the current channel when the corresponding `done` instruction is executed.

7.2.7.3 Channel Stop/Channel Status (SDMABPx_STOP_STAT)

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DE																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMABPx_STOP_STAT field descriptions

Field	Description
DE	This 32-bit register gives access to the BP (DSP) Enable bits, DE. There is one bit for every channel. • This register is a "write-ones" register to the BP. • When the BP writes 1 in bit *i* of this register, it clears the DE[i] and DSTART[i] bits. • Reading this register yields the current state of the DE[i] bits.

7.2.7.4 Channel Start (SDMABPx_DSTART)

Address: Base address + Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DSTART_DE																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMABPx_DSTART field descriptions

Field	Description
DSTART_DE	The DSTART_DE registers are 32 bits wide with one bit for every channel. • When a bit is written to 1, it enables the corresponding channel. • Two physical registers are accessed with that address (DSTART and DE), which enables the BP to trigger a channel a second time before the first trigger was processed. • This register is a "write-ones" register to the BP. Neither DSTART[i] bit can be set while the corresponding DE[i] bit is cleared. • When the BP tries to set the DSTART[i] bit by writing a one (if the corresponding DE[i] bit is clear), the bit in the DSTART[i] register will remain cleared and the DE[i] bit will be set. If the corresponding DE[i] bit was already set, the DSTART[i] bit will be set. • The next time the SDMA channel *i* attempts to clear the DE[i] bit by means of a `done` instruction, the bit in the DSTART[i] register will be cleared and the DE[i] bit will take the old value of the DSTART[i] bit. • Reading this register yields the current state of the DSTART[i] bits. This mechanism enables the BP to pipeline two DSTART commands per channel.

7.2.7.5 DMA Request Error Register (SDMABPx_EVTERR)

Address: Base address + 28h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHNERR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMABPx_EVTERR field descriptions

Field	Description
CHNERR	This register is used by the SDMA to warn the BP when an incoming DMA request was detected; it then triggers a channel that is already pending or being serviced, which may mean there is an overflow of data

SDMABPx_EVTERR field descriptions (continued)

Field	Description
	for that channel. An interrupt is sent to the BP if the corresponding channel bit is set in the INTRMASK register. • This is a "write-ones" register for the scheduler. It is only able to set the flags. The flags are cleared when the register is read by the BP or during an SDMA reset. • The CHNERR[i] bit is set when a DMA request that triggers channel *i* is received through the corresponding input pins and the EP[i] bit is already set. The EVTERR[i] bit is unaffected if the BP tries to set the EP[i] bit when that EP[i] bit is already set.

7.2.7.6 Channel DSP Interrupt Mask (SDMABPx_INTRMASK)

Address: Base address + 2Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DIMASK																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMABPx_INTRMASK field descriptions

Field	Description
DIMASK	The Interrupt Mask Register contains 32 interrupt generation mask bits. If bit DIMASK[i] is set, the DI[i] bit is set and an interrupt is sent to the BP when a DMA request error is detected on channel *i* (for example, EVTERR[i] is set).

7.2.7.7 DMA Request Error Register (SDMABPx_EVTERRDBG)

Address: Base address + 34h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHNERR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMABPx_EVTERRDBG field descriptions

Field	Description
CHNERR	This register is the same as EVTERR except reading it does not clear its contents. This address is meant to be used in debug mode. The BP OnCE may check this register value without modifying it.

7.2.8 SDMA Internal (Core) Memory Map and Internal Register Definitions

The actual SDMA memory mapped registers are summarized in the following sections; for peripherals' memory maps, refer to the respective chapters.

The following definitions serve as a key for the SDMA internal register summary.

SDMACORE memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302B_0000	Arm platform Channel 0 Pointer (SDMACORE3_MC0PTR)	32	R	0000_0000h	7.2.8.1/ 1211
302B_0008	Current Channel Pointer (SDMACORE3_CCPTR)	32	R	0000_0000h	7.2.8.2/ 1211
302B_000C	Current Channel Register (SDMACORE3_CCR)	32	R	0000_0000h	7.2.8.3/ 1212
302B_0010	Highest Pending Channel Register (SDMACORE3_NCR)	32	R	0000_0000h	7.2.8.4/ 1212
302B_0014	External DMA Requests Mirror (SDMACORE3_EVENTS)	32	R	0000_0000h	7.2.8.5/ 1213
302B_0018	Current Channel Priority (SDMACORE3_CCPRI)	32	R	0000_0000h	7.2.8.6/ 1214
302B_001C	Next Channel Priority (SDMACORE3_NCPRI)	32	R	0000_0000h	7.2.8.7/ 1214
302B_0020	OnCE Event Cell Counter (SDMACORE3_ECOUNT)	32	R/W	0000_0000h	7.2.8.8/ 1215
302B_0024	OnCE Event Cell Control Register (SDMACORE3_ECTL)	32	R/W	0000_0000h	7.2.8.9/ 1215
302B_0028	OnCE Event Address Register A (SDMACORE3_EAA)	32	R/W	0000_0000h	7.2.8.10/ 1217
302B_002C	OnCE Event Cell Address Register B (SDMACORE3_EAB)	32	R/W	0000_0000h	7.2.8.11/ 1217
302B_0030	OnCE Event Cell Address Mask (SDMACORE3_EAM)	32	R/W	0000_0000h	7.2.8.12/ 1217
302B_0034	OnCE Event Cell Data Register (SDMACORE3_ED)	32	R/W	0000_0000h	7.2.8.13/ 1218
302B_0038	OnCE Event Cell Data Mask (SDMACORE3_EDM)	32	R/W	0000_0000h	7.2.8.14/ 1218
302B_003C	OnCE Real-Time Buffer (SDMACORE3_RTB)	32	R/W	0000_0000h	7.2.8.15/ 1219
302B_0040	OnCE Trace Buffer (SDMACORE3_TB)	32	R	0000_0000h	7.2.8.16/ 1219

Table continues on the next page...

SDMACORE memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302B_0044	OnCE Status (SDMACORE3_OSTAT)	32	R	0000_0000h	7.2.8.17/ 1220
302B_0048	Channel 0 Boot Address (SDMACORE3_MCHN0ADDR)	32	R	0000_0000h	7.2.8.18/ 1222
302B_004C	ENDIAN Status Register (SDMACORE3_ENDIANNESS)	32	R	0000_0001h	7.2.8.19/ 1223
302B_0054	Lock Status Register (SDMACORE3_SDMA_LOCK)	32	R	0000_0000h	7.2.8.20/ 1224
302B_0058	External DMA Requests Mirror #2 (SDMACORE3_EVENTS2)	32	R	0000_0000h	7.2.8.21/ 1224
302C_0000	Arm platform Channel 0 Pointer (SDMACORE2_MC0PTR)	32	R	0000_0000h	7.2.8.1/ 1211
302C_0008	Current Channel Pointer (SDMACORE2_CCPTR)	32	R	0000_0000h	7.2.8.2/ 1211
302C_000C	Current Channel Register (SDMACORE2_CCR)	32	R	0000_0000h	7.2.8.3/ 1212
302C_0010	Highest Pending Channel Register (SDMACORE2_NCR)	32	R	0000_0000h	7.2.8.4/ 1212
302C_0014	External DMA Requests Mirror (SDMACORE2_EVENTS)	32	R	0000_0000h	7.2.8.5/ 1213
302C_0018	Current Channel Priority (SDMACORE2_CCPRI)	32	R	0000_0000h	7.2.8.6/ 1214
302C_001C	Next Channel Priority (SDMACORE2_NCPRI)	32	R	0000_0000h	7.2.8.7/ 1214
302C_0020	OnCE Event Cell Counter (SDMACORE2_ECOUNT)	32	R/W	0000_0000h	7.2.8.8/ 1215
302C_0024	OnCE Event Cell Control Register (SDMACORE2_ECTL)	32	R/W	0000_0000h	7.2.8.9/ 1215
302C_0028	OnCE Event Address Register A (SDMACORE2_EAA)	32	R/W	0000_0000h	7.2.8.10/ 1217
302C_002C	OnCE Event Cell Address Register B (SDMACORE2_EAB)	32	R/W	0000_0000h	7.2.8.11/ 1217
302C_0030	OnCE Event Cell Address Mask (SDMACORE2_EAM)	32	R/W	0000_0000h	7.2.8.12/ 1217
302C_0034	OnCE Event Cell Data Register (SDMACORE2_ED)	32	R/W	0000_0000h	7.2.8.13/ 1218
302C_0038	OnCE Event Cell Data Mask (SDMACORE2_EDM)	32	R/W	0000_0000h	7.2.8.14/ 1218
302C_003C	OnCE Real-Time Buffer (SDMACORE2_RTB)	32	R/W	0000_0000h	7.2.8.15/ 1219
302C_0040	OnCE Trace Buffer (SDMACORE2_TB)	32	R	0000_0000h	7.2.8.16/ 1219
302C_0044	OnCE Status (SDMACORE2_OSTAT)	32	R	0000_0000h	7.2.8.17/ 1220

Table continues on the next page...

SDMACORE memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302C_0048	Channel 0 Boot Address (SDMACORE2_MCHN0ADDR)	32	R	0000_0000h	7.2.8.18/ 1222
302C_004C	ENDIAN Status Register (SDMACORE2_ENDIANNESS)	32	R	0000_0001h	7.2.8.19/ 1223
302C_0054	Lock Status Register (SDMACORE2_SDMA_LOCK)	32	R	0000_0000h	7.2.8.20/ 1224
302C_0058	External DMA Requests Mirror #2 (SDMACORE2_EVENTS2)	32	R	0000_0000h	7.2.8.21/ 1224
30BD_0000	Arm platform Channel 0 Pointer (SDMACORE1_MC0PTR)	32	R	0000_0000h	7.2.8.1/ 1211
30BD_0008	Current Channel Pointer (SDMACORE1_CCPTR)	32	R	0000_0000h	7.2.8.2/ 1211
30BD_000C	Current Channel Register (SDMACORE1_CCR)	32	R	0000_0000h	7.2.8.3/ 1212
30BD_0010	Highest Pending Channel Register (SDMACORE1_NCR)	32	R	0000_0000h	7.2.8.4/ 1212
30BD_0014	External DMA Requests Mirror (SDMACORE1_EVENTS)	32	R	0000_0000h	7.2.8.5/ 1213
30BD_0018	Current Channel Priority (SDMACORE1_CCPRI)	32	R	0000_0000h	7.2.8.6/ 1214
30BD_001C	Next Channel Priority (SDMACORE1_NCPRI)	32	R	0000_0000h	7.2.8.7/ 1214
30BD_0020	OnCE Event Cell Counter (SDMACORE1_ECOUNT)	32	R/W	0000_0000h	7.2.8.8/ 1215
30BD_0024	OnCE Event Cell Control Register (SDMACORE1_ECTL)	32	R/W	0000_0000h	7.2.8.9/ 1215
30BD_0028	OnCE Event Address Register A (SDMACORE1_EAA)	32	R/W	0000_0000h	7.2.8.10/ 1217
30BD_002C	OnCE Event Cell Address Register B (SDMACORE1_EAB)	32	R/W	0000_0000h	7.2.8.11/ 1217
30BD_0030	OnCE Event Cell Address Mask (SDMACORE1_EAM)	32	R/W	0000_0000h	7.2.8.12/ 1217
30BD_0034	OnCE Event Cell Data Register (SDMACORE1_ED)	32	R/W	0000_0000h	7.2.8.13/ 1218
30BD_0038	OnCE Event Cell Data Mask (SDMACORE1_EDM)	32	R/W	0000_0000h	7.2.8.14/ 1218
30BD_003C	OnCE Real-Time Buffer (SDMACORE1_RTB)	32	R/W	0000_0000h	7.2.8.15/ 1219
30BD_0040	OnCE Trace Buffer (SDMACORE1_TB)	32	R	0000_0000h	7.2.8.16/ 1219
30BD_0044	OnCE Status (SDMACORE1_OSTAT)	32	R	0000_0000h	7.2.8.17/ 1220
30BD_0048	Channel 0 Boot Address (SDMACORE1_MCHN0ADDR)	32	R	0000_0000h	7.2.8.18/ 1222

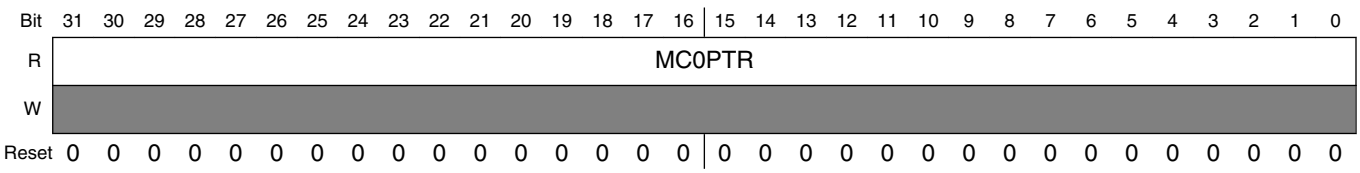
Table continues on the next page...

SDMACORE memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BD_004C	ENDIAN Status Register (SDMACORE1_ENDIANNESS)	32	R	0000_0001h	7.2.8.19/ 1223
30BD_0054	Lock Status Register (SDMACORE1_SDMA_LOCK)	32	R	0000_0000h	7.2.8.20/ 1224
30BD_0058	External DMA Requests Mirror #2 (SDMACORE1_EVENTS2)	32	R	0000_0000h	7.2.8.21/ 1224

7.2.8.1 Arm platform Channel 0 Pointer (SDMACOREx_MC0PTR)

Address: Base address + 0h offset

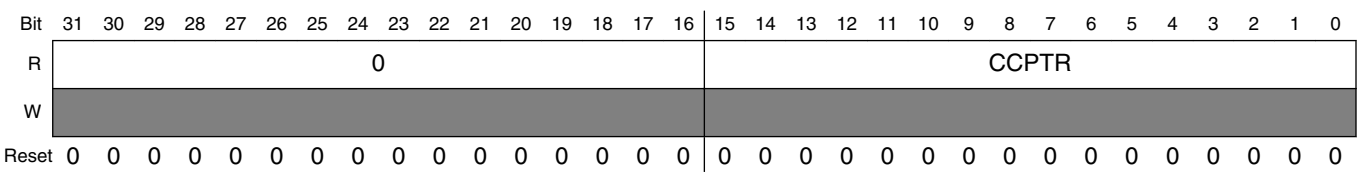


SDMACOREx_MC0PTR field descriptions

Field	Description
MC0PTR	Contains the address-in the Arm platform memory space-of the initial SDMA context and scripts that are loaded by the SDMA boot script running on channel 0.

7.2.8.2 Current Channel Pointer (SDMACOREx_CCPTR)

Address: Base address + 8h offset



SDMACOREx_CCPTR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
CCPTR	Contains the start address of the context data for the current channel: Its value is *CONTEXT_BASE* + 24* *CCR* or *CONTEXT_BASE* + 32* *CCR* where CONTEXT_BASE = 0x0800. The value 24 or 32 is selected according to the programmed channel scratch RAM size in the register shown in Channel 0 Boot Address (SDMAARM_CHN0ADDR) .

7.2.8.3 Current Channel Register (SDMACOREx_CCR)

Address: Base address + Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																											CCR				
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMACOREx_CCR field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
CCR	Contains the number of the current running channel whose context is installed. In the case that the SDMA has finished running the channel and has entered sleep state, CCR will indicate the previous running channel. The PST bits in the OSTAT register indicate when the SDMA is in sleep state.

7.2.8.4 Highest Pending Channel Register (SDMACOREx_NCR)

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																											NCR				
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMACOREx_NCR field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
NCR	Contains the number of the pending channel that the scheduler has selected to run next.

7.2.8.5 External DMA Requests Mirror (SDMACOREx_EVENTS)

NOTE

This register is very useful in the case of DMA requests that are active when a peripheral FIFO level is above the programmed watermark. The activation of the DMA request (rising edge) is detected by the SDMA logic and it can enable one or several channels. One of the channels accesses the peripheral and reads or writes a number of data that matches the watermark level (for example, if the watermark is four words, the channel reads or writes four words).

If the channel is effectively executed long after the DMA request was received, reading or writing the watermark number of data may not be sufficient to reset the DMA request (for example, if the FIFO watermark is four and at the channel execution it already contains nine pieces of data). This means no new rising edge may be detected by the SDMA, although there still remains transfers to perform. Therefore, if the channel were terminated at that time, it would not be restarted, causing potential overrun or underrun of the peripheral.

The proposed mechanism is for the channel to check this register after it has performed the "watermark" number of accesses to the peripheral. If the bit for the DMA request that triggers this channel is set, it means there is still another watermark number of data to transfer. This goes on until the bit is cleared. The same script can be used for multiple channels that require this behavior. The script can determine its channel number from the CCR register and infer the corresponding DMA request bit to check. It needs a reference table that is coherent with the request-channel matrix that the Arm platform programmed.

Address: Base address + 14h offset

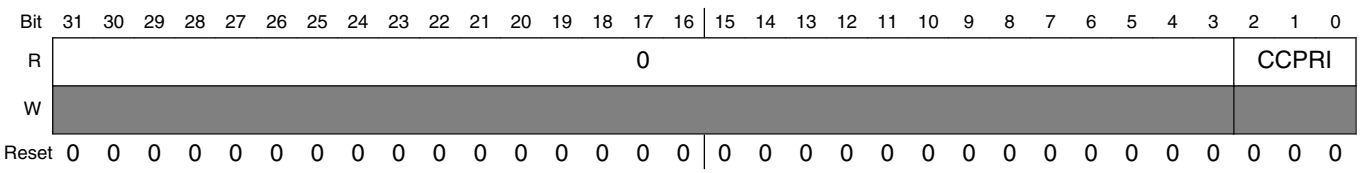
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EVENTS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMACOREx_EVENTS field descriptions

Field	Description
EVENTS	Reflects the status of the SDMA's external DMA requests. It is meant to allow any channel to monitor the states of these SDMA inputs. This register displays EVENTS 0-31. The EVENTS2 register displays events 32-47.

7.2.8.6 Current Channel Priority (SDMACOREx_CCPRI)

Address: Base address + 18h offset

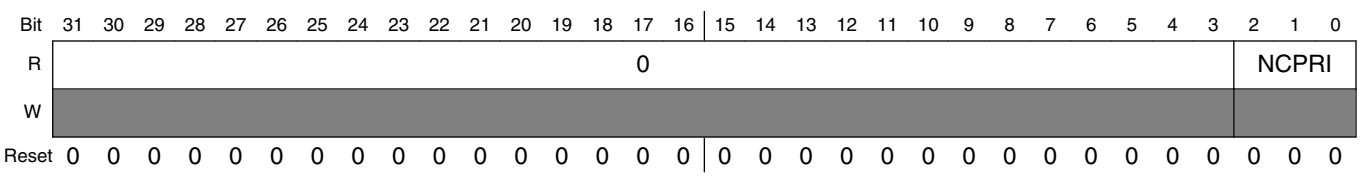


SDMACOREx_CCPRI field descriptions

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
CCPRI	Contains the 3-bit priority of the channel whose context is installed. It is 0 when no channel is running. **NOTE:** 1-7 current channel priority 0 no running channel

7.2.8.7 Next Channel Priority (SDMACOREx_NCPRI)

Address: Base address + 1Ch offset

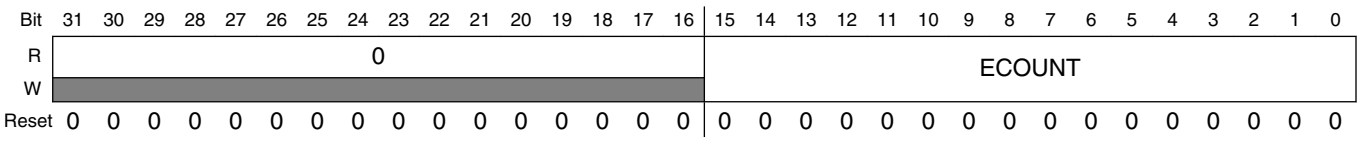


SDMACOREx_NCPRI field descriptions

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
NCPRI	Contains the 3-bit priority of the channel the scheduler has selected to run next. It is 0 when no other channel is pending.

7.2.8.8 OnCE Event Cell Counter (SDMACOREx_ECOUNT)

Address: Base address + 20h offset

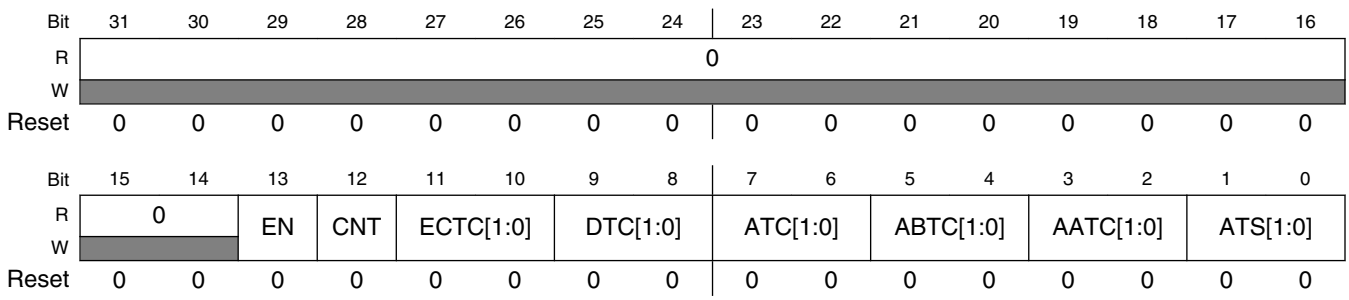


SDMACOREx_ECOUNT field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
ECOUNT	The event cell counter contains the number of times minus one that an event detection must occur before generating a debug request. • This register should be written before any attempt to use the event detection counter during an event detection process. • The counter is cleared on a JTAG reset.

7.2.8.9 OnCE Event Cell Control Register (SDMACOREx_ECTL)

Address: Base address + 24h offset



SDMACOREx_ECTL field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
13 EN	Event Cell Enable. If the EN bit is set, the event cell is allowed to generate debug requests (the cell is awakened). If it is cleared, the event detection unit is disabled and no hardware breakpoint is generated, but matching conditions are still reflected on the emulation pin. 0 Cell is disabled. 1 Cell is enabled.
12 CNT	Event Counter Enable. The event counter enable bit determines if the cell counter is used during the event detection. In order to use the event counter during an event detection process, the event cell counter register should be loaded with a value equal to the number of times minus one that an event occurs before a debug request is sent. After every event detection, the counter is decreased. When the counter reaches

Table continues on the next page...

SDMACOREx_ECTL field descriptions (continued)

Field	Description
	the value 0, the event detection cell sends a debug request to the core. The event counter register should be written and the EN bit should be set before each new event detection process uses the event counter. 0 Counter is disabled. 1 Counter is enabled.
11–10 ECTC[1:0]	The event cell trigger condition bits select the combination of address and data matching conditions that generate the final address/data condition. During program execution, if this event cell trigger condition goes to 1, a debug request is sent to the SDMA. The EN bit must be set to enable the debug request generation. 00 address ONLY 01 data ONLY 10 address AND data 11 address OR data
9–8 DTC[1:0]	The data trigger condition bits define when data is considered matching after comparison with the data register of the event detection unit. The operations are performed on unsigned values. 00 equal 01 not equal 10 greater than 11 less than
7–6 ATC[1:0]	The address trigger condition bits select how the two address conditions (addressA and addressB) are combined to define the global address matching condition. The supported combinations are described, as follows. 00 addressA ONLY 01 addrA AND addrB 10 addrA OR addrB 11 reserved
5–4 ABTC[1:0]	The Address B Trigger Condition (ABTC) controls the operations performed by address comparator B. All operations are performed on unsigned values. This comparator B outputs the addressB condition. 00 equal 01 not equal 10 greater than 11 less than
3–2 AATC[1:0]	The Address A Trigger Condition (AATC) controls the operations performed by address comparator A. All operations are performed on unsigned values. This comparator A outputs the addressA condition. 00 equal 01 not equal 10 greater than 11 less than
ATS[1:0]	The access type select bits define the memory access type required on the SDMA memory bus. 00 read ONLY 01 write ONLY 10 read or write 11 -

7.2.8.10 OnCE Event Address Register A (SDMACORE*x*_EAA)

Address: Base address + 28h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																EAA															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMACORE*x*_EAA field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
EAA	Event Cell Address Register A computes an address A condition. It is cleared on a JTAG reset.

7.2.8.11 OnCE Event Cell Address Register B (SDMACORE*x*_EAB)

Address: Base address + 2Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																EAB															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMACORE*x*_EAB field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
EAB	Event Cell Address Register B computes an address B condition. It is cleared on a JTAG reset.

7.2.8.12 OnCE Event Cell Address Mask (SDMACORE*x*_EAM)

Address: Base address + 30h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																EAM															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMACORE*x*_EAM field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

SDMACORE*x*_EAM field descriptions (continued)

Field	Description
EAM	The Event Cell Address Mask contains a user-defined address mask value. This mask is applied to the address value latched from the memory address bus before performing the address comparison. **NOTE:** There is a common address mask value for both address comparators. If bit *i* of this register is set, then bit *i* of the address value latched from the memory bus does not influence the result of the address comparison. The register is cleared on a JTAG reset.

7.2.8.13 OnCE Event Cell Data Register (SDMACORE*x*_ED)

Address: Base address + 34h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ED																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMACORE*x*_ED field descriptions

Field	Description
ED	The event cell data register contains a user defined data value. This data value is an input for the data comparator which generates the data condition. It is cleared on a JTAG reset.

7.2.8.14 OnCE Event Cell Data Mask (SDMACORE*x*_EDM)

Address: Base address + 38h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EDM																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMACORE*x*_EDM field descriptions

Field	Description
EDM	The event cell data mask register contains the user-defined data mask value. • This mask is applied to the data value latched from the memory bus before performing the data comparison. • Setting bit *i* of the event cell data mask register means that bit *i* of the data value latched from the address bus does not influence the result of the data comparison. • The data mask is cleared on a JTAG reset.

7.2.8.15 OnCE Real-Time Buffer (SDMACOREx_RTB)

Address: Base address + 3Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RTB																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMACOREx_RTB field descriptions

Field	Description
RTB	The Real Time Buffer register stores and retrieves run time information without putting the SDMA in debug mode. Writing to that register triggers a pulse on a specific real-time debug pin whose connection depends on the chip implementation. The RTB value can be accessed by the OnCE under Arm platform or JTAG control using the rbuffer command.

7.2.8.16 OnCE Trace Buffer (SDMACOREx_TB)

Address: Base address + 40h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			TBF	TADDR											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TADDR		CHFADDR													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMACOREx_TB field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value 0.
28 TBF	The Trace Buffer Flag is set when the buffer contains the addresses of a valid change of flow. The contents of the buffer should be ignored otherwise. 0 Invalid information 1 Valid information
27–14 TADDR	The target address is the address taken after the execution of the change of flow instruction.
CHFADDR	The change of flow address is the address where the change of flow is taken when executing a change of flow instruction.

7.2.8.17 OnCE Status (SDMACOREx_OSTAT)

Address: Base address + 44h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PST[3:0]				RCV	EDR	ODR	SWB	MST	0				ECDR[2:0]		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMACOREx_OSTAT field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–12 PST[3:0]	The Processor Status bits reflect the state of the SDMA RISC engine. • The "Program" state is the usual instruction execution cycle. • The "Data" state is inserted when there are wait-states during a load or a store on the data bus (ld or st). • The "Change of Flow" state is the second cycle of any instruction that breaks the sequence of instructions (jumps and channel-switching instructions). • The "Change of Flow in Loop" state is used when an error causes a hardware loop exit. • The "Debug" state means the SDMA is in debug mode. • The "Functional Unit" state is inserted when there are wait-states during a load or a store on the functional units bus (ldf or stf). • In "Sleep" modes, no script is running (this is the RISC engine idle state). The "after Reset" is slightly different because no context restoring phase will happen when a channel is triggered: The script located at address 0 will be executed (boot operation). • The "in Sleep" states are the same as above except they do not have any corresponding channel. They are used when entering debug mode after reset; the reason is that it is necessary to return to the "Sleep after Reset" state when leaving debug mode. 0 Program 1 Data 2 Change of Flow 3 Change of Flow in Loop 4 Debug 5 Functional Unit 6 Sleep 7 Save 8 Program in Sleep 9 Data in Sleep 10 Change of Flow in Sleep 11 Change Flow Loop Sleep 12 Debug in Sleep 13 Functional Unit in Sleep

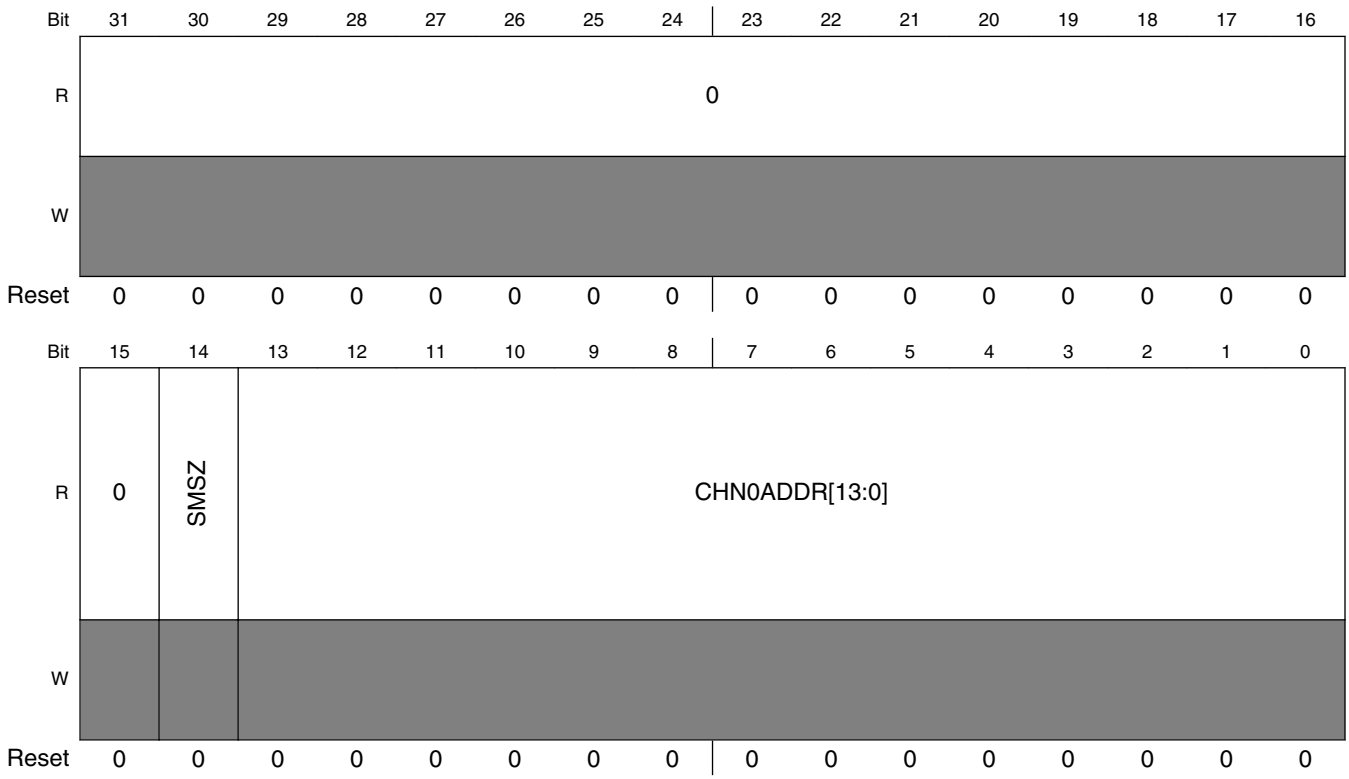
Table continues on the next page...

SDMACOREx_OSTAT field descriptions (continued)

Field	Description
	14 Sleep after Reset 15 Restore
11 RCV	After each write access to the real time buffer (RTB), the RCV bit is set. This bit is cleared after execution of an `rbuffer` command and on a JTAG reset.
10 EDR	This flag is raised when the SDMA has entered debug mode after an external debug request.
9 ODR	This flag is raised when the SDMA has entered debug mode after a OnCE debug request.
8 SWB	This flag is raised when the SDMA has entered debug mode after a software breakpoint.
7 MST	This flag is raised when the OnCE is controlled from the Arm platform peripheral interface. 0 JTAG interface controls the OnCE. 1 Arm platform peripheral interface controls the OnCE.
6–3 Reserved	This read-only field is reserved and always has the value 0.
ECDR[2:0]	Event Cell Debug Request. If the debug request comes from the event cell, the reason for entering debug mode is given by the EDR bits. The encoding of the EDR bits is useful to find out more precisely why the debug request was generated. A debug request from an event cell is generated for a specific combination of the addressA, addressB, and data conditions; the value of those fields is given by the EDR bits. If all three bits of the EDR are reset, then it did not generate any debug request. If the cell did generate a debug request, then at least one EDR bit is set; the meaning of the encoding is as follows: 0 1 matched addressA condition 1 1 matched addressB condition 2 1 matched data condition

7.2.8.18 Channel 0 Boot Address (SDMACOREx_MCHN0ADDR)

Address: Base address + 48h offset

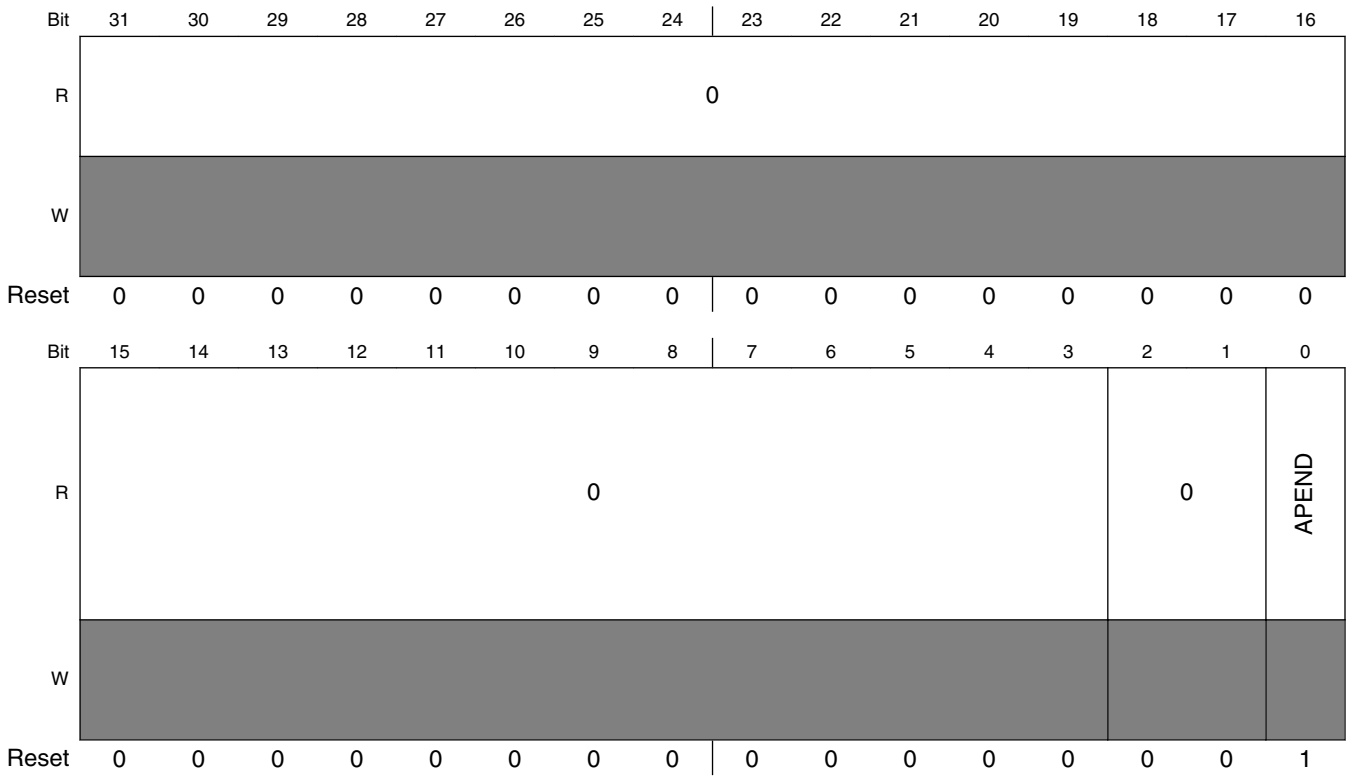


SDMACOREx_MCHN0ADDR field descriptions

Field	Description
31–15 Reserved	This read-only field is reserved and always has the value 0.
14 SMSZ	The bit 14 (Scratch Memory Size) determines if scratch memory must be available after every channel context. After reset, it is equal to 0, which defines a RAM space of 24 words for each channel. All of this area stores the channel context. By setting this bit, 32 words are reserved for every channel context, which gives eight additional words that can be used by the channel script to store any type of data. Those words are never erased by the context switching mechanism. 0 24 words per context 1 32 words per context
CHN0ADDR[13:0]	Contains the address of the channel 0 routine programmed by the Arm platform; it is loaded into a general register at the very start of the boot and the SDMA jumps to the address it contains. By default, it points to the standard boot routine in ROM.

7.2.8.19 ENDIAN Status Register (SDMACOREx_ENDIANNESS)

Address: Base address + 4Ch offset

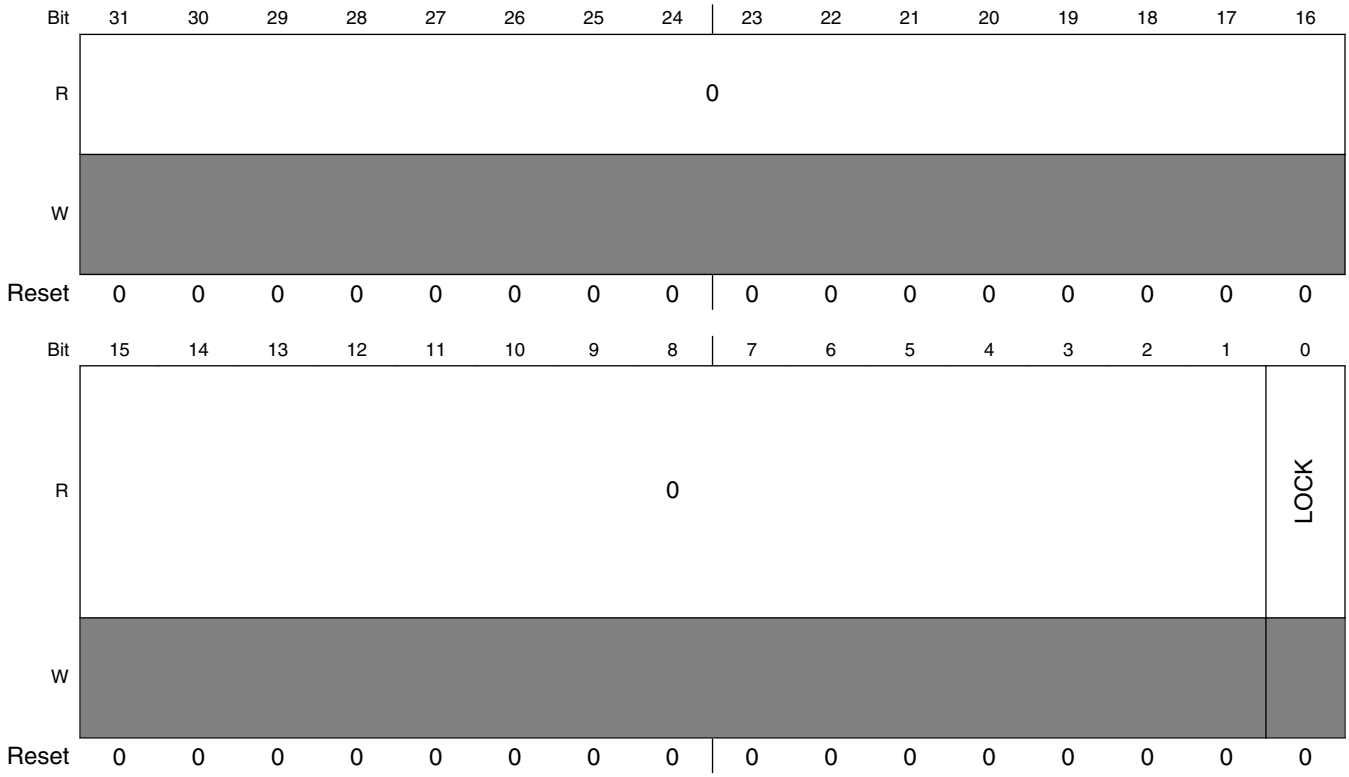


SDMACOREx_ENDIANNESS field descriptions

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 APEND	APEND indicates the endian mode of the Peripheral and Burst DMA interfaces. This bit is tied to logic '1' indicating little-endian mode. 0 - Arm platform is in big-endian mode 1 - Arm platform is in little-endian mode

7.2.8.20 Lock Status Register (SDMACOREx_SDMA_LOCK)

Address: Base address + 54h offset

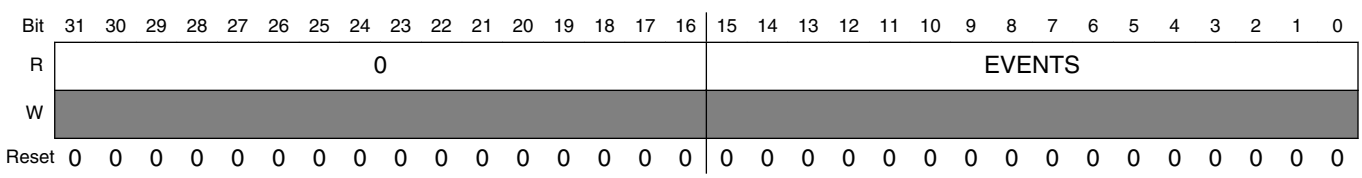


SDMACOREx_SDMA_LOCK field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 LOCK	The LOCK bit reports the value of the LOCK bit in the SDMA_LOCK status register. SDMA software may use this value to determine if certain operations such as loading of new scripts is allowed. 0 - LOCK bit clear 1 - LOCK bit set

7.2.8.21 External DMA Requests Mirror #2 (SDMACOREx_EVENTS2)

Address: Base address + 58h offset



SDMACOREx_EVENTS2 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
EVENTS	Reflects the status of the SDMA's external DMA requests. It is meant to allow any channel to monitor the states of these SDMA inputs. This register displays EVENTS 32-47. The separate EVENTS register displays events 0-31.

7.2.9 SDMA Peripheral Registers

Refer to the respective peripherals' chapters for more information.

Chapter 8
Chip IO and Pinmux

8.1 External Signals and Pin Multiplexing

8.1.1 Overview

The chip contains a limited number of pins, most of which have multiple signal options. These signal-to-pin and pin-to-signal options are selected by the input-output multiplexer called IOMUX. The IOMUX is also used to configure other pin characteristics, such as voltage level, drive strength, and hysteresis.

The muxing options table lists the external signals grouped by the module instance, the muxing options for each signal, and the registers used to route the signal to the chosen pad.

8.1.1.1 Muxing Options

Table 8-1. Muxing Options

Instance	Port	PAD	MODE
ARM PLATFORM	ARM_EVENTI	SAI1_TXC	ALT4
	ARM_EVENTO	SAI1_TXFS	ALT4
CCM	CCM_CLKO1	GPIO1_IO14	ALT6
	CCM_CLKO2	GPIO1_IO15	ALT6
	CCM_ENET_PHY_REF_CLK_ROOT	GPIO1_IO00	ALT1
	CCM_EXT_CLK1	GPIO1_IO00	ALT6
	CCM_EXT_CLK2	GPIO1_IO01	ALT6
	CCM_EXT_CLK3	GPIO1_IO06	ALT6
	CCM_EXT_CLK4	GPIO1_IO07	ALT6
	CCM_PMIC_READY	GPIO1_IO05	ALT5
	CCM_PMIC_READY	GPIO1_IO11	ALT5

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	CCM_PMIC_STBY_REQ	PMIC_STBY_REQ	ALT0
DRAM	DRAM_DQS0_P	DRAM_DQS0_P	No muxing
	DRAM_DQS0_N	DRAM_DQS0_N	No muxing
	DRAM_DM0	DRAM_DM0	No muxing
	DRAM_DQ00	DRAM_DQ00	No muxing
	DRAM_DQ01	DRAM_DQ01	No muxing
	DRAM_DQ02	DRAM_DQ02	No muxing
	DRAM_DQ03	DRAM_DQ03	No muxing
	DRAM_DQ04	DRAM_DQ04	No muxing
	DRAM_DQ05	DRAM_DQ05	No muxing
	DRAM_DQ06	DRAM_DQ06	No muxing
	DRAM_DQ07	DRAM_DQ07	No muxing
	DRAM_DQS1_P	DRAM_DQS1_P	No muxing
	DRAM_DQS1_N	DRAM_DQS1_N	No muxing
	DRAM_DM1	DRAM_DM1	No muxing
	DRAM_DQ08	DRAM_DQ08	No muxing
	DRAM_DQ09	DRAM_DQ09	No muxing
	DRAM_DQ10	DRAM_DQ10	No muxing
	DRAM_DQ11	DRAM_DQ11	No muxing
	DRAM_DQ12	DRAM_DQ12	No muxing
	DRAM_DQ13	DRAM_DQ13	No muxing
	DRAM_DQ14	DRAM_DQ14	No muxing
	DRAM_DQ15	DRAM_DQ15	No muxing
	DRAM_DQS2_P	DRAM_DQS2_P	No muxing
	DRAM_DQS2_N	DRAM_DQS2_N	No muxing
	DRAM_DM2	DRAM_DM2	No muxing
	DRAM_DQ16	DRAM_DQ16	No muxing
	DRAM_DQ17	DRAM_DQ17	No muxing
	DRAM_DQ18	DRAM_DQ18	No muxing
	DRAM_DQ19	DRAM_DQ19	No muxing
	DRAM_DQ20	DRAM_DQ20	No muxing
	DRAM_DQ21	DRAM_DQ21	No muxing
	DRAM_DQ22	DRAM_DQ22	No muxing
	DRAM_DQ23	DRAM_DQ23	No muxing
	DRAM_DQS3_P	DRAM_DQS3_P	No muxing
	DRAM_DQS3_N	DRAM_DQS3_N	No muxing
	DRAM_DM3	DRAM_DM3	No muxing
	DRAM_DQ24	DRAM_DQ24	No muxing
	DRAM_DQ25	DRAM_DQ25	No muxing

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	DRAM_DQ26	DRAM_DQ26	No muxing
	DRAM_DQ27	DRAM_DQ27	No muxing
	DRAM_DQ28	DRAM_DQ28	No muxing
	DRAM_DQ29	DRAM_DQ29	No muxing
	DRAM_DQ30	DRAM_DQ30	No muxing
	DRAM_DQ31	DRAM_DQ31	No muxing
	DRAM_RESET_N	DRAM_RESET_N	No muxing
	DRAM_ALERT_N	DRAM_ALERT_N	No muxing
	DRAM_AC00	DRAM_AC00	No muxing
	DRAM_AC01	DRAM_AC01	No muxing
	DRAM_AC02	DRAM_AC02	No muxing
	DRAM_AC03	DRAM_AC03	No muxing
	DRAM_AC04	DRAM_AC04	No muxing
	DRAM_AC05	DRAM_AC05	No muxing
	DRAM_AC06	DRAM_AC06	No muxing
	DRAM_AC07	DRAM_AC07	No muxing
	DRAM_AC08	DRAM_AC08	No muxing
	DRAM_AC09	DRAM_AC09	No muxing
	DRAM_AC10	DRAM_AC10	No muxing
	DRAM_AC11	DRAM_AC11	No muxing
	DRAM_AC12	DRAM_AC12	No muxing
	DRAM_AC13	DRAM_AC13	No muxing
	DRAM_AC14	DRAM_AC14	No muxing
	DRAM_AC15	DRAM_AC15	No muxing
	DRAM_AC16	DRAM_AC16	No muxing
	DRAM_AC17	DRAM_AC17	No muxing
	DRAM_AC19	DRAM_AC19	No muxing
	DRAM_AC20	DRAM_AC20	No muxing
	DRAM_AC21	DRAM_AC21	No muxing
	DRAM_AC22	DRAM_AC22	No muxing
	DRAM_AC23	DRAM_AC23	No muxing
	DRAM_AC24	DRAM_AC24	No muxing
	DRAM_AC25	DRAM_AC25	No muxing
	DRAM_AC26	DRAM_AC26	No muxing
	DRAM_AC27	DRAM_AC27	No muxing
	DRAM_AC28	DRAM_AC28	No muxing
	DRAM_AC29	DRAM_AC29	No muxing
	DRAM_AC30	DRAM_AC30	No muxing
	DRAM_AC31	DRAM_AC31	No muxing

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	DRAM_AC32	DRAM_AC32	No muxing
	DRAM_AC33	DRAM_AC33	No muxing
	DRAM_AC34	DRAM_AC34	No muxing
	DRAM_AC35	DRAM_AC35	No muxing
	DRAM_AC36	DRAM_AC36	No muxing
	DRAM_AC37	DRAM_AC37	No muxing
	DRAM_AC38	DRAM_AC38	No muxing
	DRAM_ZN	DRAM_ZN	No muxing
	DRAM_VREF	DRAM_VREF	No muxing
ECSPI1	ECSPI1_MISO	ECSPI1_MISO	ALT0
	ECSPI1_MOSI	ECSPI1_MOSI	ALT0
	ECSPI1_SCLK	ECSPI1_SCLK	ALT0
	ECSPI1_SS0	ECSPI1_SS0	ALT0
ECSPI2	ECSPI2_MISO	ECSPI2_MISO	ALT0
	ECSPI2_MOSI	ECSPI2_MOSI	ALT0
	ECSPI2_SCLK	ECSPI2_SCLK	ALT0
	ECSPI2_SS0	ECSPI2_SS0	ALT0
ECSPI3	ECSPI3_MISO	UART2_RXD	ALT1
	ECSPI3_MOSI	UART1_TXD	ALT1
	ECSPI3_SCLK	UART1_RXD	ALT1
	ECSPI3_SS0	UART2_TXD	ALT1
ENET11	ENET1_1588_EVENT0_IN	GPIO1_IO08	ALT1
	ENET1_1588_EVENT0_OUT	GPIO1_IO09	ALT1
	ENET1_1588_EVENT1_IN	I2C2_SCL	ALT1
	ENET1_1588_EVENT1_OUT	I2C2_SDA	ALT1
	ENET1_MDC	ENET_MDC	ALT0
	ENET1_MDC	GPIO1_IO06	ALT1
	ENET1_MDC	I2C1_SCL	ALT1
	ENET1_MDIO	ENET_MDIO	ALT0
	ENET1_MDIO	GPIO1_IO07	ALT1
	ENET1_MDIO	I2C1_SDA	ALT1
	ENET1_RGMII_RD0	ENET_RD0	ALT0
	ENET1_RGMII_RD1	ENET_RD1	ALT0
	ENET1_RGMII_RD2	ENET_RD2	ALT0
	ENET1_RGMII_RD3	ENET_RD3	ALT0
	ENET1_RGMII_RX_CTL	ENET_RX_CTL	ALT0
	ENET1_RGMII_RXC	ENET_RXC	ALT0
	ENET1_RGMII_TD0	ENET_TD0	ALT0
	ENET1_RGMII_TD1	ENET_TD1	ALT0

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	ENET1_RGMII_TD2	ENET_TD2	ALT0
	ENET1_RGMII_TD3	ENET_TD3	ALT0
	ENET1_RGMII_TX_CTL	ENET_TX_CTL	ALT0
	ENET1_RGMII_TXC	ENET_TXC	ALT0
	ENET1_RX_ER	ENET_RXC	ALT1
	ENET1_TX_ER	ENET_TXC	ALT1
	ENET1_TX_CLK	ENET_TD2	ALT1
GPIO1	GPIO1_IO00	GPIO1_IO00	ALT0
	GPIO1_IO01	GPIO1_IO01	ALT0
	GPIO1_IO10	GPIO1_IO10	ALT0
	GPIO1_IO11	GPIO1_IO11	ALT0
	GPIO1_IO12	GPIO1_IO12	ALT0
	GPIO1_IO13	GPIO1_IO13	ALT0
	GPIO1_IO14	GPIO1_IO14	ALT0
	GPIO1_IO15	GPIO1_IO15	ALT0
	GPIO1_IO16	ENET_MDC	ALT5
	GPIO1_IO17	ENET_MDIO	ALT5
	GPIO1_IO18	ENET_TD3	ALT5
	GPIO1_IO19	ENET_TD2	ALT5
	GPIO1_IO02	GPIO1_IO02	ALT0
	GPIO1_IO20	ENET_TD1	ALT5
	GPIO1_IO21	ENET_TD0	ALT5
	GPIO1_IO22	ENET_TX_CTL	ALT5
	GPIO1_IO23	ENET_TXC	ALT5
	GPIO1_IO24	ENET_RX_CTL	ALT5
	GPIO1_IO25	ENET_RXC	ALT5
	GPIO1_IO26	ENET_RD0	ALT5
	GPIO1_IO27	ENET_RD1	ALT5
	GPIO1_IO28	ENET_RD2	ALT5
	GPIO1_IO29	ENET_RD3	ALT5
	GPIO1_IO03	GPIO1_IO03	ALT0
	GPIO1_IO04	GPIO1_IO04	ALT0
	GPIO1_IO05	GPIO1_IO05	ALT0
	GPIO1_IO06	GPIO1_IO06	ALT0
	GPIO1_IO07	GPIO1_IO07	ALT0
	GPIO1_IO08	GPIO1_IO08	ALT0
	GPIO1_IO09	GPIO1_IO09	ALT0
GPIO2	GPIO2_IO00	SD1_CLK	ALT5
	GPIO2_IO01	SD1_CMD	ALT5

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	GPIO2_IO10	SD1_RESET_B	ALT5
	GPIO2_IO11	SD1_STROBE	ALT5
	GPIO2_IO12	SD2_CD_B	ALT5
	GPIO2_IO13	SD2_CLK	ALT5
	GPIO2_IO14	SD2_CMD	ALT5
	GPIO2_IO15	SD2_DATA0	ALT5
	GPIO2_IO16	SD2_DATA1	ALT5
	GPIO2_IO17	SD2_DATA2	ALT5
	GPIO2_IO18	SD2_DATA3	ALT5
	GPIO2_IO19	SD2_RESET_B	ALT5
	GPIO2_IO02	SD1_DATA0	ALT5
	GPIO2_IO20	SD2_WP	ALT5
	GPIO2_IO03	SD1_DATA1	ALT5
	GPIO2_IO04	SD1_DATA2	ALT5
	GPIO2_IO05	SD1_DATA3	ALT5
	GPIO2_IO06	SD1_DATA4	ALT5
	GPIO2_IO07	SD1_DATA5	ALT5
	GPIO2_IO08	SD1_DATA6	ALT5
	GPIO2_IO09	SD1_DATA7	ALT5
GPIO3	GPIO3_IO00	NAND_ALE	ALT5
	GPIO3_IO01	NAND_CE0_B	ALT5
	GPIO3_IO10	NAND_DATA04	ALT5
	GPIO3_IO11	NAND_DATA05	ALT5
	GPIO3_IO12	NAND_DATA06	ALT5
	GPIO3_IO13	NAND_DATA07	ALT5
	GPIO3_IO14	NAND_DQS	ALT5
	GPIO3_IO15	NAND_RE_B	ALT5
	GPIO3_IO16	NAND_READY_B	ALT5
	GPIO3_IO17	NAND_WE_B	ALT5
	GPIO3_IO18	NAND_WP_B	ALT5
	GPIO3_IO19	SAI5_RXFS	ALT5
	GPIO3_IO02	NAND_CE1_B	ALT5
	GPIO3_IO20	SAI5_RXC	ALT5
	GPIO3_IO21	SAI5_RXD0	ALT5
	GPIO3_IO22	SAI5_RXD1	ALT5
	GPIO3_IO23	SAI5_RXD2	ALT5
	GPIO3_IO24	SAI5_RXD3	ALT5
	GPIO3_IO25	SAI5_MCLK	ALT5
	GPIO3_IO03	NAND_CE2_B	ALT5

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	GPIO3_IO04	NAND_CE3_B	ALT5
	GPIO3_IO05	NAND_CLE	ALT5
	GPIO3_IO06	NAND_DATA00	ALT5
	GPIO3_IO07	NAND_DATA01	ALT5
	GPIO3_IO08	NAND_DATA02	ALT5
	GPIO3_IO09	NAND_DATA03	ALT5
GPIO4	GPIO4_IO00	SAI1_RXFS	ALT5
	GPIO4_IO01	SAI1_RXC	ALT5
	GPIO4_IO10	SAI1_TXFS	ALT5
	GPIO4_IO11	SAI1_TXC	ALT5
	GPIO4_IO12	SAI1_TXD0	ALT5
	GPIO4_IO13	SAI1_TXD1	ALT5
	GPIO4_IO14	SAI1_TXD2	ALT5
	GPIO4_IO15	SAI1_TXD3	ALT5
	GPIO4_IO16	SAI1_TXD4	ALT5
	GPIO4_IO17	SAI1_TXD5	ALT5
	GPIO4_IO18	SAI1_TXD6	ALT5
	GPIO4_IO19	SAI1_TXD7	ALT5
	GPIO4_IO02	SAI1_RXD0	ALT5
	GPIO4_IO20	SAI1_MCLK	ALT5
	GPIO4_IO21	SAI2_RXFS	ALT5
	GPIO4_IO22	SAI2_RXC	ALT5
	GPIO4_IO23	SAI2_RXD0	ALT5
	GPIO4_IO24	SAI2_TXFS	ALT5
	GPIO4_IO25	SAI2_TXC	ALT5
	GPIO4_IO26	SAI2_TXD0	ALT5
	GPIO4_IO27	SAI2_MCLK	ALT5
	GPIO4_IO28	SAI3_RXFS	ALT5
	GPIO4_IO29	SAI3_RXC	ALT5
	GPIO4_IO03	SAI1_RXD1	ALT5
	GPIO4_IO30	SAI3_RXD	ALT5
	GPIO4_IO31	SAI3_TXFS	ALT5
	GPIO4_IO04	SAI1_RXD2	ALT5
	GPIO4_IO05	SAI1_RXD3	ALT5
	GPIO4_IO06	SAI1_RXD4	ALT5
	GPIO4_IO07	SAI1_RXD5	ALT5
	GPIO4_IO08	SAI1_RXD6	ALT5
	GPIO4_IO09	SAI1_RXD7	ALT5
GPIO5	GPIO5_IO00	SAI3_TXC	ALT5

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	GPIO5_IO01	SAI3_TXD	ALT5
	GPIO5_IO10	ECSPI2_SCLK	ALT5
	GPIO5_IO11	ECSPI2_MOSI	ALT5
	GPIO5_IO12	ECSPI2_MISO	ALT5
	GPIO5_IO13	ECSPI2_SS0	ALT5
	GPIO5_IO14	I2C1_SCL	ALT5
	GPIO5_IO15	I2C1_SDA	ALT5
	GPIO5_IO16	I2C2_SCL	ALT5
	GPIO5_IO17	I2C2_SDA	ALT5
	GPIO5_IO18	I2C3_SCL	ALT5
	GPIO5_IO19	I2C3_SDA	ALT5
	GPIO5_IO02	SAI3_MCLK	ALT5
	GPIO5_IO20	I2C4_SCL	ALT5
	GPIO5_IO21	I2C4_SDA	ALT5
	GPIO5_IO22	UART1_RXD	ALT5
	GPIO5_IO23	UART1_TXD	ALT5
	GPIO5_IO24	UART2_RXD	ALT5
	GPIO5_IO25	UART2_TXD	ALT5
	GPIO5_IO26	UART3_RXD	ALT5
	GPIO5_IO27	UART3_TXD	ALT5
	GPIO5_IO28	UART4_RXD	ALT5
	GPIO5_IO29	UART4_TXD	ALT5
	GPIO5_IO03	SPDIF_TX	ALT5
	GPIO5_IO04	SPDIF_RX	ALT5
	GPIO5_IO05	SPDIF_EXT_CLK	ALT5
	GPIO5_IO06	ECSPI1_SCLK	ALT5
	GPIO5_IO07	ECSPI1_MOSI	ALT5
	GPIO5_IO08	ECSPI1_MISO	ALT5
	GPIO5_IO09	ECSPI1_SS0	ALT5
GPT1	GPT1_CAPTURE1	SAI3_RXFS	ALT1
	GPT1_CAPTURE2	SAI3_TXFS	ALT1
	GPT1_CLK	SAI3_RXC	ALT1
	GPT1_COMPARE1	SAI3_RXD	ALT1
	GPT1_COMPARE2	SAI3_TXC	ALT1
	GPT1_COMPARE3	SAI3_TXD	ALT1
GPT2	GPT2_CLK	I2C3_SCL	ALT2
GPT3	GPT3_CLK	I2C3_SDA	ALT2
I2C1	I2C1_SCL	I2C1_SCL	ALT0
	I2C1_SDA	I2C1_SDA	ALT0

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
I2C2	I2C2_SCL	I2C2_SCL	ALT0
	I2C2_SDA	I2C2_SDA	ALT0
I2C3	I2C3_SCL	I2C3_SCL	ALT0
	I2C3_SDA	I2C3_SDA	ALT0
I2C4	I2C4_SCL	I2C4_SCL	ALT0
	I2C4_SDA	I2C4_SDA	ALT0
CJTAG	CJTAG_MODE	JTAG_MOD	ALT0
	CJTAG_TCK	JTAG_TCK	ALT0
	CJTAG_TDI	JTAG_TDI	ALT0
	CJTAG_TDO	JTAG_TDO	ALT0
	CJTAG_TMS	JTAG_TMS	ALT0
	CJTAG_TRST_B	JTAG_TRST_B	ALT0
M4	M4_NMI	GPIO1_IO05	ALT1
MIPI DSI	MIPI_DSI_CLK_P	MIPI_DSI_CLK_P	No muxing
	MIPI_DSI_CLK_N	MIPI_DSI_CLK_N	No muxing
	MIPI_DSI_REXT	MIPI_DSI_REXT	No muxing
	MIPI_DSI_D0_P	MIPI_DSI_D0_P	No muxing
	MIPI_DSI_D0_N	MIPI_DSI_D0_N	No muxing
	MIPI_DSI_D1_P	MIPI_DSI_D1_P	No muxing
	MIPI_DSI_D1_N	MIPI_DSI_D1_N	No muxing
	MIPI_DSI_D2_P	MIPI_DSI_D2_P	No muxing
	MIPI_DSI_D2_N	MIPI_DSI_D2_N	No muxing
	MIPI_DSI_D3_P	MIPI_DSI_D3_P	No muxing
	MIPI_DSI_D3_N	MIPI_DSI_D3_N	No muxing
MIPI CSI	MIPI_CSI_CLK_P	MIPI_CSI_CLK_P	No muxing
	MIPI_CSI_CLK_N	MIPI_CSI_CLK_N	No muxing
	MIPI_CSI_D0_P	MIPI_CSI_D0_P	No muxing
	MIPI_CSI_D0_N	MIPI_CSI_D0_N	No muxing
	MIPI_CSI_D1_P	MIPI_CSI_D1_P	No muxing
	MIPI_CSI_D1_N	MIPI_CSI_D1_N	No muxing
	MIPI_CSI_D2_P	MIPI_CSI_D2_P	No muxing
	MIPI_CSI_D2_N	MIPI_CSI_D2_N	No muxing
	MIPI_CSI_D3_P	MIPI_CSI_D3_P	No muxing
	MIPI_CSI_D3_N	MIPI_CSI_D3_N	No muxing
PCIE1	PCIE1_CLKREQ_B	I2C4_SCL	ALT2
	PCIE1_CLKREQ_B	UART4_RXD	ALT2
	PCIE_REF_PAD_CLK_P	PCIE_REF_PAD_CLK_P	No muxing
	PCIE_REF_PAD_CLK_N	PCIE_REF_PAD_CLK_N	No muxing
	PCIE_RESREF	PCIE_RESREF	No muxing

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	PCIE_TXN_P	PCIE_TXN_P	No muxing
	PCIE_TXN_N	PCIE_TXN_N	No muxing
	PCIE_RXN_N	PCIE_RXN_N	No muxing
	PCIE_RXN_P	PCIE_RXN_P	No muxing
PDM	PDM_BIT_STREAM0	SAI1_RXD0	ALT3
	PDM_BIT_STREAM0	SAI5_RXD0	ALT4
	PDM_BIT_STREAM1	SAI1_RXD1	ALT3
	PDM_BIT_STREAM1	SAI5_RXD1	ALT4
	PDM_BIT_STREAM2	SAI1_RXD2	ALT3
	PDM_BIT_STREAM2	SAI5_RXD2	ALT4
	PDM_BIT_STREAM3	SAI1_RXD3	ALT3
	PDM_BIT_STREAM3	SAI5_RXD3	ALT4
	PDM_CLK	SAI1_TXD7	ALT3
	PDM_CLK	SAI1_MCLK	ALT3
	PDM_CLK	SAI5_RXC	ALT4
PWM1	PWM1_OUT	GPIO1_IO01	ALT1
	PWM1_OUT	SPDIF_EXT_CLK	ALT1
	PWM1_OUT	I2C4_SDA	ALT1
PWM2	PWM2_OUT	SPDIF_RX	ALT1
	PWM2_OUT	I2C4_SCL	ALT1
	PWM2_OUT	GPIO1_IO13	ALT5
PWM3	PWM3_OUT	SPDIF_TX	ALT1
	PWM3_OUT	I2C3_SDA	ALT1
	PWM3_OUT	GPIO1_IO14	ALT5
PWM4	PWM4_OUT	SAI3_MCLK	ALT1
	PWM4_OUT	I2C3_SCL	ALT1
	PWM4_OUT	GPIO1_IO15	ALT5
QSPI	QSPI_A_DATA0	NAND_DATA00	ALT1
	QSPI_A_DATA1	NAND_DATA01	ALT1
	QSPI_A_DATA2	NAND_DATA02	ALT1
	QSPI_A_DATA3	NAND_DATA03	ALT1
	QSPI_A_DQS	NAND_DQS	ALT1
	QSPI_A_SCLK	NAND_ALE	ALT1
	QSPI_A_SS0_B	NAND_CE0_B	ALT1
	QSPI_A_SS1_B	NAND_CE1_B	ALT1
	QSPI_B_DATA0	NAND_DATA04	ALT1
	QSPI_B_DATA1	NAND_DATA05	ALT1
	QSPI_B_DATA2	NAND_DATA06	ALT1
	QSPI_B_DATA3	NAND_DATA07	ALT1

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	QSPI_B_DQS	NAND_RE_B	ALT1
	QSPI_B_SCLK	NAND_CLE	ALT1
	QSPI_B_SS0_B	NAND_CE2_B	ALT1
	QSPI_B_SS1_B	NAND_CE3_B	ALT1
RAWNAND	RAWNAND_ALE	NAND_ALE	ALT0
	RAWNAND_CE0_B	NAND_CE0_B	ALT0
	RAWNAND_CE1_B	NAND_CE1_B	ALT0
	RAWNAND_CE2_B	NAND_CE2_B	ALT0
	RAWNAND_CE3_B	NAND_CE3_B	ALT0
	RAWNAND_CLE	NAND_CLE	ALT0
	RAWNAND_DATA00	NAND_DATA00	ALT0
	RAWNAND_DATA01	NAND_DATA01	ALT0
	RAWNAND_DATA02	NAND_DATA02	ALT0
	RAWNAND_DATA03	NAND_DATA03	ALT0
	RAWNAND_DATA04	NAND_DATA04	ALT0
	RAWNAND_DATA05	NAND_DATA05	ALT0
	RAWNAND_DATA06	NAND_DATA06	ALT0
	RAWNAND_DATA07	NAND_DATA07	ALT0
	RAWNAND_DQS	NAND_DQS	ALT0
	RAWNAND_RE_B	NAND_RE_B	ALT0
	RAWNAND_READY_B	NAND_READY_B	ALT0
	RAWNAND_WE_B	NAND_WE_B	ALT0
	RAWNAND_WP_B	NAND_WP_B	ALT0
SAI1	SAI1_MCLK	SAI1_MCLK	ALT0
	SAI1_RX_BCLK	SAI1_RXC	ALT0
	SAI1_RX_DATA0	SAI1_RXD0	ALT0
	SAI1_RX_DATA1	SAI1_RXD1	ALT0
	SAI1_RX_DATA2	SAI1_RXD2	ALT0
	SAI1_RX_DATA3	SAI1_RXD3	ALT0
	SAI1_RX_DATA4	SAI1_RXD4	ALT0
	SAI1_RX_DATA5	SAI1_RXD5	ALT0
	SAI1_RX_DATA6	SAI1_RXD6	ALT0
	SAI1_RX_DATA7	SAI1_RXD7	ALT0
	SAI1_RX_SYNC	SAI1_RXFS	ALT0
		SAI1_RXD5	ALT3
	SAI1_TX_BCLK	SAI1_TXC	ALT0
		SAI5_MCLK	ALT1
		SAI1_MCLK	ALT2
	SAI1_TX_DATA0	SAI1_TXD0	ALT0

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
		SAI5_RXFS	ALT1
	SAI1_TX_DATA1	SAI1_TXD1	ALT0
		SAI5_RXC	ALT1
		SAI1_RXD0	ALT2
	SAI1_TX_DATA2	SAI1_TXD2	ALT0
		SAI5_RXD0	ALT1
	SAI1_TX_DATA3	SAI1_TXD3	ALT0
		SAI5_RXD1	ALT1
	SAI1_TX_DATA4	SAI1_TXD4	ALT0
		SAI5_RXD2	ALT1
		SAI1_RXD7	ALT3
	SAI1_TX_DATA5	SAI1_TXD5	ALT0
		SAI5_RXD3	ALT1
	SAI1_TX_DATA6	SAI1_TXD6	ALT0
	SAI1_TX_DATA7	SAI1_TXD7	ALT0
	SAI1_TX_SYNC	SAI1_TXFS	ALT0
		SAI5_RXD1	ALT2
		SAI5_RXD2	ALT2
		SAI5_RXD3	ALT2
		SAI1_RXD7	ALT2
SAI2	SAI2_MCLK	SAI2_MCLK	ALT0
	SAI2_RX_BCLK	SAI2_RXC	ALT0
	SAI2_RX_DATA0	SAI2_RXD0	ALT0
	SAI2_RX_DATA1	SAI2_RXFS	ALT3
	SAI2_RX_SYNC	SAI2_RXFS	ALT0
	SAI2_TX_BCLK	SAI2_TXC	ALT0
	SAI2_TX_DATA0	SAI2_TXD0	ALT0
	SAI2_TX_DATA1	SAI2_TXFS	ALT3
	SAI2_TX_SYNC	SAI2_TXFS	ALT0
SAI3	SAI3_MCLK	SAI3_MCLK	ALT0
	SAI3_RX_BCLK	SAI3_RXC	ALT0
	SAI3_RX_DATA0	SAI3_RXD	ALT0
	SAI3_RX_DATA1	SAI3_RXFS	ALT3
	SAI3_RX_SYNC	SAI3_RXFS	ALT0
	SAI3_TX_BCLK	SAI3_TXC	ALT0
	SAI3_TX_DATA0	SAI3_TXD	ALT0
	SAI3_TX_DATA1	SAI3_TXFS	ALT3
	SAI3_TX_SYNC	SAI3_TXFS	ALT0
SAI5	SAI5_MCLK	SAI5_MCLK	ALT0

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
		SAI1_MCLK	ALT1
		SAI2_MCLK	ALT1
		SAI3_MCLK	ALT2
	SAI5_RX_BCLK	SAI5_RXC	ALT0
		SAI1_RXC	ALT1
		SAI3_RXC	ALT2
	SAI5_RX_DATA0	SAI5_RXD0	ALT0
		SAI1_RXD0	ALT1
		SAI3_RXD	ALT2
	SAI5_RX_DATA1	SAI5_RXD1	ALT0
		SAI1_RXD1	ALT1
		SAI3_TXFS	ALT2
	SAI5_RX_DATA2	SAI5_RXD2	ALT0
		SAI1_RXD2	ALT1
		SAI3_TXC	ALT2
	SAI5_RX_DATA3	SAI5_RXD3	ALT0
		SAI1_RXD3	ALT1
		SAI3_TXD	ALT2
	SAI5_RX_SYNC	SAI5_RXFS	ALT0
		SAI1_RXFS	ALT1
		SAI3_RXFS	ALT2
	SAI5_TX_BCLK	SAI1_TXC	ALT1
		SAI2_RXC	ALT1
		SAI5_RXD2	ALT3
	SAI5_TX_DATA0	SAI1_TXD0	ALT1
		SAI2_RXD0	ALT1
		SAI5_RXD3	ALT3
	SAI5_TX_DATA1	SAI1_TXD1	ALT1
		SAI2_TXFS	ALT1
		SAI2_RXFS	ALT2
	SAI5_TX_DATA2	SAI1_TXD2	ALT1
		SAI2_TXC	ALT1
	SAI5_TX_DATA3	SAI1_TXD3	ALT1
		SAI2_TXD0	ALT1
	SAI5_TX_SYNC	SAI1_TXFS	ALT1
		SAI2_RXFS	ALT1
		SAI5_RXD1	ALT3
SAI6	SAI6_MCLK	SAI1_RXD7	ALT1
		SAI1_TXD7	ALT1

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	SAI6_RX_BCLK	SAI1_TXD4	ALT1
		SAI1_RXD4	ALT2
	SAI6_RX_DATA0	SAI1_TXD5	ALT1
		SAI1_RXD5	ALT2
	SAI6_RX_SYNC	SAI1_TXD6	ALT1
		SAI1_RXD6	ALT2
	SAI6_TX_BCLK	SAI1_RXD4	ALT1
		SAI1_TXD4	ALT2
	SAI6_TX_DATA0	SAI1_RXD5	ALT1
		SAI1_TXD5	ALT2
	SAI6_TX_SYNC	SAI1_RXD6	ALT1
		SAI1_TXD6	ALT2
SDMA	SDMA1_EXT_EVENT0	GPIO1_IO03	ALT5
	SDMA1_EXT_EVENT1	GPIO1_IO04	ALT5
	SDMA2_EXT_EVENT0	GPIO1_IO09	ALT5
	SDMA2_EXT_EVENT1	GPIO1_IO12	ALT5
SNVS	SNVS_ONOFF	ONOFF	ALT0
	SNVS_PMIC_ON_REQ	PMIC_ON_REQ	ALT0
	SNVS_POR_B	POR_B	ALT0
	SNVS_RTC	RTC_XTALI	ALT0
	SNVS_RTC_RESET_B	RTC_RESET_B	ALT0
SPDIF1	SPDIF1_EXT_CLK	SPDIF_EXT_CLK	ALT0
	SPDIF1_IN	SPDIF_RX	ALT0
	SPDIF1_OUT	SPDIF_TX	ALT0
SRC	SRC_BOOT_CFG0	SAI1_RXD0	ALT6
	SRC_BOOT_CFG1	SAI1_RXD1	ALT6
	SRC_BOOT_CFG10	SAI1_TXD2	ALT6
	SRC_BOOT_CFG11	SAI1_TXD3	ALT6
	SRC_BOOT_CFG12	SAI1_TXD4	ALT6
	SRC_BOOT_CFG13	SAI1_TXD5	ALT6
	SRC_BOOT_CFG14	SAI1_TXD6	ALT6
	SRC_BOOT_CFG15	SAI1_TXD7	ALT6
	SRC_BOOT_CFG2	SAI1_RXD2	ALT6
	SRC_BOOT_CFG3	SAI1_RXD3	ALT6
	SRC_BOOT_CFG4	SAI1_RXD4	ALT6
	SRC_BOOT_CFG5	SAI1_RXD5	ALT6
	SRC_BOOT_CFG6	SAI1_RXD6	ALT6
	SRC_BOOT_CFG7	SAI1_RXD7	ALT6
	SRC_BOOT_CFG8	SAI1_TXD0	ALT6

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	SRC_BOOT_CFG9	SAI1_TXD1	ALT6
	SRC_BOOT_MODE0	BOOT_MODE0	ALT0
	SRC_BOOT_MODE1	BOOT_MODE1	ALT0
	SRC_EARLY_RESET	SD2_DATA3	ALT6
	SRC_SYSTEM_RESET	SD2_RESET_B	ALT6
UART1	UART1_CTS_B	UART3_RXD	ALT1
		SAI2_TXFS	ALT4
	UART1_RTS_B	UART3_TXD	ALT1
		SAI2_RXD0	ALT4
	UART1_RX	UART1_RXD	ALT0
		SAI2_RXC	ALT4
	UART1_TX	UART1_TXD	ALT0
		SAI2_RXFS	ALT4
UART2	UART2_CTS_B	UART4_RXD	ALT1
		SAI3_RXC	ALT4
	UART2_RTS_B	UART4_TXD	ALT1
		SAI3_RXD	ALT4
	UART2_RX	UART2_RXD	ALT0
		SAI3_TXFS	ALT4
	UART2_TX	UART2_TXD	ALT0
		SAI3_TXC	ALT4
UART3	UART3_CTS_B	ECSPI1_MISO	ALT1
	UART3_RTS_B	ECSPI1_SS0	ALT1
	UART3_RX	UART3_RXD	ALT0
		ECSPI1_SCLK	ALT1
	UART3_TX	UART3_TXD	ALT0
		ECSPI1_MOSI	ALT1
UART4	UART4_CTS_B	ECSPI2_MISO	ALT1
	UART4_RTS_B	ECSPI2_SS0	ALT1
	UART4_RX	UART4_RXD	ALT0
		ECSPI2_SCLK	ALT1
	UART4_TX	UART4_TXD	ALT0
		ECSPI2_MOSI	ALT1
USB1	USB1_OTG_ID	GPIO1_IO10	ALT1
	USB1_OTG_OC	GPIO1_IO13	ALT1
	USB1_OTG_PWR	GPIO1_IO12	ALT1
	USB1_P1_ID	USB1_ID	No muxing
	USB1_P1_VBUS	USB1_VBUS	No muxing
	USB1_P1_DP	USB1_DP	No muxing

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	USB1_P1_DN	USB1_DN	No muxing
USB2	USB2_OTG_ID	GPIO1_IO11	ALT1
	USB2_OTG_OC	GPIO1_IO15	ALT1
	USB2_OTG_PWR	GPIO1_IO14	ALT1
	USB2_P2_ID	USB2_ID	No muxing
	USB2_P2_VBUS	USB2_VBUS	No muxing
	USB2_P2_DP	USB2_DP	No muxing
	USB2_P2_DN	USB2_DN	No muxing
USDHC1	USDHC1_CD_B	GPIO1_IO06	ALT5
	USDHC1_CLK	SD1_CLK	ALT0
	USDHC1_CMD	SD1_CMD	ALT0
	USDHC1_DATA0	SD1_DATA0	ALT0
	USDHC1_DATA1	SD1_DATA1	ALT0
	USDHC1_DATA2	SD1_DATA2	ALT0
	USDHC1_DATA3	SD1_DATA3	ALT0
	USDHC1_DATA4	SD1_DATA4	ALT0
	USDHC1_DATA5	SD1_DATA5	ALT0
	USDHC1_DATA6	SD1_DATA6	ALT0
	USDHC1_DATA7	SD1_DATA7	ALT0
	USDHC1_RESET_B	SD1_RESET_B	ALT0
	USDHC1_STROBE	SD1_STROBE	ALT0
	USDHC1_VSELECT	GPIO1_IO03	ALT1
	USDHC1_WP	GPIO1_IO07	ALT5
USDHC2	USDHC2_CD_B	SD2_CD_B	ALT0
	USDHC2_CLK	SD2_CLK	ALT0
	USDHC2_CMD	SD2_CMD	ALT0
	USDHC2_DATA0	SD2_DATA0	ALT0
	USDHC2_DATA1	SD2_DATA1	ALT0
	USDHC2_DATA2	SD2_DATA2	ALT0
	USDHC2_DATA3	SD2_DATA3	ALT0
	USDHC2_RESET_B	SD2_RESET_B	ALT0
		GPIO1_IO08	ALT5
	USDHC2_VSELECT	GPIO1_IO04	ALT1
	USDHC2_WP	SD2_WP	ALT0
USDHC3	USDHC3_CD_B	NAND_DATA02	ALT2
		I2C2_SCL	ALT2
		GPIO1_IO14	ALT4
	USDHC3_CLK	NAND_WE_B	ALT2
	USDHC3_CMD	NAND_WP_B	ALT2

Table continues on the next page...

Table 8-1. Muxing Options (continued)

Instance	Port	PAD	MODE
	USDHC3_DATA0	NAND_DATA04	ALT2
	USDHC3_DATA1	NAND_DATA05	ALT2
	USDHC3_DATA2	NAND_DATA06	ALT2
	USDHC3_DATA3	NAND_DATA07	ALT2
	USDHC3_DATA4	NAND_RE_B	ALT2
	USDHC3_DATA5	NAND_CE2_B	ALT2
	USDHC3_DATA6	NAND_CE3_B	ALT2
	USDHC3_DATA7	NAND_CLE	ALT2
	USDHC3_RESET_B	NAND_READY_B	ALT2
		UART3_RXD	ALT2
		GPIO1_IO09	ALT4
	USDHC3_STROBE	NAND_CE1_B	ALT2
	USDHC3_VSELECT	UART3_TXD	ALT2
		GPIO1_IO11	ALT4
	USDHC3_WP	NAND_DATA03	ALT2
		I2C2_SDA	ALT2
		GPIO1_IO15	ALT4
WDOG1	WDOG1_ANY	GPIO1_IO02	ALT5
	WDOG1_B	GPIO1_IO02	ALT1
XTALOSC	REF_CLK_24M	GPIO1_IO01	ALT0
	REF_CLK_32K	GPIO1_IO00	ALT5
	XTALI_24M	XTALI_24M	ALT0
	XTALO_24M	XTALO_24M	No muxing

8.2 IOMUX Controller (IOMUXC)

8.2.1 Overview

The IOMUX Controller (IOMUXC), together with the IOMUX, enables the IC to share one pad to several functional blocks. This sharing is done by multiplexing the pad's input and output signals.

Every module requires a specific pad setting (such as pull up or keeper), and for each pad, there are up to 8 muxing options (called ALT modes). The pad settings parameters are controlled by the IOMUXC.

The IOMUX consists only of combinatorial logic combined from several basic IOMUX cells. Each basic IOMUX cell handles only one pad signal's muxing.

Figure 8-1 illustrates the IOMUX/IOMUXC connectivity in the system.

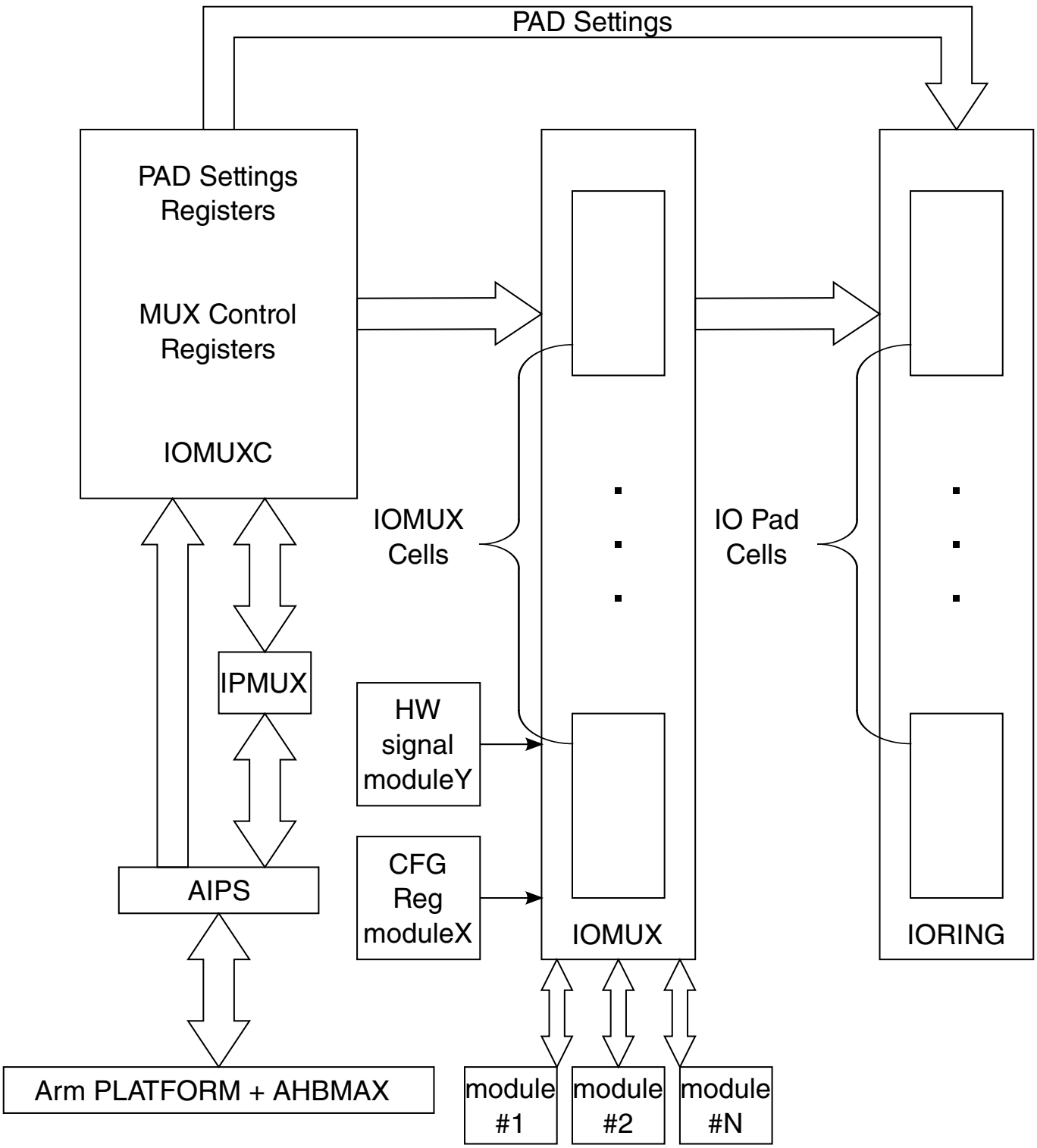


Figure 8-1. IOMUX SoC Level Block Diagram

8.2.1.1 Features

The IOMUXC features are:

- 32-bit software mux control registers (IOMUXC_SW_MUX_CTL_PAD_<PAD NAME> or IOMUXC_SW_MUX_CTL_GRP_<GROUP NAME>) to configure 1 of 8 alternate (ALT) MUX_MODE feilds of each pad or a predefined group of pads and to enable the forcing of an input path of the pad(s) (SION bit).
- 32-bit software pad control registers (IOMUXC_SW_PAD_CTL_PAD_<PAD_NAME> or IOMUXC_SW_PAD_CTL_GRP_<GROUP NAME>) to configure specific pad settings of each pad, or a predefined group of pads.
- 32-bit general purpose registers - 32-bit registers according to SoC requirements for any usage.
- 32-bit input select control registers to control the input path to a module when more than one pad drives this module input.

Each SW MUX/PAD CTL IOMUXC register handles only one pad or one pad's group.

Only the minimum number of registers required by software are implemented by hardware. For example, if only ALT0 and ALT1 modes are used on Pad x then only one bit register will be generated as the MUX_MODE control field in the software mux control register of Pad x.

The software mux control registers may allow the forcing of pads to become input (input path enabled) regardless of the functional direction driven. This may be useful for loopback and GPIO data capture.

8.2.2 Clocks

The table found here describes the clock sources for IOMUXC.

Please see Clock Controller Module (CCM) for clock setting, configuration and gating information.

Table 8-2. IOMUXC Clocks

Clock name	Clock Root	Description
ipg_clk_s	ipg_clk_root	Peripheral access clock

8.2.3 Functional description

This section provides a complete functional description of the block.

The IOMUXC consists of two sub-blocks:

- IOMUXC_REGISTERS includes all of the IOMUXC registers (see Features).
- IOMUXC_LOGIC includes all of the IOMUXC combinatorial logic (IP interface controls, address decoder, observability muxes).

The IOMUX consists of a number (about the number of pads in the SoC) of basic iomux_cell units. If only one functional mode is required for a specific pad, there is no need for IOMUX and the signals can be connected directly from the module to the I/O. The IOMUX cell is required whenever two or more functional modes are required for a specific pad or when one functional mode and the one test mode are required.

The basic iomux_cell design, which allows two levels of HW signal control (in ALT6 and ALT7 modes - ALT7 gets highest priority) is shown in Figure 8-2.

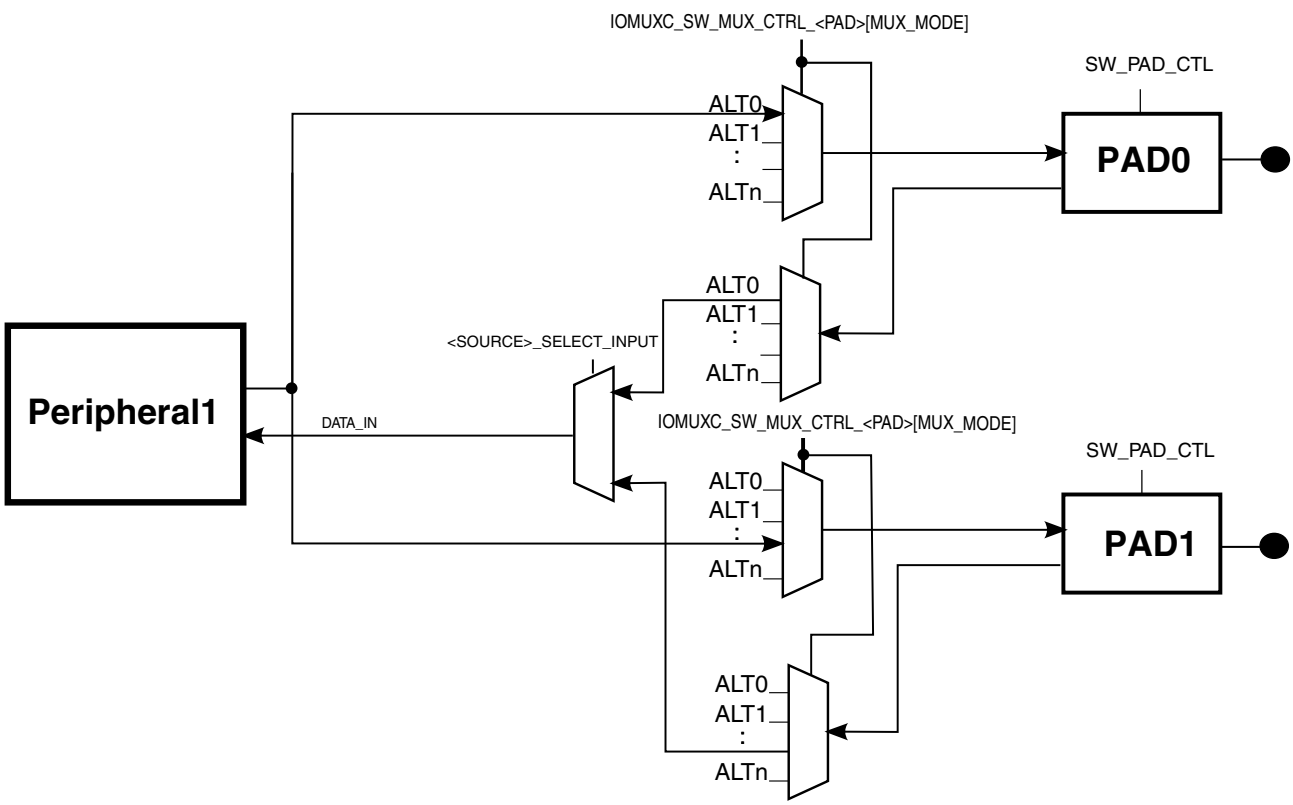


Figure 8-2. IOMUX Cell Block Diagram

8.2.3.1 GPIO pad features

The GPIO pad includes the following features:

- Wide-range voltage interface
  - 1.8V ~ 3.3V I/O interface
- CMOS input / Schmitt trigger Input
- 3-state and open-drain output
- Two slew rate control levels
- Programmable feature support
  - Controllable input enable
  - Controllable CMOS/Schmitt trigger input
  - Controllable pull-up/pull-down resistor
  - Controllable output drive strength (x1 / x2 / x4 / x6)
  - Controllable slew rate control (slow slew / fast slew)

8.2.3.1.1 Pull up/Pull down control

Pull up/Pull down function is controlled by the PE and PS pin.

Table 8-3. Pull up / Pull down control truth table

Mode	State	
	PE	PS
Disable	0	X
Pull-down enable	1	0
Pull-up enable	1	1

8.2.3.1.2 Input control

The IS pin selects CMOS and Schmitt trigger.

Table 8-4. Input control truth table

Mode	State	
	IE	IS
Disable	0	X
CMOS input	1	0
Schmitt trigger input	1	1

8.2.3.1.3 Output driver control

Output drive strength is controlled by the DS0, DS1 pin.

Table 8-5. Drive strength control truth table

State		Driver Strength
DS0	DS1	
0	0	X1
0	1	X2
1	0	X4
1	1	X6

The SR pin controls slew-rate of output driver.

Table 8-6. Slew-rate of output driver

State	Driver Slew
SR	
0	Fast Slew
1	Slow Slew

8.2.3.2 ALT6 and ALT7 extended muxing modes

The ALT7 and ALT6 extended muxing modes allow any signal in the system (such as fuse, pad input, JTAG, or software register) to override any software configuration and to force the ALT6/ALT7 muxing mode.

It also allows an IOMUX software register to control a group of pads.

8.2.3.3 SW Loopback through SION bit

A limited option exists to override the default pad functionality and force the input path to be active (ipp_ibe==1'b1) regardless of the value driven by the corresponding module. This can be done by setting the SION (Software Input On) bit in the IOMUXC_SW_MUX_CTL register (when available) to "1".

Uses include:

- LoopBack - Module x drives the pad and also receives pad value as an input.

8.2.3.4 Daisy chain - multi pads driving same module input pin

In some cases, more than one pad may drive a single module input pin. Such cases require the addition of one more level of IOMUXing; all of these input signals are muxed, and a dedicated software controlled register controls the mux in order to select the required input path.

A module port involved in "daisy chain" requires two software configuration commands, one for selecting the mode for this pad (programable via the IOMUXC_SW_MUX_CTL_<PAD> registers) and one for defining it as the input path (via the daisy chain registers).

This means that a module port involved in "daisy chain" requires two software configuration commands, one for selecting the mode for this pad (programable via the IOMUXC_SW_MUX_CTL_<PAD> registers) and one for defining it as the input path (via the daisy chain registers). The daisy chain is illustrated in the figure below.

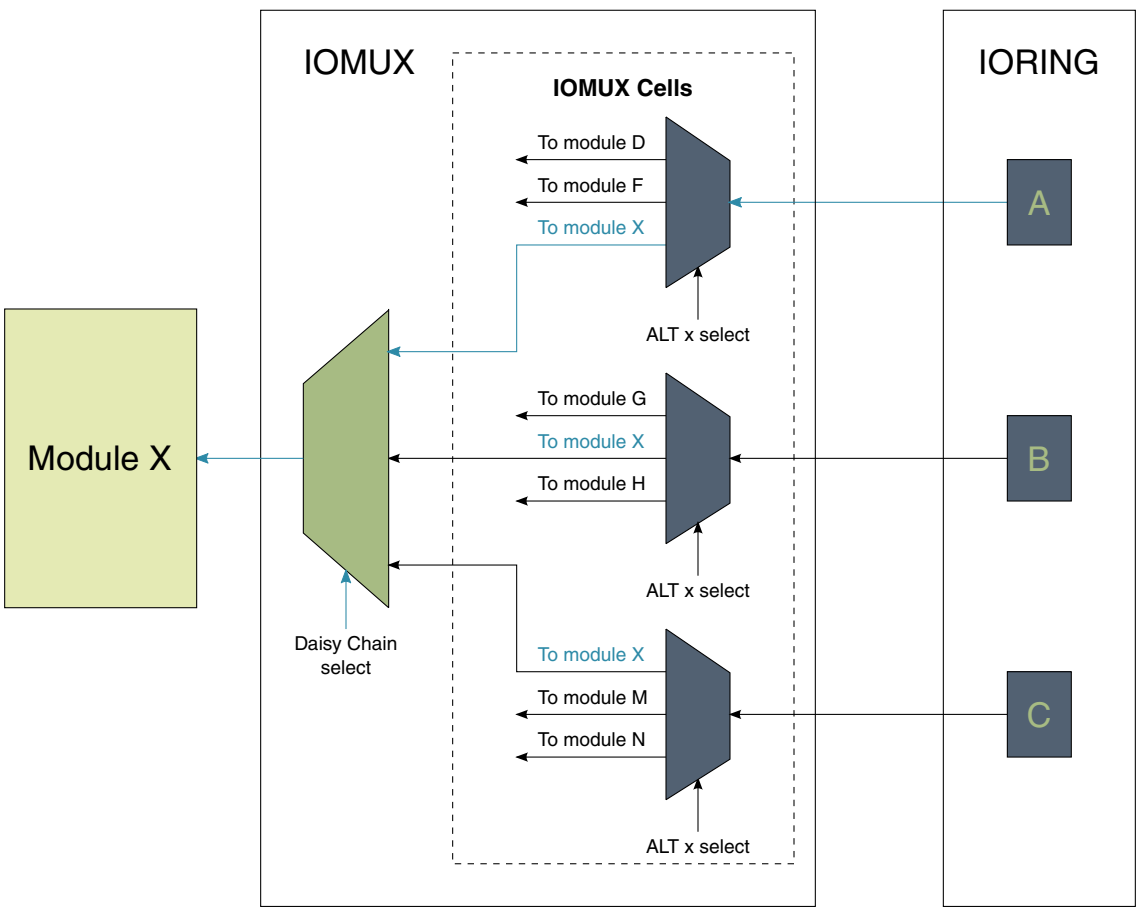


Figure 8-3. Daisy chain illustration

8.2.4 IOMUXC GPR Memory Map/Register Definition

IOMUXC_GPR memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3034_0000	General Purpose Register 0 (IOMUXC_GPR_GPR0)	32	R/W	0000_0000h	8.2.4.1/ 1252
3034_0004	General Purpose Register 1 (IOMUXC_GPR_GPR1)	32	R/W	0000_0000h	8.2.4.2/ 1253
3034_0008	General Purpose Register 2 (IOMUXC_GPR_GPR2)	32	R/W	0000_0000h	8.2.4.3/ 1254
3034_000C	General Purpose Register 3 (IOMUXC_GPR_GPR3)	32	R/W	See section	8.2.4.4/ 1257
3034_0010	General Purpose Register 4 (IOMUXC_GPR_GPR4)	32	R/W	0000_0000h	8.2.4.5/ 1261
3034_0014	General Purpose Register 5 (IOMUXC_GPR_GPR5)	32	R/W	0000_0000h	8.2.4.6/ 1264
3034_0018	General Purpose Register 6 (IOMUXC_GPR_GPR6)	32	R/W	0000_0000h	8.2.4.7/ 1265
3034_001C	General Purpose Register 7 (IOMUXC_GPR_GPR7)	32	R/W	0000_0000h	8.2.4.8/ 1267
3034_0020	General Purpose Register 8 (IOMUXC_GPR_GPR8)	32	R/W	0000_0000h	8.2.4.9/ 1268
3034_0024	General Purpose Register 9 (IOMUXC_GPR_GPR9)	32	R/W	0000_0000h	8.2.4.10/ 1269
3034_0028	General Purpose Register 10 (IOMUXC_GPR_GPR10)	32	R/W	0000_0008h	8.2.4.11/ 1270
3034_002C	General Purpose Register 11 (IOMUXC_GPR_GPR11)	32	R/W	0000_0000h	8.2.4.12/ 1272
3034_0030	General Purpose Register 12 (IOMUXC_GPR_GPR12)	32	R/W	0000_4000h	8.2.4.13/ 1274
3034_0034	General Purpose Register 13 (IOMUXC_GPR_GPR13)	32	R/W	See section	8.2.4.14/ 1275
3034_0038	General Purpose Register 14 (IOMUXC_GPR_GPR14)	32	R/W	4B49_9100h	8.2.4.15/ 1277
3034_003C	General Purpose Register 15 (IOMUXC_GPR_GPR15)	32	R/W	6188_FFFFh	8.2.4.16/ 1278
3034_0040	General Purpose Register 16 (IOMUXC_GPR_GPR16)	32	R/W	4940_9100h	8.2.4.17/ 1279
3034_0044	General Purpose Register 17 (IOMUXC_GPR_GPR17)	32	R/W	6188_FFFFh	8.2.4.18/ 1280
3034_0048	General Purpose Register 18 (IOMUXC_GPR_GPR18)	32	R/W	0000_0000h	8.2.4.19/ 1281
3034_004C	General Purpose Register 19 (IOMUXC_GPR_GPR19)	32	R	See section	8.2.4.20/ 1281

Table continues on the next page...

IOMUXC_GPR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3034_0050	General Purpose Register 20 (IOMUXC_GPR_GPR20)	32	R/W	0000_0000h	8.2.4.21/ 1281
3034_0054	General Purpose Register 21 (IOMUXC_GPR_GPR21)	32	R/W	0000_0000h	8.2.4.22/ 1282
3034_0058	General Purpose Register 22 (IOMUXC_GPR_GPR22)	32	R/W	See section	8.2.4.23/ 1283
3034_005C	General Purpose Register (IOMUXC_GPR_GPR23)	32	R/W	Undefined	8.2.4.24/ 1283
3034_0060	General Purpose Register (IOMUXC_GPR_GPR24)	32	R/W	Undefined	8.2.4.24/ 1283
3034_0064	General Purpose Register (IOMUXC_GPR_GPR25)	32	R/W	Undefined	8.2.4.24/ 1283
3034_0068	General Purpose Register (IOMUXC_GPR_GPR26)	32	R/W	Undefined	8.2.4.24/ 1283
3034_006C	General Purpose Register (IOMUXC_GPR_GPR27)	32	R/W	Undefined	8.2.4.24/ 1283
3034_0070	General Purpose Register (IOMUXC_GPR_GPR28)	32	R/W	Undefined	8.2.4.24/ 1283
3034_0074	General Purpose Register (IOMUXC_GPR_GPR29)	32	R/W	Undefined	8.2.4.24/ 1283
3034_0078	General Purpose Register (IOMUXC_GPR_GPR30)	32	R/W	Undefined	8.2.4.24/ 1283
3034_007C	General Purpose Register (IOMUXC_GPR_GPR31)	32	R/W	Undefined	8.2.4.24/ 1283
3034_0080	General Purpose Register (IOMUXC_GPR_GPR32)	32	R/W	Undefined	8.2.4.24/ 1283
3034_0084	General Purpose Register (IOMUXC_GPR_GPR33)	32	R/W	Undefined	8.2.4.24/ 1283
3034_0088	General Purpose Register (IOMUXC_GPR_GPR34)	32	R/W	Undefined	8.2.4.24/ 1283
3034_008C	General Purpose Register (IOMUXC_GPR_GPR35)	32	R/W	Undefined	8.2.4.24/ 1283
3034_0090	General Purpose Register (IOMUXC_GPR_GPR36)	32	R/W	Undefined	8.2.4.24/ 1283
3034_0094	General Purpose Register (IOMUXC_GPR_GPR37)	32	R/W	Undefined	8.2.4.24/ 1283
3034_0098	General Purpose Register (IOMUXC_GPR_GPR38)	32	R/W	Undefined	8.2.4.24/ 1283
3034_009C	General Purpose Register (IOMUXC_GPR_GPR39)	32	R/W	Undefined	8.2.4.24/ 1283
3034_00A0	General Purpose Register (IOMUXC_GPR_GPR40)	32	R/W	Undefined	8.2.4.24/ 1283
3034_00A4	General Purpose Register (IOMUXC_GPR_GPR41)	32	R/W	Undefined	8.2.4.24/ 1283

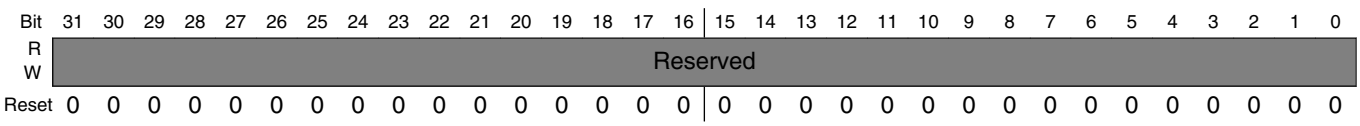
Table continues on the next page...

IOMUXC_GPR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3034_00A8	General Purpose Register (IOMUXC_GPR_GPR42)	32	R/W	Undefined	8.2.4.24/ 1283
3034_00AC	General Purpose Register (IOMUXC_GPR_GPR43)	32	R/W	Undefined	8.2.4.24/ 1283
3034_00B0	General Purpose Register (IOMUXC_GPR_GPR44)	32	R/W	Undefined	8.2.4.24/ 1283
3034_00B4	General Purpose Register (IOMUXC_GPR_GPR45)	32	R/W	Undefined	8.2.4.24/ 1283
3034_00B8	General Purpose Register (IOMUXC_GPR_GPR46)	32	R/W	Undefined	8.2.4.24/ 1283
3034_00BC	General Purpose Register (IOMUXC_GPR_GPR47)	32	R/W	Undefined	8.2.4.24/ 1283

8.2.4.1 General Purpose Register 0 (IOMUXC_GPR_GPR0)

Address: 3034_0000h base + 0h offset = 3034_0000h

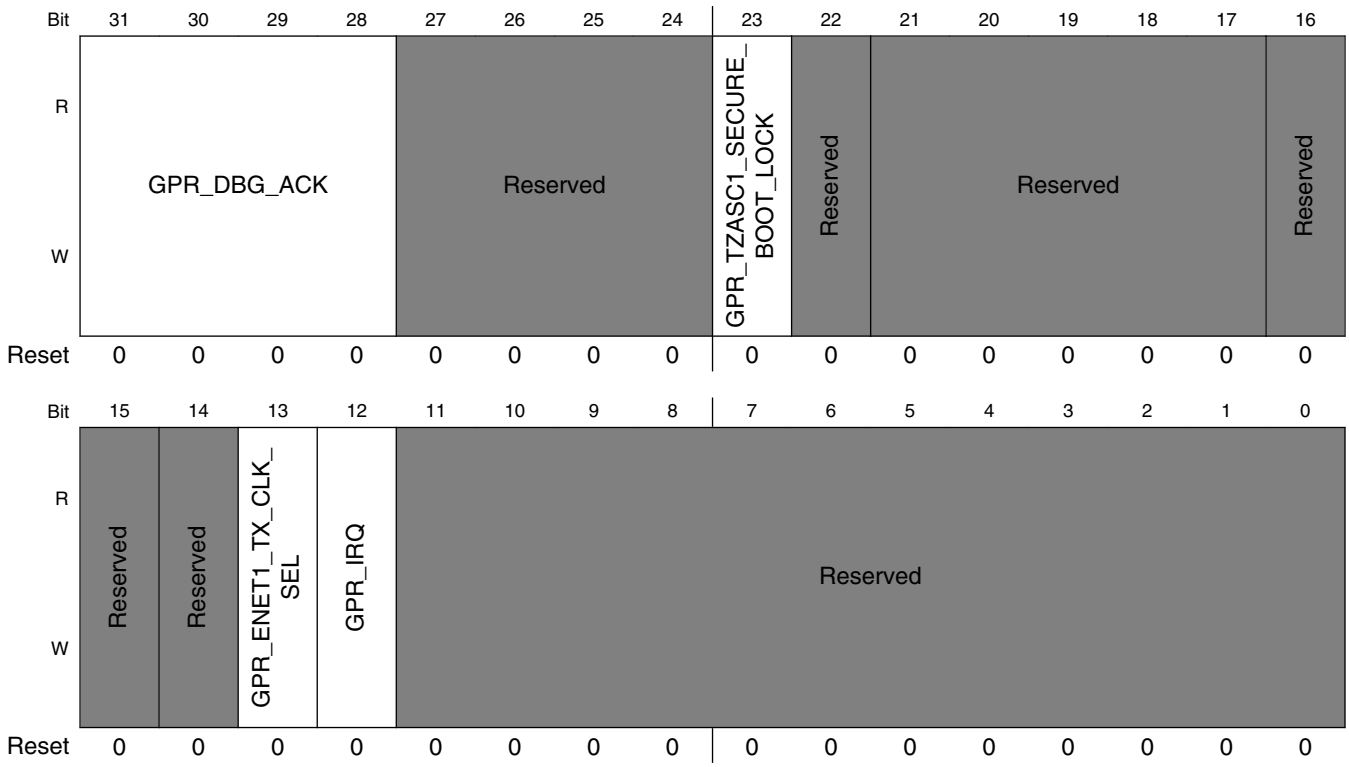


IOMUXC_GPR_GPR0 field descriptions

Field	Description
-	This field is reserved.

8.2.4.2 General Purpose Register 1 (IOMUXC_GPR_GPR1)

Address: 3034_0000h base + 4h offset = 3034_0004h



IOMUXC_GPR_GPR1 field descriptions

Field	Description
31–28 GPR_DBG_ACK	For CA53 cores.
27–24 -	This field is reserved.
23 GPR_TZASC1_SECURE_BOOT_LOCK	TZASC-1 Secure Boot Lock.
22 -	This field is reserved.
21–17 -	This field is reserved.
16 -	This field is reserved.
15 -	This field is reserved.
14 -	This field is reserved.

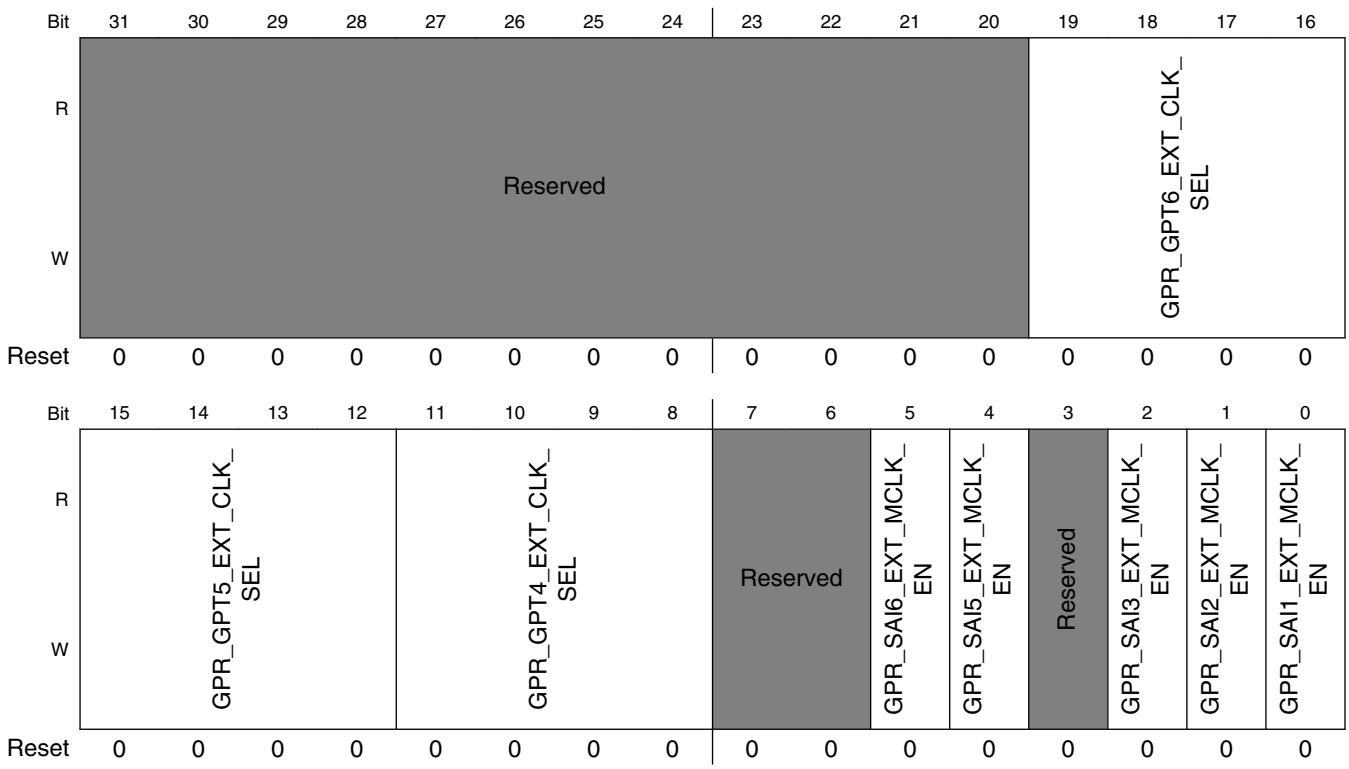
Table continues on the next page...

IOMUXC_GPR_GPR1 field descriptions (continued)

Field	Description
13 GPR_ENET1_ TX_CLK_SEL	1 ENET1 RMII clock comes from CCM->pad->loopback. SOI bit for the pad (IOMUXC_SW_INPUT_ON_PAD_ENET_TD2) should be set also. 0 ENET1 RMII clock comes from external PHY or OSC
12 GPR_IRQ	Generate IRQ on IRQ0.
-	This field is reserved.

8.2.4.3 General Purpose Register 2 (IOMUXC_GPR_GPR2)

Address: 3034_0000h base + 8h offset = 3034_0008h



IOMUXC_GPR_GPR2 field descriptions

Field	Description
31–20 -	This field is reserved.
19–16 GPR_GPT6_ EXT_CLK_SEL	GPT6 IPP_IND_CLKIN source select. 4'h0 SAI1_TX_SYNC 4'h1 SAI2_TX_SYNC 4'h2 SAI3_TX_SYNC 4'h3 Reserved

Table continues on the next page...

IOMUXC_GPR_GPR2 field descriptions (continued)

Field	Description
	4'h4 SAI5_TX_SYNC 4'h5 SAI6_TX_SYNC 4'h6 SAI1_RX_SYNC 4'h7 SAI2_RX_SYNC 4'h8 SAI3_RX_SYNC 4'h9 Reserved 4'ha SAI5_RX_SYNC 4'hb SAI6_RX_SYNC
15–12 GPR_GPT5_ EXT_CLK_SEL	GPT5 IPP_IND_CLKIN source select. 4'h0 SAI1_TX_SYNC 4'h1 SAI2_TX_SYNC 4'h2 SAI3_TX_SYNC 4'h3 Reserved 4'h4 SAI5_TX_SYNC 4'h5 SAI6_TX_SYNC 4'h6 SAI1_RX_SYNC 4'h7 SAI2_RX_SYNC 4'h8 SAI3_RX_SYNC 4'h9 Reserved 4'ha SAI5_RX_SYNC 4'hb SAI6_RX_SYNC
11–8 GPR_GPT4_ EXT_CLK_SEL	GPT4 IPP_IND_CLKIN source select. 4'h0 SAI1_TX_SYNC 4'h1 SAI2_TX_SYNC 4'h2 SAI3_TX_SYNC 4'h3 Reserved 4'h4 SAI5_TX_SYNC 4'h5 SAI6_TX_SYNC 4'h6 SAI1_RX_SYNC 4'h7 SAI2_RX_SYNC 4'h8 SAI3_RX_SYNC 4'h9 Reserved 4'ha SAI5_RX_SYNC 4'hb SAI6_RX_SYNC
7–6 -	This field is reserved.
5 GPR_SAI6_ EXT_MCLK_EN	
4 GPR_SAI5_ EXT_MCLK_EN	
3 -	This field is reserved.

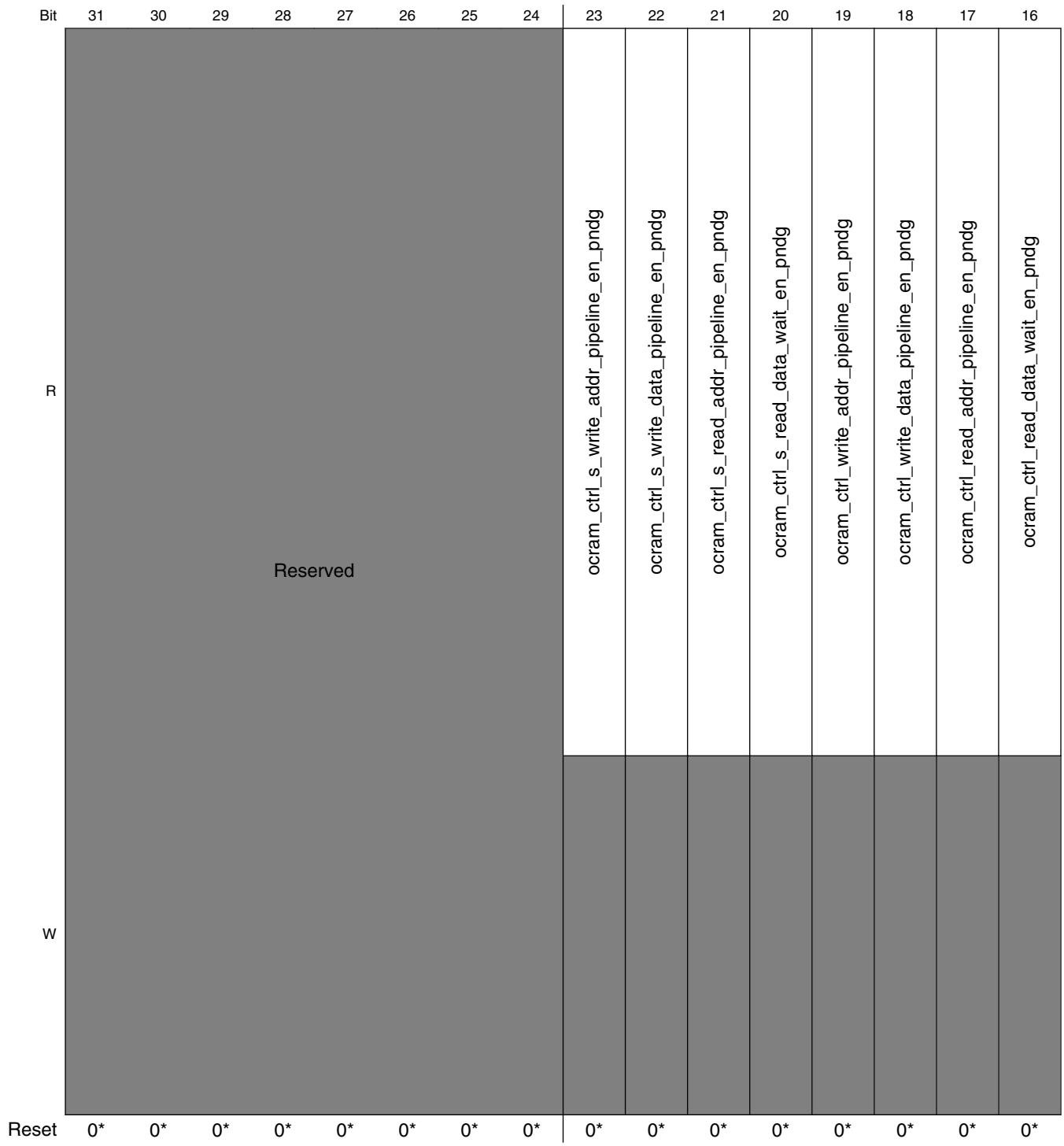
Table continues on the next page...

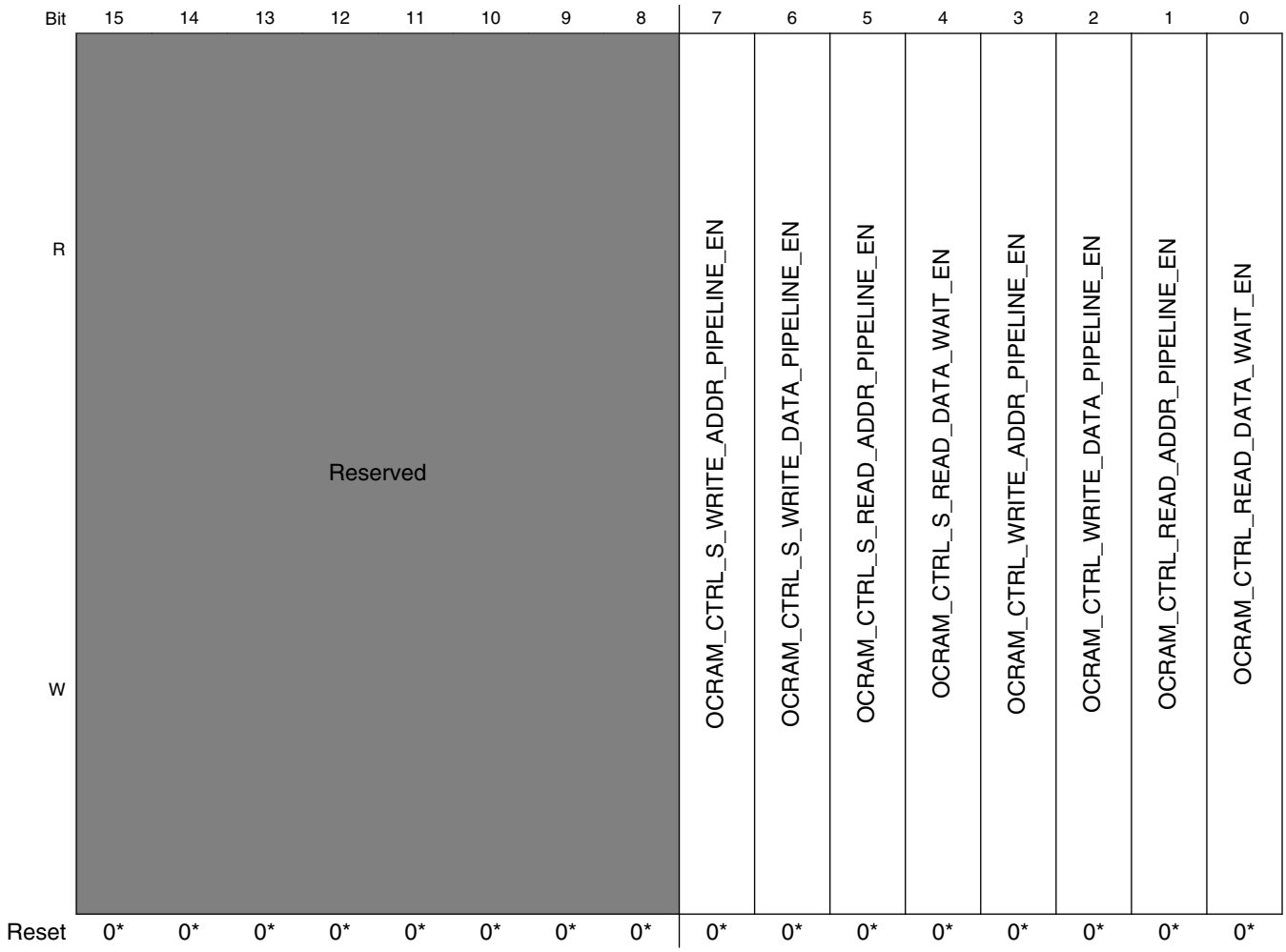
IOMUXC_GPR_GPR2 field descriptions (continued)

Field	Description
2 GPR_SAI3_ EXT_MCLK_EN	
1 GPR_SAI2_ EXT_MCLK_EN	
0 GPR_SAI1_ EXT_MCLK_EN	

8.2.4.4 General Purpose Register 3 (IOMUXC_GPR_GPR3)

Address: 3034_0000h base + Ch offset = 3034_000Ch





* Notes:
- For reset:

  [31:16] - N/A

  [15:0] - 16'b0000000011111111

IOMUXC_GPR_GPR3 field descriptions

Field	Description
31–24 -	This field is reserved.
23 ocram_ctrl_s_ write_addr_ pipeline_en_pndg	Write address pipeline enable update is pending.
22 ocram_ctrl_s_ write_data_ pipeline_en_pndg	Write data pipeline enable update is pending.

Table continues on the next page...

IOMUXC_GPR_GPR3 field descriptions (continued)

Field	Description
21 ocram_ctrl_s_read_addr_pipeline_en_pndg	Read address pipeline enable update is pending.
20 ocram_ctrl_s_read_data_wait_en_pndg	Read data wait state control update is pending.
19 ocram_ctrl_write_addr_pipeline_en_pndg	Write address pipeline enable update is pending.
18 ocram_ctrl_write_data_pipeline_en_pndg	Write data pipeline enable update is pending.
17 ocram_ctrl_read_addr_pipeline_en_pndg	Read address pipeline enable update is pending.
16 ocram_ctrl_read_data_wait_en_pndg	Read data wait state control update is pending.
15–8 -	This field is reserved.
7 OCRAM_CTRL_S_WRITE_ADDR_PIPELINE_EN	Write address pipeline enable.
6 OCRAM_CTRL_S_WRITE_DATA_PIPELINE_EN	Write data pipeline enable.
5 OCRAM_CTRL_S_READ_ADDR_PIPELINE_EN	Read address pipeline enable.
4 OCRAM_CTRL_S_READ_DATA_WAIT_EN	Read data wait state control.
3 OCRAM_CTRL_WRITE_ADDR_PIPELINE_EN	Write address pipeline enable.

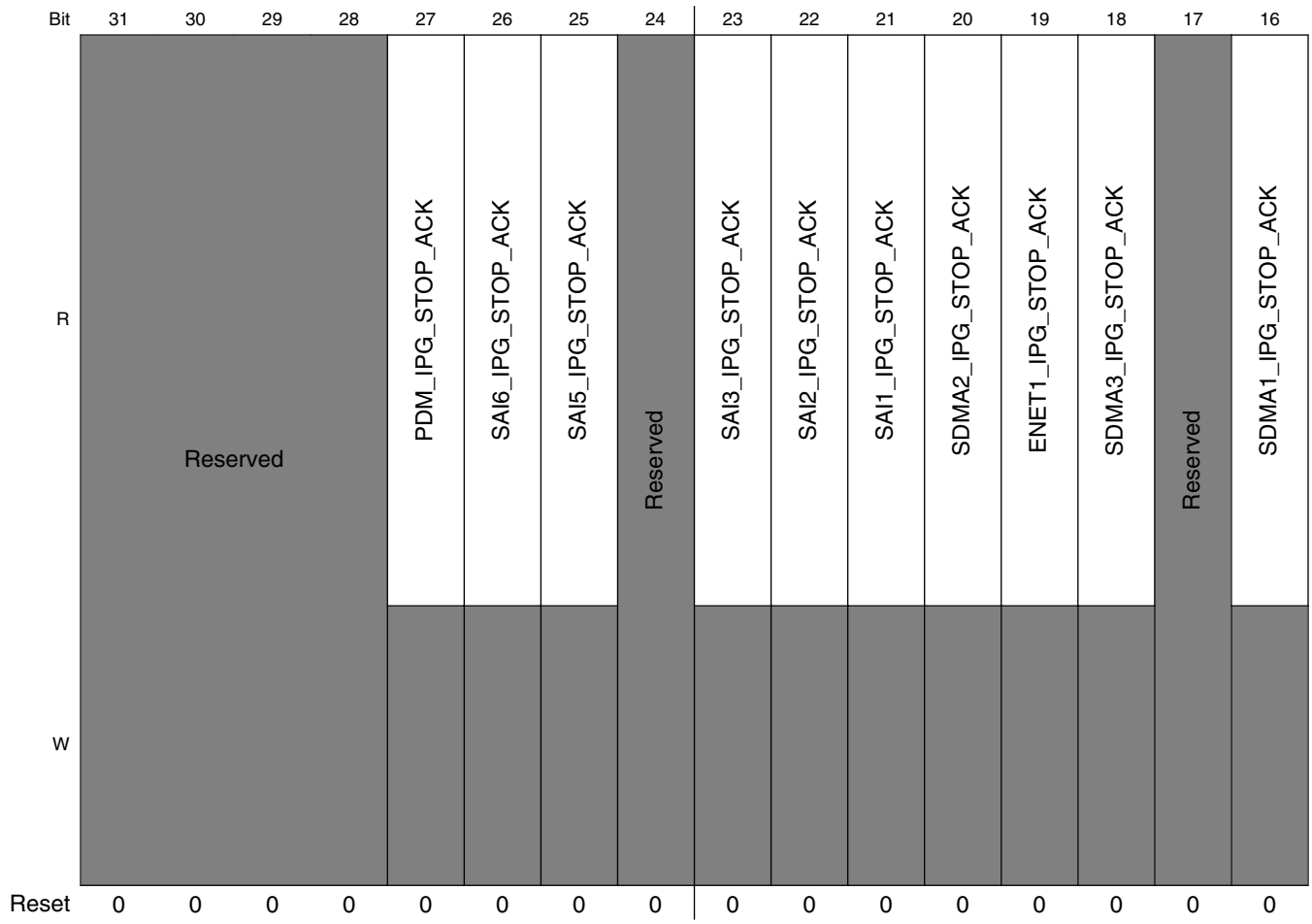
Table continues on the next page...

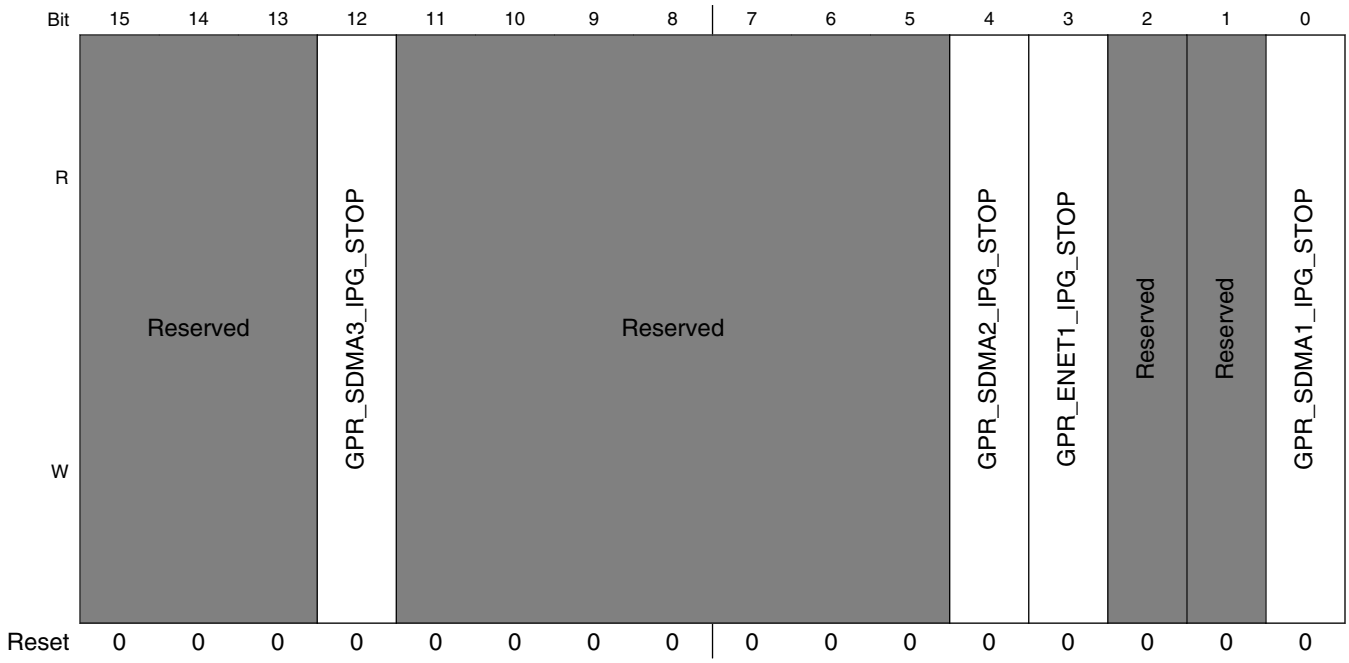
IOMUXC_GPR_GPR3 field descriptions (continued)

Field	Description
2 OCRAM_CTRL_ WRITE_DATA_ PIPELINE_EN	Write data pipeline enable.
1 OCRAM_CTRL_ READ_ADDR_ PIPELINE_EN	Read address pipeline enable.
0 OCRAM_CTRL_ READ_DATA_ WAIT_EN	Read data wait state control.

8.2.4.5 General Purpose Register 4 (IOMUXC_GPR_GPR4)

Address: 3034_0000h base + 10h offset = 3034_0010h





IOMUXC_GPR_GPR4 field descriptions

Field	Description
31–28 -	This field is reserved.
27 PDM_IPG_ STOP_ACK	PDM stop acknowledge.
26 SAI6_IPG_ STOP_ACK	SAI6 stop acknowledge.
25 SAI5_IPG_ STOP_ACK	SAI5 stop acknowledge.
24 -	This field is reserved.
23 SAI3_IPG_ STOP_ACK	SAI3 stop acknowledge.
22 SAI2_IPG_ STOP_ACK	SAI2 stop acknowledge.
21 SAI1_IPG_ STOP_ACK	SAI1 stop acknowledge.
20 SDMA2_IPG_ STOP_ACK	SDMA2 stop acknowledge.

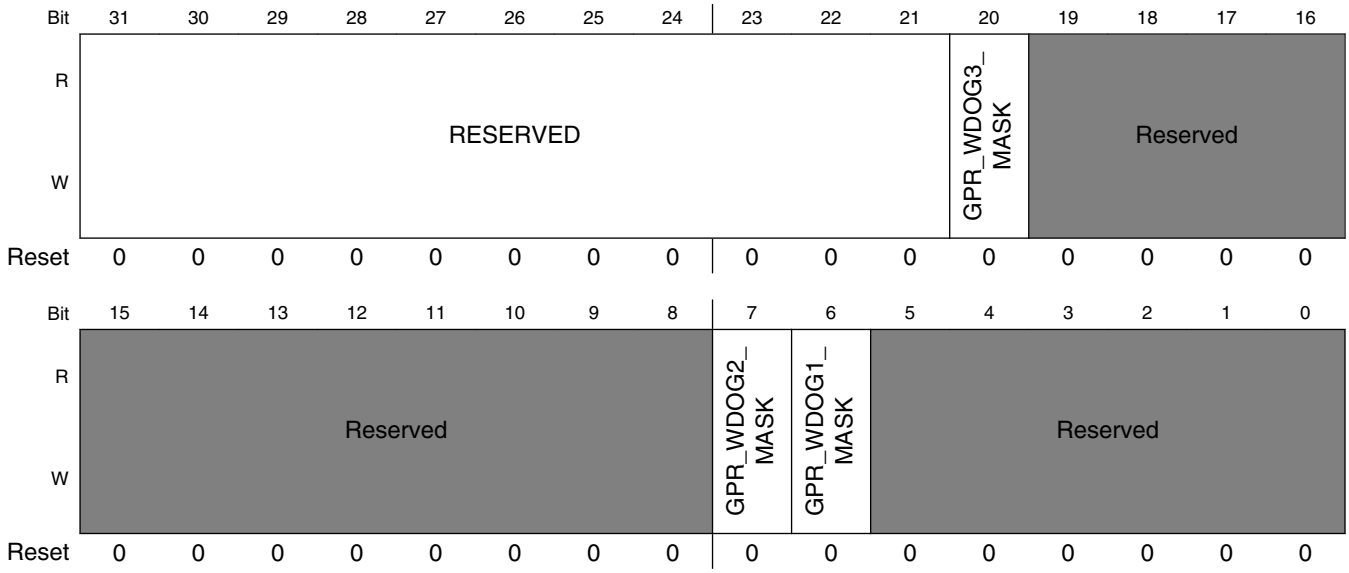
Table continues on the next page...

IOMUXC_GPR_GPR4 field descriptions (continued)

Field	Description
19 ENET1_IPG_ STOP_ACK	ENET1 stop acknowledge.
18 SDMA3_IPG_ STOP_ACK	SDMA3 stop acknowledge.
17 -	This field is reserved.
16 SDMA1_IPG_ STOP_ACK	SDMA1 stop acknowledge.
15–13 -	This field is reserved.
12 GPR_SDMA3_ IPG_STOP	SDMA3 stop request.
11–5 -	This field is reserved.
4 GPR_SDMA2_ IPG_STOP	SDMA2 stop request.
3 GPR_ENET1_ IPG_STOP	ENET1 stop request.
2 -	This field is reserved.
1 -	This field is reserved.
0 GPR_SDMA1_ IPG_STOP	SDMA1 stop request.

8.2.4.6 General Purpose Register 5 (IOMUXC_GPR_GPR5)

Address: 3034_0000h base + 14h offset = 3034_0014h

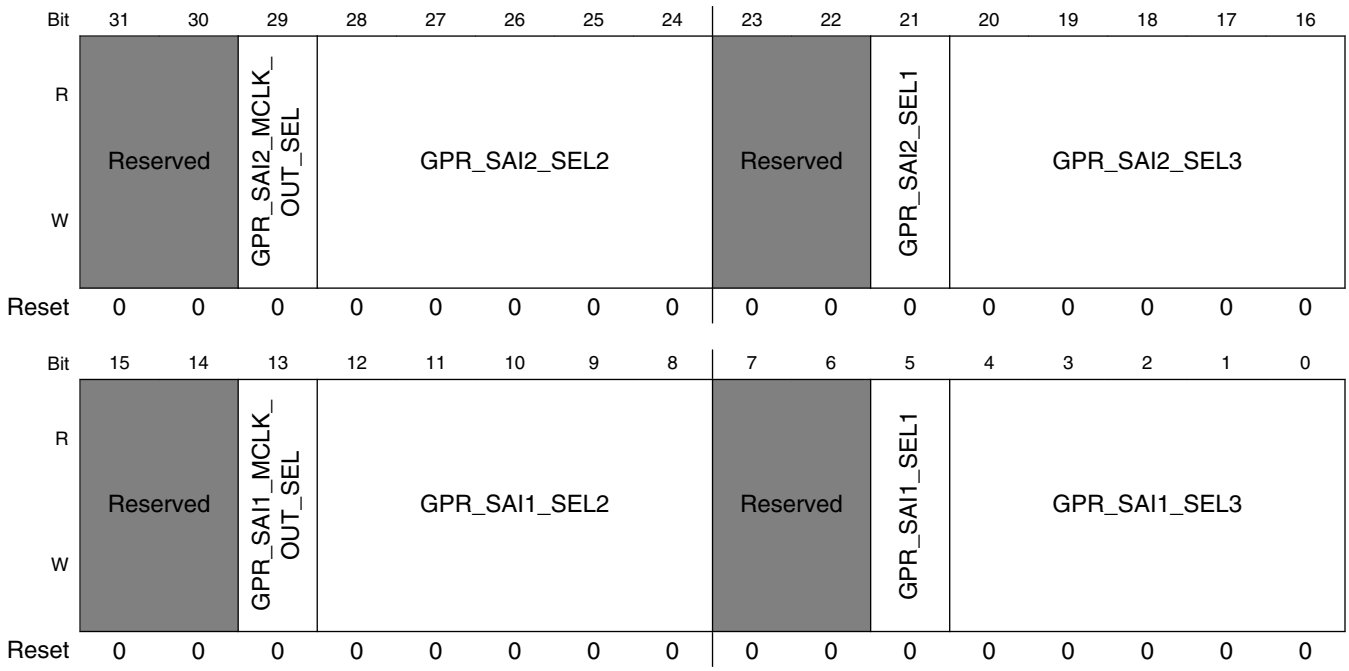


IOMUXC_GPR_GPR5 field descriptions

Field	Description
31–21 RESERVED	This field is reserved.
20 GPR_WDOG3_MASK	This bit is only used to mask the internal WDOG3 int signal for the output of GPIO1_IO02.ALT5_OUT, which is combined with WDOG1/2/3. 0 WDOG3 low will make the GPIO1_IO02.ALT5_OUT low 1 WDOG3 low will not impact the GPIO1_IO02.ALT5_OUT
19–8 -	This field is reserved.
7 GPR_WDOG2_MASK	This bit is only used to mask the internal WDOG2 int signal for the output of GPIO1_IO02.ALT5_OUT, which is combined with WDOG1/2/3. 0 WDOG2 low will make the GPIO1_IO02.ALT5_OUT low 1 WDOG2 low will not impact the GPIO1_IO02.ALT5_OUT
6 GPR_WDOG1_MASK	Normally, WDOG1 output is in GPIO1_IO02.ALT1_OUT. This bit is only used to mask the internal WDOG1 int signal for the output of GPIO1_IO02.ALT5_OUT, which is combined with WDOG1/2/3. 0 WDOG1 low will make the GPIO1_IO02.ALT5_OUT low 1 WDOG1 low will not impact the GPIO1_IO02.ALT5_OUT
-	This field is reserved.

8.2.4.7 General Purpose Register 6 (IOMUXC_GPR_GPR6)

Address: 3034_0000h base + 18h offset = 3034_0018h



IOMUXC_GPR_GPR6 field descriptions

Field	Description
31–30 -	This field is reserved.
29 GPR_SAI2_ MCLK_OUT_SEL	Please refer to SAI1 control bit.
28–24 GPR_SAI2_SEL2	Please refer to SAI1 control bit.
23–22 -	This field is reserved.
21 GPR_SAI2_SEL1	Please refer to SAI1 control bit.
20–16 GPR_SAI2_SEL3	Please refer to SAI1 control bit.
15–14 -	This field is reserved.
13 GPR_SAI1_ MCLK_OUT_SEL	SAIn MCLK output select. 1'd0 SAIn_CLK_ROOT 1'd1 IPP_DO_SAI_MCLK of SAIn
12–8 GPR_SAI1_SEL2	IPG_CLK_SAI_MCLK[2] of SAIn source select. 5'd0 SAI1_CLK_ROOT

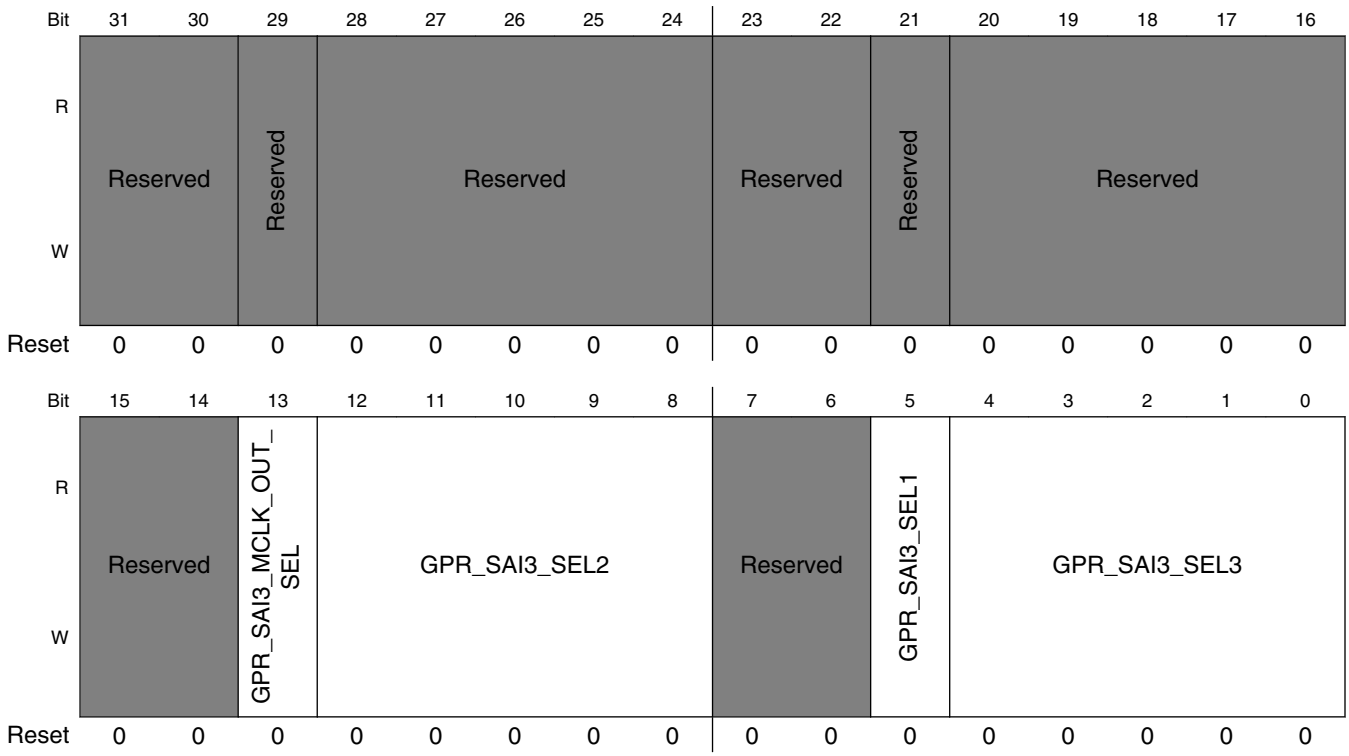
Table continues on the next page...

IOMUXC_GPR_GPR6 field descriptions (continued)

Field	Description
	5'd1 SAI2_CLK_ROOT 5'd2 SAI3_CLK_ROOT 5'd3 Reserved 5'd4 SAI5_CLK_ROOT 5'd5 SAI6_CLK_ROOT 5'd6 SAI1.MCLK 5'd7 SAI2.MCLK 5'd8 SAI3.MCLK 5'd9 Reserved 5'd10 SAI5.MCLK 5'd11 SAI6.MCLK 5'd12 SPDIF1_CLK_ROOT 5'd13 Reserved 5'd14 SPDIF1.EXTCLK 5'd15 SPDIF1.SRCCLK 5'd16 SPDIF1.OUTCLK 5'd17 Reserved
7–6 -	This field is reserved.
5 GPR_SAI1_SEL1	IPG_CLK_SAI_MCLK of SAIn source select. 1'd0 SAIn_CLK_ROOT 1'd1 SAIn.MCLK
GPR_SAI1_SEL3	IPG_CLK_SAI_MCLK[3] of SAIn source select. 5'd0 SAI1_CLK_ROOT 5'd1 SAI2_CLK_ROOT 5'd2 SAI3_CLK_ROOT 5'd3 Reserved 5'd4 SAI5_CLK_ROOT 5'd5 SAI6_CLK_ROOT 5'd6 SAI1.MCLK 5'd7 SAI2.MCLK 5'd8 SAI3.MCLK 5'd9 Reserved 5'd10 SAI5.MCLK 5'd11 SAI6.MCLK 5'd12 SPDIF1_CLK_ROOT 5'd13 Reserved 5'd14 SPDIF1.EXTCLK 5'd15 SPDIF1.SRCCLK 5'd16 SPDIF1.OUTCLK 5'd17 Reserved

8.2.4.8 General Purpose Register 7 (IOMUXC_GPR_GPR7)

Address: 3034_0000h base + 1Ch offset = 3034_001Ch



IOMUXC_GPR_GPR7 field descriptions

Field	Description
31–30 -	This field is reserved.
29 -	This field is reserved.
28–24 -	This field is reserved.
23–22 -	This field is reserved.
21 -	This field is reserved.
20–16 -	This field is reserved.
15–14 -	This field is reserved.
13 GPR_SAI3_ MCLK_OUT_SEL	Please refer to SAI1 control bit.
12–8 GPR_SAI3_SEL2	Please refer to SAI1 control bit.

Table continues on the next page...

IOMUXC_GPR_GPR7 field descriptions (continued)

Field	Description
7–6 -	This field is reserved.
5 GPR_SAI3_SEL1	Please refer to SAI1 control bit.
GPR_SAI3_SEL3	Please refer to SAI1 control bit.

8.2.4.9 General Purpose Register 8 (IOMUXC_GPR_GPR8)

Address: 3034_0000h base + 20h offset = 3034_0020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		GPR_SAI6_MCLK_OUT_SEL	GPR_SAI6_SEL2					Reserved		GPR_SAI6_SEL1	GPR_SAI6_SEL3				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		GPR_SAI5_MCLK_OUT_SEL	GPR_SAI5_SEL2					Reserved		GPR_SAI5_SEL1	GPR_SAI5_SEL3				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_GPR_GPR8 field descriptions

Field	Description
31–30 -	This field is reserved.
29 GPR_SAI6_ MCLK_OUT_SEL	Please refer to SAI1 control bit.
28–24 GPR_SAI6_SEL2	Please refer to SAI1 control bit.
23–22 -	This field is reserved.
21 GPR_SAI6_SEL1	Please refer to SAI1 control bit.

Table continues on the next page...

IOMUXC_GPR_GPR8 field descriptions (continued)

Field	Description
20–16 GPR_SAI6_SEL3	Please refer to SAI1 control bit.
15–14 -	This field is reserved.
13 GPR_SAI5_ MCLK_OUT_SEL	Please refer to SAI1 control bit.
12–8 GPR_SAI5_SEL2	Please refer to SAI1 control bit.
7–6 -	This field is reserved.
5 GPR_SAI5_SEL1	Please refer to SAI1 control bit.
GPR_SAI5_SEL3	Please refer to SAI1 control bit.

8.2.4.10 General Purpose Register 9 (IOMUXC_GPR_GPR9)

Address: 3034_0000h base + 24h offset = 3034_0024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_GPR_GPR9 field descriptions

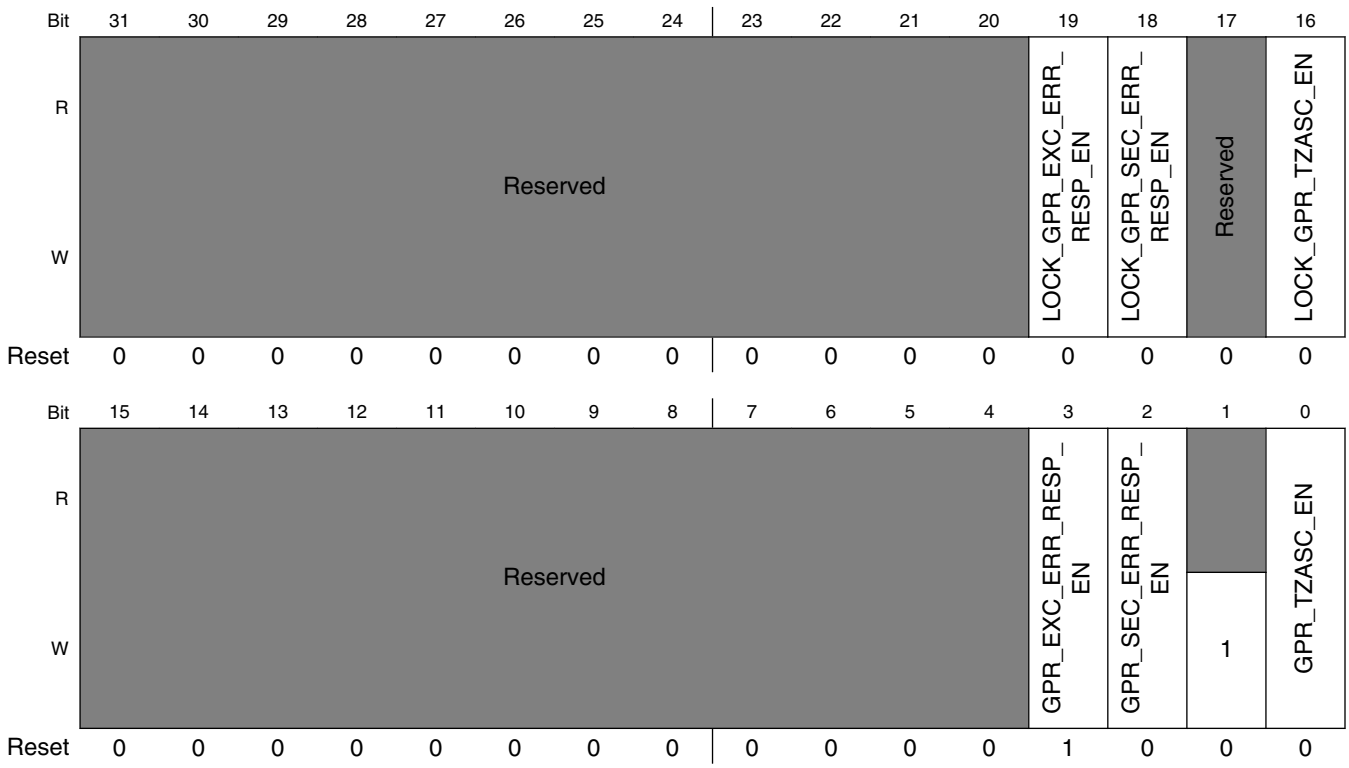
Field	Description
-	This field is reserved.

8.2.4.11 General Purpose Register 10 (IOMUXC_GPR_GPR10)

NOTE

Set GPR10[1] to 1 when TZASC_EN is enabled.

Address: 3034_0000h base + 28h offset = 3034_0028h



IOMUXC_GPR_GPR10 field descriptions

Field	Description
31–20 -	This field is reserved.
19 LOCK_GPR_EXC_ERR_RESP_EN	This is a "sticky" type bit. Lock bit for GPR_EXC_ERR_RESP_EN.
18 LOCK_GPR_SEC_ERR_RESP_EN	This is a "sticky" type bit. Lock bit for GPR_SEC_ERR_RESP_EN.
17 -	This field is reserved. Reserved

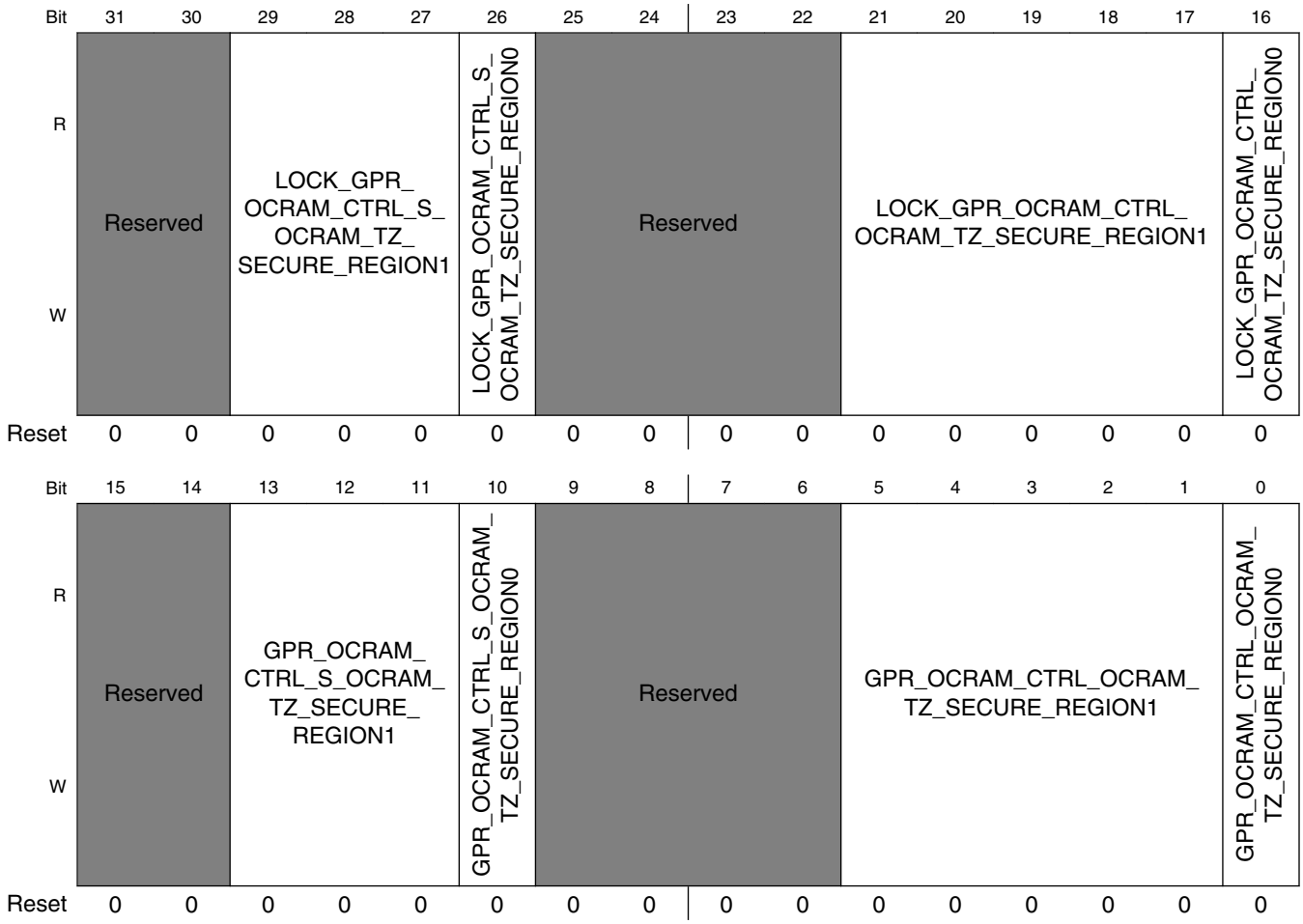
Table continues on the next page...

IOMUXC_GPR_GPR10 field descriptions (continued)

Field	Description
16 LOCK_GPR_ TZASC_EN	This is a "sticky" type bit. Lock bit for GPR_TZASC_EN.
15–4 -	This field is reserved.
3 GPR_EXC_ ERR_RESP_EN	This is a "lock" type bit.
2 GPR_SEC_ ERR_RESP_EN	This is a "lock" type bit.
1 -	Reserved. Set this bit to 1.
0 GPR_TZASC_EN	This is a "lock" type bit. Connect to TZASC_EN input on TZASC_ID_WRAP. **NOTE:** Ensure GPR10[1] is set to 1 when TZASC_EN is enabled. 0 Do not use the TZASC module. All transactions are routed around the TZASC block. 1 Enable the TZASC module. All transactions are processed by this block as per the TZASC TRM describes.

8.2.4.12 General Purpose Register 11 (IOMUXC_GPR_GPR11)

Address: 3034_0000h base + 2Ch offset = 3034_002Ch



IOMUXC_GPR_GPR11 field descriptions

Field	Description
31–30 -	This field is reserved.
29–27 LOCK_GPR_OCRAM_CTRL_S_OCRAM_TZ_SECURE_REGION1	This is a "sticky" type bit. Lock bit for GPR_OCRAM_CTRL_S_OCRAM_TZ_SECURE_REGION1[13:11].
26 LOCK_GPR_OCRAM_CTRL_S_OCRAM_TZ_SECURE_REGION0	This is a "sticky" type bit. Lock bit for GPR_OCRAM_CTRL_S_OCRAM_TZ_SECURE_REGION0[10].

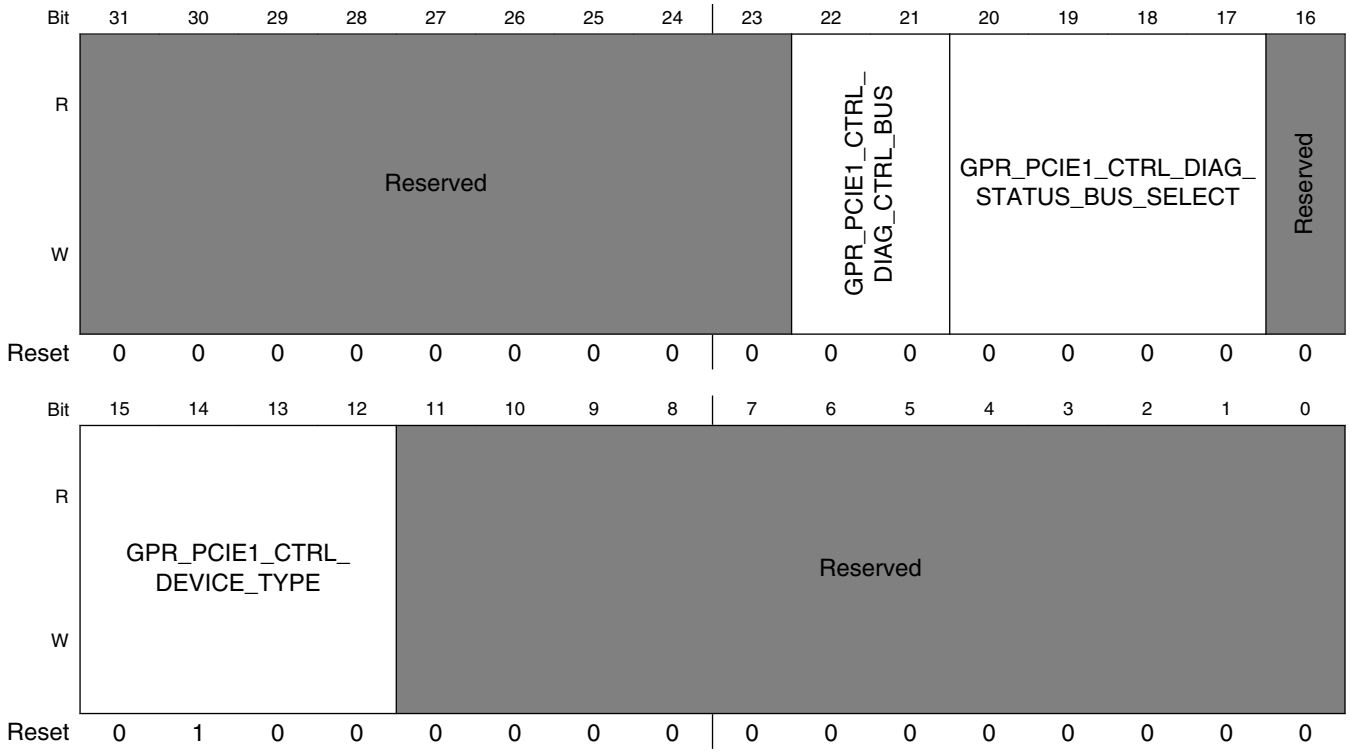
Table continues on the next page...

IOMUXC_GPR_GPR11 field descriptions (continued)

Field	Description
25–22 -	This field is reserved.
21–17 LOCK_GPR_OCRAM_CTRL_OCRAM_TZ_SECURE_REGION1	This is a "sticky" type bit. Lock bit for GPR_OCRAM_CTRL_OCRAM_TZ_SECURE_REGION1[5:1].
16 LOCK_GPR_OCRAM_CTRL_OCRAM_TZ_SECURE_REGION0	This is a "sticky" type bit. Lock bit for GPR_OCRAM_CTRL_OCRAM_TZ_SECURE_REGION0[0].
15–14 -	This field is reserved.
13–11 GPR_OCRAM_CTRL_S_OCRAM_TZ_SECURE_REGION1	This is a "lock" type bit.
10 GPR_OCRAM_CTRL_S_OCRAM_TZ_SECURE_REGION0	This is a "lock" type bit.
9–6 -	This field is reserved.
5–1 GPR_OCRAM_CTRL_OCRAM_TZ_SECURE_REGION1	This is a "lock" type bit.
0 GPR_OCRAM_CTRL_OCRAM_TZ_SECURE_REGION0	This is a "lock" type bit.

8.2.4.13 General Purpose Register 12 (IOMUXC_GPR_GPR12)

Address: 3034_0000h base + 30h offset = 3034_0030h

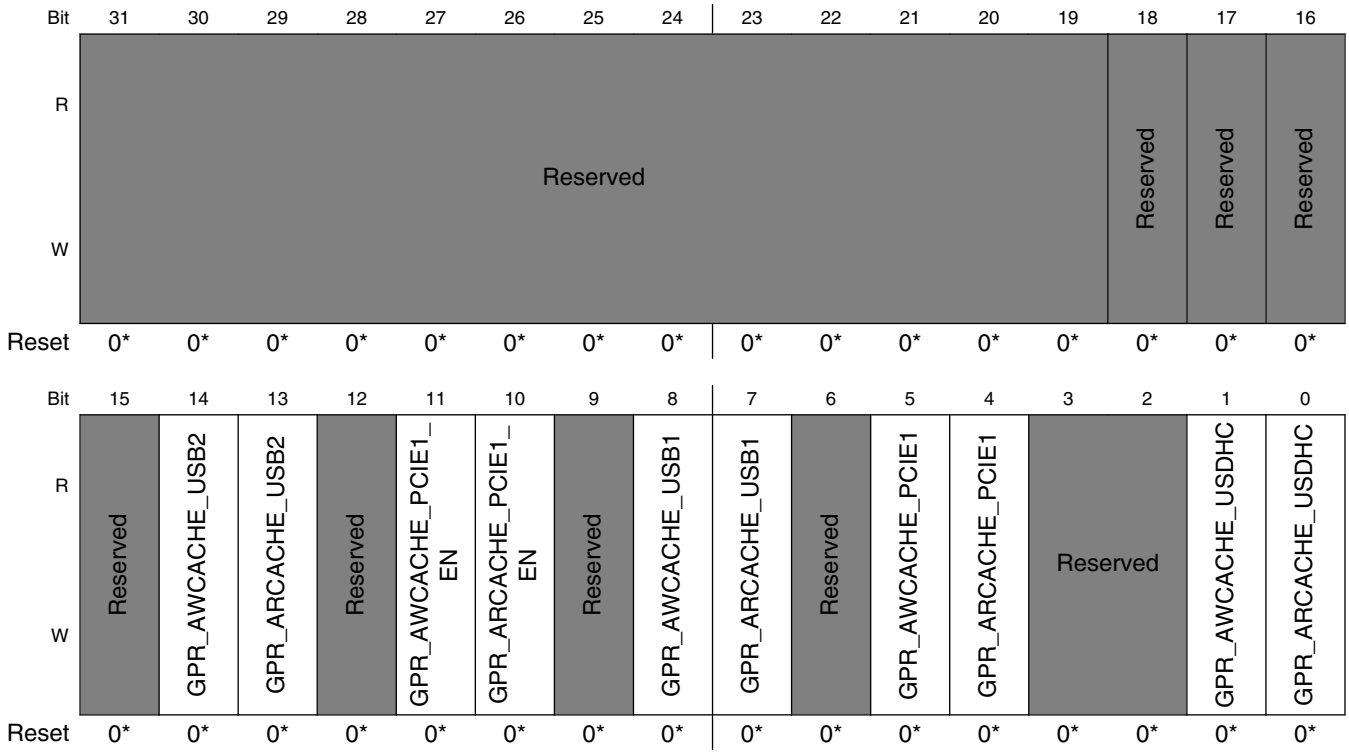


IOMUXC_GPR_GPR12 field descriptions

Field	Description
31–23 -	This field is reserved.
22–21 GPR_PCIE1_CTRL_DIAG_CTRL_BUS	PCIe Diagnostic Control Bus.
20–17 GPR_PCIE1_CTRL_DIAG_STATUS_BUS_SELECT	PCIe Diagnostic Status Bus Select.
16 -	This field is reserved.
15–12 GPR_PCIE1_CTRL_DEVICE_TYPE	PCIe device/port type.
-	This field is reserved.

8.2.4.14 General Purpose Register 13 (IOMUXC_GPR_GPR13)

Address: 3034_0000h base + 34h offset = 3034_0034h



* Notes:
- For reset:

  [31:19] - N/A

  [18:0] - 19'b0

IOMUXC_GPR_GPR13 field descriptions

Field	Description
31–19 -	This field is reserved.
18 -	This field is reserved.
17 -	This field is reserved.
16 -	This field is reserved.
15 -	This field is reserved.

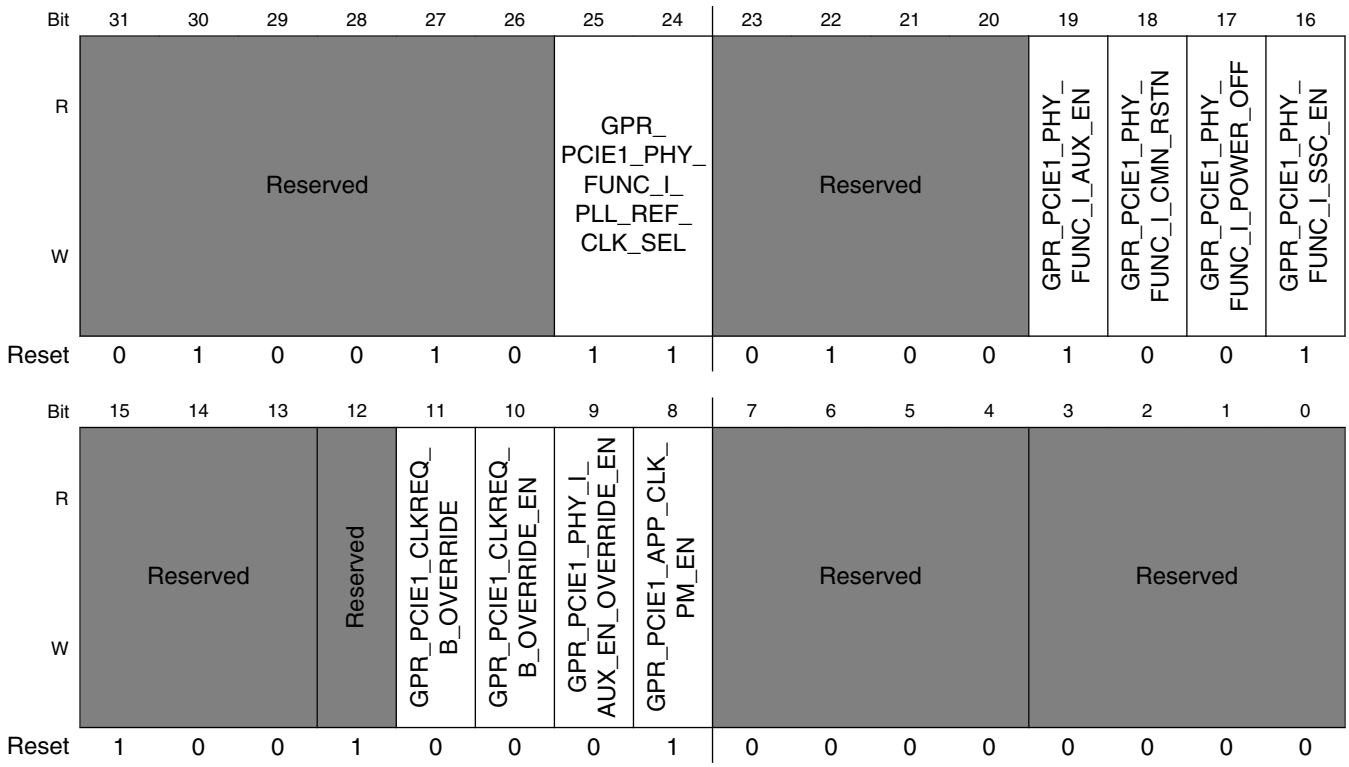
Table continues on the next page...

IOMUXC_GPR_GPR13 field descriptions (continued)

Field	Description
14 GPR_ AWCACHE_ USB2	Control the AWCACHE[1] of USB2 master transaction.
13 GPR_ ARCACHE_ USB2	Control the ARCACHE[1] of USB2 master transaction.
12 -	This field is reserved.
11 GPR_ AWCACHE_ PCIE1_EN	Enable the GPR control of AWCACHE[1] of PCIE1 master transaction.
10 GPR_ ARCACHE_ PCIE1_EN	Enable the GPR control of ARCACHE[1] of PCIE1 master transaction.
9 -	This field is reserved.
8 GPR_ AWCACHE_ USB1	Control the AWCACHE[1] of USB1 master transaction.
7 GPR_ ARCACHE_ USB1	Control the ARCACHE[1] of USB1 master transaction.
6 -	This field is reserved.
5 GPR_ AWCACHE_ PCIE1	Control the AWCACHE[1] of PCIE1 master transaction.
4 GPR_ ARCACHE_ PCIE1	Control the ARCACHE[1] of PCIE1 master transaction.
3–2 -	This field is reserved.
1 GPR_ AWCACHE_ USDHC	Control the AWCACHE[1] of USDHC master transaction.
0 GPR_ ARCACHE_ USDHC	Control the ARCACHE[1] of USDHC master transaction.

8.2.4.15 General Purpose Register 14 (IOMUXC_GPR_GPR14)

Address: 3034_0000h base + 38h offset = 3034_0038h



IOMUXC_GPR_GPR14 field descriptions

Field	Description
31–26 -	This field is reserved.
25–24 GPR_PCIE1_ PHY_FUNC_I_ PLL_REF_CLK_ SEL	00 N/A 01 Selects reference clock from XO (pll_refclk_from_xo) 10 Selects reference clock from IO (ext_ref_clkp/n) 11 Selects reference clock from SOC PLL (pll_refclk_from_syspll)
23–20 -	This field is reserved.
19 GPR_PCIE1_ PHY_FUNC_I_ AUX_EN	External Reference Clock I/O (for PLL) Enable Signal. **NOTE:** Please see bit field 9 for more details. 1 Enable 0 Disable
18 GPR_PCIE1_ PHY_FUNC_I_ CMN_RSTN	Resets the PCIe PHY Common Block Reset. This is an Active low reset for PCIe PHY Common Block.

Table continues on the next page...

IOMUXC_GPR_GPR14 field descriptions (continued)

Field	Description
17 GPR_PCIE1_PHY_FUNC_I_POWER_OFF	PMA power down signal. 1 Power Down 0 Power Up
16 GPR_PCIE1_PHY_FUNC_I_SSC_EN	SSC enable signal. 1 Enable 0 Disable
15–13 -	This field is reserved.
12 -	This field is reserved.
11 GPR_PCIE1_CLKREQ_B_OVERRIDE	Control PCIE_CLKREQ_B to the pad together with CLKREQ_B from controller.
10 GPR_PCIE1_CLKREQ_B_OVERRIDE_EN	Control PCIE_CLKREQ_B to the pad together with CLKREQ_B from controller.
9 GPR_PCIE1_PHY_I_AUX_EN_OVERRIDE_EN	{GPR_PCIE1_PHY_I_AUX_EN_OVERRIDE_EN, GPR_PCIE1_PHY_FUNC_I_AUX_EN} 2'b00 External Reference Clock I/O (for PLL) Disable 2'b01 External Reference Clock I/O (for PLL) Enable 2'b10 External Reference Clock I/O (for PLL) Disable 2'b11 External Reference Clock I/O (for PLL) output is controlled by CLKREQ#
8 GPR_PCIE1_APP_CLK_PM_EN	To PCIe CTRL.
7–4 -	This field is reserved.
-	This field is reserved.

8.2.4.16 General Purpose Register 15 (IOMUXC_GPR_GPR15)

Address: 3034_0000h base + 3Ch offset = 3034_003Ch

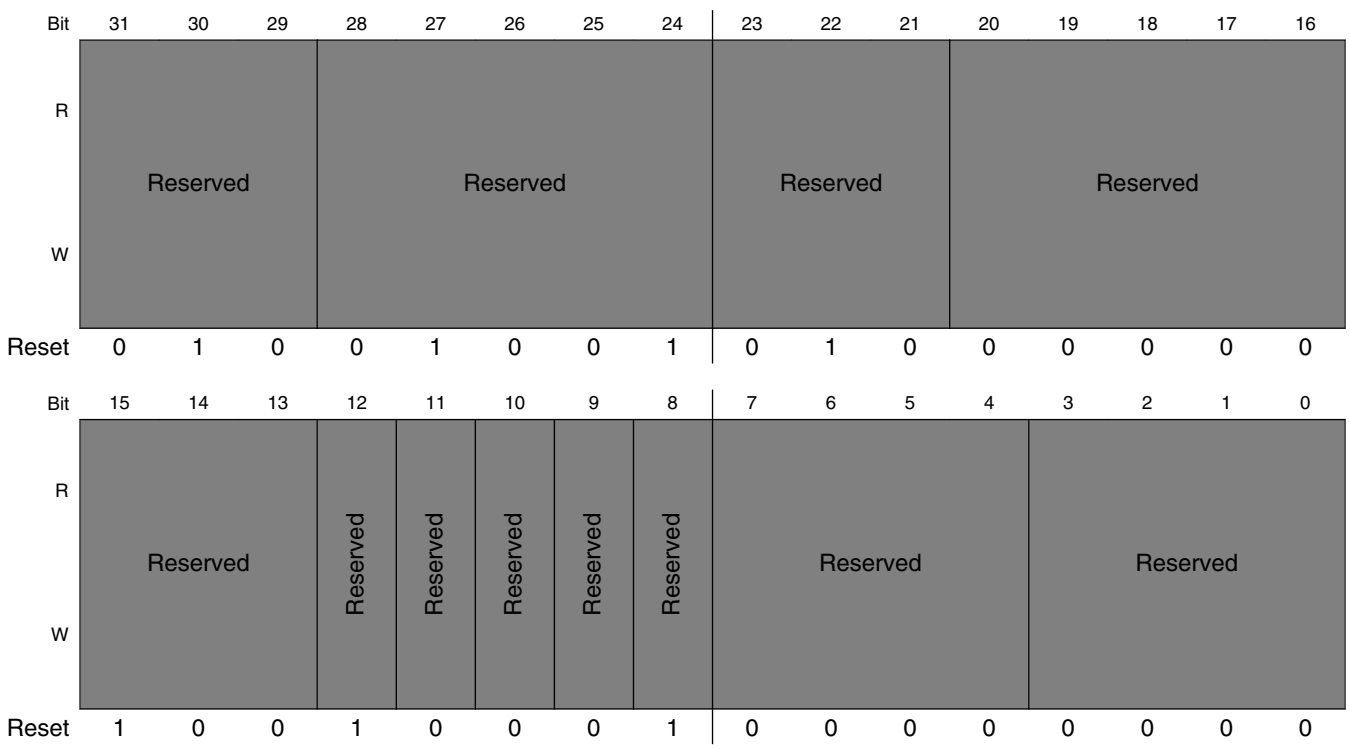
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						Reserved						Reserved				Reserved									Reserved							
Reset	0	1	1	0	0	0	0	1	1	0	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

IOMUXC_GPR_GPR15 field descriptions

Field	Description
31–26 -	This field is reserved.
25–20 -	This field is reserved.
19–14 -	This field is reserved.
13–7 -	This field is reserved.
-	This field is reserved.

8.2.4.17 General Purpose Register 16 (IOMUXC_GPR_GPR16)

Address: 3034_0000h base + 40h offset = 3034_0040h



IOMUXC_GPR_GPR16 field descriptions

Field	Description
31–29 -	This field is reserved.
28–24 -	This field is reserved.
23–21 -	This field is reserved.

Table continues on the next page...

IOMUXC_GPR_GPR16 field descriptions (continued)

Field	Description
20–16 -	This field is reserved.
15–13 -	This field is reserved.
12 -	This field is reserved.
11 -	This field is reserved.
10 -	This field is reserved.
9 -	This field is reserved.
8 -	This field is reserved.
7–4 -	This field is reserved.
-	This field is reserved.

8.2.4.18 General Purpose Register 17 (IOMUXC_GPR_GPR17)

Address: 3034_0000h base + 44h offset = 3034_0044h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						Reserved						Reserved				Reserved									Reserved							
Reset	0	1	1	0	0	0	0	1	1	0	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

IOMUXC_GPR_GPR17 field descriptions

Field	Description
31–26 -	This field is reserved.
25–20 -	This field is reserved.
19–14 -	This field is reserved.
13–7 -	This field is reserved.
-	This field is reserved.

8.2.4.19 General Purpose Register 18 (IOMUXC_GPR_GPR18)

Address: 3034_0000h base + 48h offset = 3034_0048h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_GPR_GPR18 field descriptions

Field	Description
-	This field is reserved.

8.2.4.20 General Purpose Register 19 (IOMUXC_GPR_GPR19)

Address: 3034_0000h base + 4Ch offset = 3034_004Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCIE_DIAG_STATUS																															
W																																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

* Notes:
- For reset:

  [31:0] - N/A

IOMUXC_GPR_GPR19 field descriptions

Field	Description
PCIE_DIAG_ STATUS	From PCIE.

8.2.4.21 General Purpose Register 20 (IOMUXC_GPR_GPR20)

Address: 3034_0000h base + 50h offset = 3034_0050h

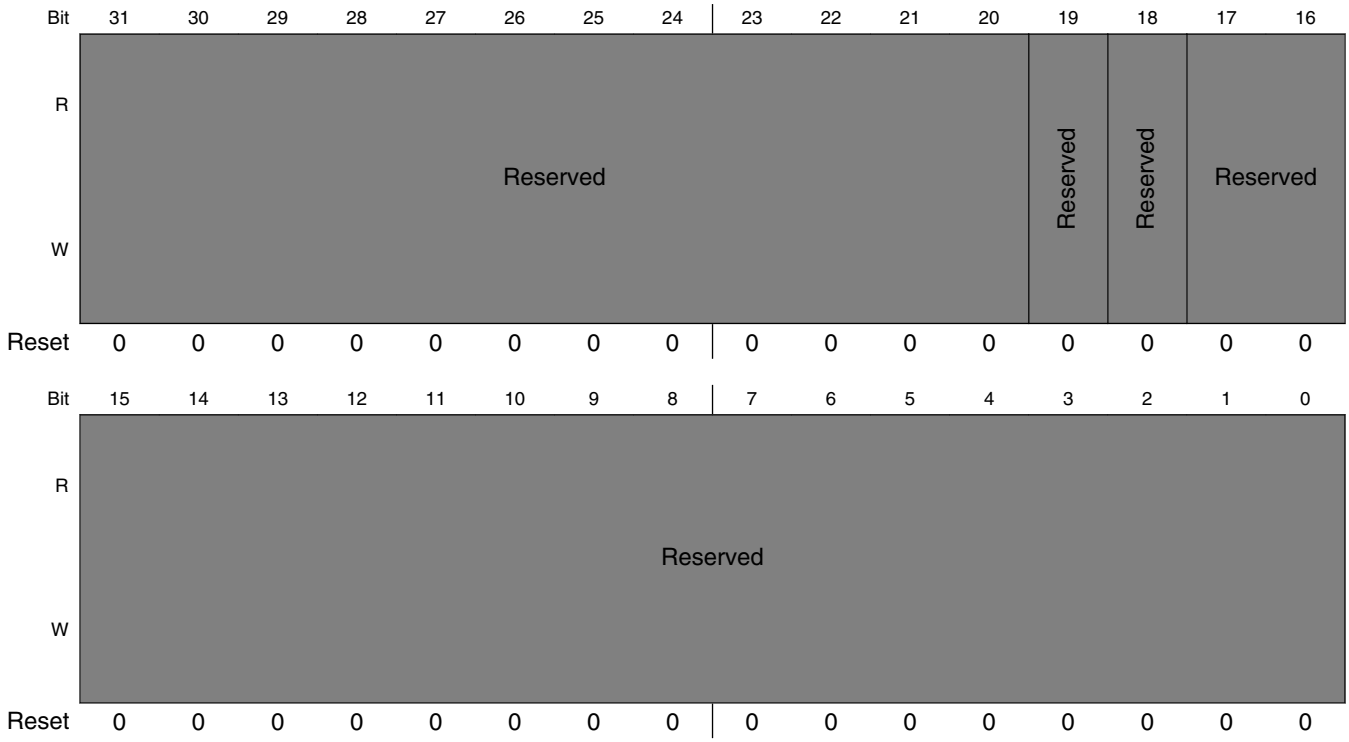
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_GPR_GPR20 field descriptions

Field	Description
-	This field is reserved.

8.2.4.22 General Purpose Register 21 (IOMUXC_GPR_GPR21)

Address: 3034_0000h base + 54h offset = 3034_0054h



IOMUXC_GPR_GPR21 field descriptions

Field	Description
31–20 -	This field is reserved.
19 -	This field is reserved.
18 -	This field is reserved.
- 	This field is reserved.

8.2.4.23 General Purpose Register 22 (IOMUXC_GPR_GPR22)

Address: 3034_0000h base + 58h offset = 3034_0058h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								CPU_STANDBYWFE				CPU_STANDBYWFI				Reserved															
W																																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

* Notes:
- For reset:

  [31:16] - N/A

  [15:0] - 16'b0

IOMUXC_GPR_GPR22 field descriptions

Field	Description
31–24 -	This field is reserved.
23–20 CPU_ STANDBYWFE	From CA53.
19–16 CPU_ STANDBYWFI	From CA53.
-	This field is reserved.

8.2.4.24 General Purpose Register (IOMUXC_GPR_GPR*n*)

Reserved.

Address: 3034_0000h base + 5Ch offset + (4d × i), where i=0d to 24d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	-																															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

IOMUXC_GPR_GPR*n* field descriptions

Field	Description
-	Reserved.

8.2.5 IOMUXC Memory Map/Register Definition

IOMUXC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_0014	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_PMIC_STBY_REQ)	32	R/W	0000_0000h	8.2.5.1/ 1299
3033_0018	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_PMIC_ON_REQ)	32	R/W	0000_0000h	8.2.5.2/ 1300
3033_001C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ONOFF)	32	R/W	0000_0000h	8.2.5.3/ 1300
3033_0020	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_POR_B)	32	R/W	0000_0000h	8.2.5.4/ 1301
3033_0024	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_RTC_RESET_B)	32	R/W	0000_0000h	8.2.5.5/ 1301
3033_0028	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO00)	32	R/W	0000_0000h	8.2.5.6/ 1302
3033_002C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO01)	32	R/W	0000_0000h	8.2.5.7/ 1303
3033_0030	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO02)	32	R/W	0000_0000h	8.2.5.8/ 1304
3033_0034	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO03)	32	R/W	0000_0000h	8.2.5.9/ 1305
3033_0038	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO04)	32	R/W	0000_0000h	8.2.5.10/ 1306
3033_003C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO05)	32	R/W	0000_0000h	8.2.5.11/ 1307
3033_0040	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO06)	32	R/W	0000_0000h	8.2.5.12/ 1308
3033_0044	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO07)	32	R/W	0000_0000h	8.2.5.13/ 1309
3033_0048	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO08)	32	R/W	0000_0000h	8.2.5.14/ 1310
3033_004C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO09)	32	R/W	0000_0000h	8.2.5.15/ 1311
3033_0050	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO10)	32	R/W	0000_0000h	8.2.5.16/ 1312
3033_0054	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO11)	32	R/W	0000_0000h	8.2.5.17/ 1313

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_0058	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO12)	32	R/W	0000_0000h	8.2.5.18/ 1314
3033_005C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO13)	32	R/W	0000_0000h	8.2.5.19/ 1315
3033_0060	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO14)	32	R/W	0000_0000h	8.2.5.20/ 1316
3033_0064	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO15)	32	R/W	0000_0000h	8.2.5.21/ 1317
3033_0068	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_MDC)	32	R/W	0000_0005h	8.2.5.22/ 1318
3033_006C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_MDIO)	32	R/W	0000_0005h	8.2.5.23/ 1319
3033_0070	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_TD3)	32	R/W	0000_0005h	8.2.5.24/ 1320
3033_0074	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_TD2)	32	R/W	0000_0005h	8.2.5.25/ 1321
3033_0078	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_TD1)	32	R/W	0000_0005h	8.2.5.26/ 1322
3033_007C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_TD0)	32	R/W	0000_0005h	8.2.5.27/ 1323
3033_0080	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_TX_CTL)	32	R/W	0000_0005h	8.2.5.28/ 1324
3033_0084	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_TXC)	32	R/W	0000_0005h	8.2.5.29/ 1325
3033_0088	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_RX_CTL)	32	R/W	0000_0005h	8.2.5.30/ 1326
3033_008C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_RXC)	32	R/W	0000_0005h	8.2.5.31/ 1327
3033_0090	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_RD0)	32	R/W	0000_0005h	8.2.5.32/ 1328
3033_0094	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_RD1)	32	R/W	0000_0005h	8.2.5.33/ 1329
3033_0098	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_RD2)	32	R/W	0000_0005h	8.2.5.34/ 1330
3033_009C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_RD3)	32	R/W	0000_0005h	8.2.5.35/ 1331
3033_00A0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_CLK)	32	R/W	0000_0005h	8.2.5.36/ 1332
3033_00A4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_CMD)	32	R/W	0000_0005h	8.2.5.37/ 1333
3033_00A8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0)	32	R/W	0000_0005h	8.2.5.38/ 1334
3033_00AC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1)	32	R/W	0000_0005h	8.2.5.39/ 1335

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_00B0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2)	32	R/W	0000_0005h	8.2.5.40/ 1336
3033_00B4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3)	32	R/W	0000_0005h	8.2.5.41/ 1337
3033_00B8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA4)	32	R/W	0000_0005h	8.2.5.42/ 1338
3033_00BC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA5)	32	R/W	0000_0005h	8.2.5.43/ 1339
3033_00C0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA6)	32	R/W	0000_0005h	8.2.5.44/ 1340
3033_00C4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA7)	32	R/W	0000_0005h	8.2.5.45/ 1341
3033_00C8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_RESET_B)	32	R/W	0000_0005h	8.2.5.46/ 1342
3033_00CC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_STROBE)	32	R/W	0000_0005h	8.2.5.47/ 1343
3033_00D0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_CD_B)	32	R/W	0000_0005h	8.2.5.48/ 1344
3033_00D4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_CLK)	32	R/W	0000_0005h	8.2.5.49/ 1345
3033_00D8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_CMD)	32	R/W	0000_0005h	8.2.5.50/ 1346
3033_00DC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0)	32	R/W	0000_0005h	8.2.5.51/ 1347
3033_00E0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1)	32	R/W	0000_0005h	8.2.5.52/ 1348
3033_00E4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2)	32	R/W	0000_0005h	8.2.5.53/ 1349
3033_00E8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3)	32	R/W	0000_0005h	8.2.5.54/ 1350
3033_00EC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_RESET_B)	32	R/W	0000_0005h	8.2.5.55/ 1351
3033_00F0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_WP)	32	R/W	0000_0005h	8.2.5.56/ 1352
3033_00F4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_ALE)	32	R/W	0000_0005h	8.2.5.57/ 1353
3033_00F8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_CE0_B)	32	R/W	0000_0005h	8.2.5.58/ 1354
3033_00FC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_CE1_B)	32	R/W	0000_0005h	8.2.5.59/ 1355
3033_0100	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_CE2_B)	32	R/W	0000_0005h	8.2.5.60/ 1356
3033_0104	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_CE3_B)	32	R/W	0000_0005h	8.2.5.61/ 1357

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_0108	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_CLE)	32	R/W	0000_0005h	8.2.5.62/ 1358
3033_010C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA00)	32	R/W	0000_0005h	8.2.5.63/ 1359
3033_0110	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA01)	32	R/W	0000_0005h	8.2.5.64/ 1360
3033_0114	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA02)	32	R/W	0000_0005h	8.2.5.65/ 1361
3033_0118	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA03)	32	R/W	0000_0005h	8.2.5.66/ 1362
3033_011C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA04)	32	R/W	0000_0005h	8.2.5.67/ 1363
3033_0120	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA05)	32	R/W	0000_0005h	8.2.5.68/ 1364
3033_0124	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA06)	32	R/W	0000_0005h	8.2.5.69/ 1365
3033_0128	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA07)	32	R/W	0000_0005h	8.2.5.70/ 1366
3033_012C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DQS)	32	R/W	0000_0005h	8.2.5.71/ 1367
3033_0130	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_RE_B)	32	R/W	0000_0005h	8.2.5.72/ 1368
3033_0134	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_READY_B)	32	R/W	0000_0005h	8.2.5.73/ 1369
3033_0138	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_WE_B)	32	R/W	0000_0005h	8.2.5.74/ 1370
3033_013C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_WP_B)	32	R/W	0000_0005h	8.2.5.75/ 1371
3033_0140	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_RXFS)	32	R/W	0000_0005h	8.2.5.76/ 1372
3033_0144	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_RXC)	32	R/W	0000_0005h	8.2.5.77/ 1373
3033_0148	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD0)	32	R/W	0000_0005h	8.2.5.78/ 1374
3033_014C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD1)	32	R/W	0000_0005h	8.2.5.79/ 1376
3033_0150	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD2)	32	R/W	0000_0005h	8.2.5.80/ 1377
3033_0154	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD3)	32	R/W	0000_0005h	8.2.5.81/ 1378
3033_0158	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_MCLK)	32	R/W	0000_0005h	8.2.5.82/ 1380
3033_015C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXFS)	32	R/W	0000_0005h	8.2.5.83/ 1381

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_0160	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXC)	32	R/W	0000_0005h	8.2.5.84/ 1382
3033_0164	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD0)	32	R/W	0000_0005h	8.2.5.85/ 1383
3033_0168	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD1)	32	R/W	0000_0005h	8.2.5.86/ 1384
3033_016C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD2)	32	R/W	0000_0005h	8.2.5.87/ 1386
3033_0170	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD3)	32	R/W	0000_0005h	8.2.5.88/ 1387
3033_0174	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD4)	32	R/W	0000_0005h	8.2.5.89/ 1388
3033_0178	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD5)	32	R/W	0000_0005h	8.2.5.90/ 1390
3033_017C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD6)	32	R/W	0000_0005h	8.2.5.91/ 1391
3033_0180	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD7)	32	R/W	0000_0005h	8.2.5.92/ 1392
3033_0184	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXFS)	32	R/W	0000_0005h	8.2.5.93/ 1394
3033_0188	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXC)	32	R/W	0000_0005h	8.2.5.94/ 1395
3033_018C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD0)	32	R/W	0000_0005h	8.2.5.95/ 1396
3033_0190	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD1)	32	R/W	0000_0005h	8.2.5.96/ 1397
3033_0194	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD2)	32	R/W	0000_0005h	8.2.5.97/ 1398
3033_0198	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD3)	32	R/W	0000_0005h	8.2.5.98/ 1399
3033_019C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD4)	32	R/W	0000_0005h	8.2.5.99/ 1400
3033_01A0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD5)	32	R/W	0000_0005h	8.2.5.100/ 1402
3033_01A4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD6)	32	R/W	0000_0005h	8.2.5.101/ 1403
3033_01A8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD7)	32	R/W	0000_0005h	8.2.5.102/ 1404
3033_01AC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_MCLK)	32	R/W	0000_0005h	8.2.5.103/ 1405
3033_01B0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_RXFS)	32	R/W	0000_0005h	8.2.5.104/ 1407
3033_01B4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_RXC)	32	R/W	0000_0005h	8.2.5.105/ 1408

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_01B8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_RXD0)	32	R/W	0000_0005h	8.2.5.106/ 1409
3033_01BC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_TXFS)	32	R/W	0000_0005h	8.2.5.107/ 1410
3033_01C0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_TXC)	32	R/W	0000_0005h	8.2.5.108/ 1411
3033_01C4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_TXD0)	32	R/W	0000_0005h	8.2.5.109/ 1412
3033_01C8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_MCLK)	32	R/W	0000_0005h	8.2.5.110/ 1413
3033_01CC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_RXFS)	32	R/W	0000_0005h	8.2.5.111/ 1414
3033_01D0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_RXC)	32	R/W	0000_0005h	8.2.5.112/ 1416
3033_01D4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_RXD)	32	R/W	0000_0005h	8.2.5.113/ 1417
3033_01D8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_TXFS)	32	R/W	0000_0005h	8.2.5.114/ 1418
3033_01DC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_TXC)	32	R/W	0000_0005h	8.2.5.115/ 1419
3033_01E0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_TXD)	32	R/W	0000_0005h	8.2.5.116/ 1421
3033_01E4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_MCLK)	32	R/W	0000_0005h	8.2.5.117/ 1422
3033_01E8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SPDIF_TX)	32	R/W	0000_0005h	8.2.5.118/ 1423
3033_01EC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SPDIF_RX)	32	R/W	0000_0005h	8.2.5.119/ 1424
3033_01F0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SPDIF_EXT_CLK)	32	R/W	0000_0005h	8.2.5.120/ 1425
3033_01F4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI1_SCLK)	32	R/W	0000_0005h	8.2.5.121/ 1426
3033_01F8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI1_MOSI)	32	R/W	0000_0005h	8.2.5.122/ 1427
3033_01FC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI1_MISO)	32	R/W	0000_0005h	8.2.5.123/ 1428
3033_0200	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI1_SS0)	32	R/W	0000_0005h	8.2.5.124/ 1429
3033_0204	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI2_SCLK)	32	R/W	0000_0005h	8.2.5.125/ 1430
3033_0208	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI2_MOSI)	32	R/W	0000_0005h	8.2.5.126/ 1431
3033_020C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI2_MISO)	32	R/W	0000_0005h	8.2.5.127/ 1432

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_0210	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI2_SS0)	32	R/W	0000_0005h	8.2.5.128/ 1433
3033_0214	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C1_SCL)	32	R/W	0000_0005h	8.2.5.129/ 1434
3033_0218	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C1_SDA)	32	R/W	0000_0005h	8.2.5.130/ 1435
3033_021C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C2_SCL)	32	R/W	0000_0005h	8.2.5.131/ 1436
3033_0220	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C2_SDA)	32	R/W	0000_0005h	8.2.5.132/ 1437
3033_0224	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C3_SCL)	32	R/W	0000_0005h	8.2.5.133/ 1438
3033_0228	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C3_SDA)	32	R/W	0000_0005h	8.2.5.134/ 1439
3033_022C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C4_SCL)	32	R/W	0000_0005h	8.2.5.135/ 1440
3033_0230	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C4_SDA)	32	R/W	0000_0005h	8.2.5.136/ 1441
3033_0234	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART1_RXD)	32	R/W	0000_0005h	8.2.5.137/ 1442
3033_0238	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART1_TXD)	32	R/W	0000_0005h	8.2.5.138/ 1443
3033_023C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART2_RXD)	32	R/W	0000_0005h	8.2.5.139/ 1444
3033_0240	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART2_TXD)	32	R/W	0000_0005h	8.2.5.140/ 1445
3033_0244	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART3_RXD)	32	R/W	0000_0005h	8.2.5.141/ 1446
3033_0248	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART3_TXD)	32	R/W	0000_0005h	8.2.5.142/ 1447
3033_024C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART4_RXD)	32	R/W	0000_0005h	8.2.5.143/ 1448
3033_0250	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART4_TXD)	32	R/W	0000_0005h	8.2.5.144/ 1449
3033_0254	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_TEST_MODE)	32	R/W	0000_0101h	8.2.5.145/ 1450
3033_0258	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_BOOT_MODE0)	32	R/W	0000_0081h	8.2.5.146/ 1451
3033_025C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_BOOT_MODE1)	32	R/W	0000_0181h	8.2.5.147/ 1452
3033_0260	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_MOD)	32	R/W	0000_1901h	8.2.5.148/ 1453
3033_0264	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TRST_B)	32	R/W	0000_0141h	8.2.5.149/ 1454

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_0268	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TDI)	32	R/W	0000_0141h	8.2.5.150/ 1454
3033_026C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TMS)	32	R/W	0000_0141h	8.2.5.151/ 1455
3033_0270	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TCK)	32	R/W	0000_0141h	8.2.5.152/ 1456
3033_0274	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TDO)	32	R/W	0000_0141h	8.2.5.153/ 1457
3033_0278	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_RTC)	32	R/W	0000_0000h	8.2.5.154/ 1458
3033_027C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_PMIC_STBY_REQ)	32	R/W	0000_010Ch	8.2.5.155/ 1458
3033_0280	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_PMIC_ON_REQ)	32	R/W	0000_196Ch	8.2.5.156/ 1459
3033_0284	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ONOFF)	32	R/W	0000_01CCh	8.2.5.157/ 1460
3033_0288	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_POR_B)	32	R/W	0000_01CCh	8.2.5.158/ 1461
3033_028C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_RTC_RESET_B)	32	R/W	0000_01CCh	8.2.5.159/ 1462
3033_0290	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO00)	32	R/W	0000_0114h	8.2.5.160/ 1463
3033_0294	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO01)	32	R/W	0000_0116h	8.2.5.161/ 1464
3033_0298	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO02)	32	R/W	0000_0156h	8.2.5.162/ 1465
3033_029C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO03)	32	R/W	0000_0116h	8.2.5.163/ 1466
3033_02A0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO04)	32	R/W	0000_0116h	8.2.5.164/ 1467
3033_02A4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO05)	32	R/W	0000_0156h	8.2.5.165/ 1468
3033_02A8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO06)	32	R/W	0000_0116h	8.2.5.166/ 1470
3033_02AC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO07)	32	R/W	0000_1916h	8.2.5.167/ 1471
3033_02B0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO08)	32	R/W	0000_0116h	8.2.5.168/ 1472
3033_02B4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO09)	32	R/W	0000_0116h	8.2.5.169/ 1473
3033_02B8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO10)	32	R/W	0000_0116h	8.2.5.170/ 1474
3033_02BC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO11)	32	R/W	0000_0116h	8.2.5.171/ 1476

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_02C0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO12)	32	R/W	0000_0116h	8.2.5.172/ 1477
3033_02C4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO13)	32	R/W	0000_0116h	8.2.5.173/ 1478
3033_02C8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO14)	32	R/W	0000_0116h	8.2.5.174/ 1479
3033_02CC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO15)	32	R/W	0000_0116h	8.2.5.175/ 1480
3033_02D0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_MDC)	32	R/W	0000_0116h	8.2.5.176/ 1481
3033_02D4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_MDIO)	32	R/W	0000_0116h	8.2.5.177/ 1483
3033_02D8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_TD3)	32	R/W	0000_0116h	8.2.5.178/ 1484
3033_02DC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_TD2)	32	R/W	0000_0116h	8.2.5.179/ 1485
3033_02E0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_TD1)	32	R/W	0000_0116h	8.2.5.180/ 1486
3033_02E4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_TD0)	32	R/W	0000_0116h	8.2.5.181/ 1487
3033_02E8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_TX_CTL)	32	R/W	0000_1916h	8.2.5.182/ 1488
3033_02EC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_TXC)	32	R/W	0000_0116h	8.2.5.183/ 1490
3033_02F0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_RX_CTL)	32	R/W	0000_0116h	8.2.5.184/ 1491
3033_02F4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_RXC)	32	R/W	0000_0116h	8.2.5.185/ 1492
3033_02F8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_RD0)	32	R/W	0000_0116h	8.2.5.186/ 1493
3033_02FC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_RD1)	32	R/W	0000_0116h	8.2.5.187/ 1494
3033_0300	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_RD2)	32	R/W	0000_0116h	8.2.5.188/ 1495
3033_0304	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_RD3)	32	R/W	0000_0116h	8.2.5.189/ 1497
3033_0308	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_CLK)	32	R/W	0000_0116h	8.2.5.190/ 1498
3033_030C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_CMD)	32	R/W	0000_0116h	8.2.5.191/ 1499
3033_0310	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA0)	32	R/W	0000_0116h	8.2.5.192/ 1500
3033_0314	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA1)	32	R/W	0000_0116h	8.2.5.193/ 1501

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_0318	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA2)	32	R/W	0000_0116h	8.2.5.194/ 1502
3033_031C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA3)	32	R/W	0000_1916h	8.2.5.195/ 1504
3033_0320	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA4)	32	R/W	0000_0116h	8.2.5.196/ 1505
3033_0324	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA5)	32	R/W	0000_0116h	8.2.5.197/ 1506
3033_0328	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA6)	32	R/W	0000_0116h	8.2.5.198/ 1507
3033_032C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA7)	32	R/W	0000_0116h	8.2.5.199/ 1508
3033_0330	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_RESET_B)	32	R/W	0000_0116h	8.2.5.200/ 1510
3033_0334	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_STROBE)	32	R/W	0000_0116h	8.2.5.201/ 1511
3033_0338	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_CD_B)	32	R/W	0000_0116h	8.2.5.202/ 1512
3033_033C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_CLK)	32	R/W	0000_0116h	8.2.5.203/ 1513
3033_0340	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_CMD)	32	R/W	0000_0116h	8.2.5.204/ 1514
3033_0344	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA0)	32	R/W	0000_1916h	8.2.5.205/ 1515
3033_0348	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA1)	32	R/W	0000_0116h	8.2.5.206/ 1517
3033_034C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA2)	32	R/W	0000_0116h	8.2.5.207/ 1518
3033_0350	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA3)	32	R/W	0000_0116h	8.2.5.208/ 1519
3033_0354	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_RESET_B)	32	R/W	0000_0116h	8.2.5.209/ 1520
3033_0358	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_WP)	32	R/W	0000_0116h	8.2.5.210/ 1521
3033_035C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_ALE)	32	R/W	0000_0116h	8.2.5.211/ 1522
3033_0360	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_CE0_B)	32	R/W	0000_0116h	8.2.5.212/ 1524
3033_0364	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_CE1_B)	32	R/W	0000_0116h	8.2.5.213/ 1525
3033_0368	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_CE2_B)	32	R/W	0000_0116h	8.2.5.214/ 1526
3033_036C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_CE3_B)	32	R/W	0000_0116h	8.2.5.215/ 1527

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_0370	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_CLE)	32	R/W	0000_0116h	8.2.5.216/ 1528
3033_0374	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA00)	32	R/W	0000_0116h	8.2.5.217/ 1529
3033_0378	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA01)	32	R/W	0000_0116h	8.2.5.218/ 1531
3033_037C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA02)	32	R/W	0000_0116h	8.2.5.219/ 1532
3033_0380	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA03)	32	R/W	0000_1916h	8.2.5.220/ 1533
3033_0384	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA04)	32	R/W	0000_0116h	8.2.5.221/ 1534
3033_0388	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA05)	32	R/W	0000_0116h	8.2.5.222/ 1535
3033_038C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA06)	32	R/W	0000_0116h	8.2.5.223/ 1537
3033_0390	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA07)	32	R/W	0000_0116h	8.2.5.224/ 1538
3033_0394	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DQS)	32	R/W	0000_0116h	8.2.5.225/ 1539
3033_0398	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_RE_B)	32	R/W	0000_0116h	8.2.5.226/ 1540
3033_039C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_READY_B)	32	R/W	0000_0116h	8.2.5.227/ 1541
3033_03A0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_WE_B)	32	R/W	0000_0116h	8.2.5.228/ 1542
3033_03A4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_WP_B)	32	R/W	0000_0116h	8.2.5.229/ 1544
3033_03A8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_RXFS)	32	R/W	0000_0116h	8.2.5.230/ 1545
3033_03AC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_RXC)	32	R/W	0000_0116h	8.2.5.231/ 1546
3033_03B0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD0)	32	R/W	0000_0116h	8.2.5.232/ 1547
3033_03B4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD1)	32	R/W	0000_1916h	8.2.5.233/ 1548
3033_03B8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD2)	32	R/W	0000_0116h	8.2.5.234/ 1550
3033_03BC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD3)	32	R/W	0000_0116h	8.2.5.235/ 1551
3033_03C0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_MCLK)	32	R/W	0000_0116h	8.2.5.236/ 1552
3033_03C4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXFS)	32	R/W	0000_0116h	8.2.5.237/ 1553

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_03C8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXC)	32	R/W	0000_0116h	8.2.5.238/ 1554
3033_03CC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD0)	32	R/W	0000_0116h	8.2.5.239/ 1555
3033_03D0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD1)	32	R/W	0000_0116h	8.2.5.240/ 1557
3033_03D4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD2)	32	R/W	0000_0116h	8.2.5.241/ 1558
3033_03D8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD3)	32	R/W	0000_0116h	8.2.5.242/ 1559
3033_03DC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD4)	32	R/W	0000_0116h	8.2.5.243/ 1560
3033_03E0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD5)	32	R/W	0000_0116h	8.2.5.244/ 1561
3033_03E4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD6)	32	R/W	0000_0116h	8.2.5.245/ 1562
3033_03E8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD7)	32	R/W	0000_0116h	8.2.5.246/ 1564
3033_03EC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXFS)	32	R/W	0000_1916h	8.2.5.247/ 1565
3033_03F0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXC)	32	R/W	0000_0116h	8.2.5.248/ 1566
3033_03F4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD0)	32	R/W	0000_0116h	8.2.5.249/ 1567
3033_03F8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD1)	32	R/W	0000_0116h	8.2.5.250/ 1568
3033_03FC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD2)	32	R/W	0000_0116h	8.2.5.251/ 1570
3033_0400	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD3)	32	R/W	0000_0116h	8.2.5.252/ 1571
3033_0404	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD4)	32	R/W	0000_0116h	8.2.5.253/ 1572
3033_0408	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD5)	32	R/W	0000_0116h	8.2.5.254/ 1573
3033_040C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD6)	32	R/W	0000_0116h	8.2.5.255/ 1574
3033_0410	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD7)	32	R/W	0000_0116h	8.2.5.256/ 1575
3033_0414	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_MCLK)	32	R/W	0000_0116h	8.2.5.257/ 1577
3033_0418	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_RXFS)	32	R/W	0000_0116h	8.2.5.258/ 1578
3033_041C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_RXC)	32	R/W	0000_0116h	8.2.5.259/ 1579

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_0420	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_RXD0)	32	R/W	0000_0116h	8.2.5.260/ 1580
3033_0424	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_TXFS)	32	R/W	0000_1916h	8.2.5.261/ 1581
3033_0428	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_TXC)	32	R/W	0000_0116h	8.2.5.262/ 1583
3033_042C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_TXD0)	32	R/W	0000_0116h	8.2.5.263/ 1584
3033_0430	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_MCLK)	32	R/W	0000_0116h	8.2.5.264/ 1585
3033_0434	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_RXFS)	32	R/W	0000_0116h	8.2.5.265/ 1586
3033_0438	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_RXC)	32	R/W	0000_0116h	8.2.5.266/ 1587
3033_043C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_RXD)	32	R/W	0000_0116h	8.2.5.267/ 1588
3033_0440	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_TXFS)	32	R/W	0000_0116h	8.2.5.268/ 1590
3033_0444	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_TXC)	32	R/W	0000_1916h	8.2.5.269/ 1591
3033_0448	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_TXD)	32	R/W	0000_0116h	8.2.5.270/ 1592
3033_044C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_MCLK)	32	R/W	0000_0116h	8.2.5.271/ 1593
3033_0450	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SPDIF_TX)	32	R/W	0000_0116h	8.2.5.272/ 1594
3033_0454	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SPDIF_RX)	32	R/W	0000_0116h	8.2.5.273/ 1596
3033_0458	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SPDIF_EXT_CLK)	32	R/W	0000_0116h	8.2.5.274/ 1597
3033_045C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI1_SCLK)	32	R/W	0000_0116h	8.2.5.275/ 1598
3033_0460	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI1_MOSI)	32	R/W	0000_0116h	8.2.5.276/ 1599
3033_0464	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI1_MISO)	32	R/W	0000_0116h	8.2.5.277/ 1600
3033_0468	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI1_SS0)	32	R/W	0000_0116h	8.2.5.278/ 1601
3033_046C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI2_SCLK)	32	R/W	0000_1916h	8.2.5.279/ 1603
3033_0470	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI2_MOSI)	32	R/W	0000_0116h	8.2.5.280/ 1604
3033_0474	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI2_MISO)	32	R/W	0000_0116h	8.2.5.281/ 1605

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_0478	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI2_SS0)	32	R/W	0000_0116h	8.2.5.282/ 1606
3033_047C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C1_SCL)	32	R/W	0000_0116h	8.2.5.283/ 1607
3033_0480	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C1_SDA)	32	R/W	0000_0116h	8.2.5.284/ 1609
3033_0484	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C2_SCL)	32	R/W	0000_0116h	8.2.5.285/ 1610
3033_0488	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C2_SDA)	32	R/W	0000_0116h	8.2.5.286/ 1611
3033_048C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C3_SCL)	32	R/W	0000_1916h	8.2.5.287/ 1612
3033_0490	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C3_SDA)	32	R/W	0000_0116h	8.2.5.288/ 1613
3033_0494	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C4_SCL)	32	R/W	0000_0116h	8.2.5.289/ 1615
3033_0498	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C4_SDA)	32	R/W	0000_0116h	8.2.5.290/ 1616
3033_049C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART1_RXD)	32	R/W	0000_0116h	8.2.5.291/ 1617
3033_04A0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART1_TXD)	32	R/W	0000_0116h	8.2.5.292/ 1618
3033_04A4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART2_RXD)	32	R/W	0000_0116h	8.2.5.293/ 1619
3033_04A8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART2_TXD)	32	R/W	0000_0116h	8.2.5.294/ 1620
3033_04AC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART3_RXD)	32	R/W	0000_1916h	8.2.5.295/ 1622
3033_04B0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART3_TXD)	32	R/W	0000_0116h	8.2.5.296/ 1623
3033_04B4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART4_RXD)	32	R/W	0000_0116h	8.2.5.297/ 1624
3033_04B8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART4_TXD)	32	R/W	0000_0116h	8.2.5.298/ 1625
3033_04BC	Select Input Register (IOMUXC_CCM_PMIC_READY_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.299/ 1627
3033_04C0	Select Input Register (IOMUXC_ENET1_MDIO_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.300/ 1627
3033_04C4	Select Input Register (IOMUXC_SAI1_RX_SYNC_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.301/ 1628
3033_04C8	Select Input Register (IOMUXC_SAI1_TX_BCLK_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.302/ 1629
3033_04CC	Select Input Register (IOMUXC_SAI1_TX_SYNC_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.303/ 1629

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_04D0	Select Input Register (IOMUXC_SAI5_RX_BCLK_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.304/ 1630
3033_04D4	Select Input Register (IOMUXC_SAI5_RX_DATA_SELECT_INPUT_0)	32	R/W	0000_0000h	8.2.5.305/ 1631
3033_04D8	Select Input Register (IOMUXC_SAI5_RX_DATA_SELECT_INPUT_1)	32	R/W	0000_0000h	8.2.5.306/ 1631
3033_04DC	Select Input Register (IOMUXC_SAI5_RX_DATA_SELECT_INPUT_2)	32	R/W	0000_0000h	8.2.5.307/ 1632
3033_04E0	Select Input Register (IOMUXC_SAI5_RX_DATA_SELECT_INPUT_3)	32	R/W	0000_0000h	8.2.5.308/ 1633
3033_04E4	Select Input Register (IOMUXC_SAI5_RX_SYNC_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.309/ 1633
3033_04E8	Select Input Register (IOMUXC_SAI5_TX_BCLK_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.310/ 1634
3033_04EC	Select Input Register (IOMUXC_SAI5_TX_SYNC_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.311/ 1635
3033_04F0	Select Input Register (IOMUXC_UART1_RTS_B_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.312/ 1635
3033_04F4	Select Input Register (IOMUXC_UART1_RXD_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.313/ 1636
3033_04F8	Select Input Register (IOMUXC_UART2_RTS_B_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.314/ 1637
3033_04FC	Select Input Register (IOMUXC_UART2_RXD_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.315/ 1637
3033_0500	Select Input Register (IOMUXC_UART3_RTS_B_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.316/ 1638
3033_0504	Select Input Register (IOMUXC_UART3_RXD_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.317/ 1639
3033_0508	Select Input Register (IOMUXC_UART4_RTS_B_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.318/ 1639
3033_050C	Select Input Register (IOMUXC_UART4_RXD_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.319/ 1640
3033_0510	Select Input Register (IOMUXC_SAI6_RX_BCLK_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.320/ 1641
3033_0514	Select Input Register (IOMUXC_SAI6_RX_DATA_SELECT_INPUT_0)	32	R/W	0000_0000h	8.2.5.321/ 1642
3033_0518	Select Input Register (IOMUXC_SAI6_RX_SYNC_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.322/ 1643
3033_051C	Select Input Register (IOMUXC_SAI6_TX_BCLK_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.323/ 1644
3033_0520	Select Input Register (IOMUXC_SAI6_TX_SYNC_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.324/ 1645
3033_0524	Select Input Register (IOMUXC_PCIE1_CLKREQ_B_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.325/ 1646

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_052C	Select Input Register (IOMUXC_SAI5_MCLK_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.326/ 1646
3033_0530	Select Input Register (IOMUXC_SAI6_MCLK_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.327/ 1647
3033_0534	Select Input Register (IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_0)	32	R/W	0000_0000h	8.2.5.328/ 1648
3033_0538	Select Input Register (IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_1)	32	R/W	0000_0000h	8.2.5.329/ 1649
3033_053C	Select Input Register (IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_2)	32	R/W	0000_0000h	8.2.5.330/ 1650
3033_0540	Select Input Register (IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_3)	32	R/W	0000_0000h	8.2.5.331/ 1651
3033_0544	Select Input Register (IOMUXC_USDHC3_CD_B_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.332/ 1651
3033_0548	Select Input Register (IOMUXC_USDHC3_WP_SELECT_INPUT)	32	R/W	0000_0000h	8.2.5.333/ 1652

8.2.5.1 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_PMIC_STBY_REQ)

Address: 3033_0000h base + 14h offset = 3033_0014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved									SION	Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SW_MUX_CTL_PAD_PMIC_STBY_REQ field descriptions

Field	Description
31–7 -	This field is reserved. Reserved
6 SION	Not used for IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.2 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_PMIC_ON_REQ)

Address: 3033_0000h base + 18h offset = 3033_0018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved									SION	Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SW_MUX_CTL_PAD_PMIC_ON_REQ field descriptions

Field	Description
31–7 -	This field is reserved. Reserved
6 SION	Not used for IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.3 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ONOFF)

Address: 3033_0000h base + 1Ch offset = 3033_001Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved									SION	Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SW_MUX_CTL_PAD_ONOFF field descriptions

Field	Description
31–7 -	This field is reserved. Reserved
6 SION	Not used for IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.4 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_POR_B)

Address: 3033_0000h base + 20h offset = 3033_0020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved									SION	Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SW_MUX_CTL_PAD_POR_B field descriptions

Field	Description
31–7 -	This field is reserved. Reserved
6 SION	Not used for IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.5 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_RTC_RESET_B)

Address: 3033_0000h base + 24h offset = 3033_0024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

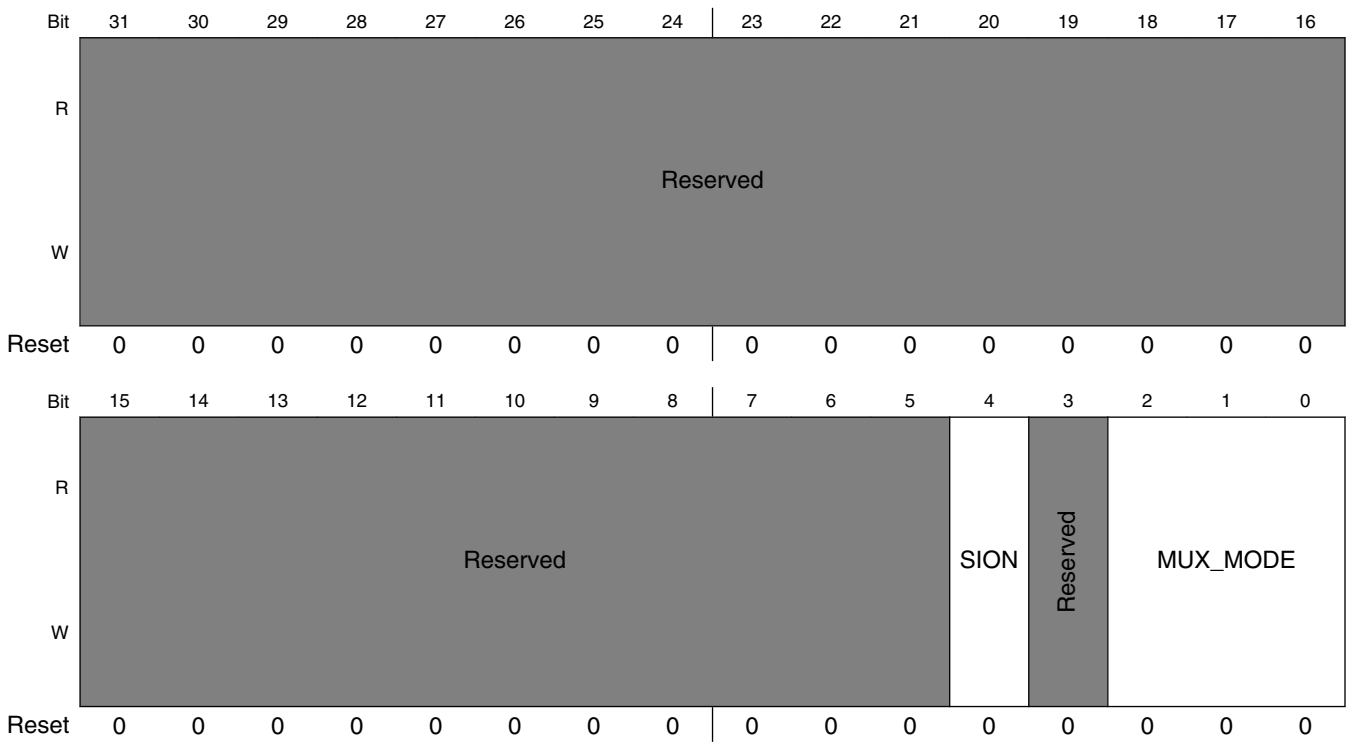
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved									SION	Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SW_MUX_CTL_PAD_RTC_RESET_B field descriptions

Field	Description
31–7 -	This field is reserved. Reserved
6 SION	Not used for IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.6 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO00)

Address: 3033_0000h base + 28h offset = 3033_0028h

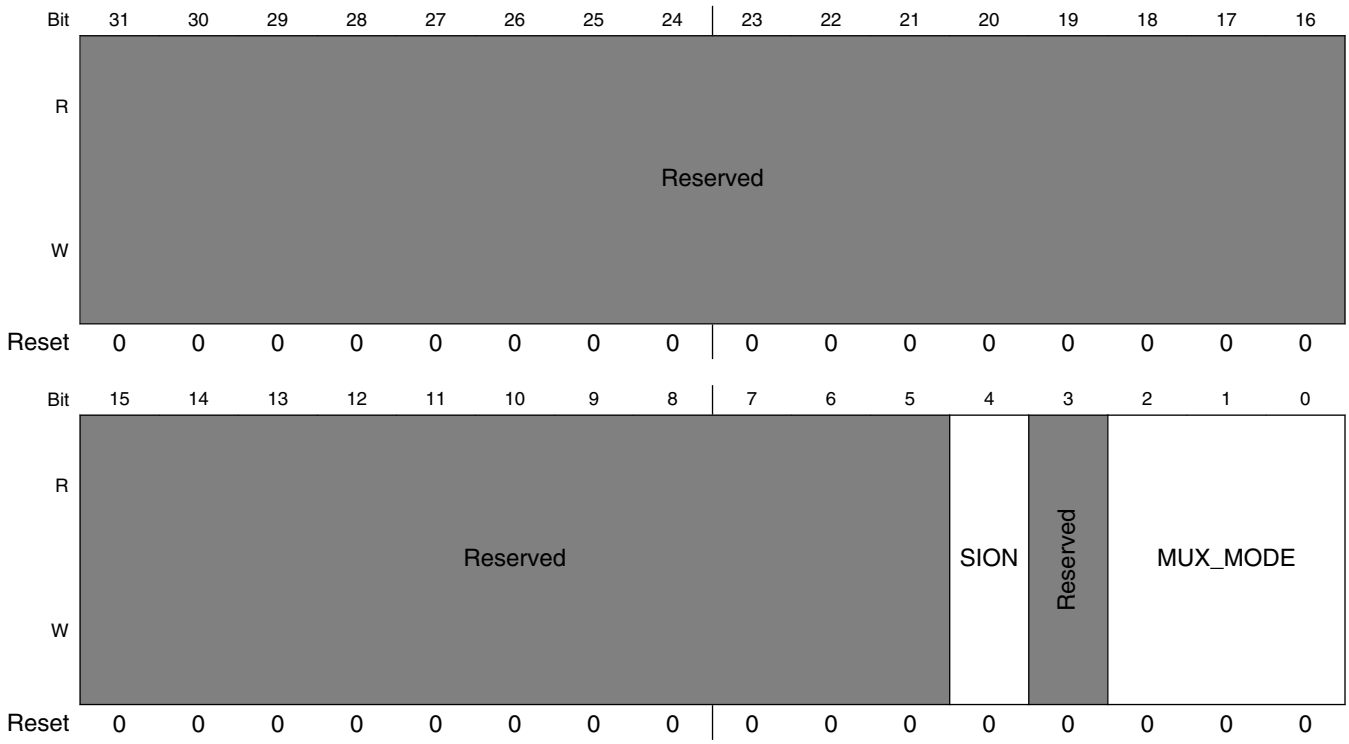


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO00 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO00 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: GPIO1_IO00. 000 **ALT0** — Select signal GPIO1_IO00 001 **ALT1** — Select signal CCM_ENET_PHY_REF_CLK_ROOT 101 **ALT5** — Select signal CCM_REF_CLK_32K 110 **ALT6** — Select signal CCM_EXT_CLK1

8.2.5.7 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO01)

Address: 3033_0000h base + 2Ch offset = 3033_002Ch

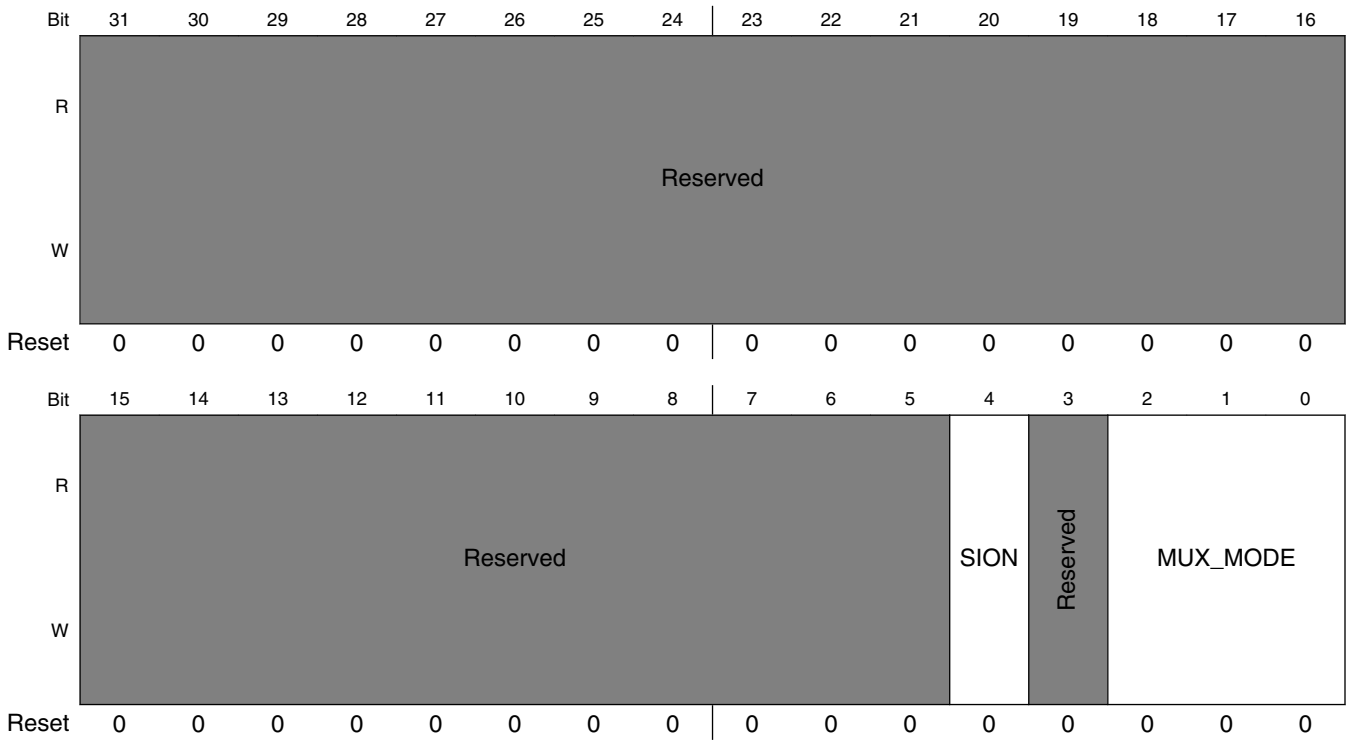


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO01 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO01 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: GPIO1_IO01. 000 **ALT0** — Select signal GPIO1_IO01 001 **ALT1** — Select signal PWM1_OUT 101 **ALT5** — Select signal CCM_REF_CLK_24M 110 **ALT6** — Select signal CCM_EXT_CLK2

8.2.5.8 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO02)

Address: 3033_0000h base + 30h offset = 3033_0030h

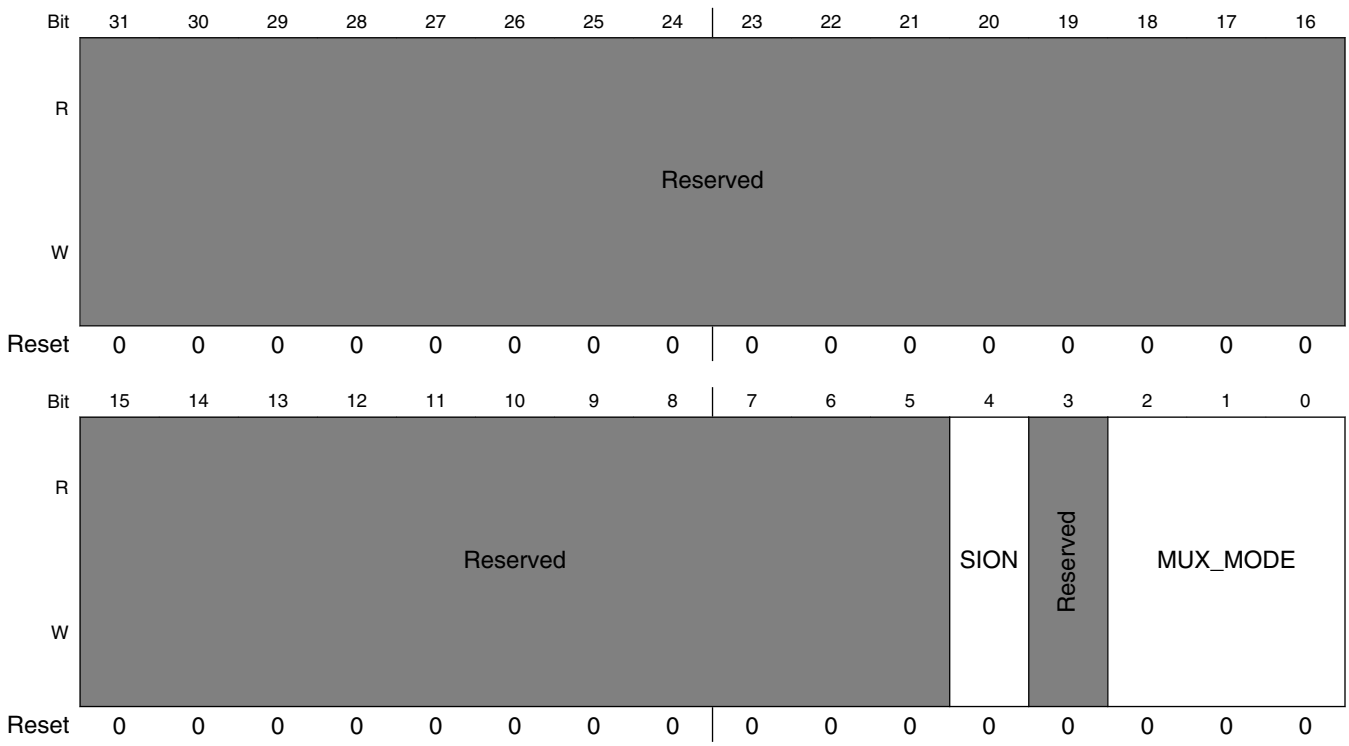


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO02 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO02 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: GPIO1_IO02. 000 **ALT0** — Select signal GPIO1_IO02 001 **ALT1** — Select signal WDOG1_WDOG_B 101 **ALT5** — Select signal WDOG1_WDOG_ANY 111 **ALT7** — Select signal SJC_DE_B

8.2.5.9 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO03)

Address: 3033_0000h base + 34h offset = 3033_0034h

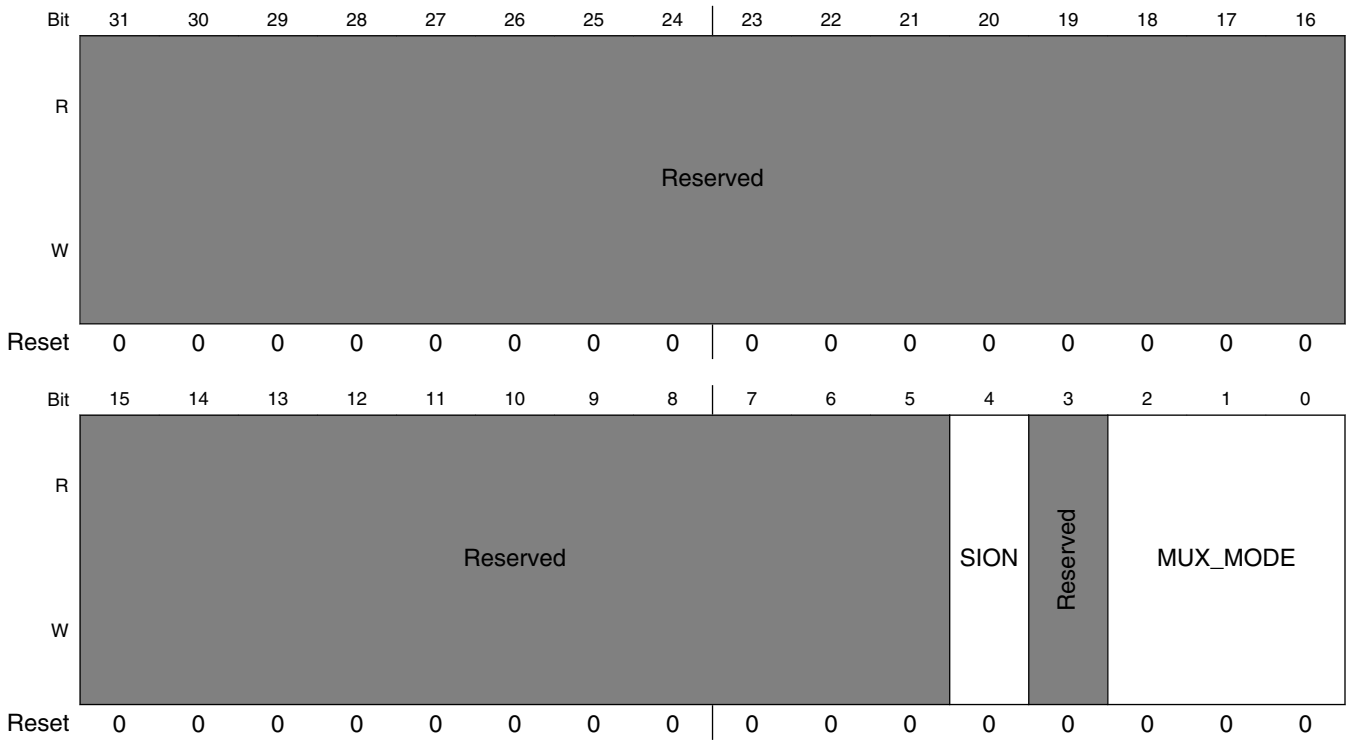


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO03 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO03 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: GPIO1_IO03. 000 **ALT0** — Select signal GPIO1_IO03 001 **ALT1** — Select signal USDHC1_VSELECT 101 **ALT5** — Select signal SDMA1_EXT_EVENT0

8.2.5.10 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO04)

Address: 3033_0000h base + 38h offset = 3033_0038h

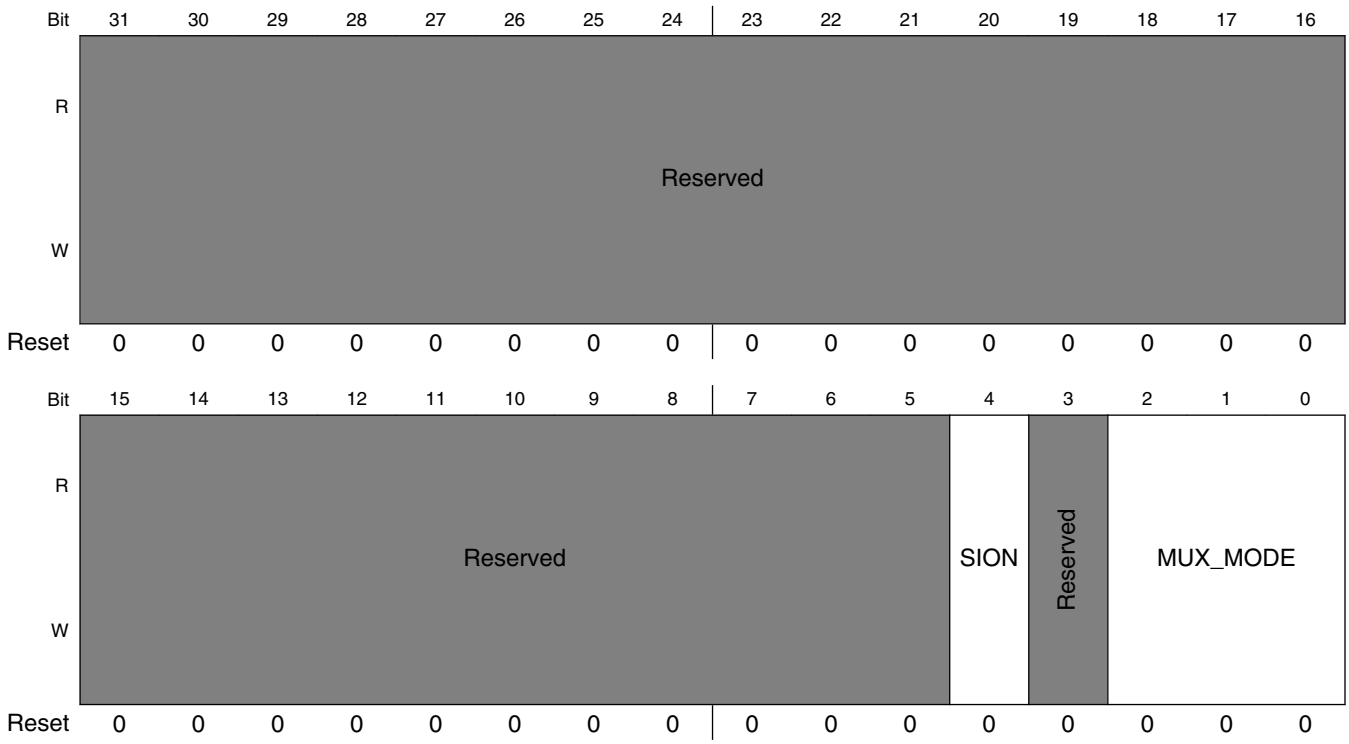


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO04 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO04 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: GPIO1_IO04. 000 **ALT0** — Select signal GPIO1_IO04 001 **ALT1** — Select signal USDHC2_VSELECT 101 **ALT5** — Select signal SDMA1_EXT_EVENT1

8.2.5.11 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO05)

Address: 3033_0000h base + 3Ch offset = 3033_003Ch

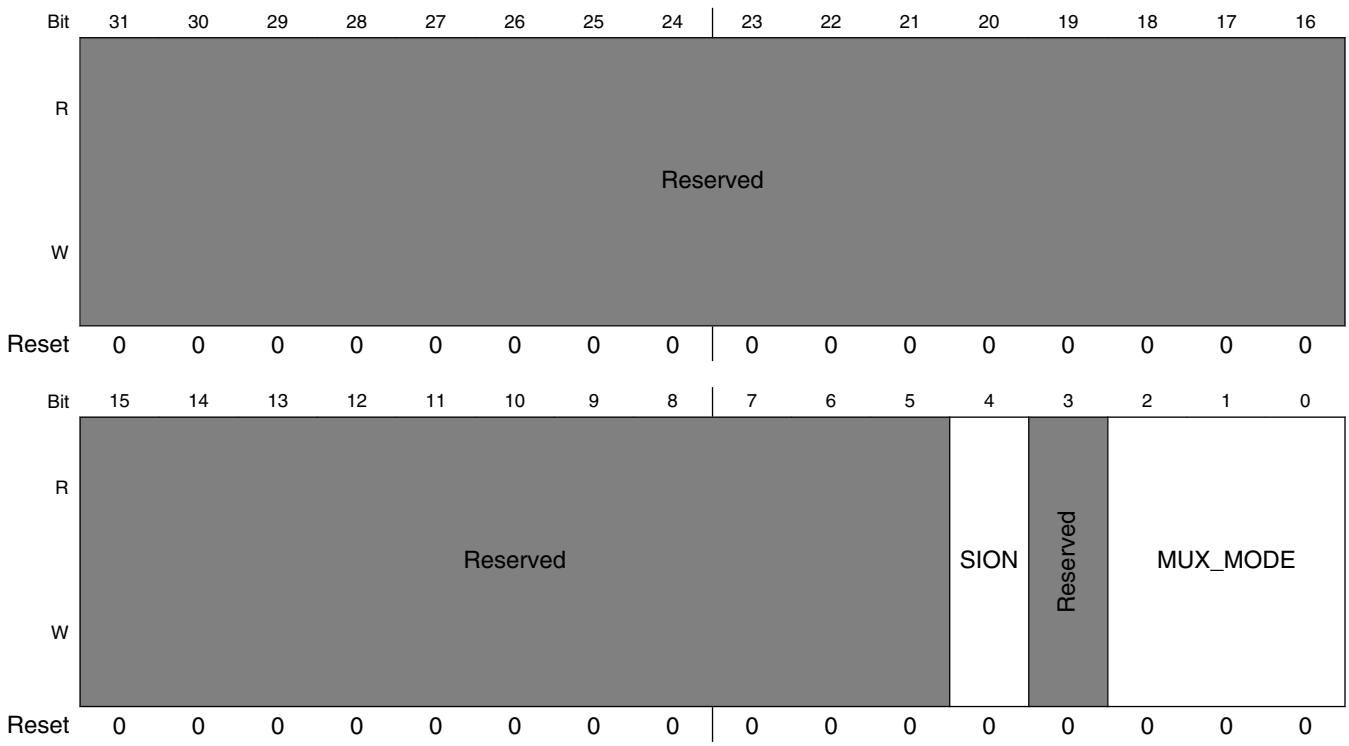


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO05 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO05 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: GPIO1_IO05. 000 **ALT0** — Select signal GPIO1_IO05 001 **ALT1** — Select signal M4_NMI 101 **ALT5** — Select signal CCM_PMIC_READY

8.2.5.12 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO06)

Address: 3033_0000h base + 40h offset = 3033_0040h

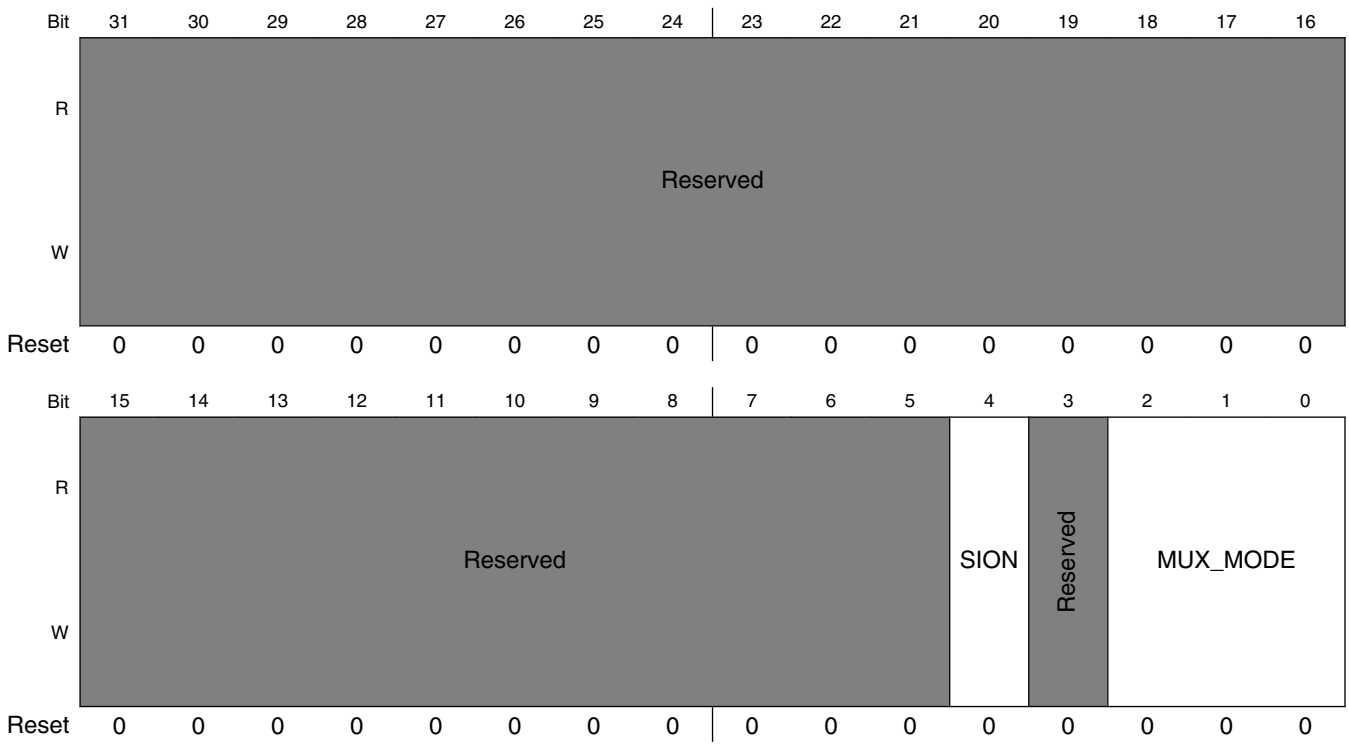


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO06 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO06 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: GPIO1_IO06. 000 **ALT0** — Select signal GPIO1_IO06 001 **ALT1** — Select signal ENET1_MDC 101 **ALT5** — Select signal USDHC1_CD_B 110 **ALT6** — Select signal CCM_EXT_CLK3

8.2.5.13 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO07)

Address: 3033_0000h base + 44h offset = 3033_0044h

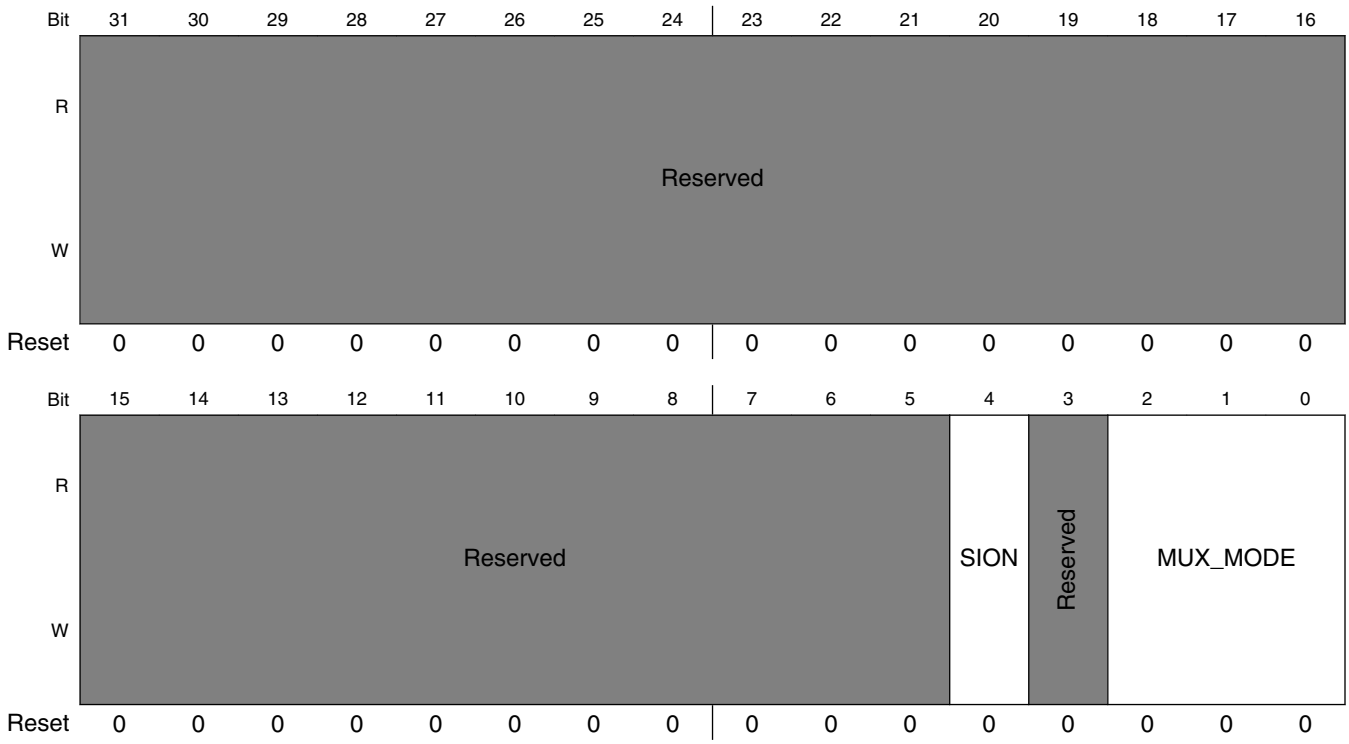


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO07 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO07 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: GPIO1_IO07. **NOTE:** Pad GPIO1_IO07 is involved in Daisy Chain. 000 **ALT0** — Select signal GPIO1_IO07 001 **ALT1** — Select signal ENET1_MDIO - Configure register IOMUXC_ENET1_MDIO_SELECT_INPUT for mode ALT1. 101 **ALT5** — Select signal USDHC1_WP 110 **ALT6** — Select signal CCM_EXT_CLK4

8.2.5.14 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO08)

Address: 3033_0000h base + 48h offset = 3033_0048h

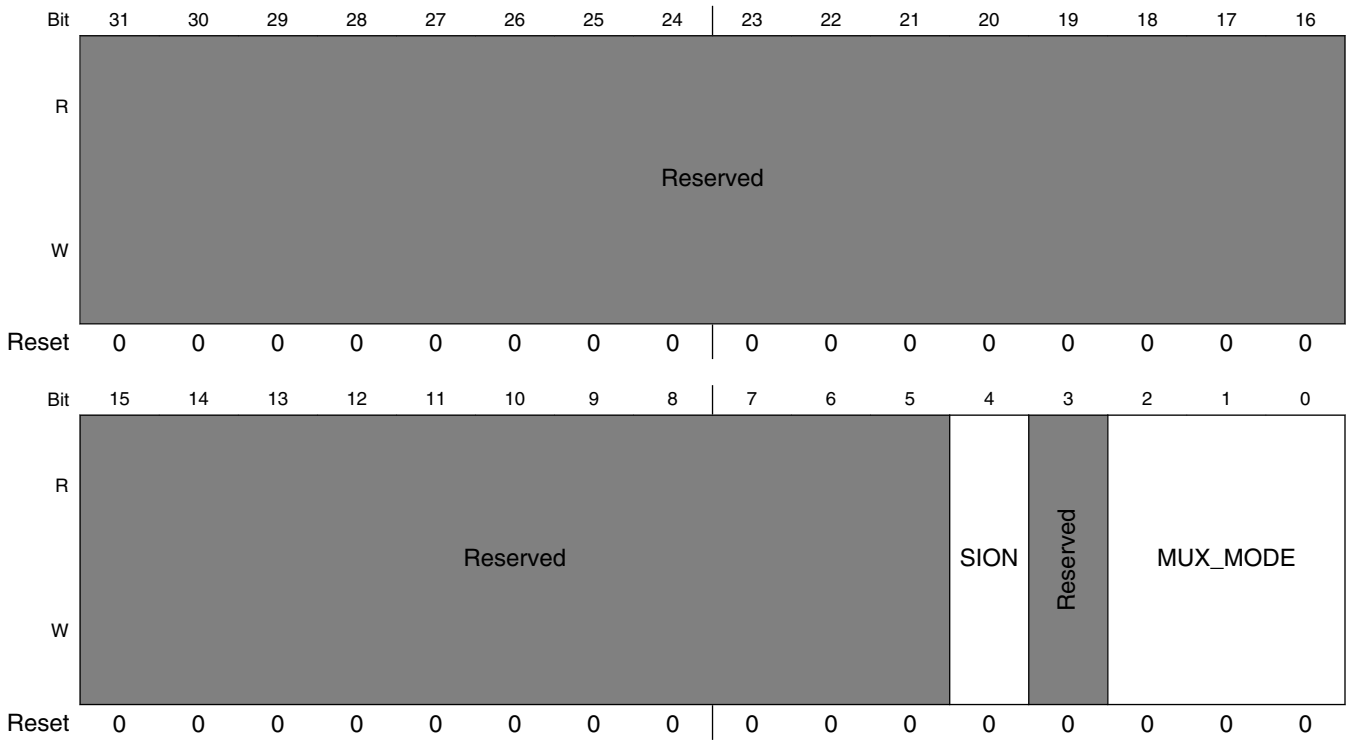


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO08 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO08 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: GPIO1_IO08. 000 **ALT0** — Select signal GPIO1_IO08 001 **ALT1** — Select signal ENET1_1588_EVENT0_IN 101 **ALT5** — Select signal USDHC2_RESET_B

8.2.5.15 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO09)

Address: 3033_0000h base + 4Ch offset = 3033_004Ch

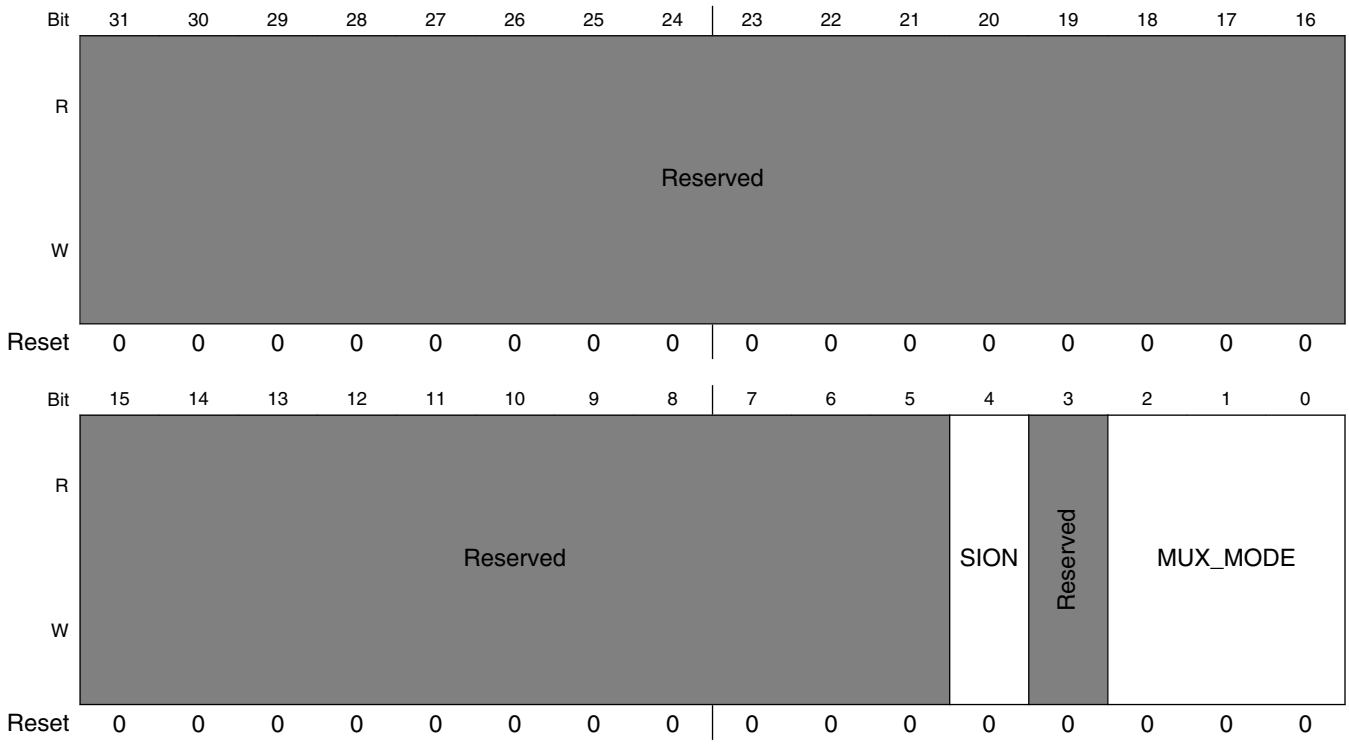


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO09 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO09 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: GPIO1_IO09. 000 **ALT0** — Select signal GPIO1_IO09 001 **ALT1** — Select signal ENET1_1588_EVENT0_OUT 100 **ALT4** — Select signal USDHC3_RESET_B 101 **ALT5** — Select signal SDMA2_EXT_EVENT0

8.2.5.16 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO10)

Address: 3033_0000h base + 50h offset = 3033_0050h

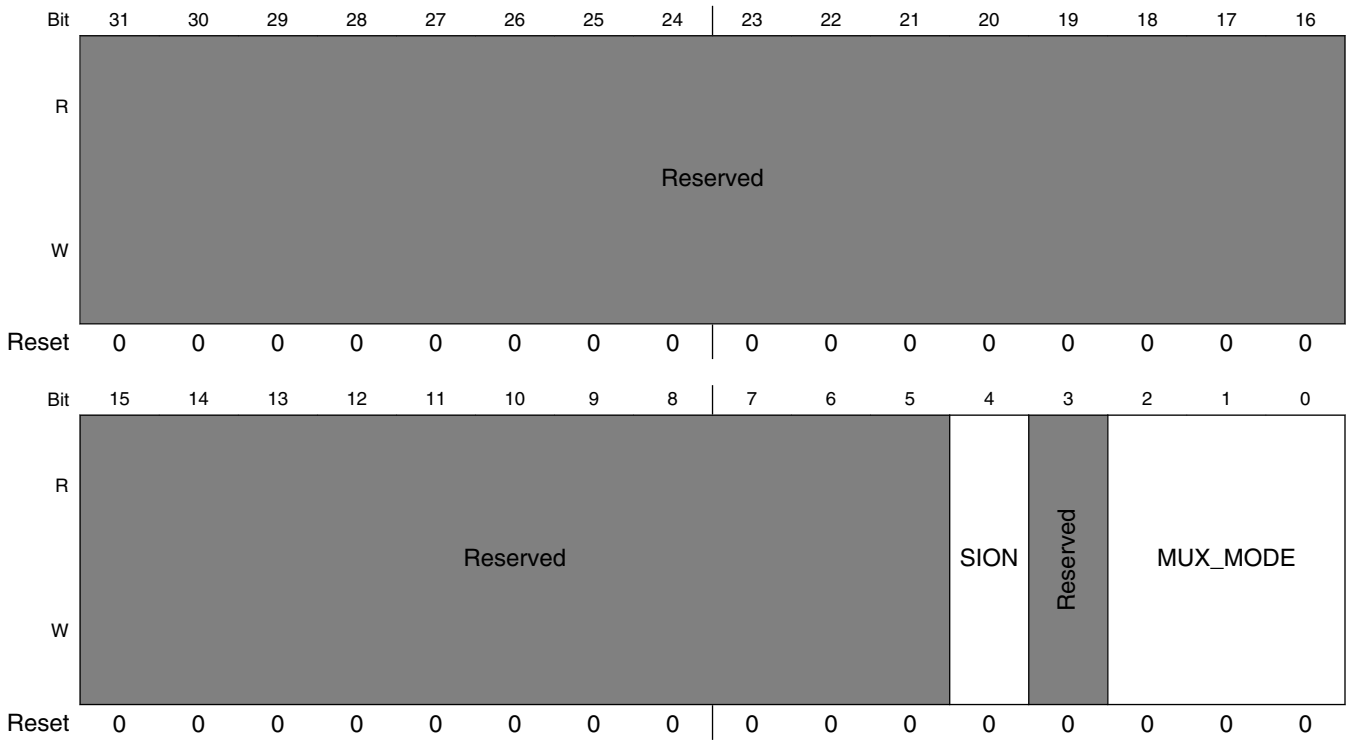


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO10 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO10 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: GPIO1_IO10. 000 **ALT0** — Select signal GPIO1_IO10 001 **ALT1** — Select signal USB1_OTG_ID

8.2.5.17 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO11)

Address: 3033_0000h base + 54h offset = 3033_0054h

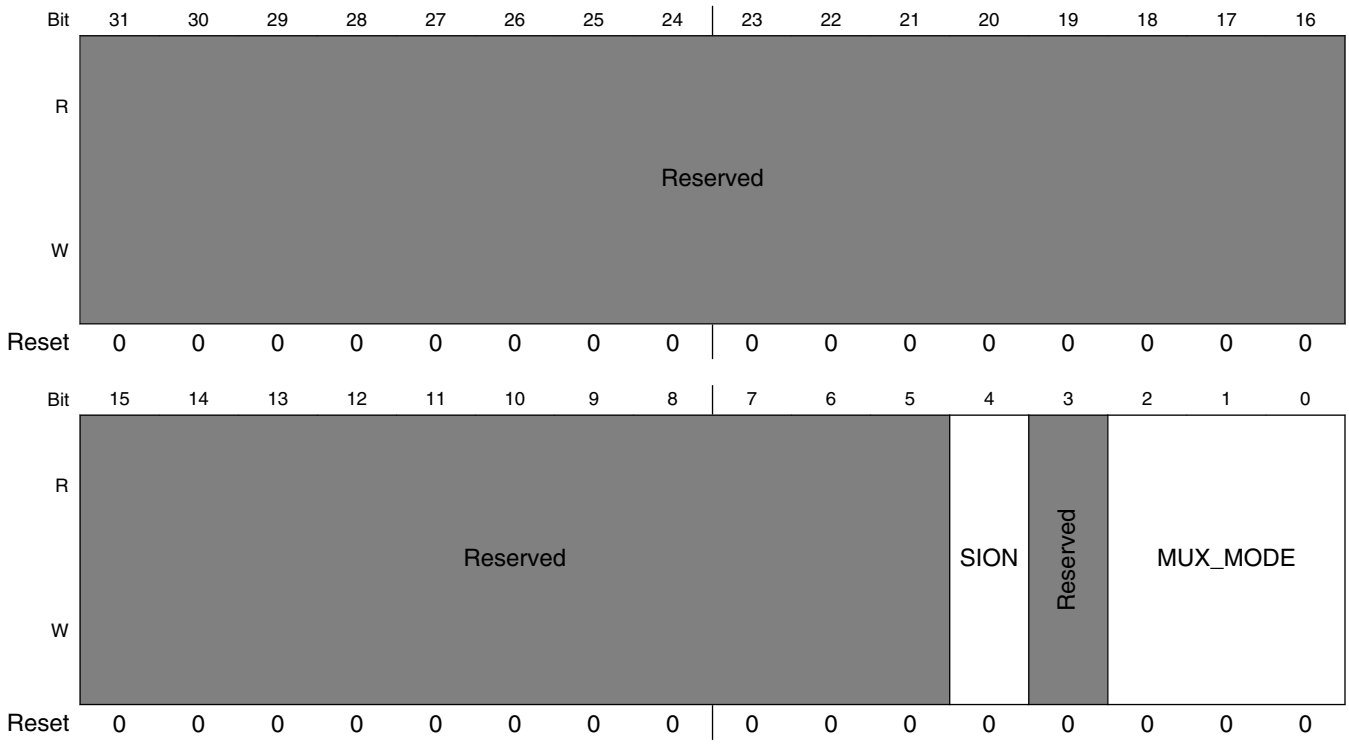


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO11 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO11 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: GPIO1_IO11. 000 **ALT0** — Select signal GPIO1_IO11 001 **ALT1** — Select signal USB2_OTG_ID 100 **ALT4** — Select signal USDHC3_VSELECT 101 **ALT5** — Select signal CCM_PMIC_READY

8.2.5.18 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO12)

Address: 3033_0000h base + 58h offset = 3033_0058h

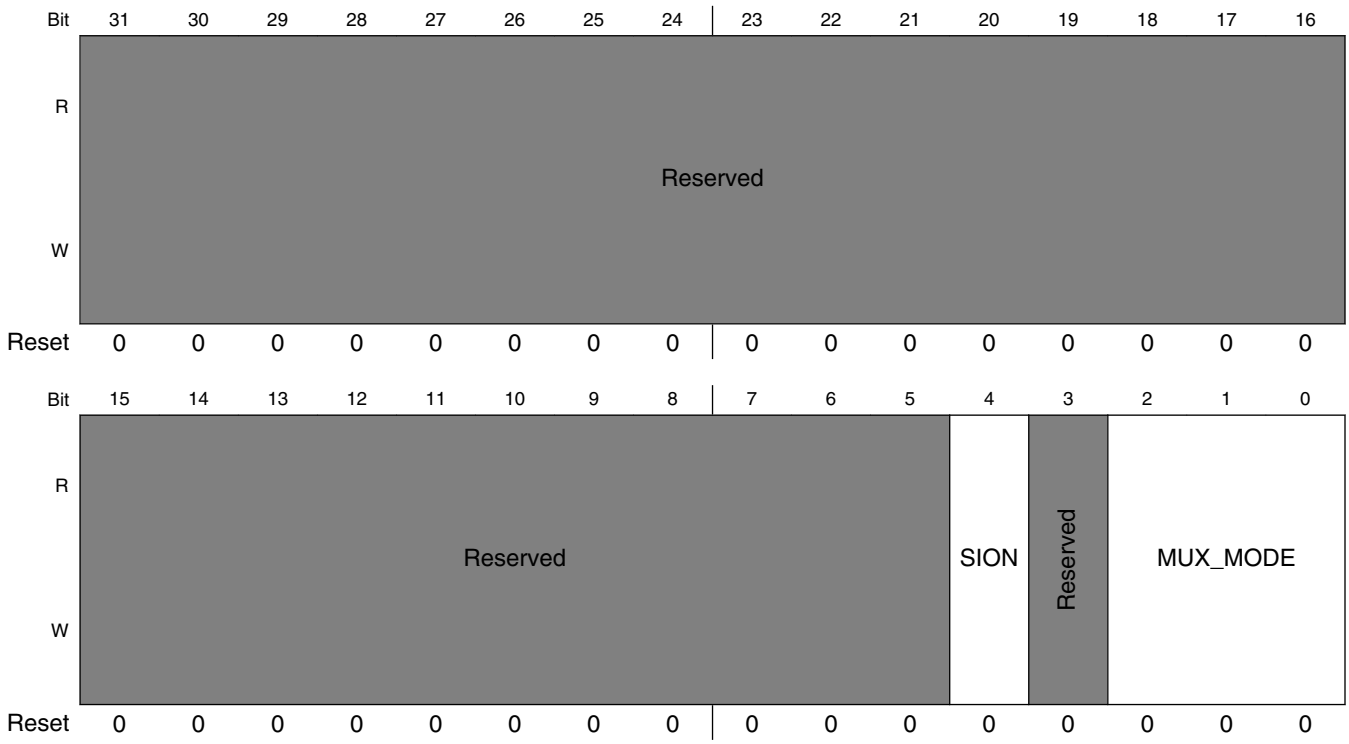


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO12 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO12 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: GPIO1_IO12. 000 **ALT0** — Select signal GPIO1_IO12 001 **ALT1** — Select signal USB1_OTG_PWR 101 **ALT5** — Select signal SDMA2_EXT_EVENT1

8.2.5.19 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO13)

Address: 3033_0000h base + 5Ch offset = 3033_005Ch

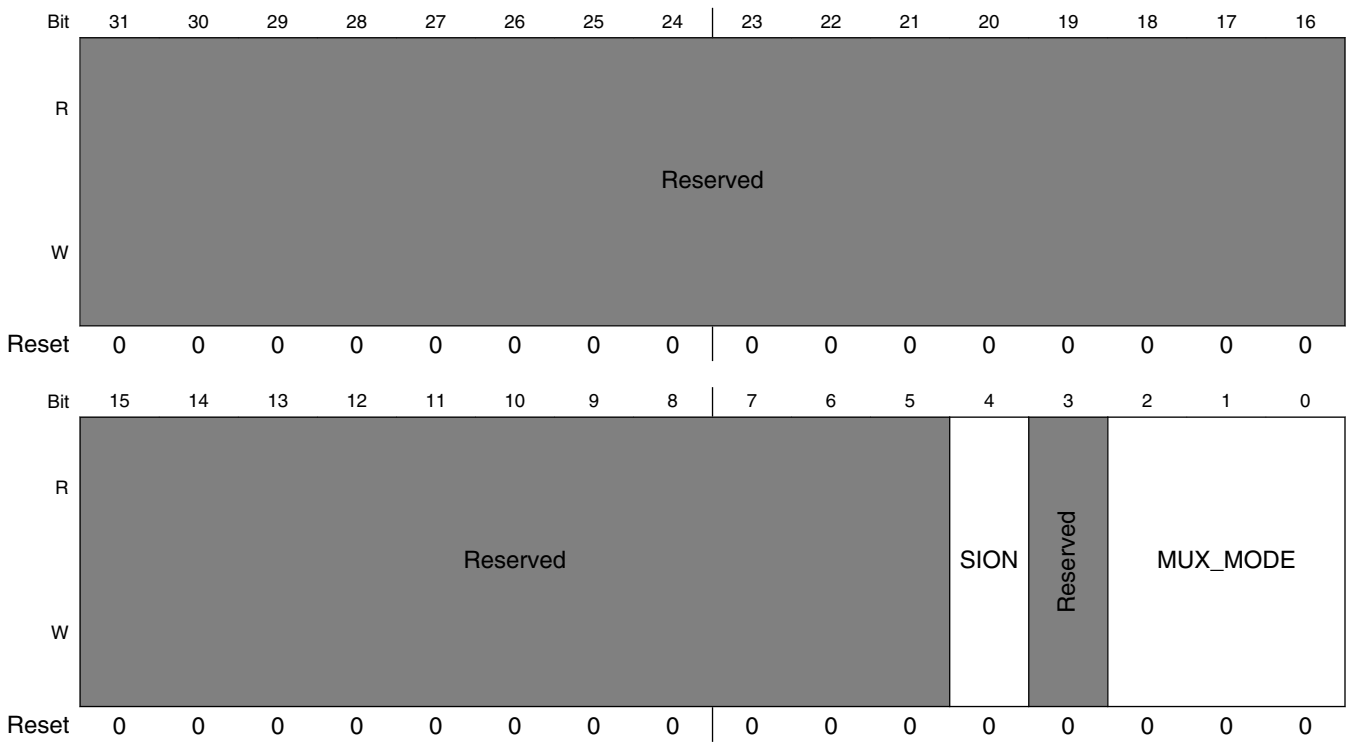


IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO13 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO13 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: GPIO1_IO13. 000 **ALT0** — Select signal GPIO1_IO13 001 **ALT1** — Select signal USB1_OTG_OC 101 **ALT5** — Select signal PWM2_OUT

8.2.5.20 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO14)

Address: 3033_0000h base + 60h offset = 3033_0060h



IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO14 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO14 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: GPIO1_IO14. **NOTE:** Pad GPIO1_IO14 is involved in Daisy Chain. 000 **ALT0** — Select signal GPIO1_IO14 001 **ALT1** — Select signal USB2_OTG_PWR 100 **ALT4** — Select signal USDHC3_CD_B - Configure register IOMUXC_USDHC3_CD_B_SELECT_INPUT for mode ALT4.

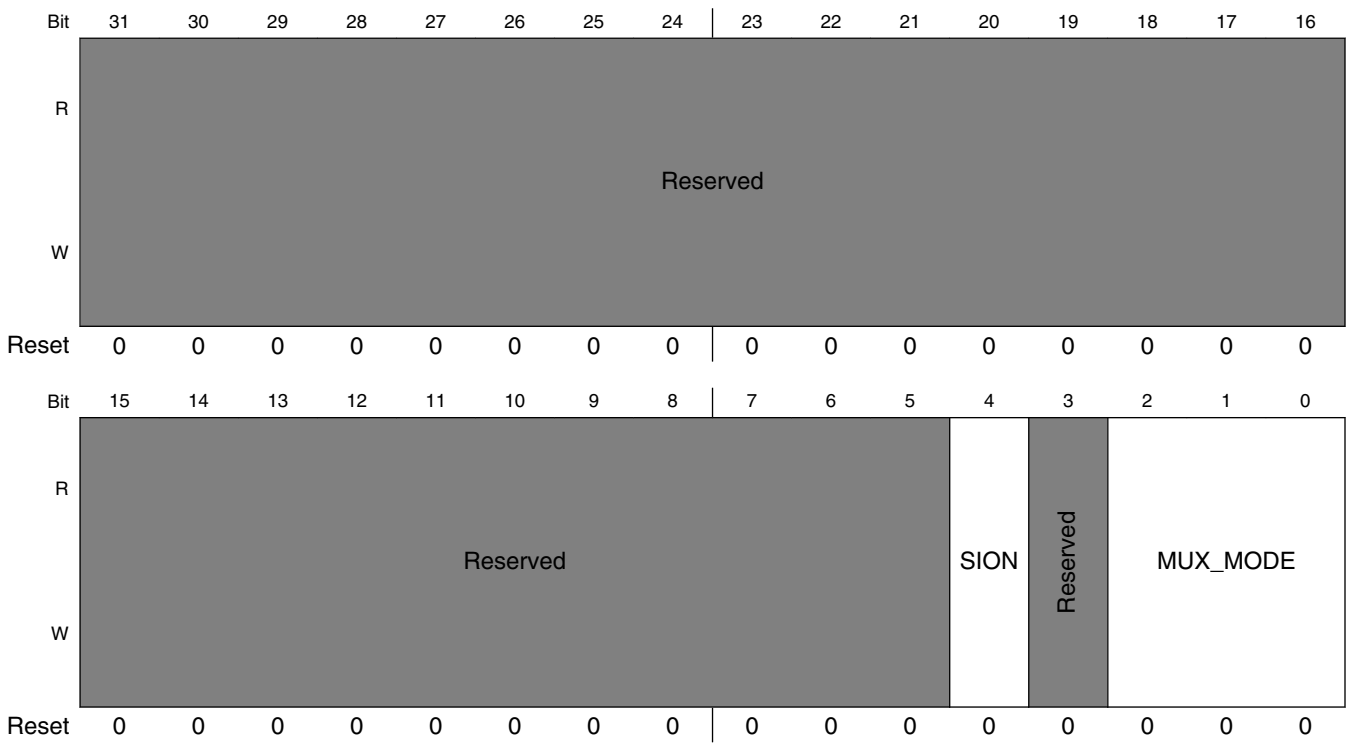
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO14 field descriptions (continued)

Field	Description
	101 **ALT5** — Select signal PWM3_OUT 110 **ALT6** — Select signal CCM_CLKO1

8.2.5.21 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO15)

Address: 3033_0000h base + 64h offset = 3033_0064h



IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO15 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad GPIO1_IO15 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: GPIO1_IO15.

Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO15 field descriptions (continued)

Field	Description
	NOTE: Pad GPIO1_IO15 is involved in Daisy Chain. 000 **ALT0** — Select signal GPIO1_IO15 001 **ALT1** — Select signal USB2_OTG_OC 100 **ALT4** — Select signal USDHC3_WP - Configure register IOMUXC_USDHC3_WP_SELECT_INPUT for mode ALT4. 101 **ALT5** — Select signal PWM4_OUT 110 **ALT6** — Select signal CCM_CLKO2

8.2.5.22 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_MDC)

Address: 3033_0000h base + 68h offset = 3033_0068h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_ENET_MDC field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality.

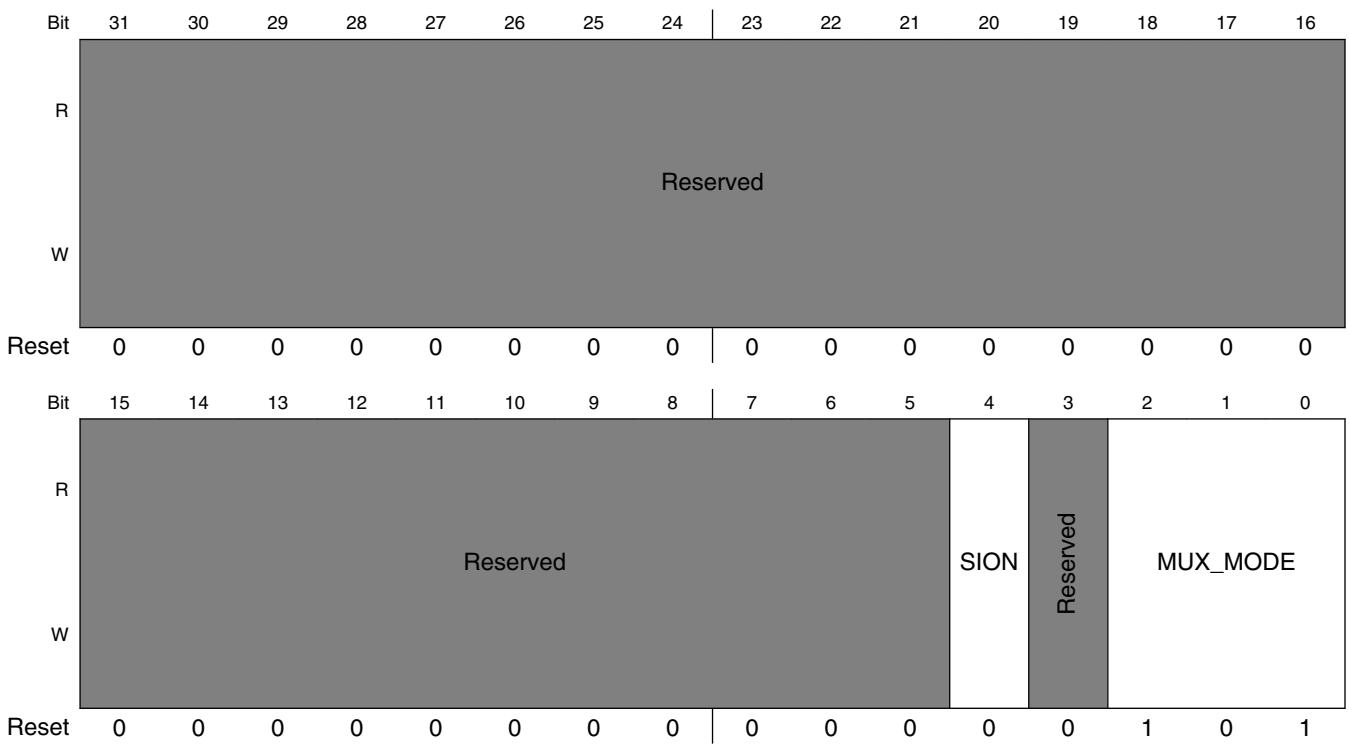
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_ENET_MDC field descriptions (continued)

Field	Description
	1 **ENABLED** — Force input path of pad ENET_MDC 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: ENET_MDC. 000 **ALT0** — Select signal ENET1_MDC 101 **ALT5** — Select signal GPIO1_IO16

8.2.5.23 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_MDIO)

Address: 3033_0000h base + 6Ch offset = 3033_006Ch



IOMUXC_SW_MUX_CTL_PAD_ENET_MDIO field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality.

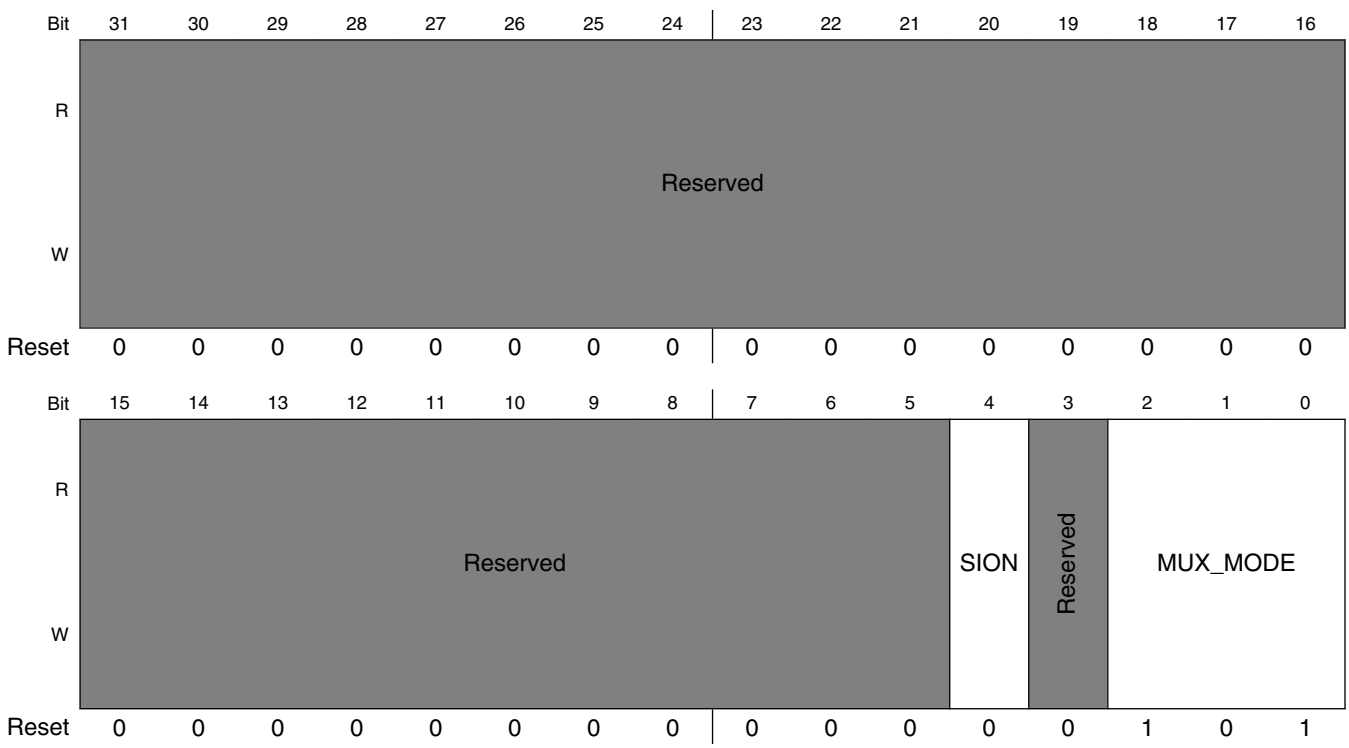
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_ENET_MDIO field descriptions (continued)

Field	Description
	1 **ENABLED** — Force input path of pad ENET_MDIO 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: ENET_MDIO. **NOTE:** Pad ENET_MDIO is involved in Daisy Chain. 000 **ALT0** — Select signal ENET1_MDIO - Configure register IOMUXC_ENET1_MDIO_SELECT_INPUT for mode ALT0. 101 **ALT5** — Select signal GPIO1_IO17

8.2.5.24 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_TD3)

Address: 3033_0000h base + 70h offset = 3033_0070h



IOMUXC_SW_MUX_CTL_PAD_ENET_TD3 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved

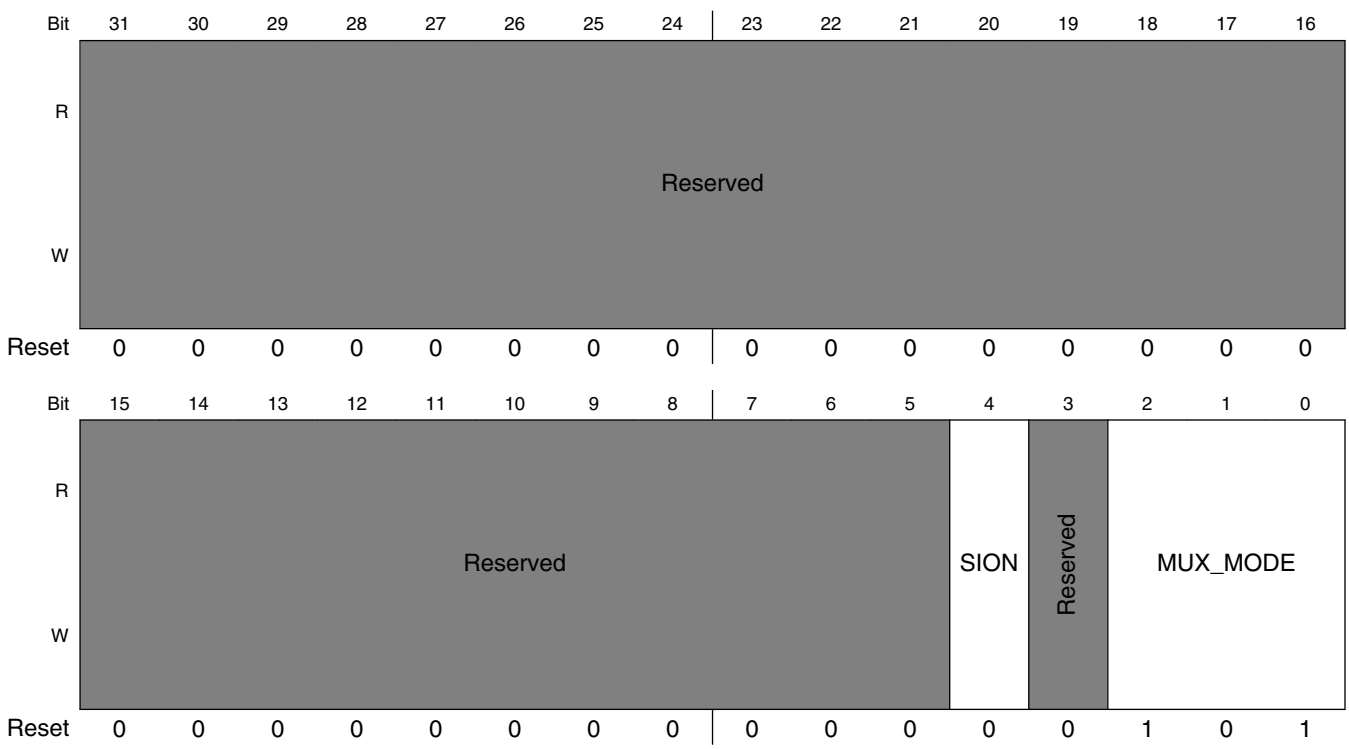
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_ENET_TD3 field descriptions (continued)

Field	Description
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ENET_TD3 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: ENET_TD3. 000 **ALT0** — Select signal ENET1_RGMII_TD3 101 **ALT5** — Select signal GPIO1_IO18

8.2.5.25 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_TD2)

Address: 3033_0000h base + 74h offset = 3033_0074h



IOMUXC_SW_MUX_CTL_PAD_ENET_TD2 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved

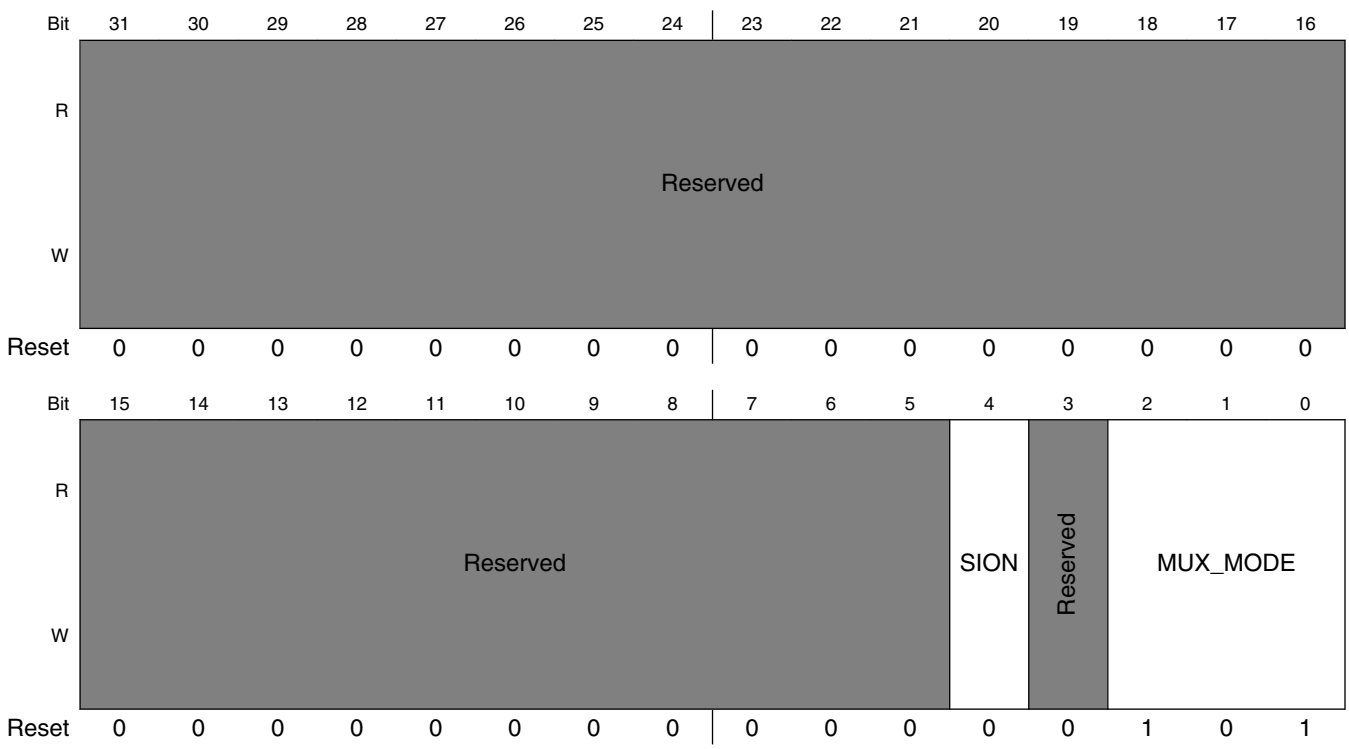
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_ENET_TD2 field descriptions (continued)

Field	Description
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ENET_TD2 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: ENET_TD2. 000 **ALT0** — Select signal ENET1_RGMII_TD2 001 **ALT1** — Select signal INPUT=ENET1_TX_CLK, OUTPUT=CCM_ENET_REF_CLK_ROOT 101 **ALT5** — Select signal GPIO1_IO19

8.2.5.26 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_TD1)

Address: 3033_0000h base + 78h offset = 3033_0078h

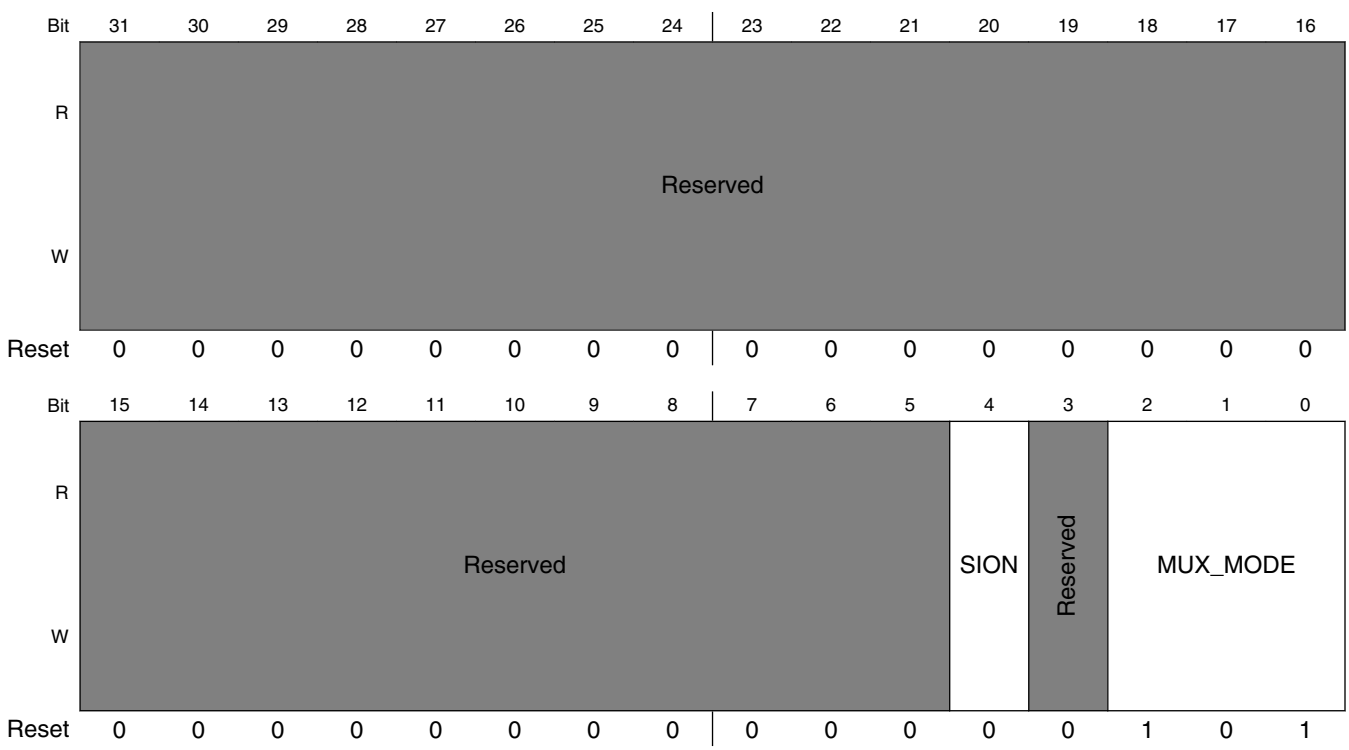


IOMUXC_SW_MUX_CTL_PAD_ENET_TD1 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ENET_TD1 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: ENET_TD1. 000 **ALT0** — Select signal ENET1_RGMII_TD1 101 **ALT5** — Select signal GPIO1_IO20

8.2.5.27 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_TD0)

Address: 3033_0000h base + 7Ch offset = 3033_007Ch

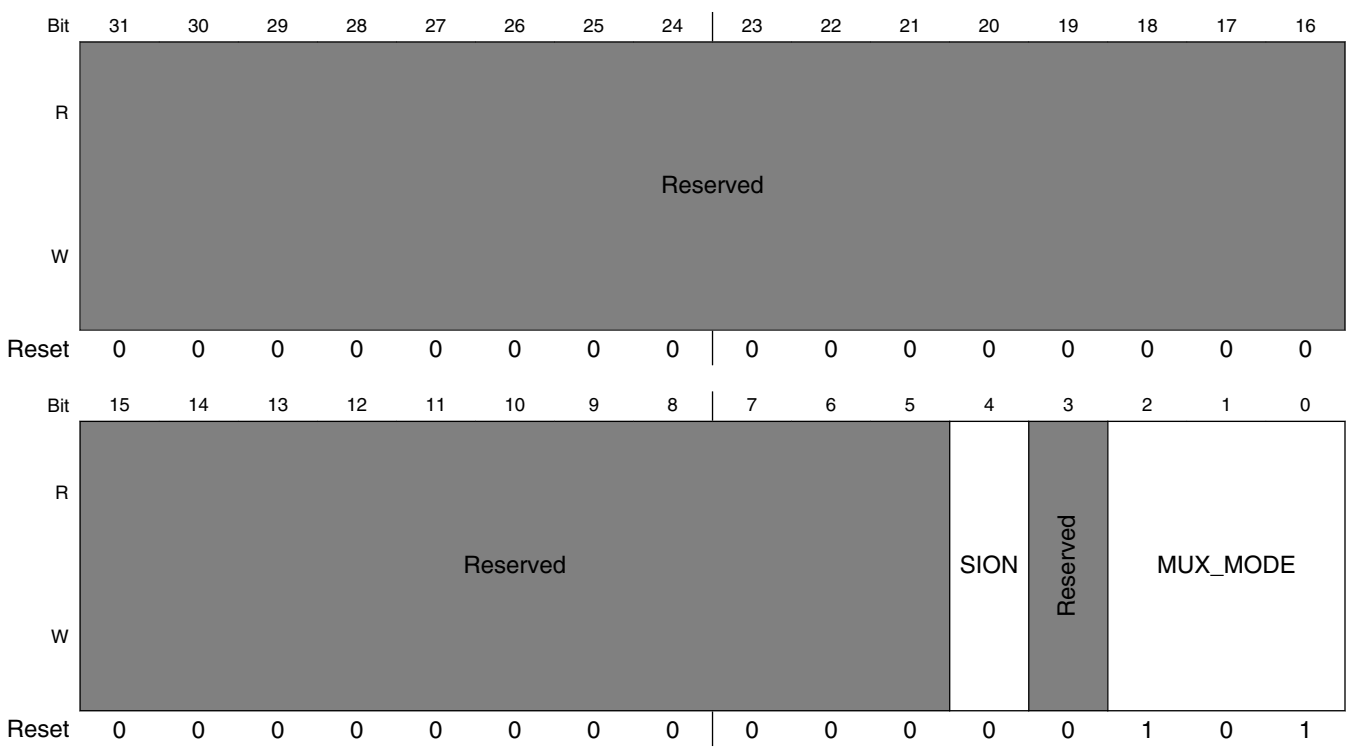


IOMUXC_SW_MUX_CTL_PAD_ENET_TD0 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ENET_TD0 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: ENET_TD0. 000 **ALT0** — Select signal ENET1_RGMII_TD0 101 **ALT5** — Select signal GPIO1_IO21

8.2.5.28 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_TX_CTL)

Address: 3033_0000h base + 80h offset = 3033_0080h



IOMUXC_SW_MUX_CTL_PAD_ENET_TX_CTL field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ENET_TX_CTL 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: ENET_TX_CTL. 000 **ALT0** — Select signal ENET1_RGMII_TX_CTL 101 **ALT5** — Select signal GPIO1_IO22

8.2.5.29 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_TXC)

Address: 3033_0000h base + 84h offset = 3033_0084h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

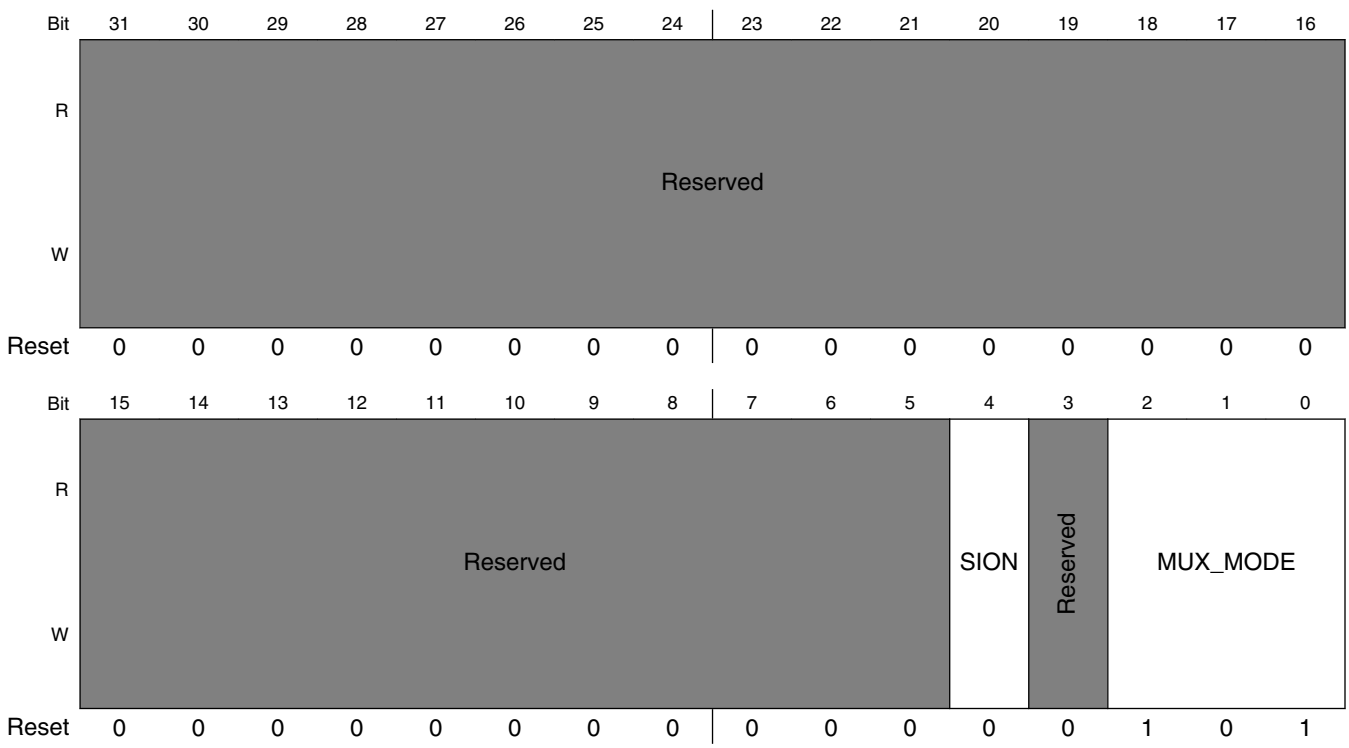
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_ENET_TXC field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ENET_TXC 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: ENET_TXC. 000 **ALT0** — Select signal ENET1_RGMII_TXC 001 **ALT1** — Select signal ENET1_TX_ER 101 **ALT5** — Select signal GPIO1_IO23

8.2.5.30 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_RX_CTL)

Address: 3033_0000h base + 88h offset = 3033_0088h

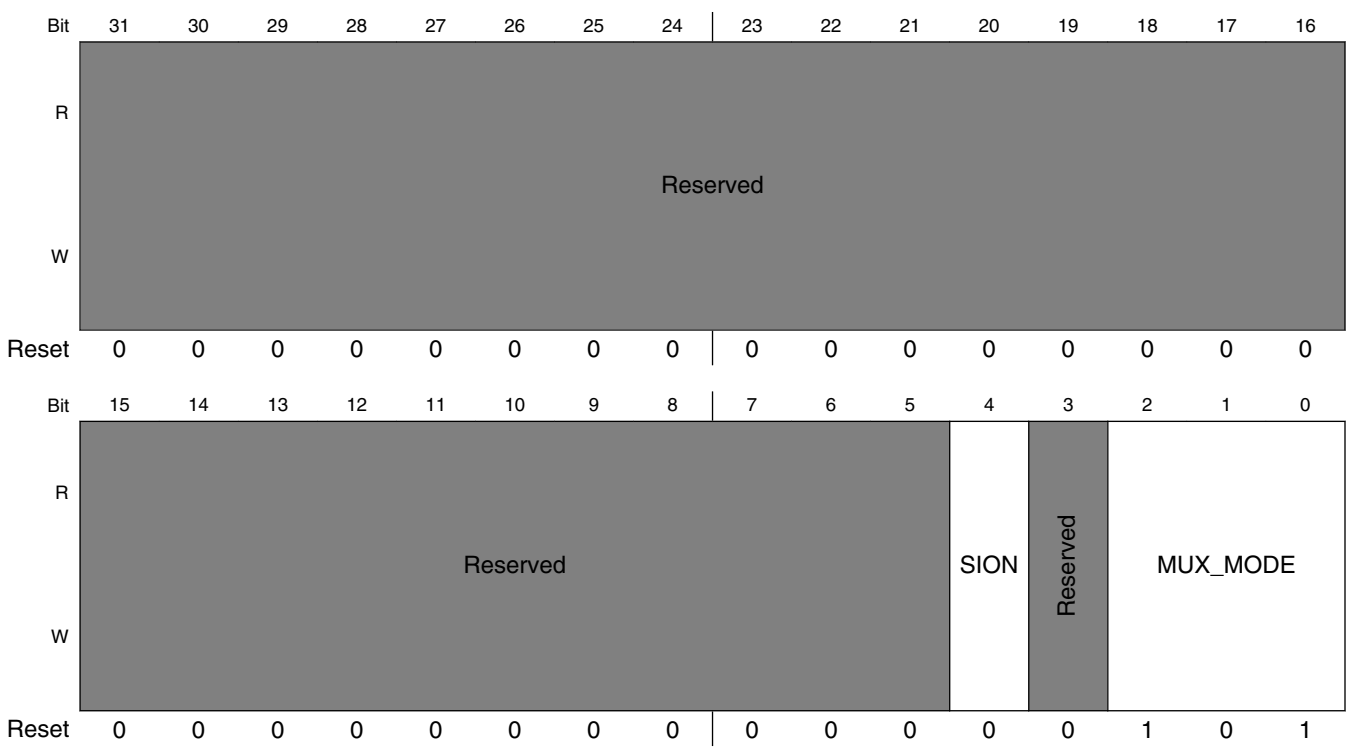


IOMUXC_SW_MUX_CTL_PAD_ENET_RX_CTL field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ENET_RX_CTL 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: ENET_RX_CTL. 000 **ALT0** — Select signal ENET1_RGMII_RX_CTL 101 **ALT5** — Select signal GPIO1_IO24

8.2.5.31 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_RXC)

Address: 3033_0000h base + 8Ch offset = 3033_008Ch

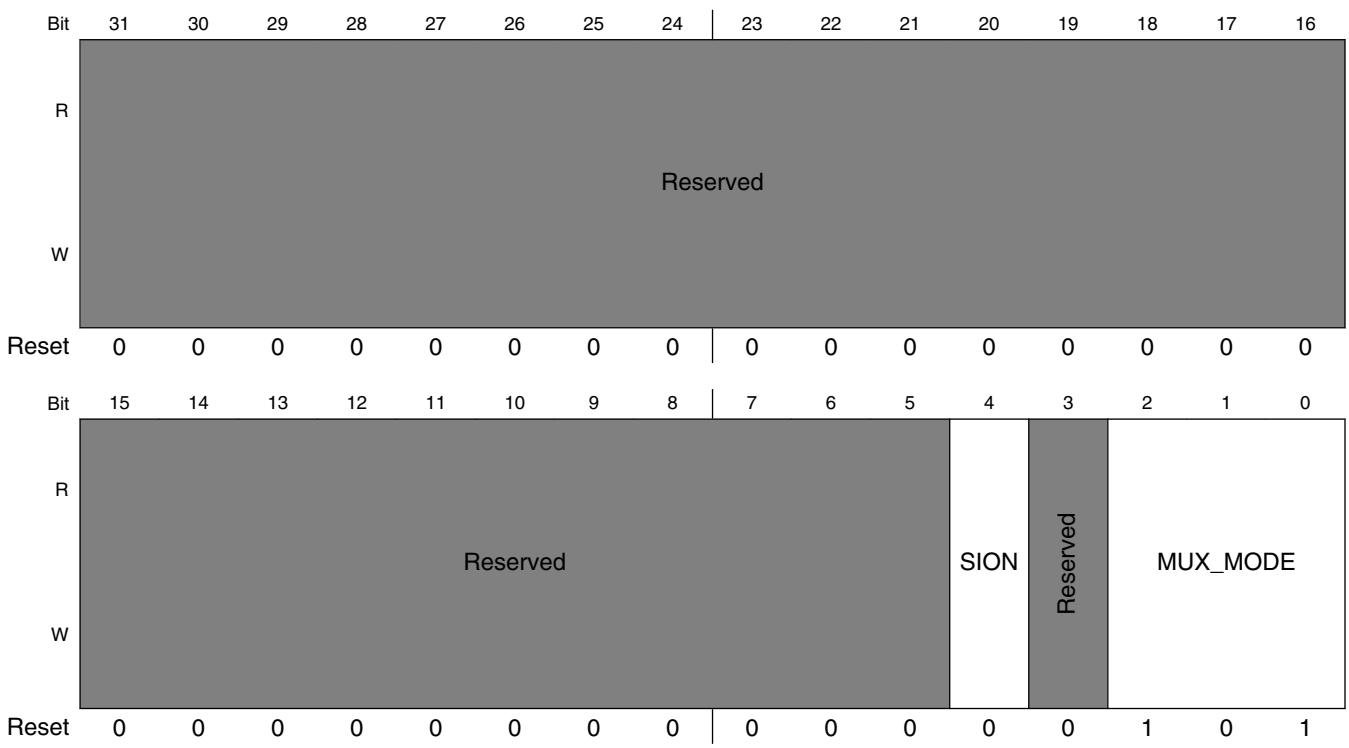


IOMUXC_SW_MUX_CTL_PAD_ENET_RXC field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ENET_RXC 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: ENET_RXC. 000 **ALT0** — Select signal ENET1_RGMII_RXC 001 **ALT1** — Select signal ENET1_RX_ER 101 **ALT5** — Select signal GPIO1_IO25

8.2.5.32 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_RD0)

Address: 3033_0000h base + 90h offset = 3033_0090h

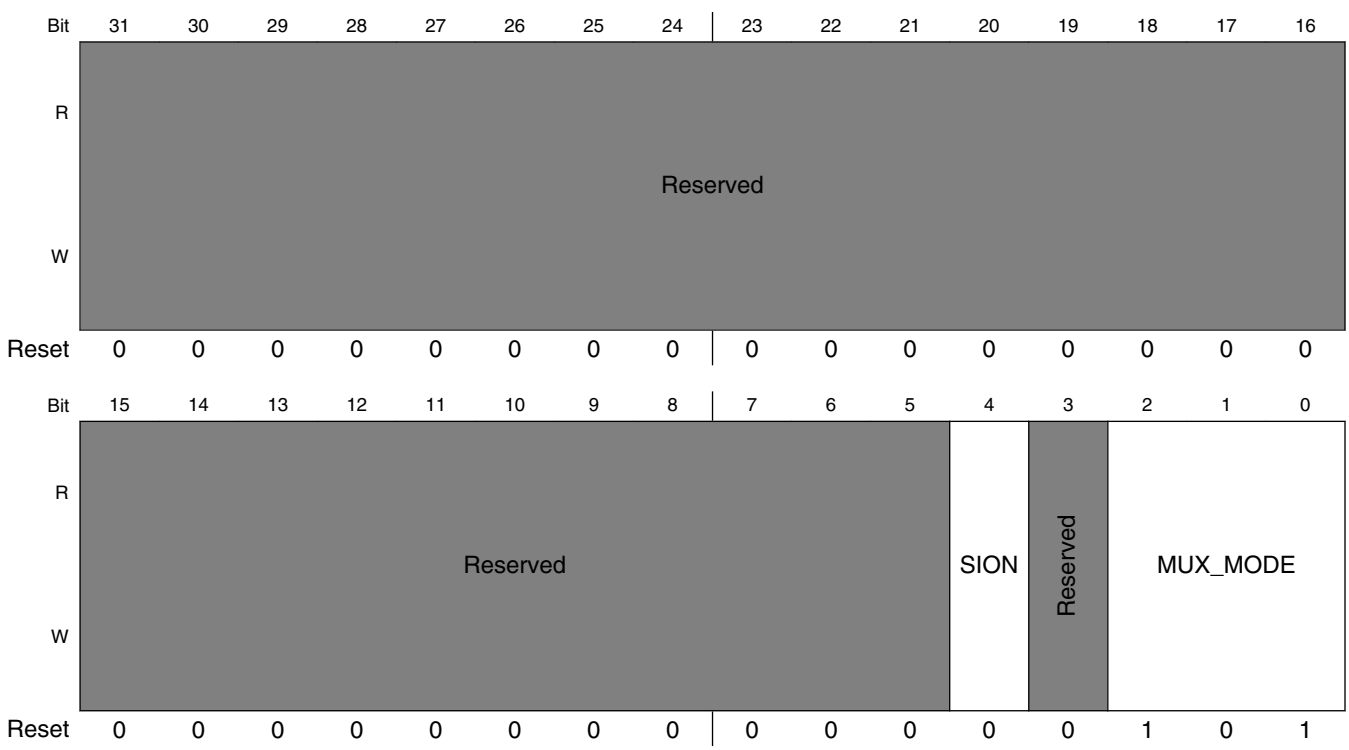


IOMUXC_SW_MUX_CTL_PAD_ENET_RD0 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ENET_RD0 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: ENET_RD0. 000 **ALT0** — Select signal ENET1_RGMII_RD0 101 **ALT5** — Select signal GPIO1_IO26

8.2.5.33 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_RD1)

Address: 3033_0000h base + 94h offset = 3033_0094h

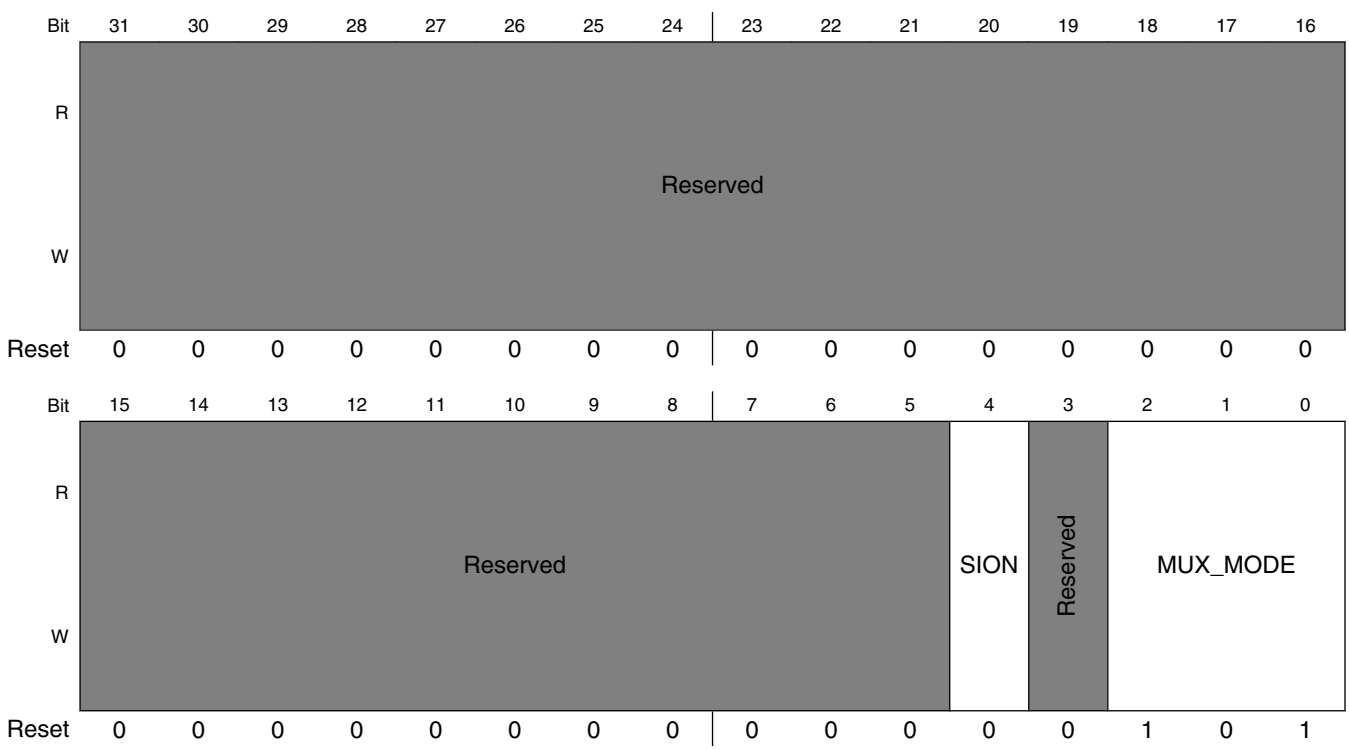


IOMUXC_SW_MUX_CTL_PAD_ENET_RD1 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ENET_RD1 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: ENET_RD1. 000 **ALT0** — Select signal ENET1_RGMII_RD1 101 **ALT5** — Select signal GPIO1_IO27

8.2.5.34 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_RD2)

Address: 3033_0000h base + 98h offset = 3033_0098h

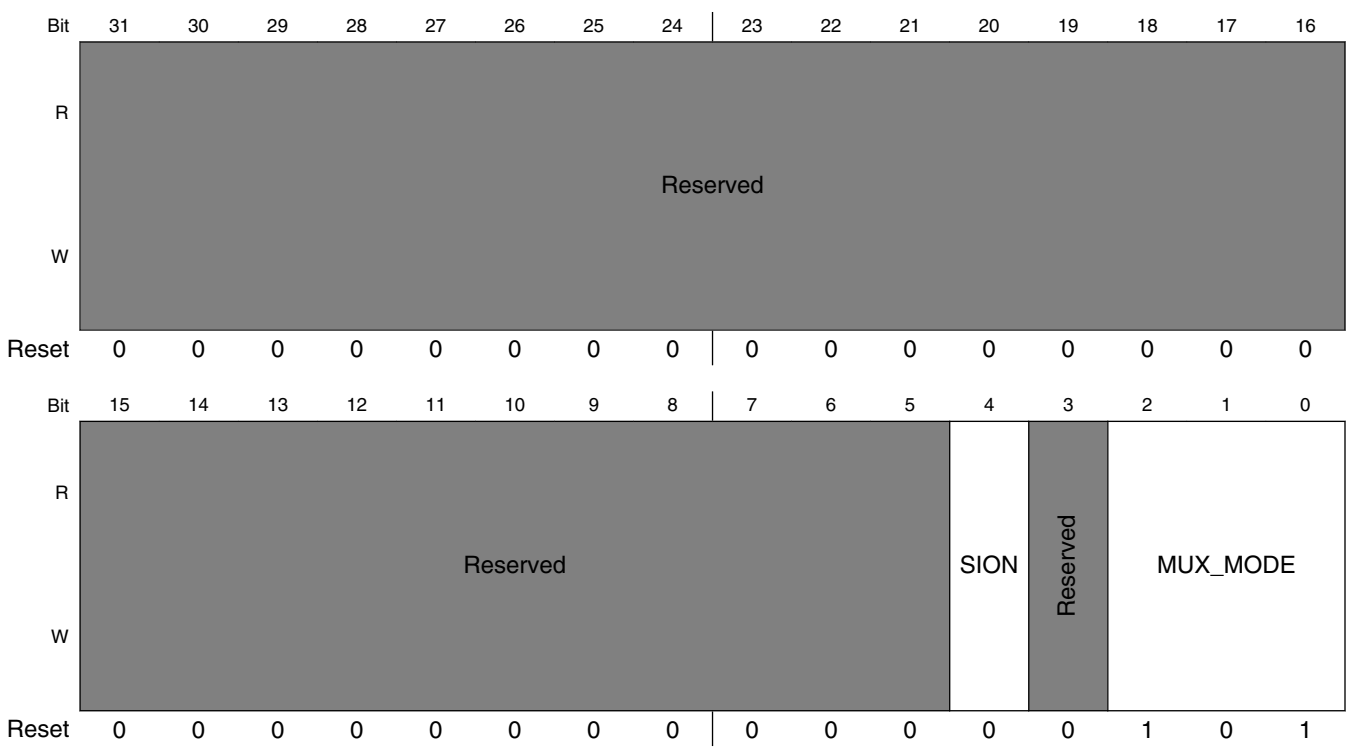


IOMUXC_SW_MUX_CTL_PAD_ENET_RD2 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ENET_RD2 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: ENET_RD2. 000 **ALT0** — Select signal ENET1_RGMII_RD2 101 **ALT5** — Select signal GPIO1_IO28

8.2.5.35 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ENET_RD3)

Address: 3033_0000h base + 9Ch offset = 3033_009Ch

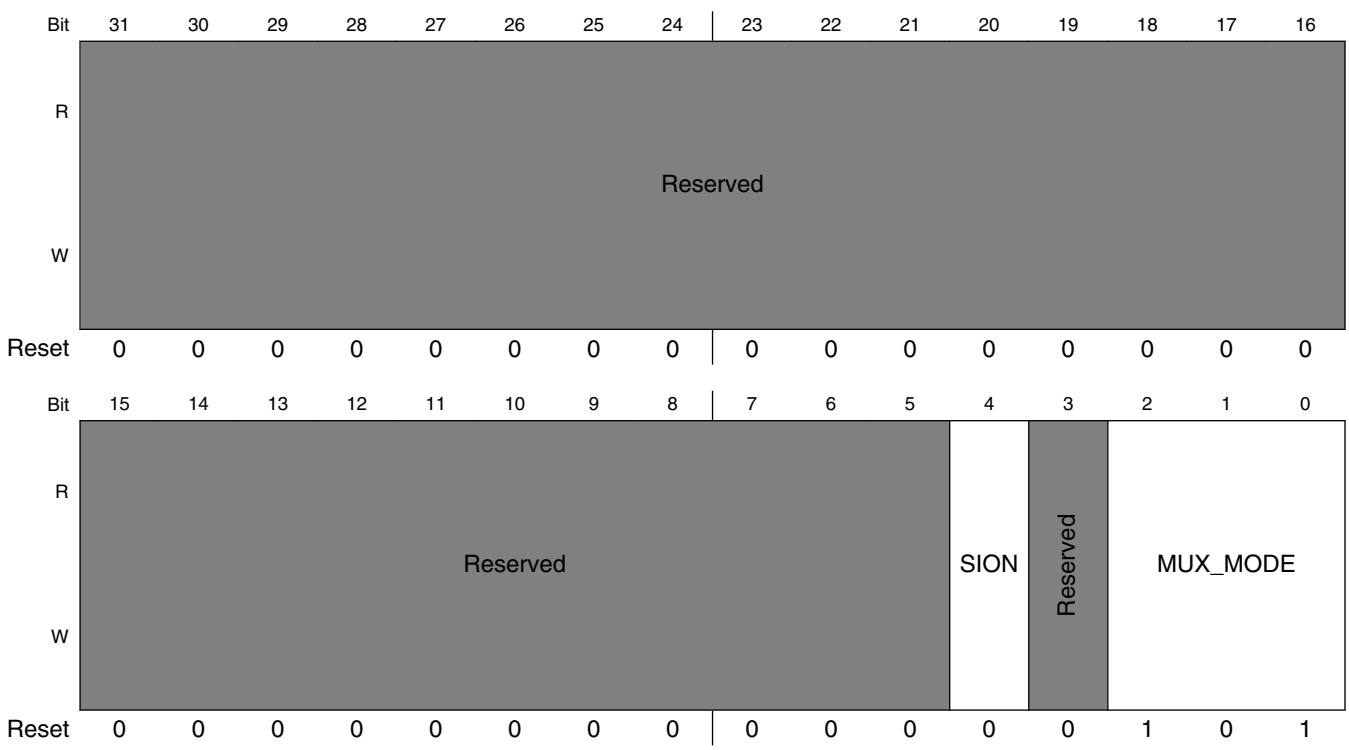


IOMUXC_SW_MUX_CTL_PAD_ENET_RD3 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ENET_RD3 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: ENET_RD3. 000 **ALT0** — Select signal ENET1_RGMII_RD3 101 **ALT5** — Select signal GPIO1_IO29

8.2.5.36 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_CLK)

Address: 3033_0000h base + A0h offset = 3033_00A0h



IOMUXC_SW_MUX_CTL_PAD_SD1_CLK field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD1_CLK 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD1_CLK. 000 **ALT0** — Select signal USDHC1_CLK 101 **ALT5** — Select signal GPIO2_IO00

8.2.5.37 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_CMD)

Address: 3033_0000h base + A4h offset = 3033_00A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

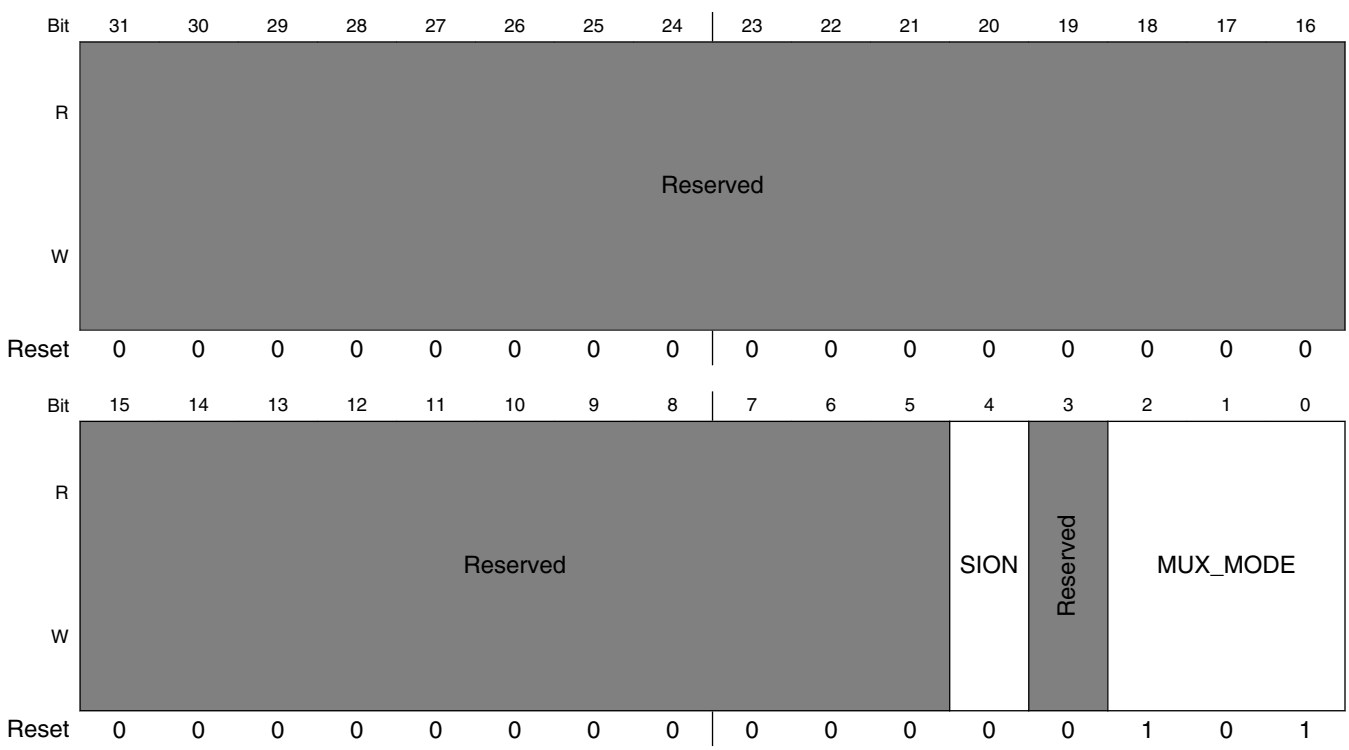
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_SD1_CMD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD1_CMD 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD1_CMD. 000 **ALT0** — Select signal USDHC1_CMD 101 **ALT5** — Select signal GPIO2_IO01

8.2.5.38 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0)

Address: 3033_0000h base + A8h offset = 3033_00A8h

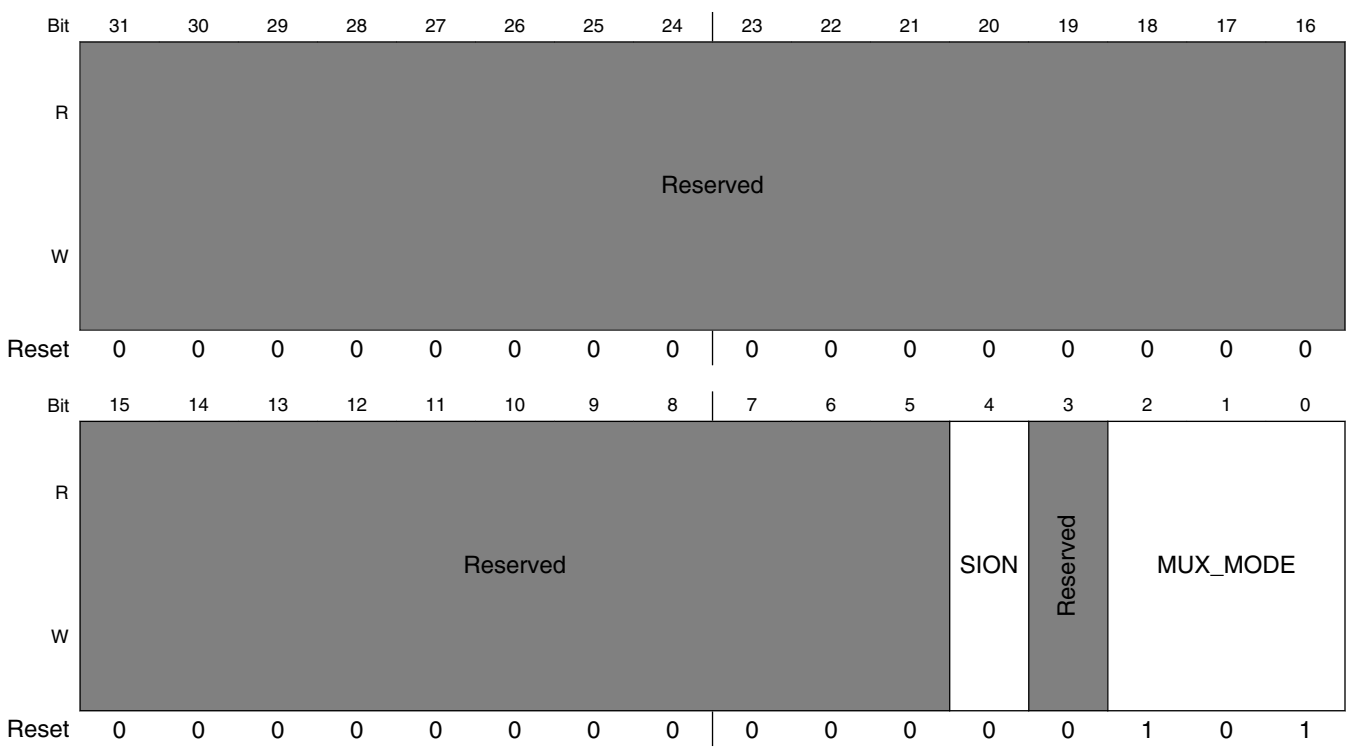


IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD1_DATA0 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD1_DATA0. 000 **ALT0** — Select signal USDHC1_DATA0 101 **ALT5** — Select signal GPIO2_IO02

8.2.5.39 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1)

Address: 3033_0000h base + ACh offset = 3033_00ACh

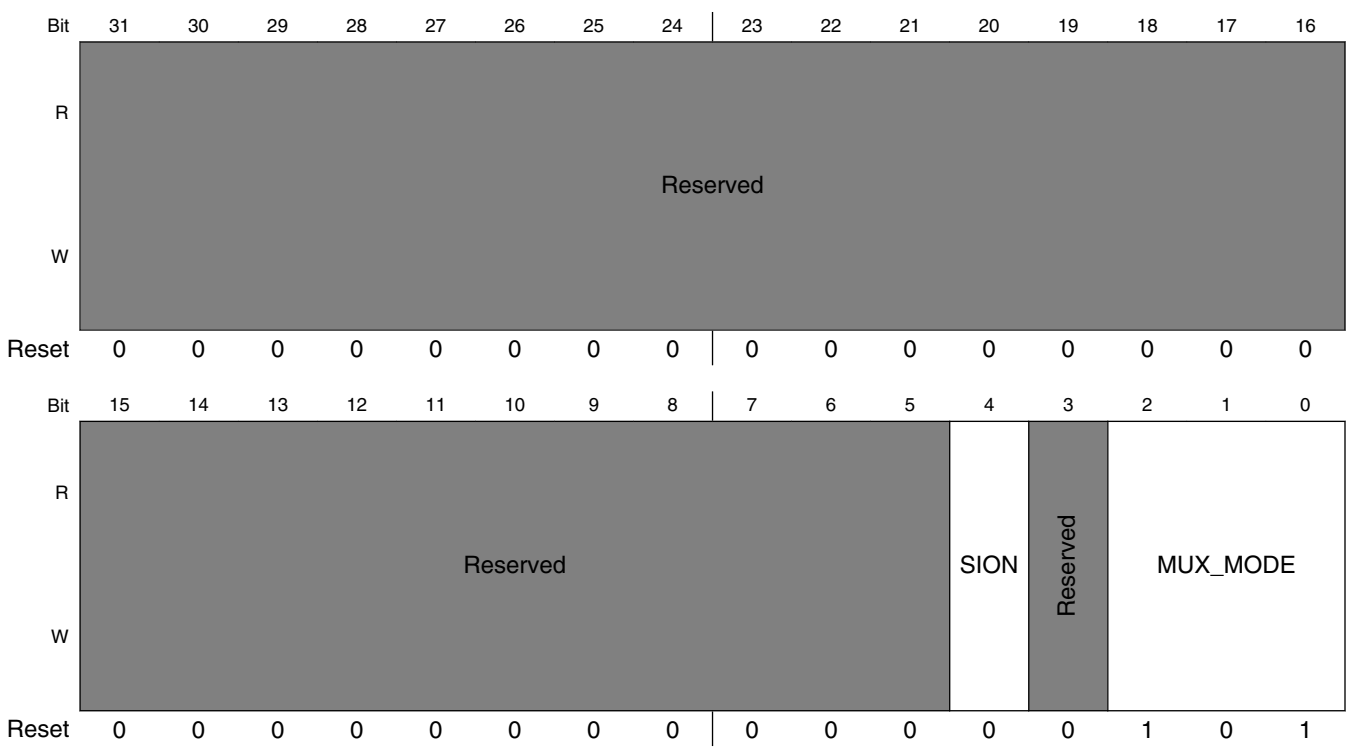


IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD1_DATA1 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD1_DATA1. 000 **ALT0** — Select signal USDHC1_DATA1 101 **ALT5** — Select signal GPIO2_IO03

8.2.5.40 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2)

Address: 3033_0000h base + B0h offset = 3033_00B0h

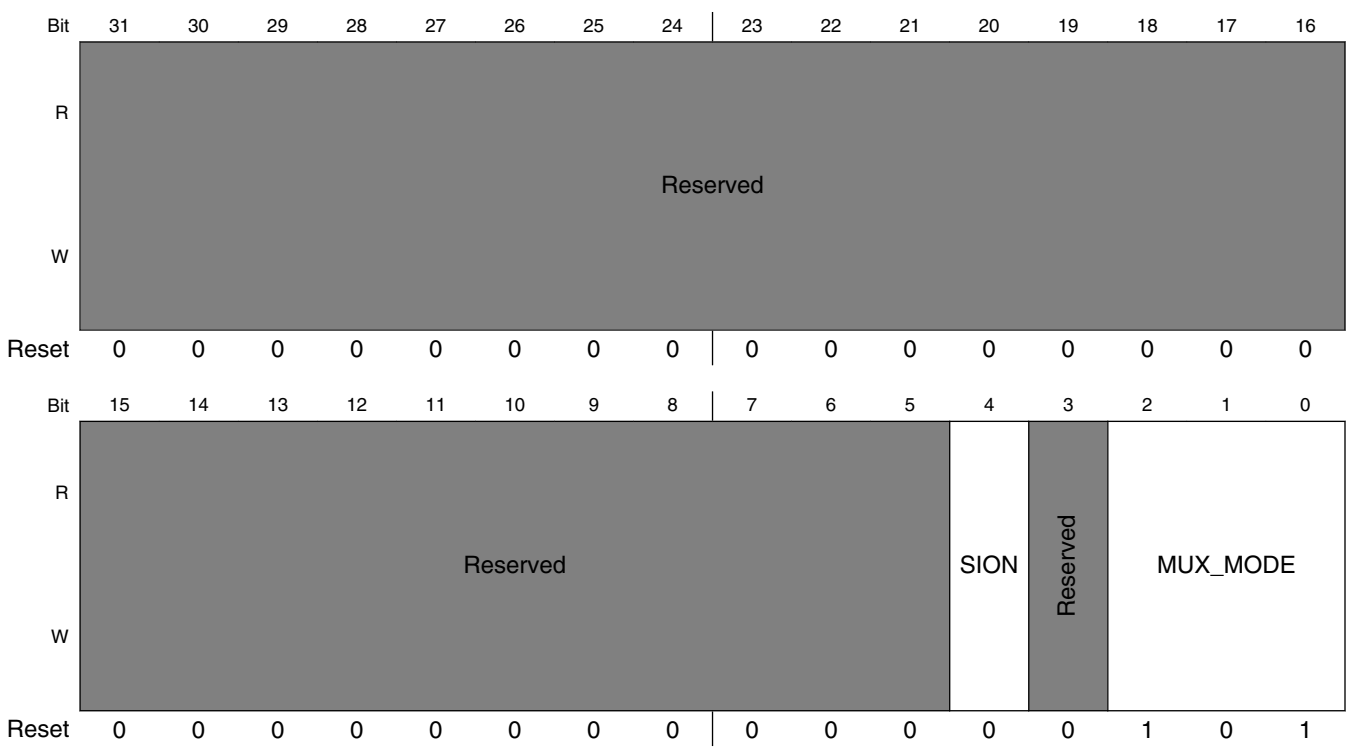


IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD1_DATA2 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD1_DATA2. 000 **ALT0** — Select signal USDHC1_DATA2 101 **ALT5** — Select signal GPIO2_IO04

8.2.5.41 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3)

Address: 3033_0000h base + B4h offset = 3033_00B4h

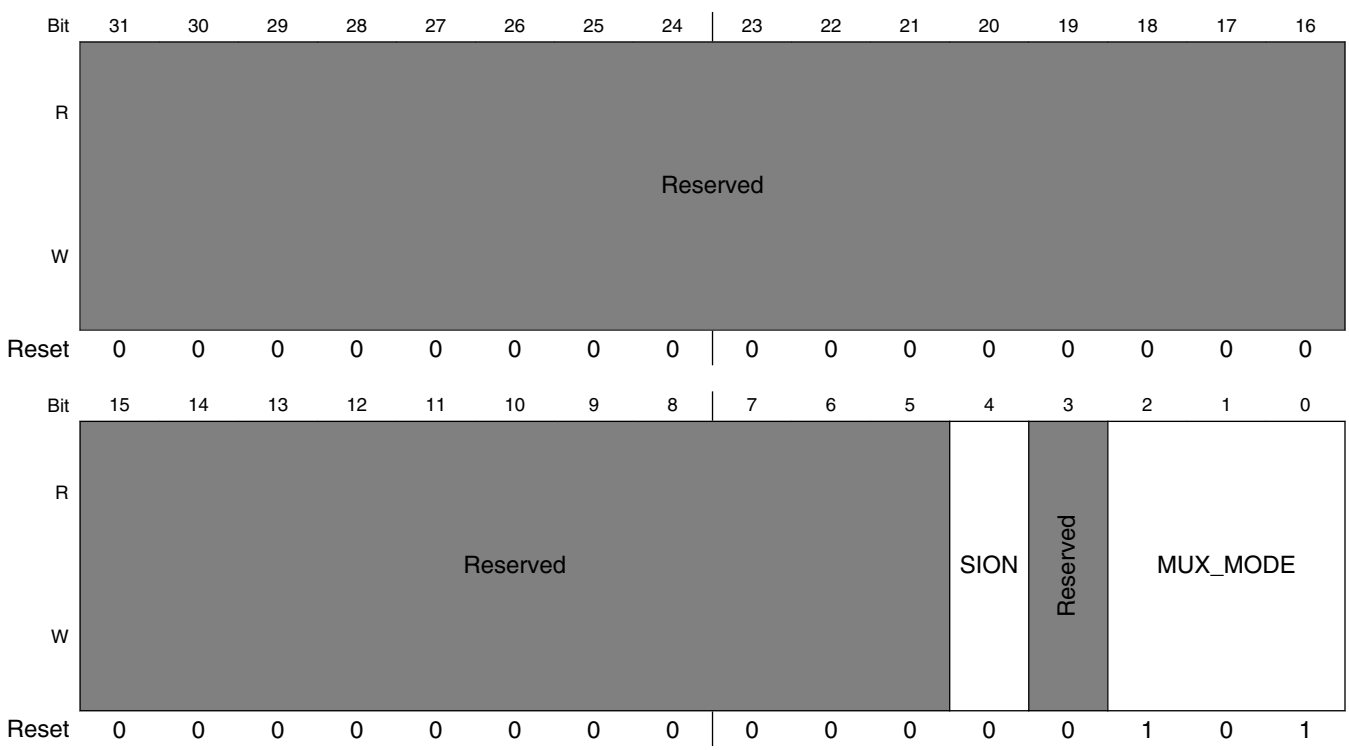


IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD1_DATA3 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD1_DATA3. 000 **ALT0** — Select signal USDHC1_DATA3 101 **ALT5** — Select signal GPIO2_IO05

8.2.5.42 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA4)

Address: 3033_0000h base + B8h offset = 3033_00B8h

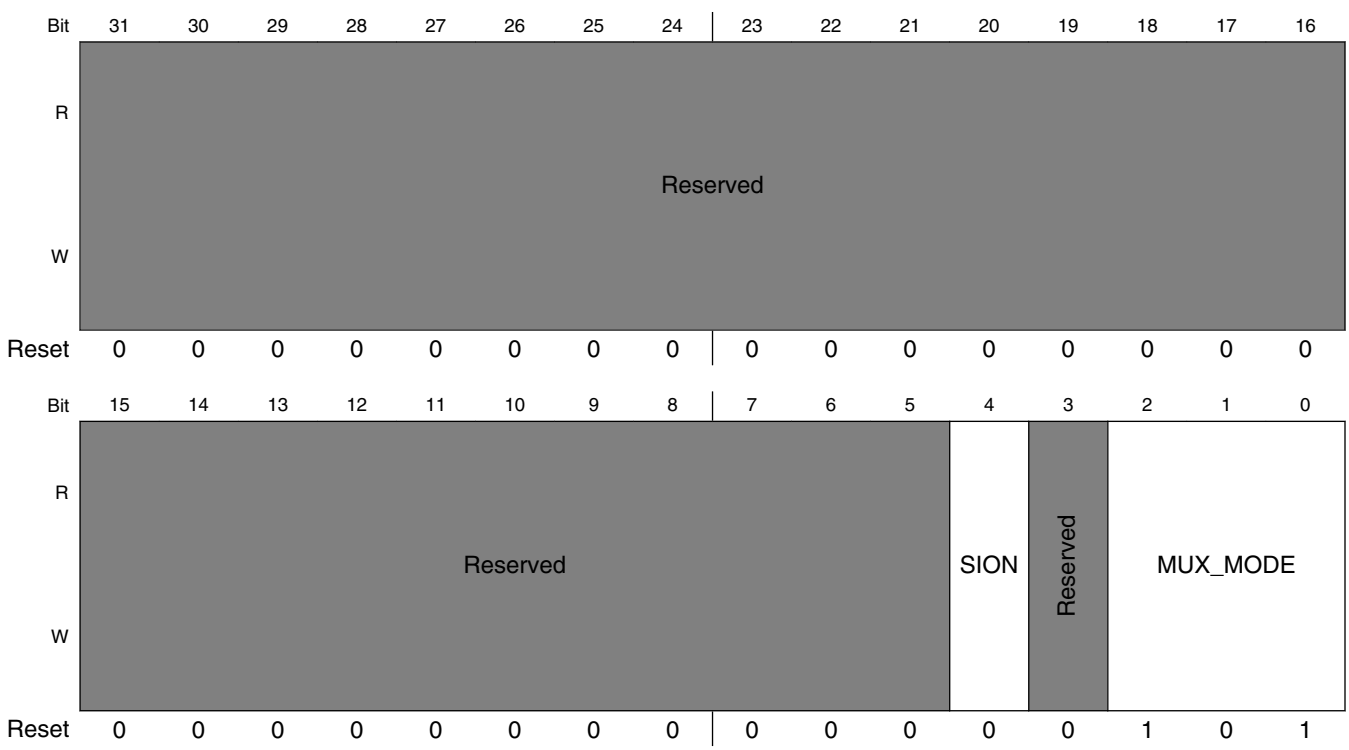


IOMUXC_SW_MUX_CTL_PAD_SD1_DATA4 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD1_DATA4 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD1_DATA4. 000 **ALT0** — Select signal USDHC1_DATA4 101 **ALT5** — Select signal GPIO2_IO06

8.2.5.43 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA5)

Address: 3033_0000h base + BCh offset = 3033_00BCh

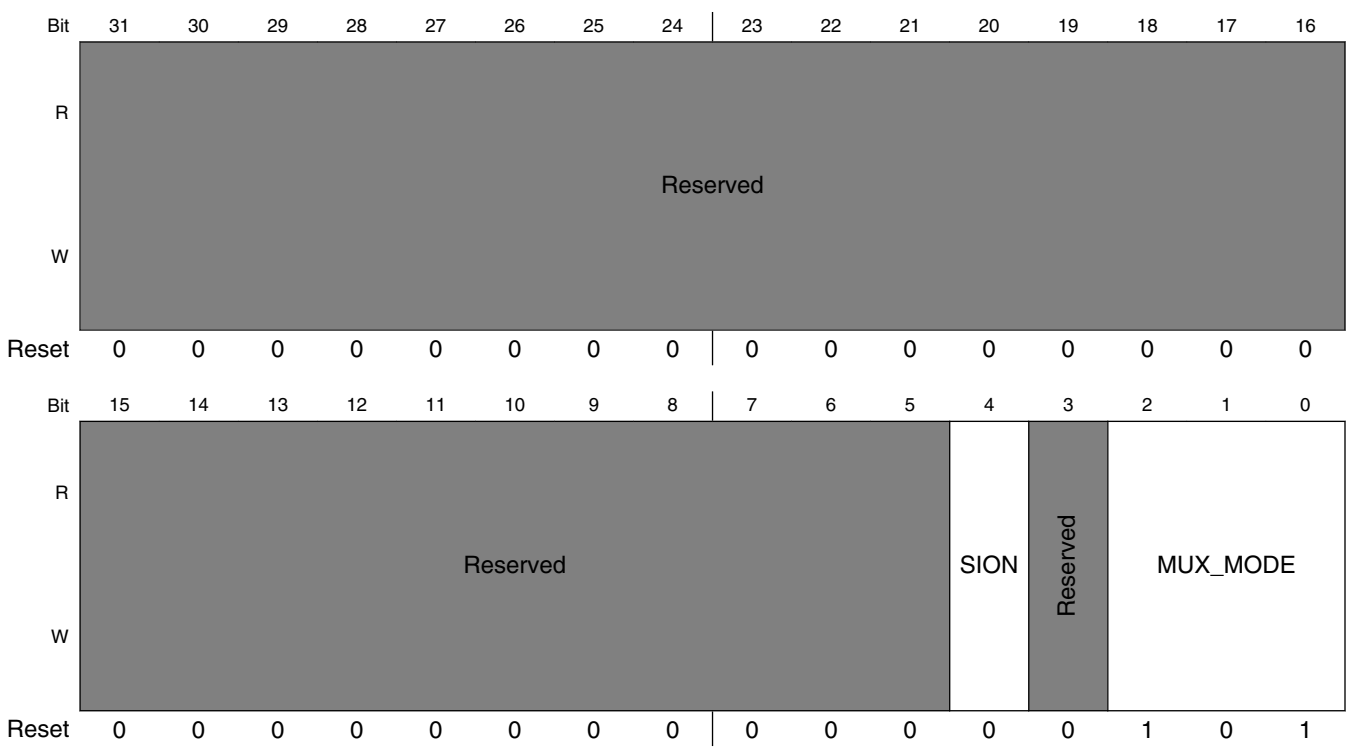


IOMUXC_SW_MUX_CTL_PAD_SD1_DATA5 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD1_DATA5 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD1_DATA5. 000 **ALT0** — Select signal USDHC1_DATA5 101 **ALT5** — Select signal GPIO2_IO07

8.2.5.44 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA6)

Address: 3033_0000h base + C0h offset = 3033_00C0h

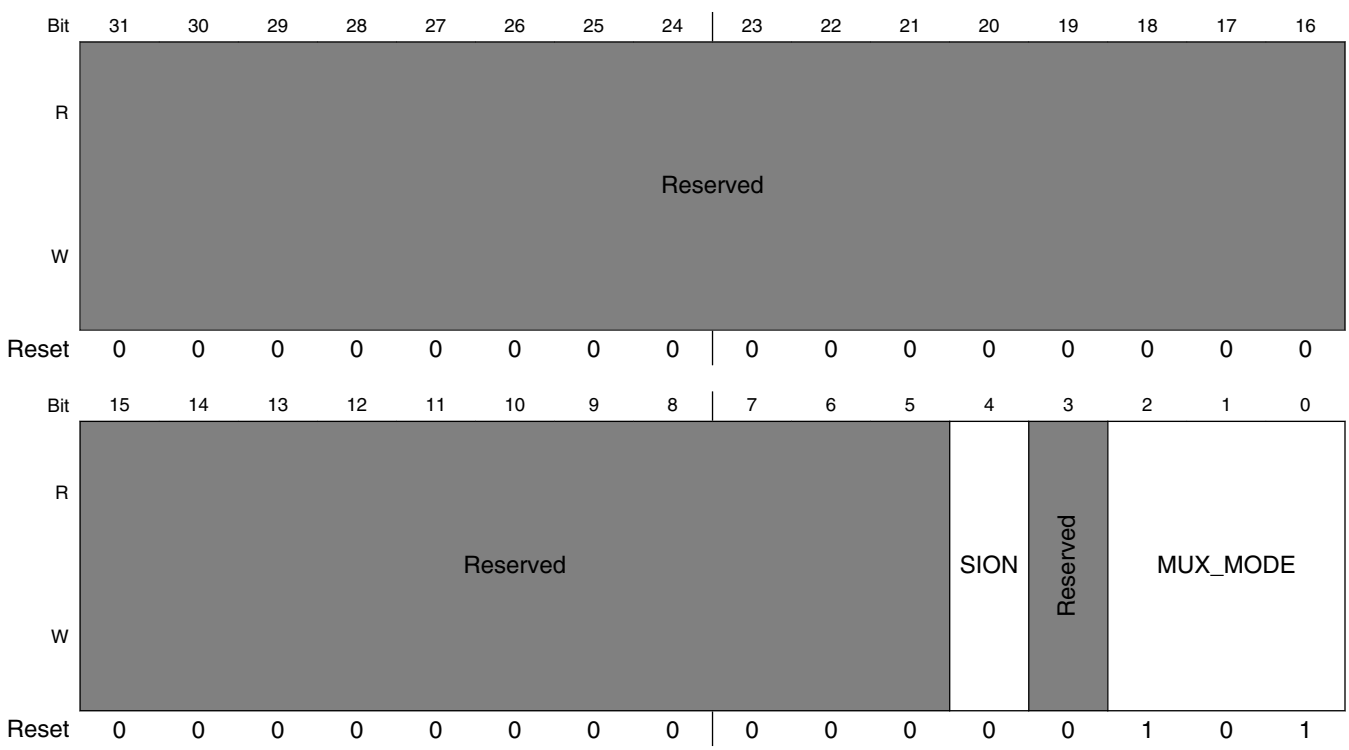


IOMUXC_SW_MUX_CTL_PAD_SD1_DATA6 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD1_DATA6 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD1_DATA6. 000 **ALT0** — Select signal USDHC1_DATA6 101 **ALT5** — Select signal GPIO2_IO08

8.2.5.45 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA7)

Address: 3033_0000h base + C4h offset = 3033_00C4h

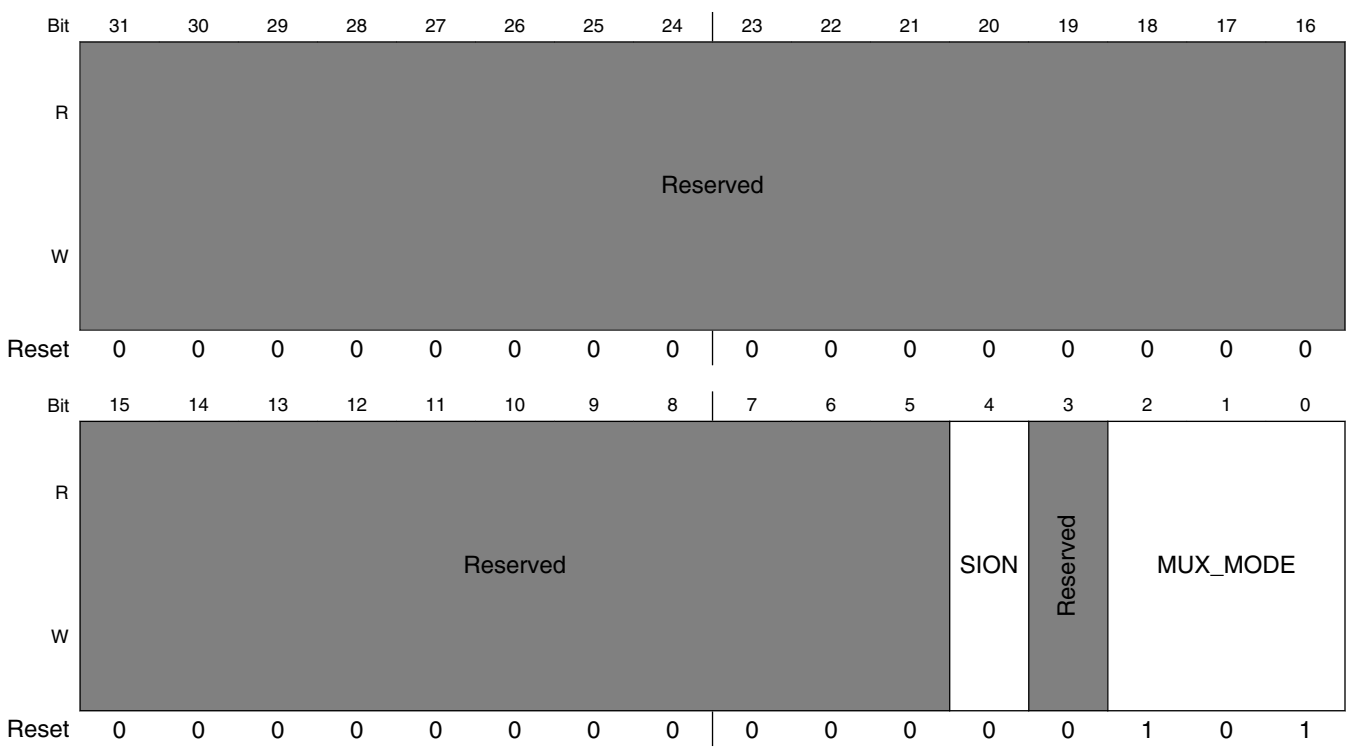


IOMUXC_SW_MUX_CTL_PAD_SD1_DATA7 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD1_DATA7 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD1_DATA7. 000 **ALT0** — Select signal USDHC1_DATA7 101 **ALT5** — Select signal GPIO2_IO09

8.2.5.46 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_RESET_B)

Address: 3033_0000h base + C8h offset = 3033_00C8h

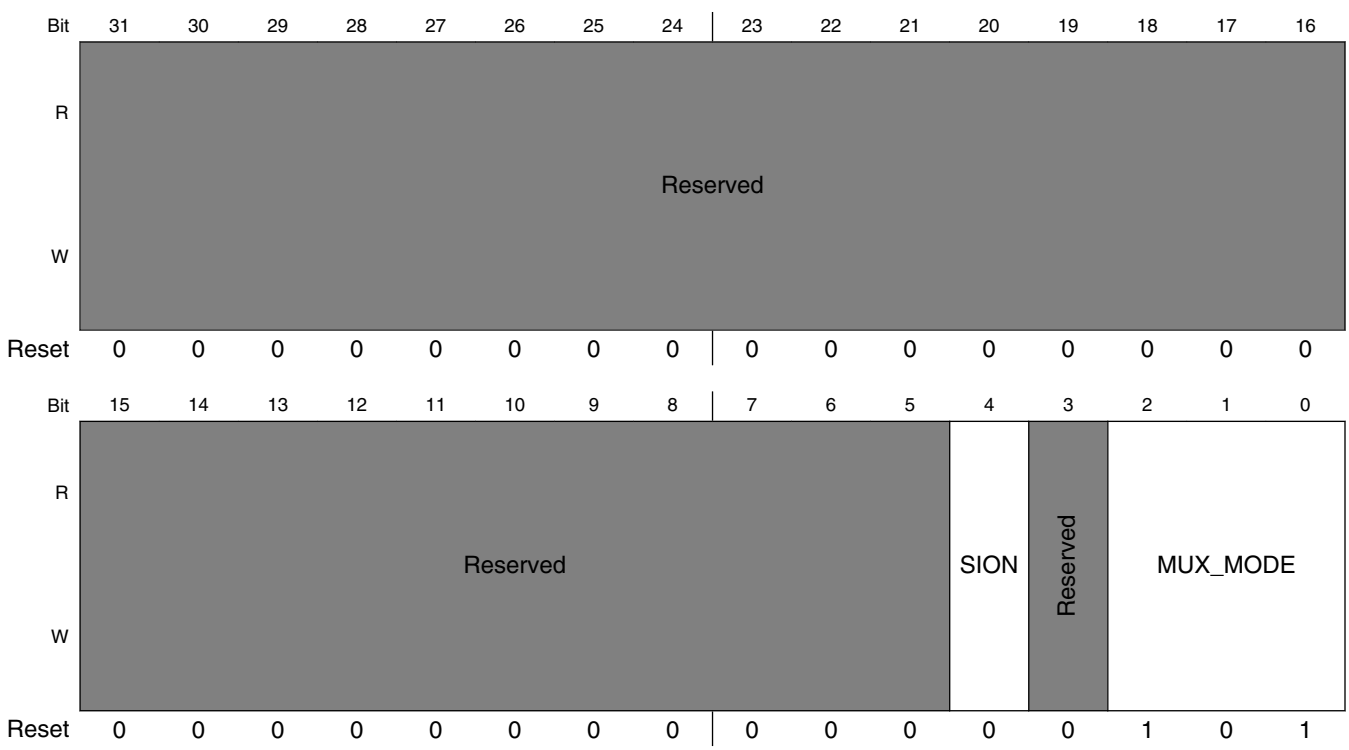


IOMUXC_SW_MUX_CTL_PAD_SD1_RESET_B field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD1_RESET_B 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD1_RESET_B. 000 **ALT0** — Select signal USDHC1_RESET_B 101 **ALT5** — Select signal GPIO2_IO10

8.2.5.47 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_STROBE)

Address: 3033_0000h base + CCh offset = 3033_00CCh

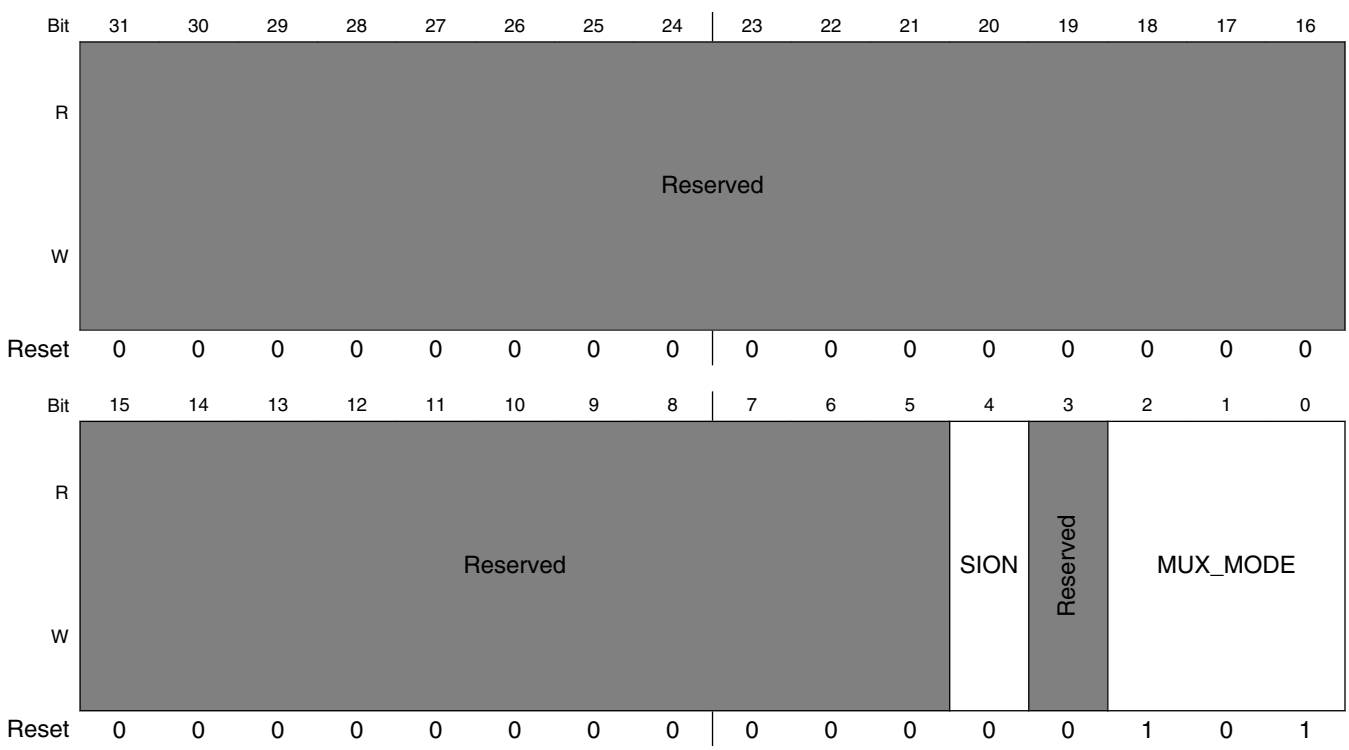


IOMUXC_SW_MUX_CTL_PAD_SD1_STROBE field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD1_STROBE 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD1_STROBE. 000 **ALT0** — Select signal USDHC1_STROBE 101 **ALT5** — Select signal GPIO2_IO11

8.2.5.48 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_CD_B)

Address: 3033_0000h base + D0h offset = 3033_00D0h

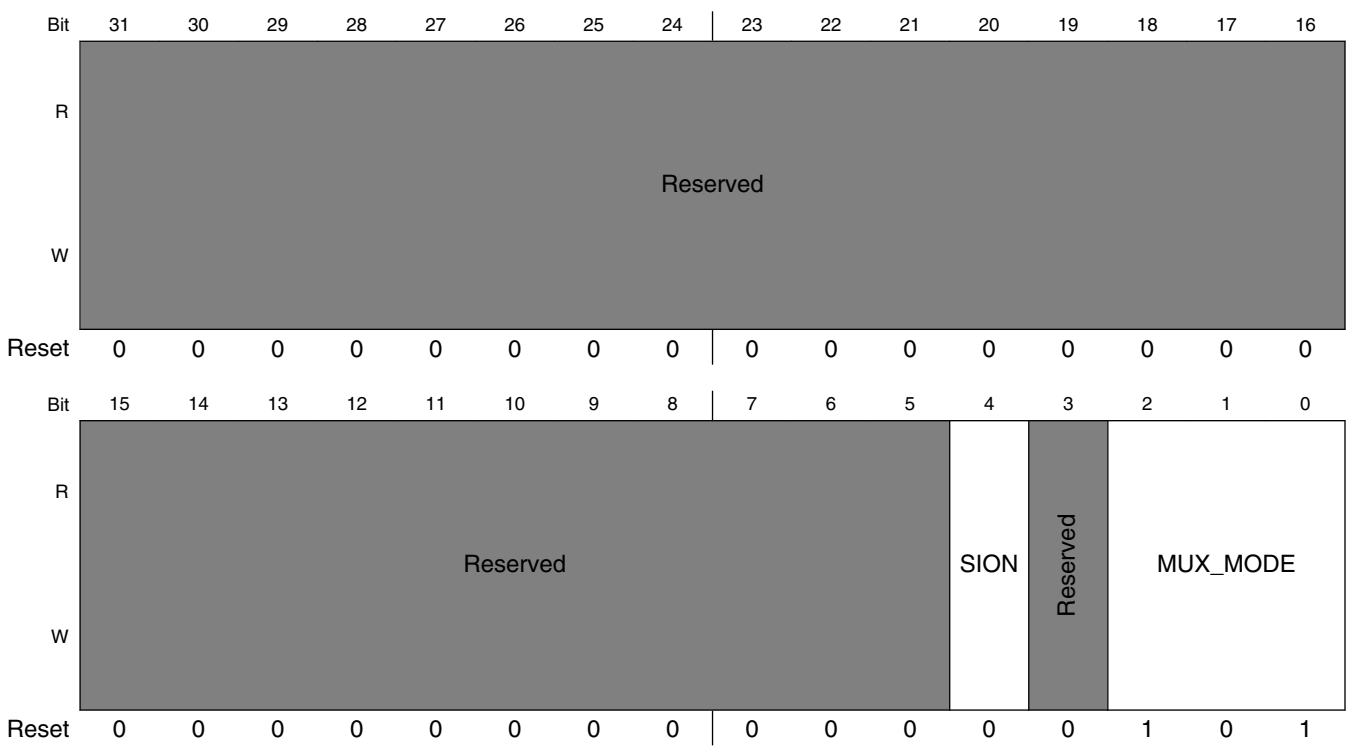


IOMUXC_SW_MUX_CTL_PAD_SD2_CD_B field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD2_CD_B 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD2_CD_B. 000 **ALT0** — Select signal USDHC2_CD_B 101 **ALT5** — Select signal GPIO2_IO12

8.2.5.49 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_CLK)

Address: 3033_0000h base + D4h offset = 3033_00D4h

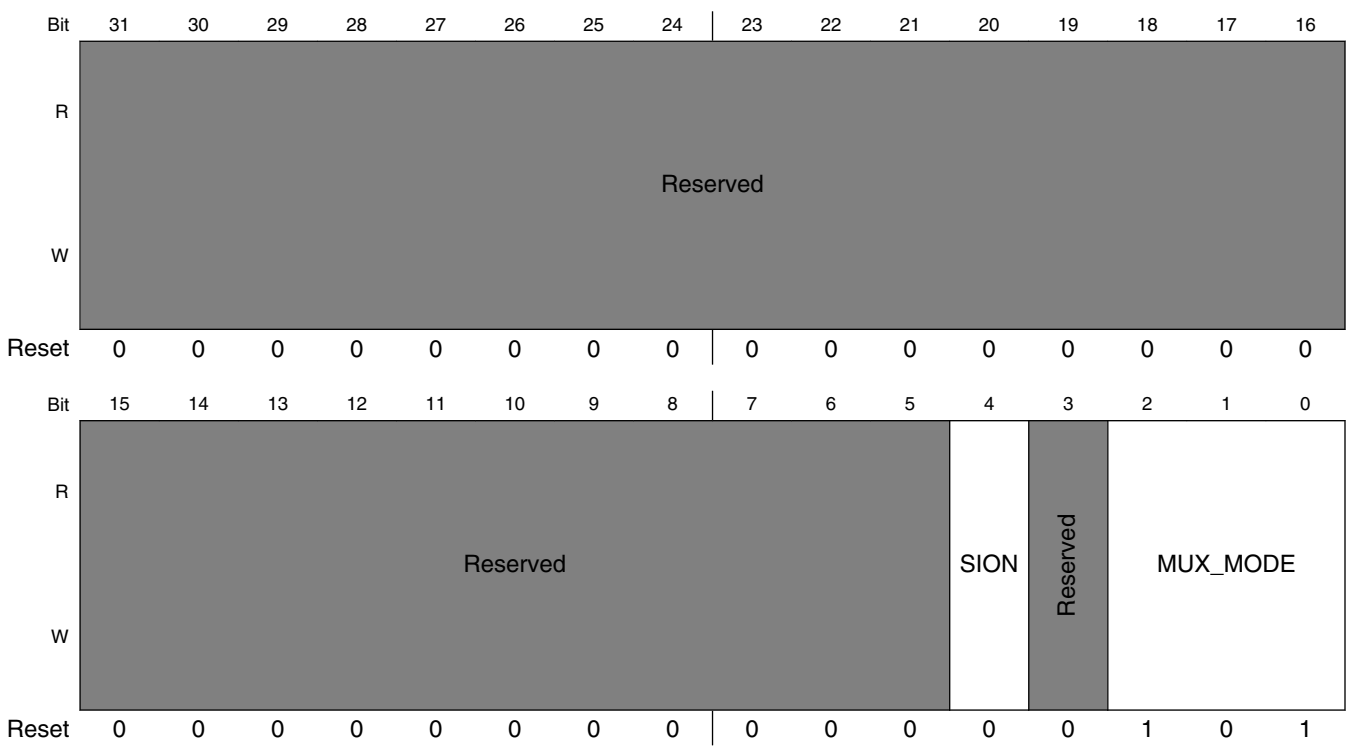


IOMUXC_SW_MUX_CTL_PAD_SD2_CLK field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD2_CLK 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD2_CLK. 000 **ALT0** — Select signal USDHC2_CLK 101 **ALT5** — Select signal GPIO2_IO13

8.2.5.50 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_CMD)

Address: 3033_0000h base + D8h offset = 3033_00D8h



IOMUXC_SW_MUX_CTL_PAD_SD2_CMD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD2_CMD 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD2_CMD. 000 **ALT0** — Select signal USDHC2_CMD 101 **ALT5** — Select signal GPIO2_IO14

8.2.5.51 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0)

Address: 3033_0000h base + DCh offset = 3033_00DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

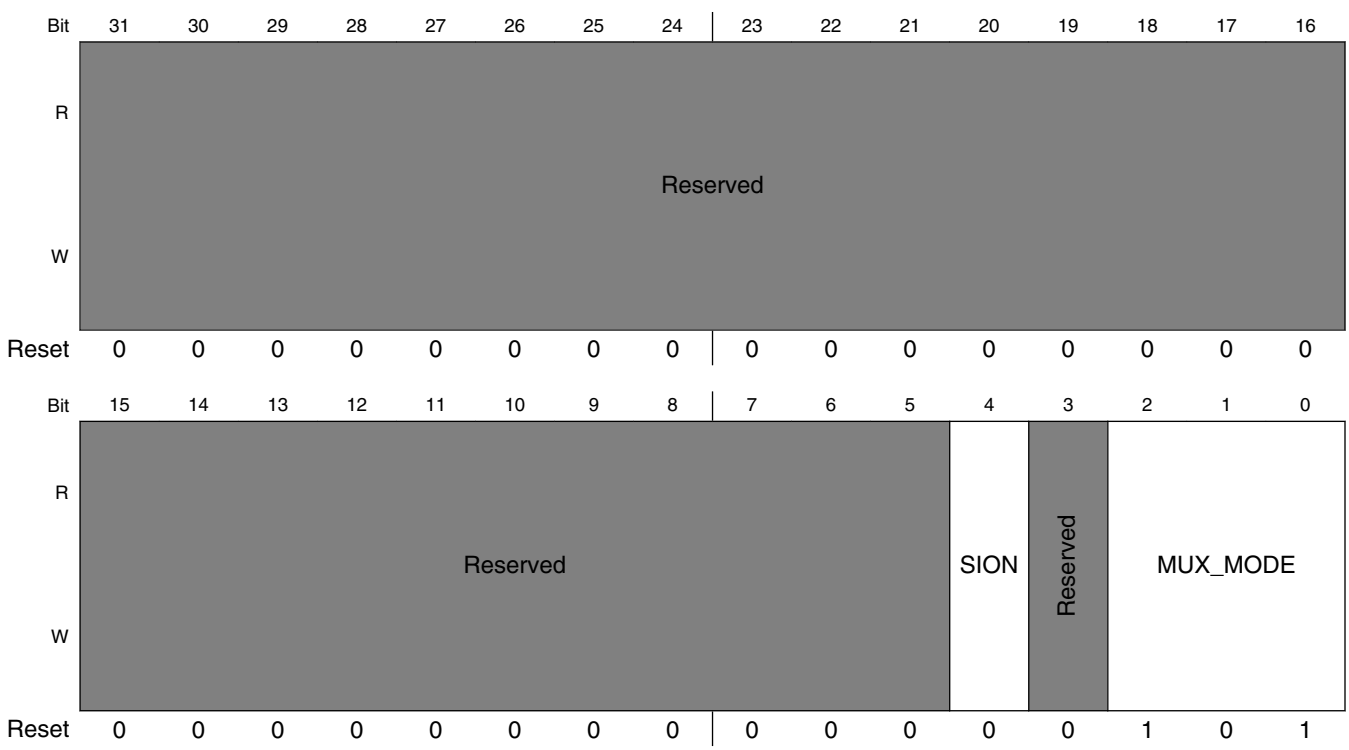
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD2_DATA0 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD2_DATA0. 000 **ALT0** — Select signal USDHC2_DATA0 101 **ALT5** — Select signal GPIO2_IO15

8.2.5.52 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1)

Address: 3033_0000h base + E0h offset = 3033_00E0h

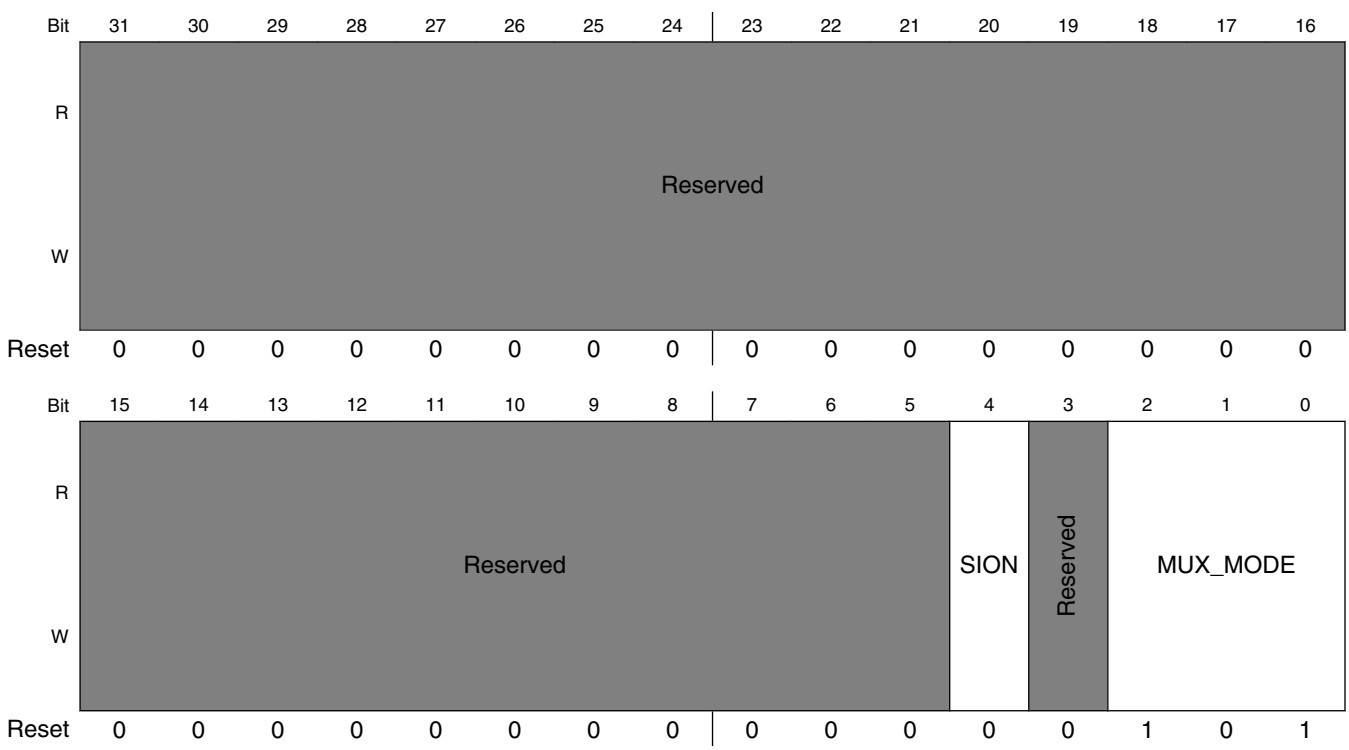


IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD2_DATA1 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD2_DATA1. 000 **ALT0** — Select signal USDHC2_DATA1 101 **ALT5** — Select signal GPIO2_IO16

8.2.5.53 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2)

Address: 3033_0000h base + E4h offset = 3033_00E4h

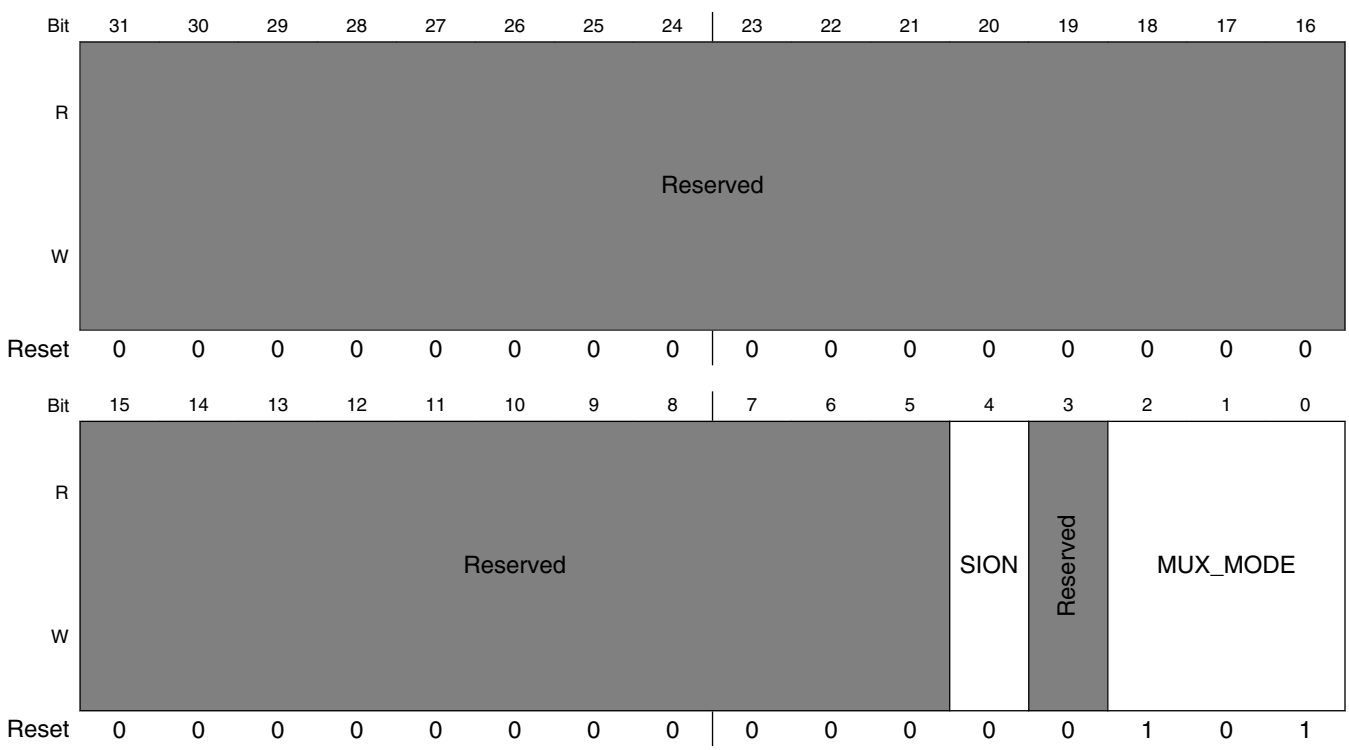


IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD2_DATA2 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD2_DATA2. 000 **ALT0** — Select signal USDHC2_DATA2 101 **ALT5** — Select signal GPIO2_IO17

8.2.5.54 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3)

Address: 3033_0000h base + E8h offset = 3033_00E8h

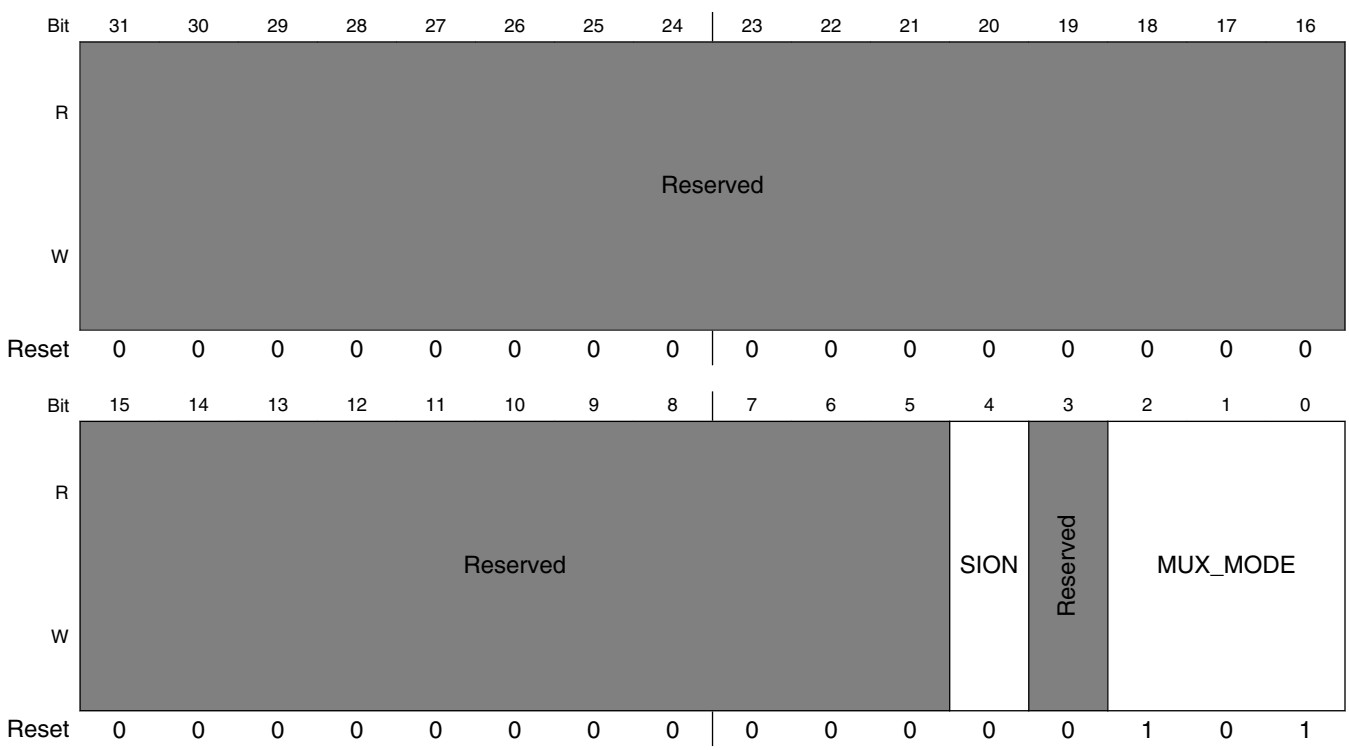


IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD2_DATA3 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD2_DATA3. 000 **ALT0** — Select signal USDHC2_DATA3 101 **ALT5** — Select signal GPIO2_IO18

8.2.5.55 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_RESET_B)

Address: 3033_0000h base + ECh offset = 3033_00ECh

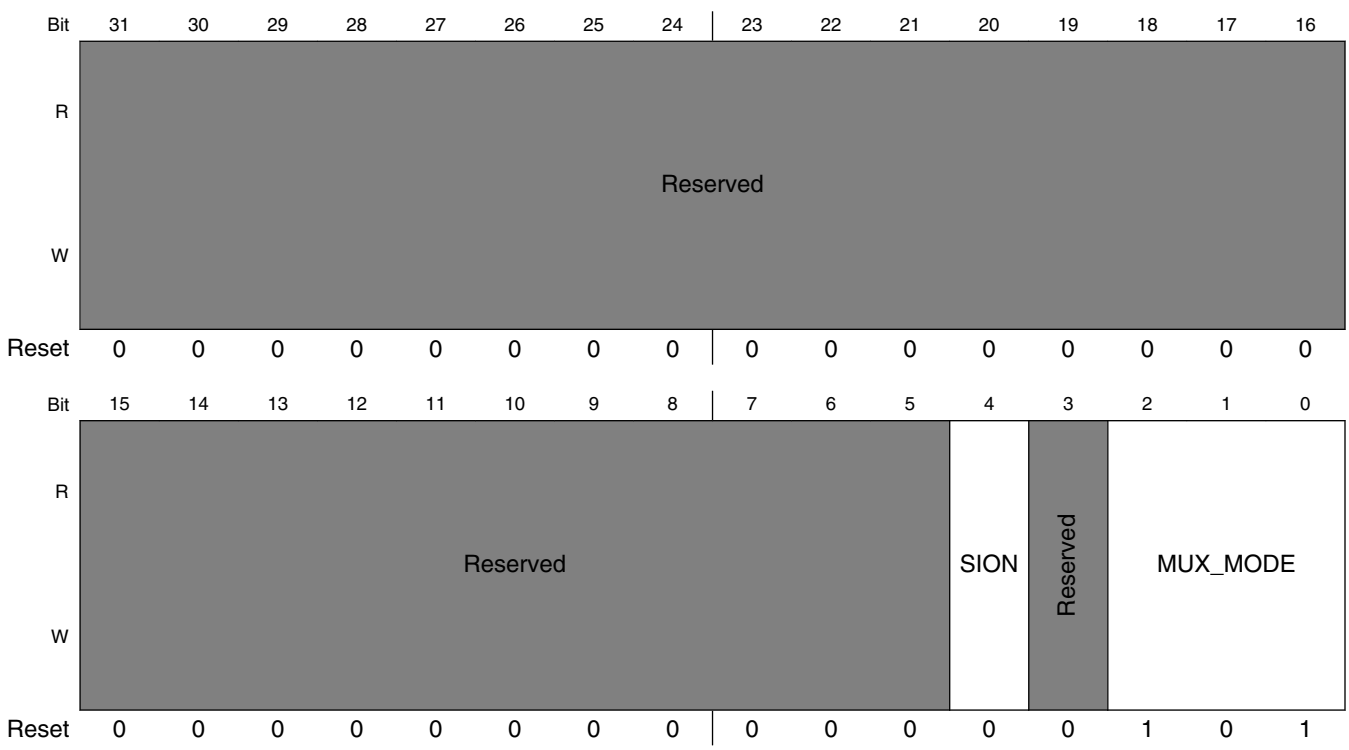


IOMUXC_SW_MUX_CTL_PAD_SD2_RESET_B field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD2_RESET_B 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD2_RESET_B. 000 **ALT0** — Select signal USDHC2_RESET_B 101 **ALT5** — Select signal GPIO2_IO19

8.2.5.56 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_WP)

Address: 3033_0000h base + F0h offset = 3033_00F0h

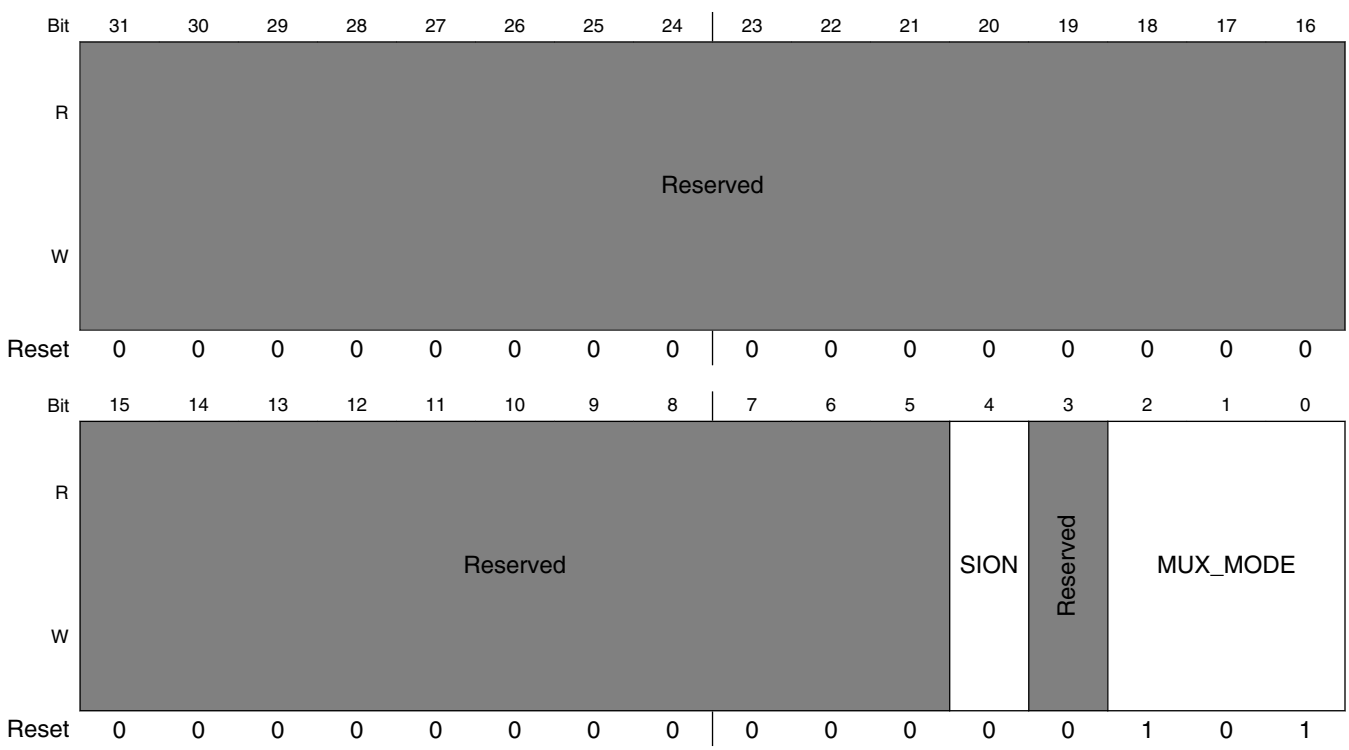


IOMUXC_SW_MUX_CTL_PAD_SD2_WP field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD2_WP 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for pad: SD2_WP. 000 **ALT0** — Select signal USDHC2_WP 101 **ALT5** — Select signal GPIO2_IO20

8.2.5.57 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_ALE)

Address: 3033_0000h base + F4h offset = 3033_00F4h

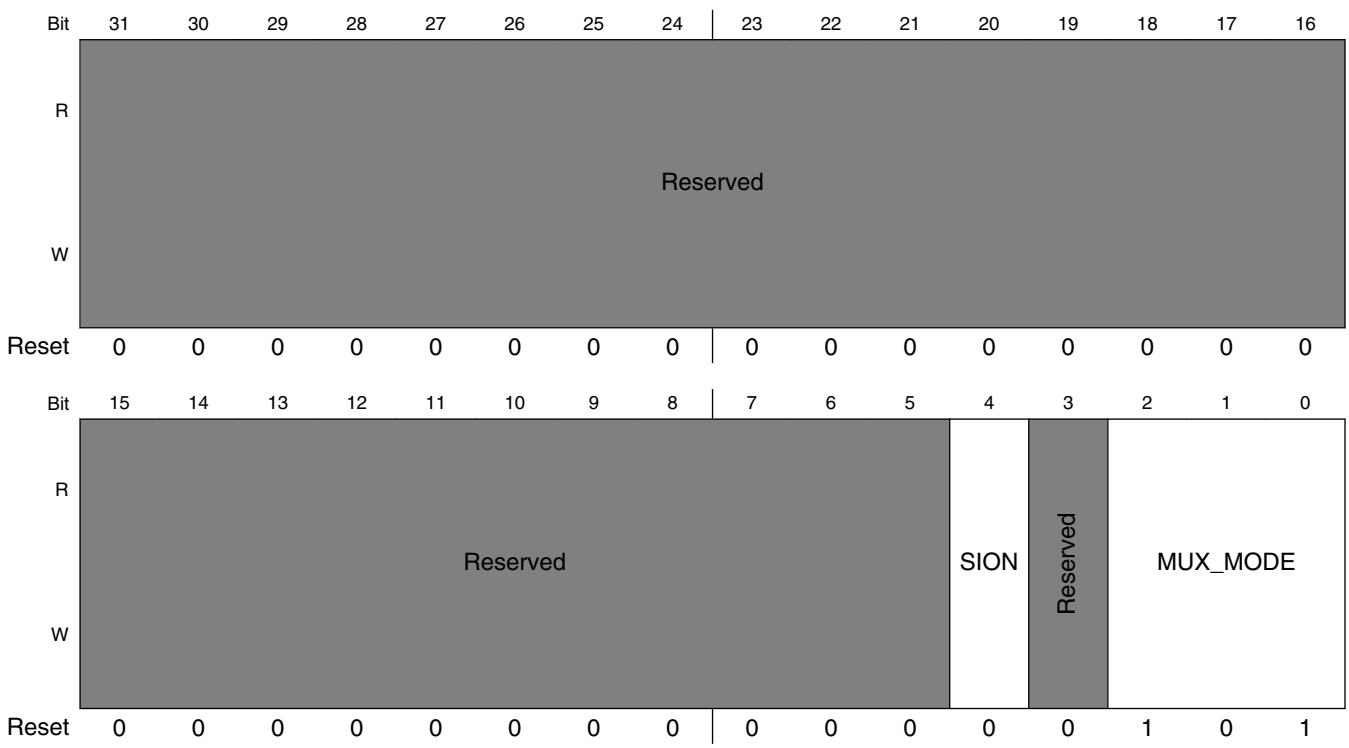


IOMUXC_SW_MUX_CTL_PAD_NAND_ALE field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_ALE 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: NAND_ALE. 000 **ALT0** — Select signal RAWNAND_ALE 001 **ALT1** — Select signal QSPI_A_SCLK 101 **ALT5** — Select signal GPIO3_IO00

8.2.5.58 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_CE0_B)

Address: 3033_0000h base + F8h offset = 3033_00F8h



IOMUXC_SW_MUX_CTL_PAD_NAND_CE0_B field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_CE0_B 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: NAND_CE0_B. 000 **ALT0** — Select signal RAWNAND_CE0_B 001 **ALT1** — Select signal QSPI_A_SS0_B 101 **ALT5** — Select signal GPIO3_IO01

8.2.5.59 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_CE1_B)

Address: 3033_0000h base + FCh offset = 3033_00FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

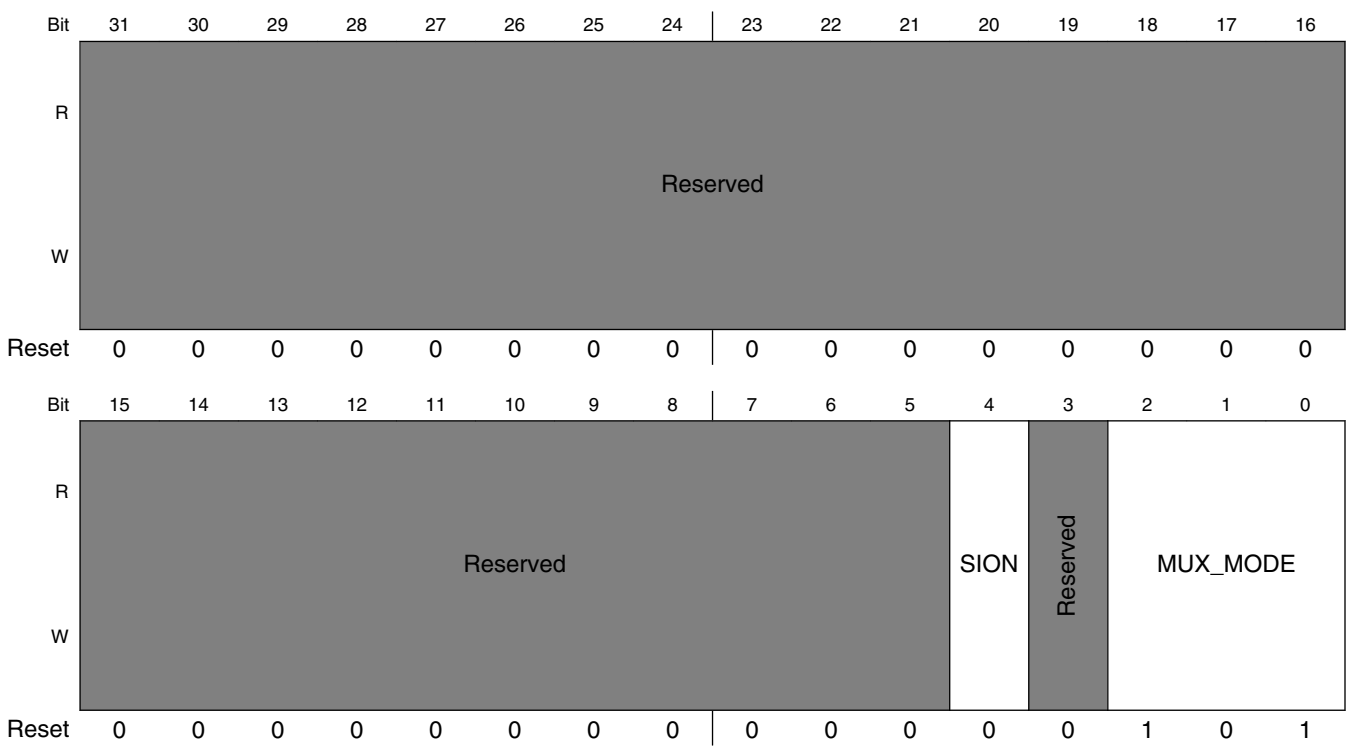
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_NAND_CE1_B field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_CE1_B 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: NAND_CE1_B. 000 **ALT0** — Select signal RAWNAND_CE1_B 001 **ALT1** — Select signal QSPI_A_SS1_B 010 **ALT2** — Select signal USDHC3_STROBE 101 **ALT5** — Select signal GPIO3_IO02

8.2.5.60 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_CE2_B)

Address: 3033_0000h base + 100h offset = 3033_0100h

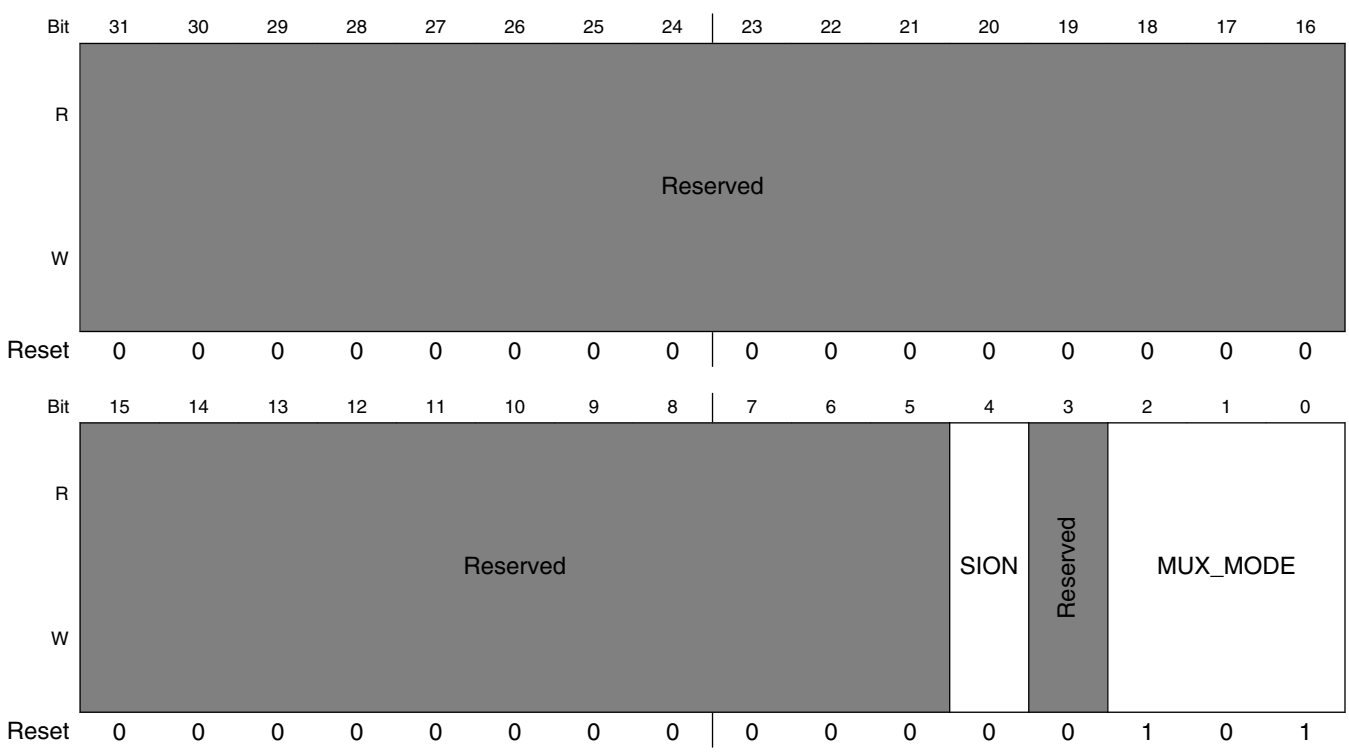


IOMUXC_SW_MUX_CTL_PAD_NAND_CE2_B field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_CE2_B 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: NAND_CE2_B. 000 **ALT0** — Select signal RAWNAND_CE2_B 001 **ALT1** — Select signal QSPI_B_SS0_B 010 **ALT2** — Select signal USDHC3_DATA5 101 **ALT5** — Select signal GPIO3_IO03

8.2.5.61 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_CE3_B)

Address: 3033_0000h base + 104h offset = 3033_0104h

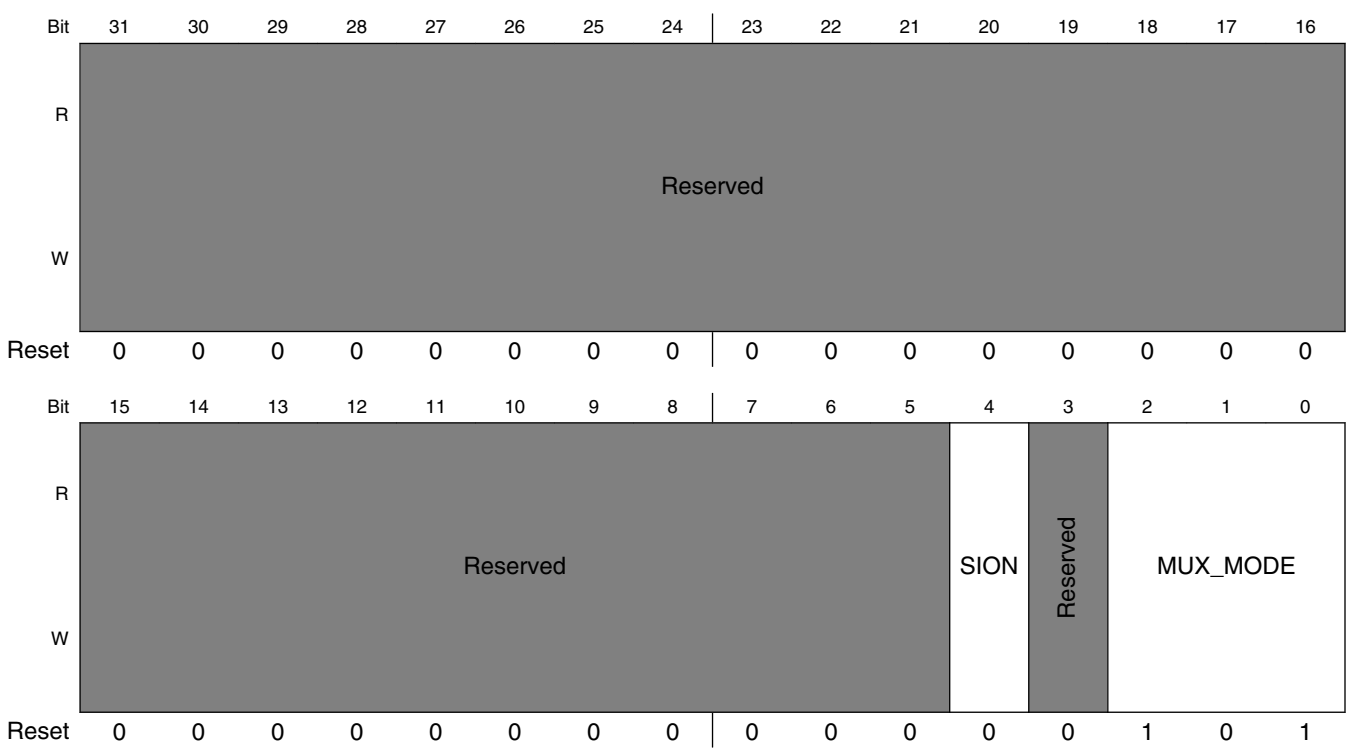


IOMUXC_SW_MUX_CTL_PAD_NAND_CE3_B field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_CE3_B 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: NAND_CE3_B. 000 **ALT0** — Select signal RAWNAND_CE3_B 001 **ALT1** — Select signal QSPI_B_SS1_B 010 **ALT2** — Select signal USDHC3_DATA6 101 **ALT5** — Select signal GPIO3_IO04

8.2.5.62 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_CLE)

Address: 3033_0000h base + 108h offset = 3033_0108h

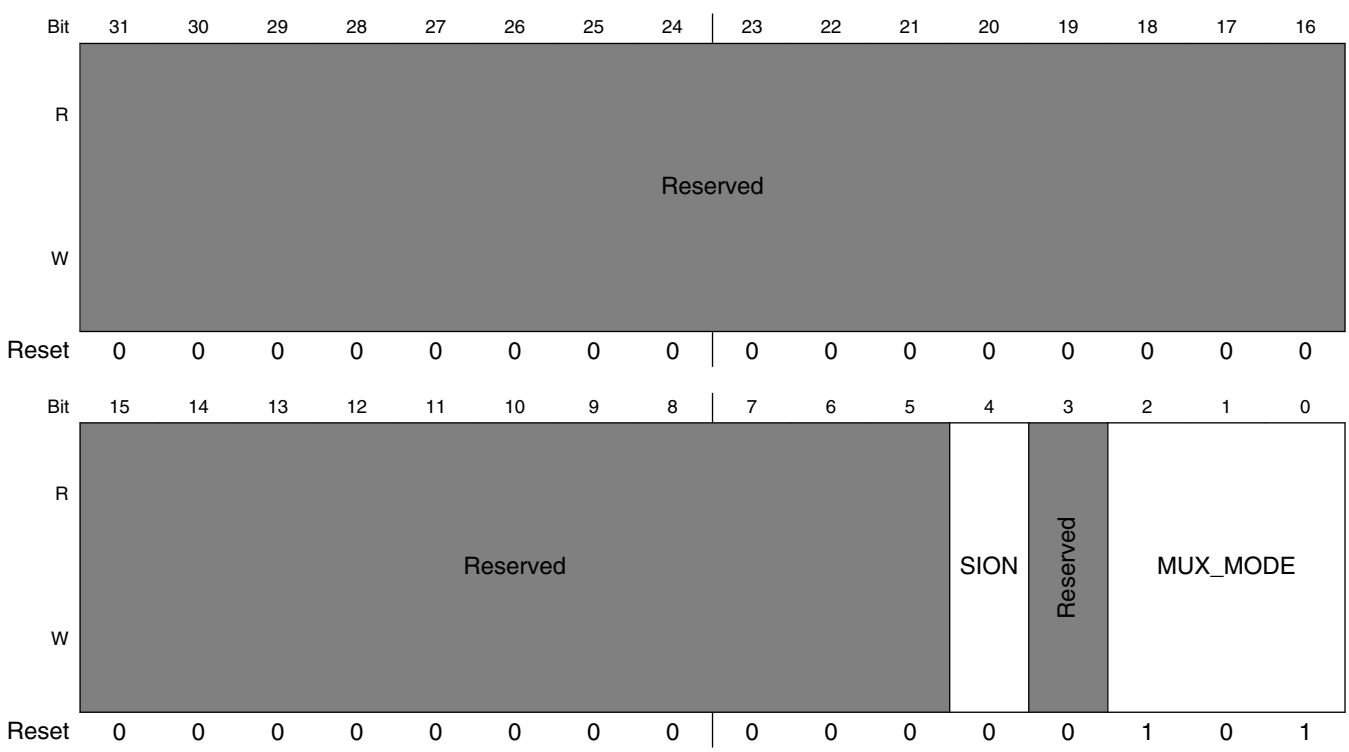


IOMUXC_SW_MUX_CTL_PAD_NAND_CLE field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_CLE 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: NAND_CLE. 000 **ALT0** — Select signal RAWNAND_CLE 001 **ALT1** — Select signal QSPI_B_SCLK 010 **ALT2** — Select signal USDHC3_DATA7 101 **ALT5** — Select signal GPIO3_IO05

8.2.5.63 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA00)

Address: 3033_0000h base + 10Ch offset = 3033_010Ch

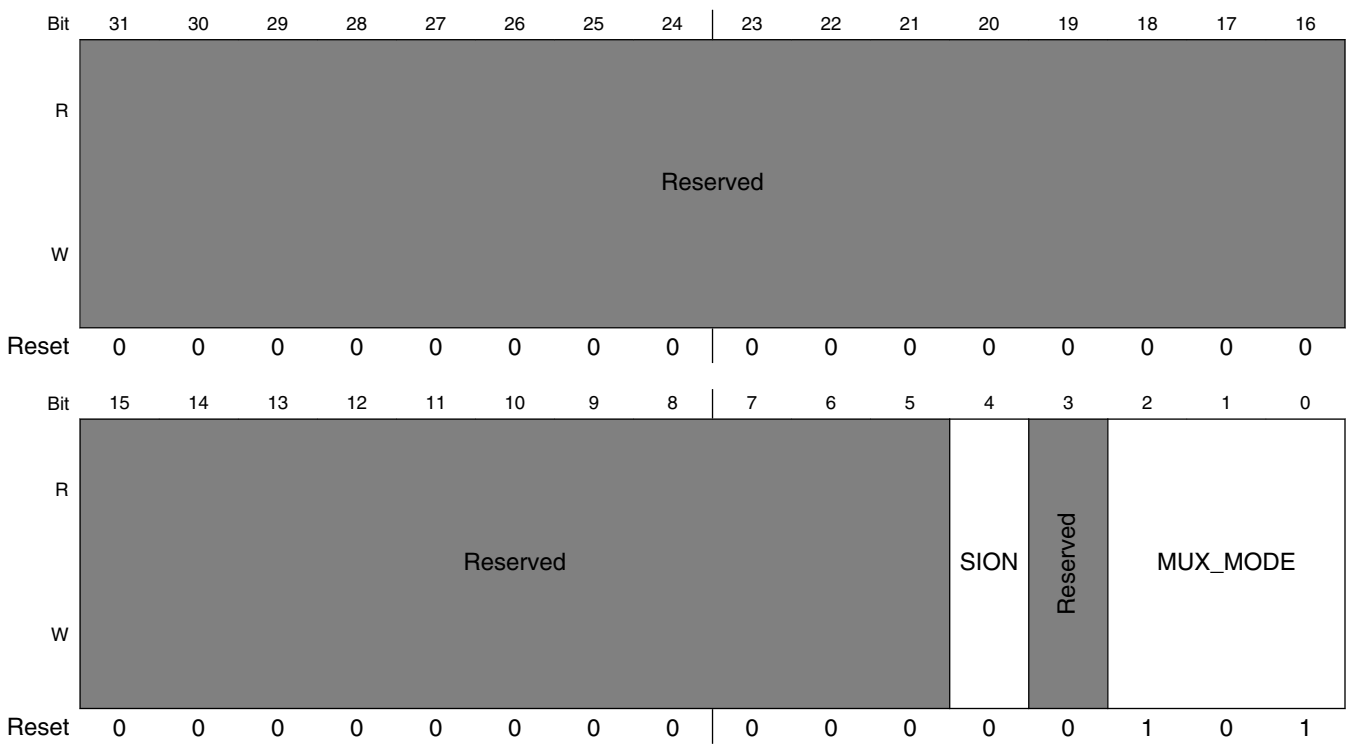


IOMUXC_SW_MUX_CTL_PAD_NAND_DATA00 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_DATA00 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: NAND_DATA00. 000 **ALT0** — Select signal RAWNAND_DATA00 001 **ALT1** — Select signal QSPI_A_DATA0 101 **ALT5** — Select signal GPIO3_IO06

8.2.5.64 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA01)

Address: 3033_0000h base + 110h offset = 3033_0110h



IOMUXC_SW_MUX_CTL_PAD_NAND_DATA01 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_DATA01 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: NAND_DATA01. 000 **ALT0** — Select signal RAWNAND_DATA01 001 **ALT1** — Select signal QSPI_A_DATA1 101 **ALT5** — Select signal GPIO3_IO07

8.2.5.65 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA02)

Address: 3033_0000h base + 114h offset = 3033_0114h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

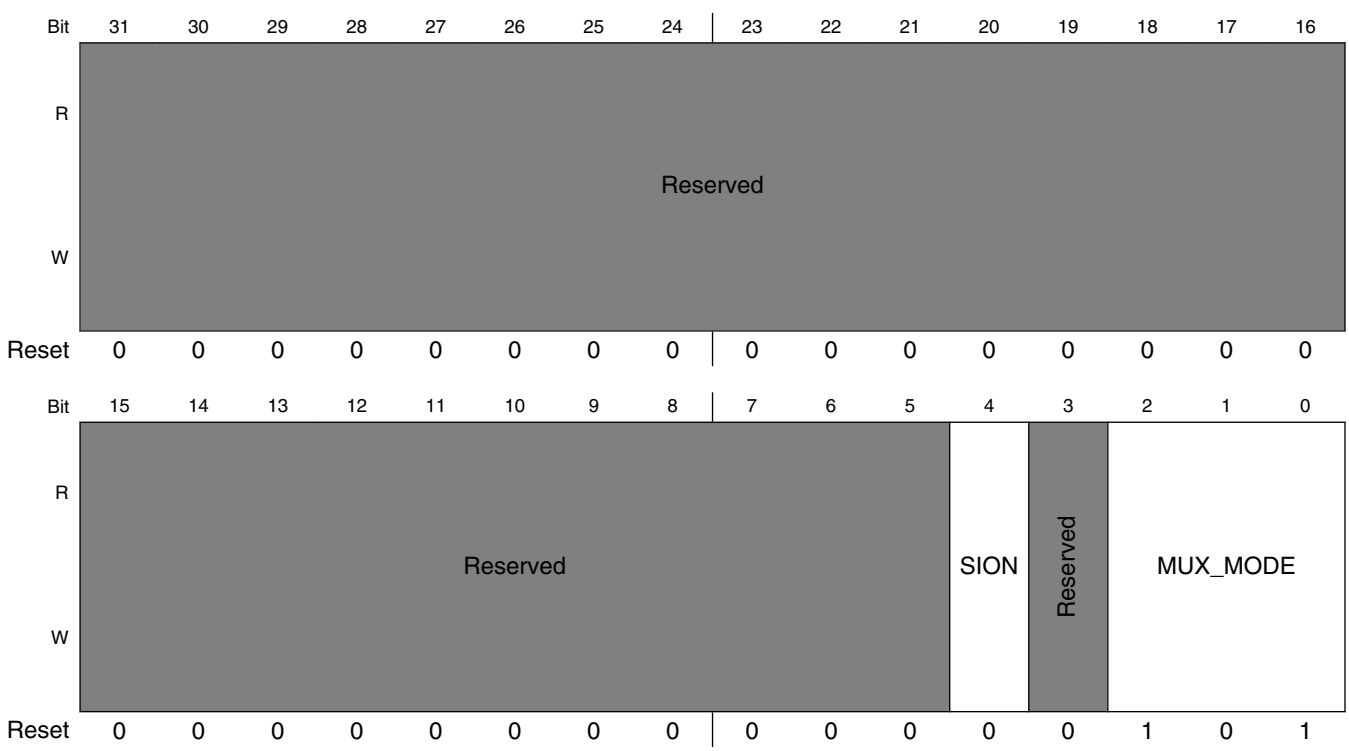
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_NAND_DATA02 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_DATA02 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: NAND_DATA02. **NOTE:** Pad NAND_DATA02 is involved in Daisy Chain. 000 **ALT0** — Select signal RAWNAND_DATA02 001 **ALT1** — Select signal QSPI_A_DATA2 010 **ALT2** — Select signal USDHC3_CD_B - Configure register IOMUXC_USDHC3_CD_B_SELECT_INPUT for mode ALT2. 101 **ALT5** — Select signal GPIO3_IO08

8.2.5.66 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA03)

Address: 3033_0000h base + 118h offset = 3033_0118h

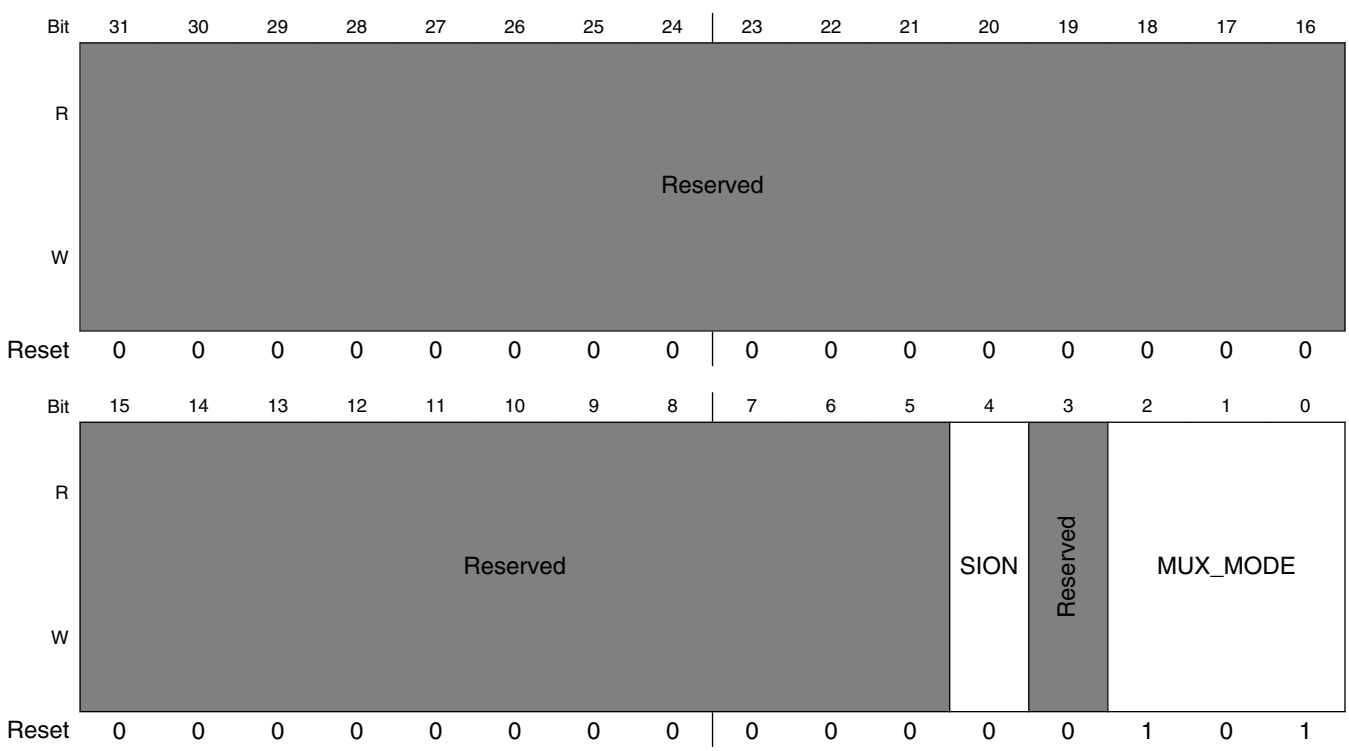


IOMUXC_SW_MUX_CTL_PAD_NAND_DATA03 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_DATA03 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: NAND_DATA03. **NOTE:** Pad NAND_DATA03 is involved in Daisy Chain. 000 **ALT0** — Select signal RAWNAND_DATA03 001 **ALT1** — Select signal QSPI_A_DATA3 010 **ALT2** — Select signal USDHC3_WP - Configure register IOMUXC_USDHC3_WP_SELECT_INPUT for mode ALT2. 101 **ALT5** — Select signal GPIO3_IO09

8.2.5.67 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA04)

Address: 3033_0000h base + 11Ch offset = 3033_011Ch



IOMUXC_SW_MUX_CTL_PAD_NAND_DATA04 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_DATA04 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: NAND_DATA04. 000 **ALT0** — Select signal RAWNAND_DATA04 001 **ALT1** — Select signal QSPI_B_DATA0 010 **ALT2** — Select signal USDHC3_DATA0 101 **ALT5** — Select signal GPIO3_IO10

8.2.5.68 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA05)

Address: 3033_0000h base + 120h offset = 3033_0120h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_NAND_DATA05 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_DATA05 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: NAND_DATA05. 000 **ALT0** — Select signal RAWNAND_DATA05 001 **ALT1** — Select signal QSPI_B_DATA1 010 **ALT2** — Select signal USDHC3_DATA1 101 **ALT5** — Select signal GPIO3_IO11

8.2.5.69 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA06)

Address: 3033_0000h base + 124h offset = 3033_0124h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_NAND_DATA06 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_DATA06 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: NAND_DATA06. 000 **ALT0** — Select signal RAWNAND_DATA06 001 **ALT1** — Select signal QSPI_B_DATA2 010 **ALT2** — Select signal USDHC3_DATA2 101 **ALT5** — Select signal GPIO3_IO12

8.2.5.70 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DATA07)

Address: 3033_0000h base + 128h offset = 3033_0128h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_NAND_DATA07 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_DATA07 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: NAND_DATA07. 000 **ALT0** — Select signal RAWNAND_DATA07 001 **ALT1** — Select signal QSPI_B_DATA3 010 **ALT2** — Select signal USDHC3_DATA3 101 **ALT5** — Select signal GPIO3_IO13

8.2.5.71 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_DQS)

Address: 3033_0000h base + 12Ch offset = 3033_012Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

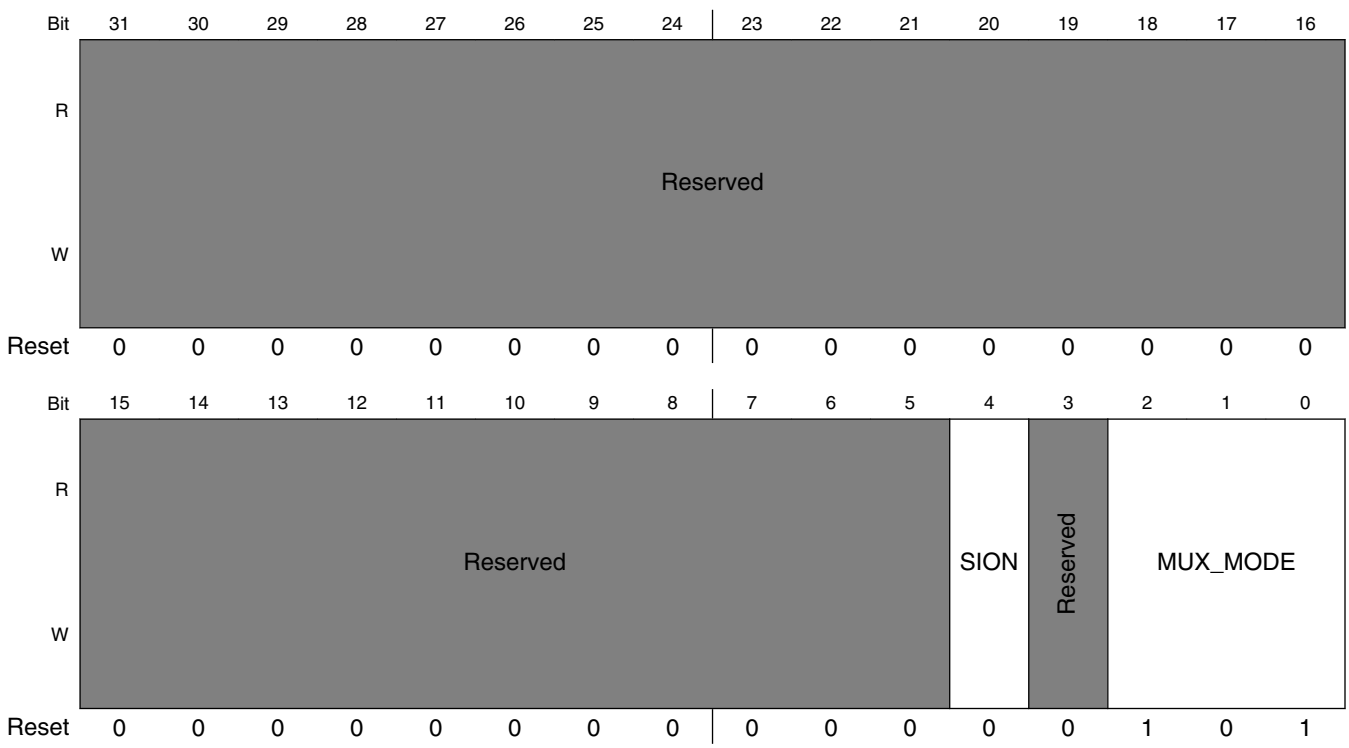
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_NAND_DQS field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_DQS 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: NAND_DQS. 000 **ALT0** — Select signal RAWNAND_DQS 001 **ALT1** — Select signal QSPI_A_DQS 101 **ALT5** — Select signal GPIO3_IO14

8.2.5.72 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_RE_B)

Address: 3033_0000h base + 130h offset = 3033_0130h

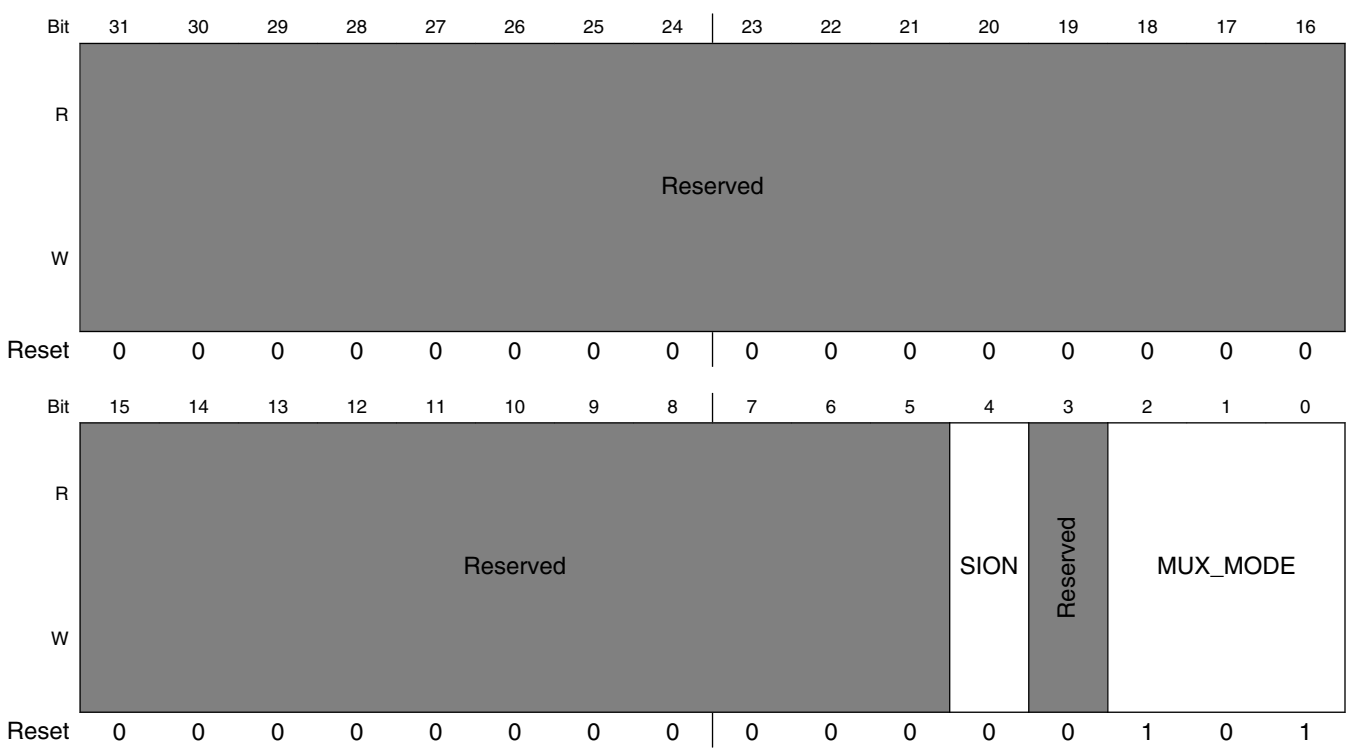


IOMUXC_SW_MUX_CTL_PAD_NAND_RE_B field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_RE_B 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: NAND_RE_B. 000 **ALT0** — Select signal RAWNAND_RE_B 001 **ALT1** — Select signal QSPI_B_DQS 010 **ALT2** — Select signal USDHC3_DATA4 101 **ALT5** — Select signal GPIO3_IO15

8.2.5.73 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_READY_B)

Address: 3033_0000h base + 134h offset = 3033_0134h

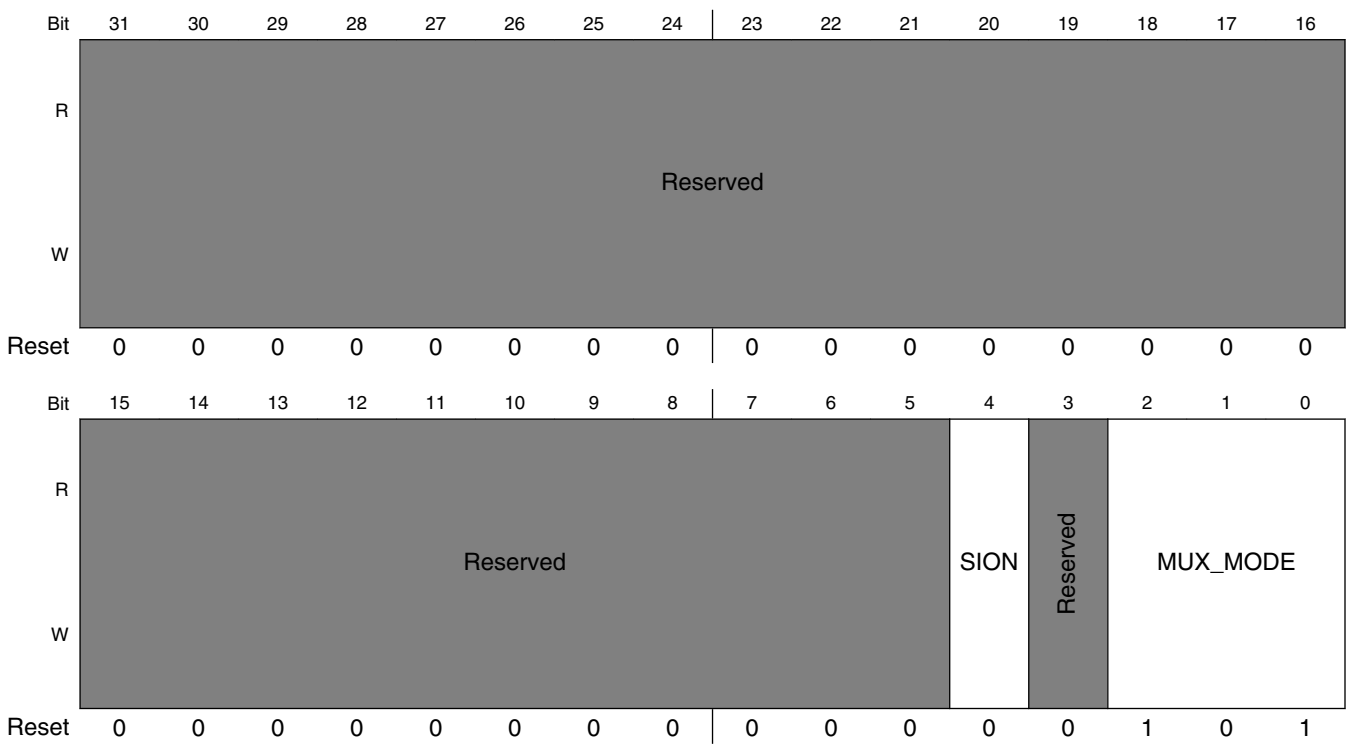


IOMUXC_SW_MUX_CTL_PAD_NAND_READY_B field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_READY_B 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: NAND_READY_B. 000 **ALT0** — Select signal RAWNAND_READY_B 010 **ALT2** — Select signal USDHC3_RESET_B 101 **ALT5** — Select signal GPIO3_IO16

8.2.5.74 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_WE_B)

Address: 3033_0000h base + 138h offset = 3033_0138h

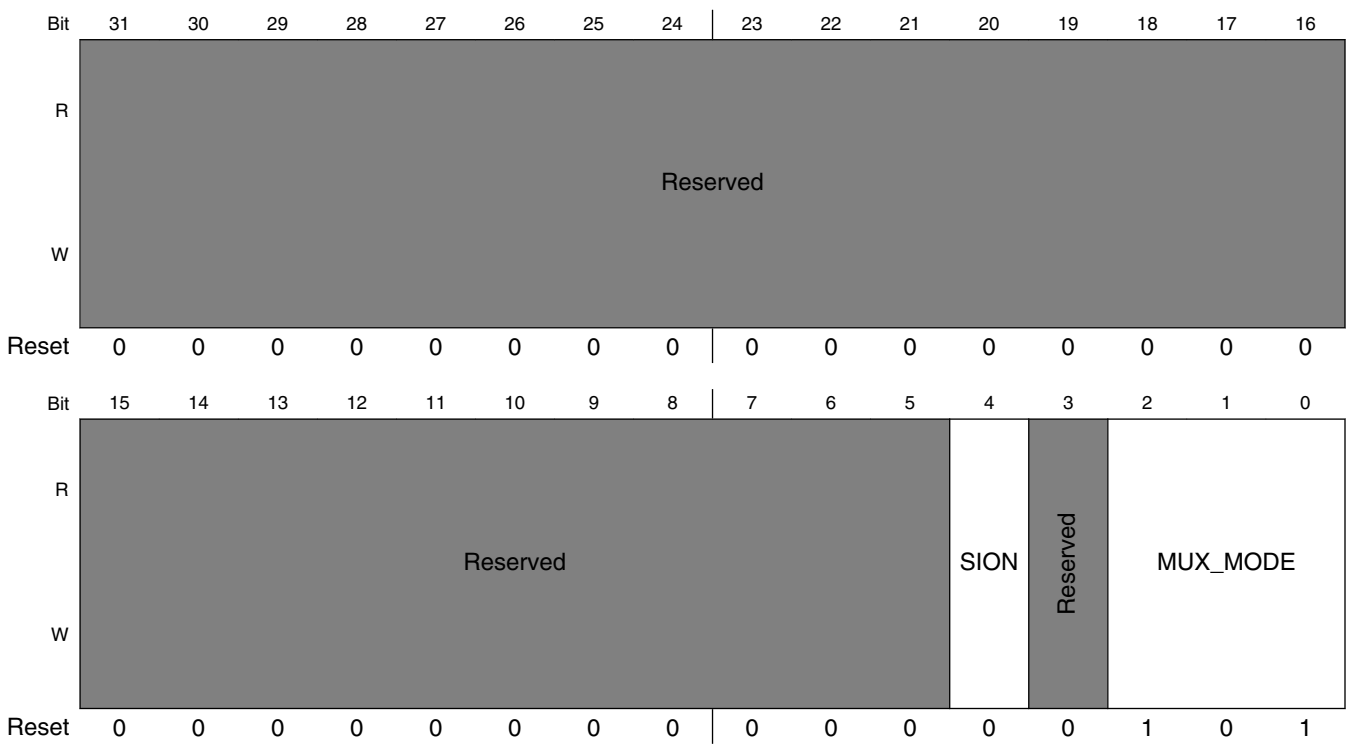


IOMUXC_SW_MUX_CTL_PAD_NAND_WE_B field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_WE_B 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: NAND_WE_B. 000 **ALT0** — Select signal RAWNAND_WE_B 010 **ALT2** — Select signal USDHC3_CLK 101 **ALT5** — Select signal GPIO3_IO17

8.2.5.75 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_NAND_WP_B)

Address: 3033_0000h base + 13Ch offset = 3033_013Ch

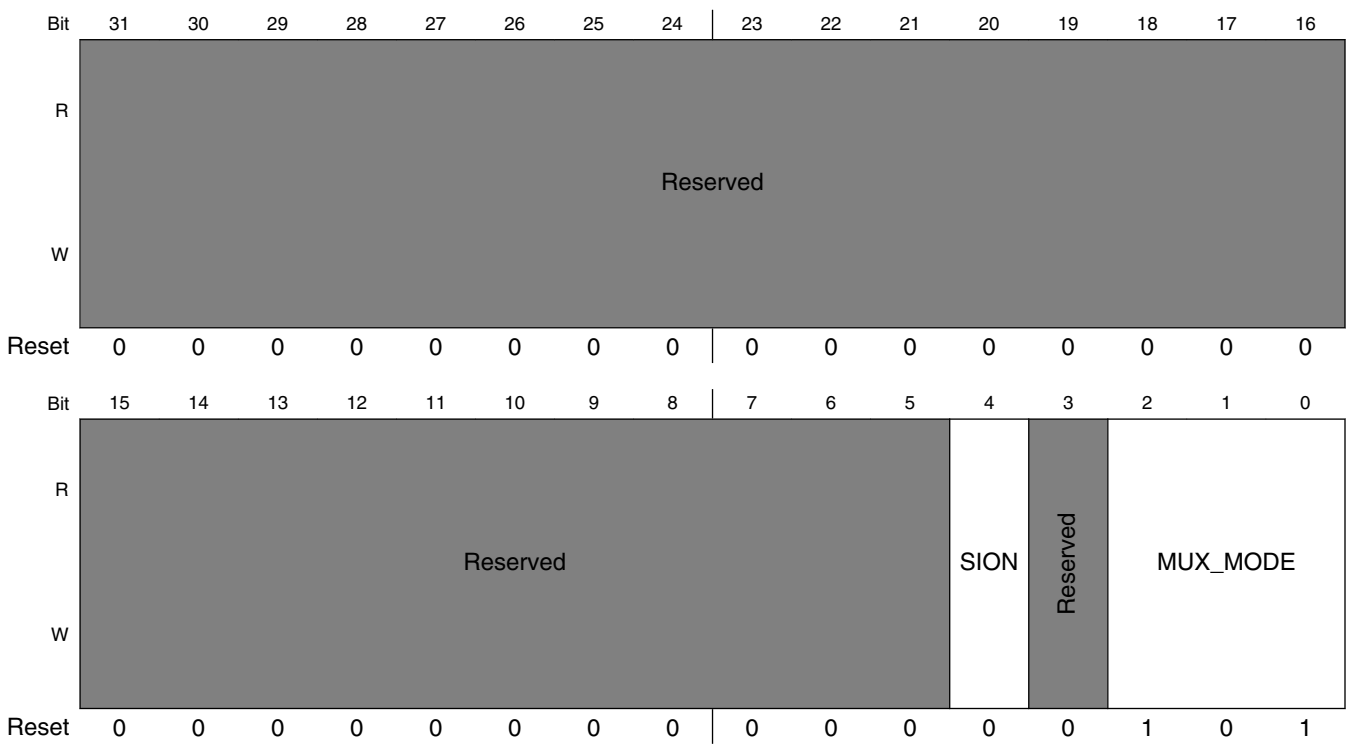


IOMUXC_SW_MUX_CTL_PAD_NAND_WP_B field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad NAND_WP_B 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: NAND_WP_B. 000 **ALT0** — Select signal RAWNAND_WP_B 010 **ALT2** — Select signal USDHC3_CMD 101 **ALT5** — Select signal GPIO3_IO18

8.2.5.76 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_RXFS)

Address: 3033_0000h base + 140h offset = 3033_0140h

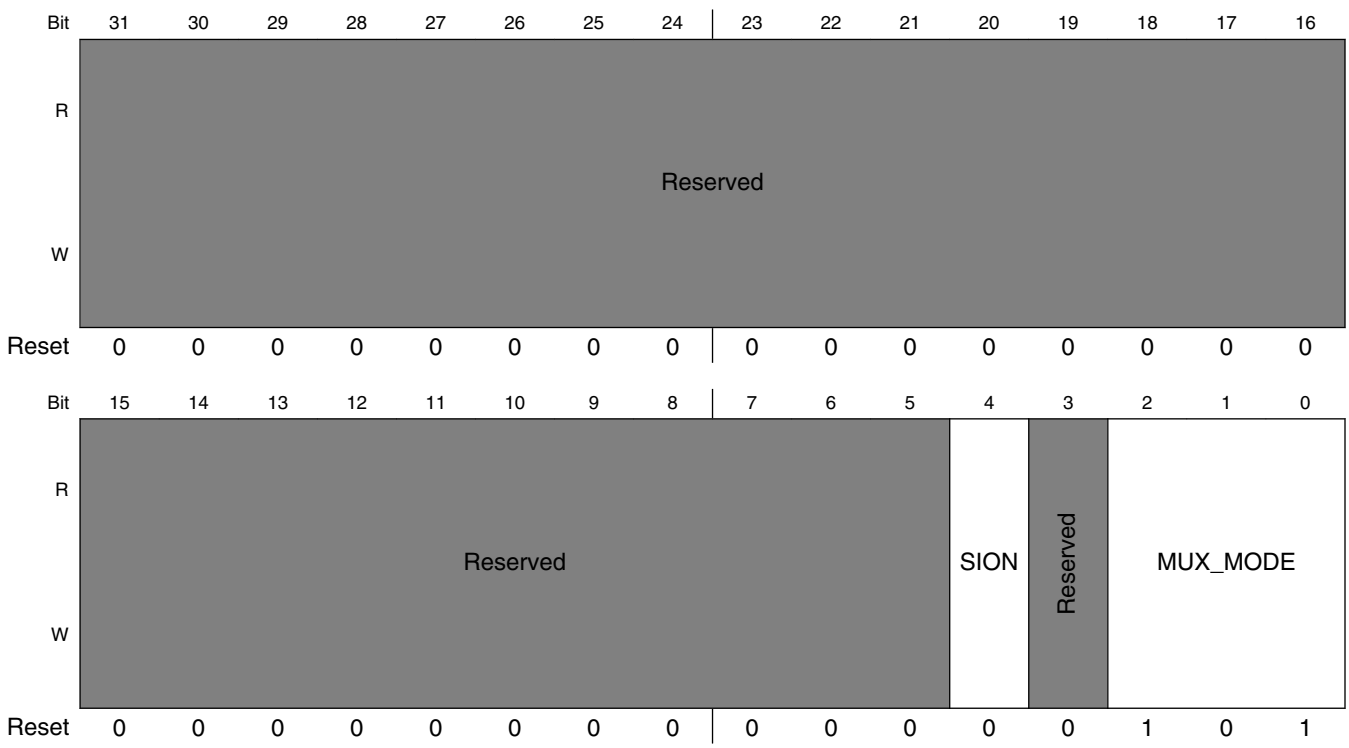


IOMUXC_SW_MUX_CTL_PAD_SAI5_RXFS field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI5_RXFS 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: SAI5_RXFS. **NOTE:** Pad SAI5_RXFS is involved in Daisy Chain. 000 **ALT0** — Select signal SAI5_RX_SYNC - Configure register IOMUXC_SAI5_RX_SYNC_SELECT_INPUT for mode ALT0. 001 **ALT1** — Select signal SAI1_TX_DATA0 101 **ALT5** — Select signal GPIO3_IO19

8.2.5.77 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_RXC)

Address: 3033_0000h base + 144h offset = 3033_0144h

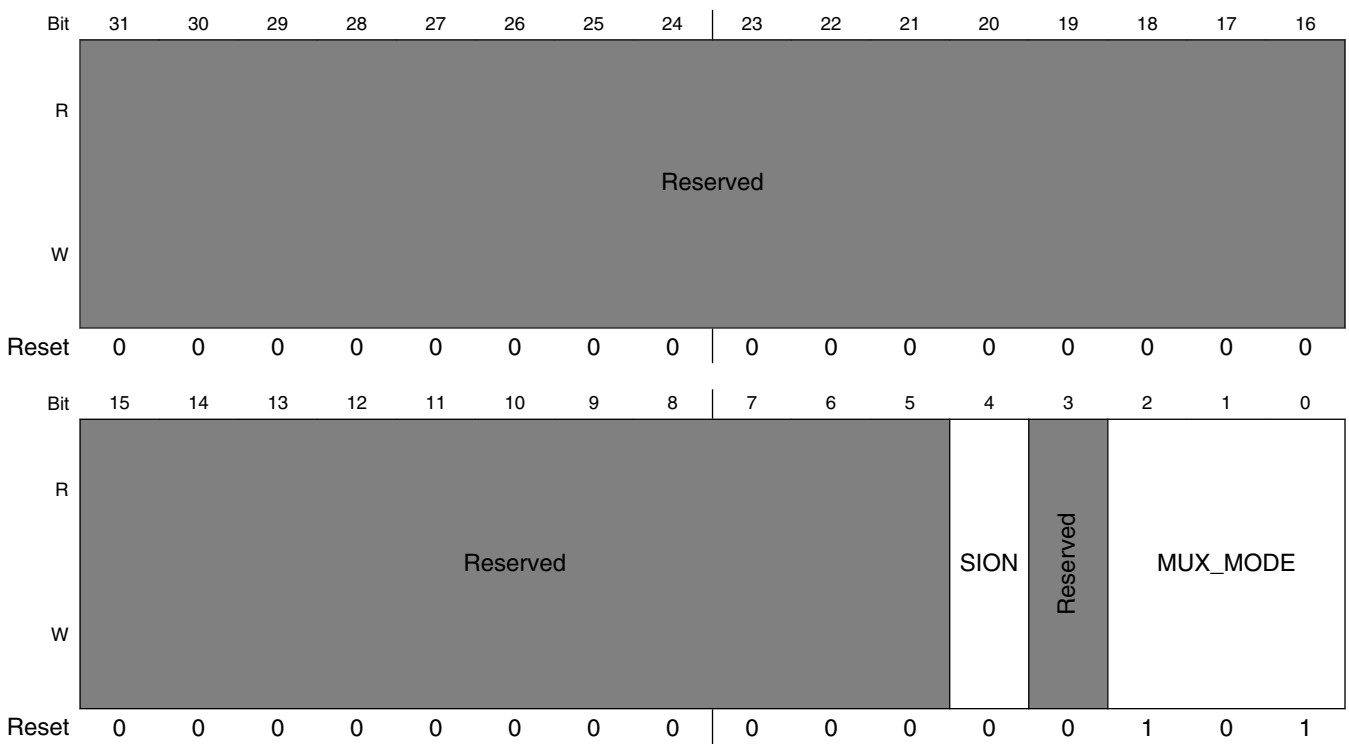


IOMUXC_SW_MUX_CTL_PAD_SAI5_RXC field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI5_RXC 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: SAI5_RXC. **NOTE:** Pad SAI5_RXC is involved in Daisy Chain. 000 **ALT0** — Select signal SAI5_RX_BCLK - Configure register IOMUXC_SAI5_RX_BCLK_SELECT_INPUT for mode ALT0. 001 **ALT1** — Select signal SAI1_TX_DATA1 100 **ALT4** — Select signal PDM_CLK 101 **ALT5** — Select signal GPIO3_IO20

8.2.5.78 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD0)

Address: 3033_0000h base + 148h offset = 3033_0148h

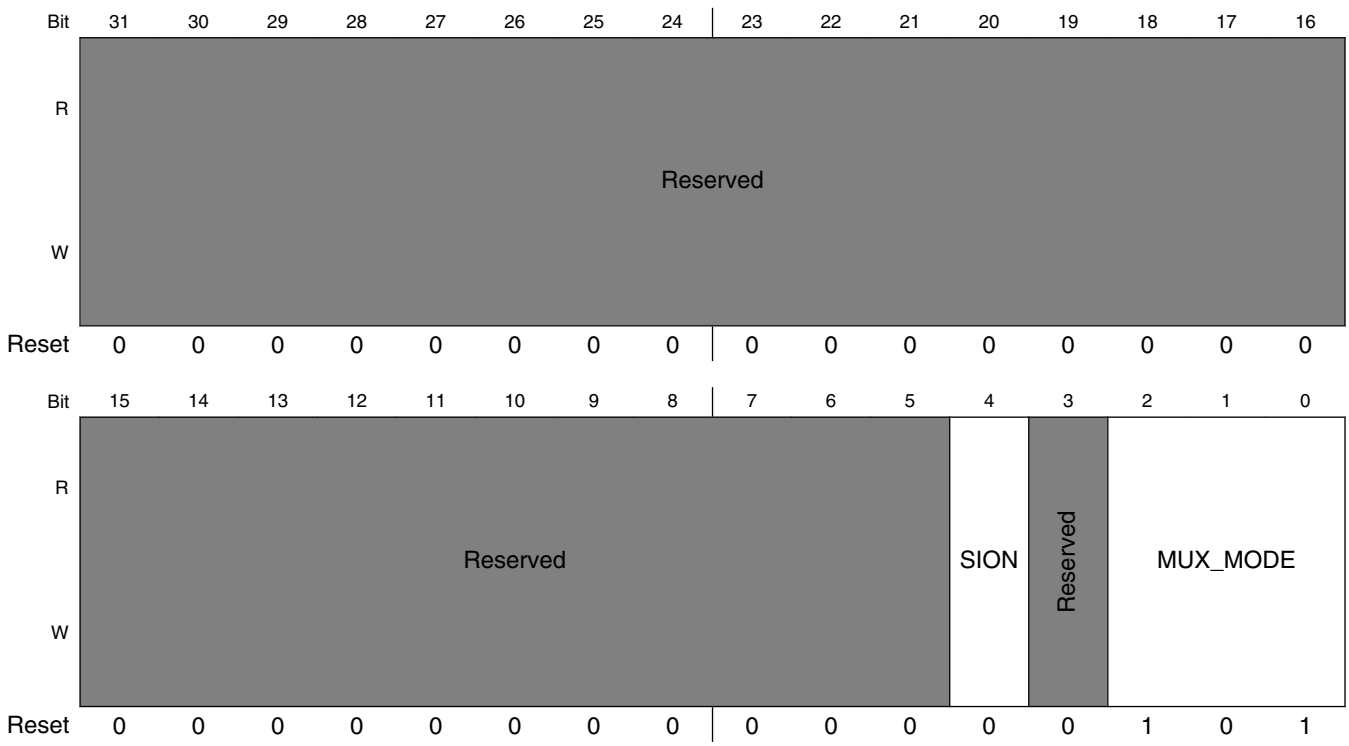


IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD0 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI5_RXD0 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: SAI5_RXD0. **NOTE:** Pad SAI5_RXD0 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI5_RX_DATA0 - Configure register IOMUXC_SAI5_RX_DATA_SELECT_INPUT_0 for mode ALT0. 001 **ALT1** — Select signal SAI1_TX_DATA2 100 **ALT4** — Select signal PDM_BIT_STREAM0 - Configure register IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_0 for mode ALT4. 101 **ALT5** — Select signal GPIO3_IO21

8.2.5.79 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD1)

Address: 3033_0000h base + 14Ch offset = 3033_014Ch



IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD1 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI5_RXD1 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI5_RXD1. **NOTE:** Pad SAI5_RXD1 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI5_RX_DATA1 - Configure register IOMUXC_SAI5_RX_DATA_SELECT_INPUT_1 for mode ALT0. 001 **ALT1** — Select signal SAI1_TX_DATA3 010 **ALT2** — Select signal SAI1_TX_SYNC

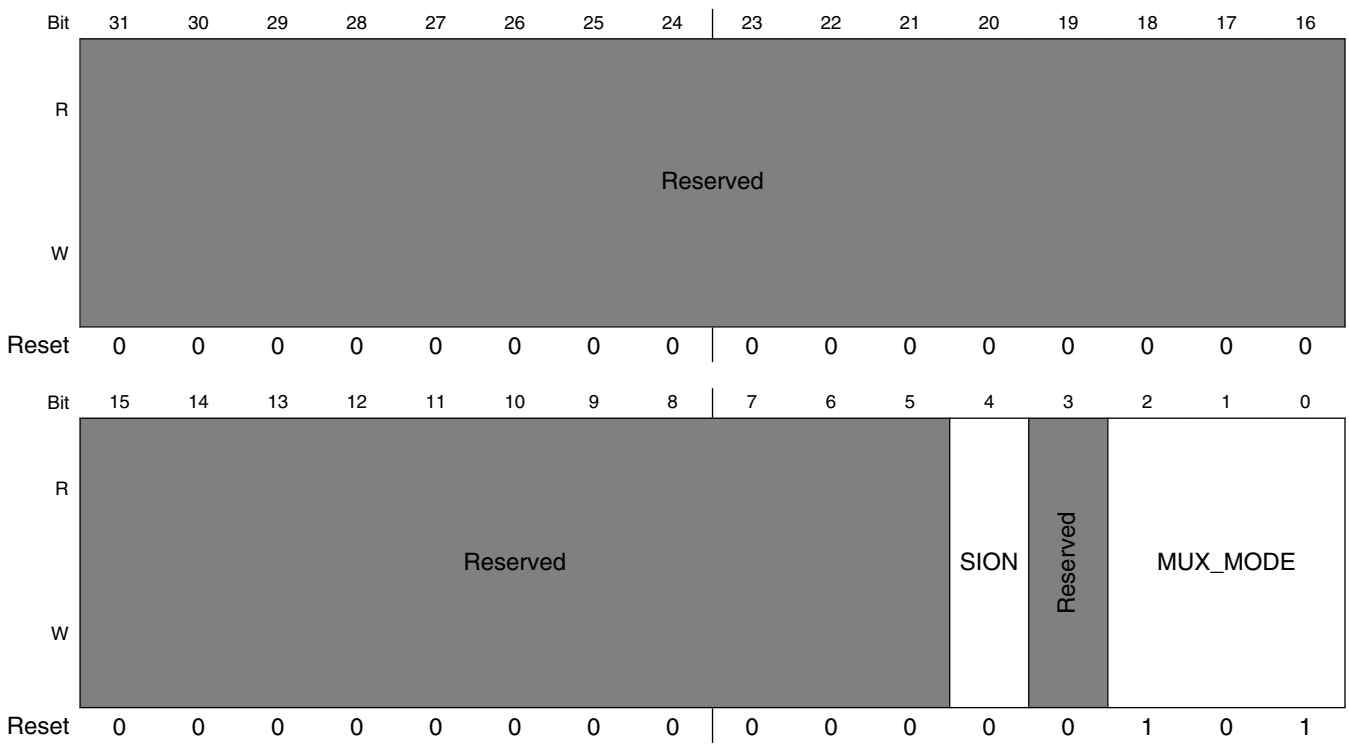
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD1 field descriptions (continued)

Field	Description
	- Configure register IOMUXC_SAI1_TX_SYNC_SELECT_INPUT for mode ALT2. 011 **ALT3** — Select signal SAI5_TX_SYNC - Configure register IOMUXC_SAI5_TX_SYNC_SELECT_INPUT for mode ALT3. 100 **ALT4** — Select signal PDM_BIT_STREAM1 - Configure register IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_1 for mode ALT4. 101 **ALT5** — Select signal GPIO3_IO22

8.2.5.80 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD2)

Address: 3033_0000h base + 150h offset = 3033_0150h



IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD2 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI5_RXD2 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).

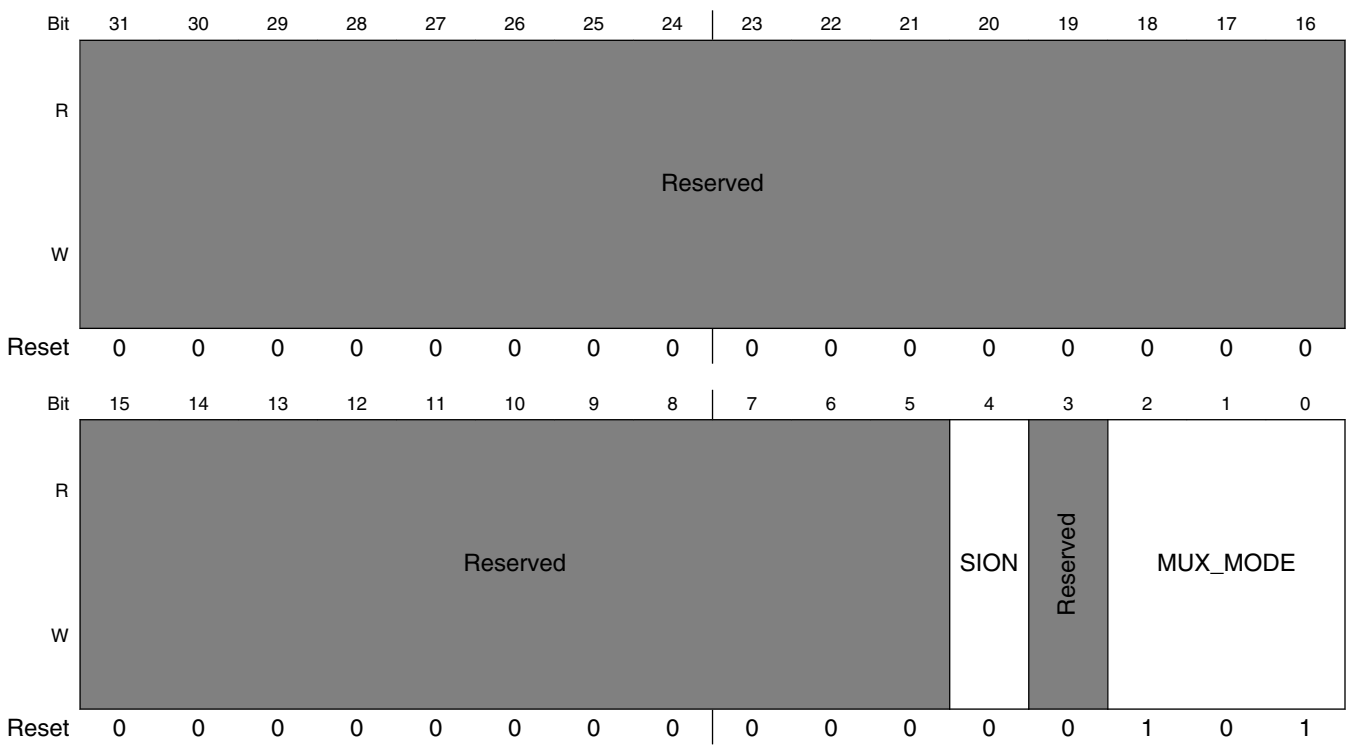
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD2 field descriptions (continued)

Field	Description
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI5_RXD2. **NOTE:** Pad SAI5_RXD2 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI5_RX_DATA2 - Configure register IOMUXC_SAI5_RX_DATA_SELECT_INPUT_2 for mode ALT0. 001 **ALT1** — Select signal SAI1_TX_DATA4 010 **ALT2** — Select signal SAI1_TX_SYNC - Configure register IOMUXC_SAI1_TX_SYNC_SELECT_INPUT for mode ALT2. 011 **ALT3** — Select signal SAI5_TX_BCLK - Configure register IOMUXC_SAI5_TX_BCLK_SELECT_INPUT for mode ALT3. 100 **ALT4** — Select signal PDM_BIT_STREAM2 - Configure register IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_2 for mode ALT4. 101 **ALT5** — Select signal GPIO3_IO23

8.2.5.81 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD3)

Address: 3033_0000h base + 154h offset = 3033_0154h

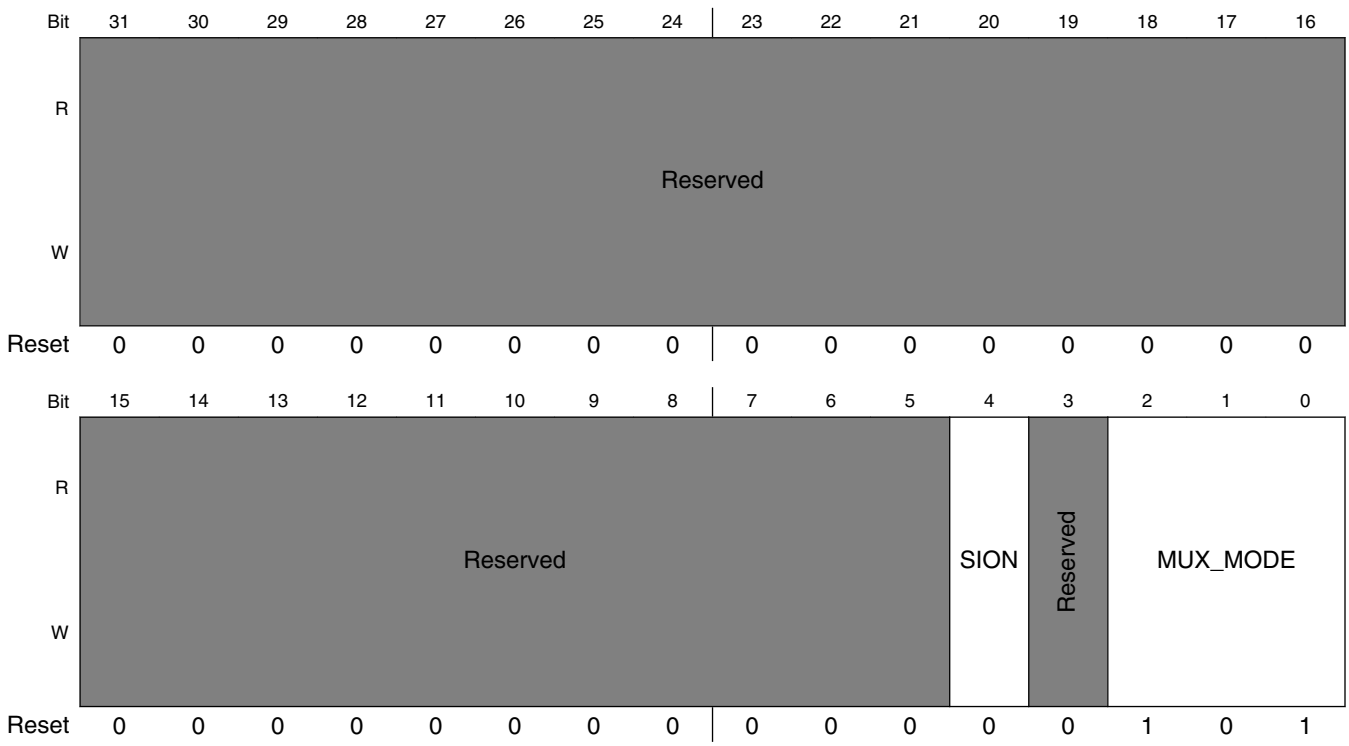


IOMUXC_SW_MUX_CTL_PAD_SAI5_RXD3 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI5_RXD3 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI5_RXD3. **NOTE:** Pad SAI5_RXD3 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI5_RX_DATA3 - Configure register IOMUXC_SAI5_RX_DATA_SELECT_INPUT_3 for mode ALT0. 001 **ALT1** — Select signal SAI1_TX_DATA5 010 **ALT2** — Select signal SAI1_TX_SYNC - Configure register IOMUXC_SAI1_TX_SYNC_SELECT_INPUT for mode ALT2. 011 **ALT3** — Select signal SAI5_TX_DATA0 100 **ALT4** — Select signal PDM_BIT_STREAM3 - Configure register IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_3 for mode ALT4. 101 **ALT5** — Select signal GPIO3_IO24

8.2.5.82 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI5_MCLK)

Address: 3033_0000h base + 158h offset = 3033_0158h

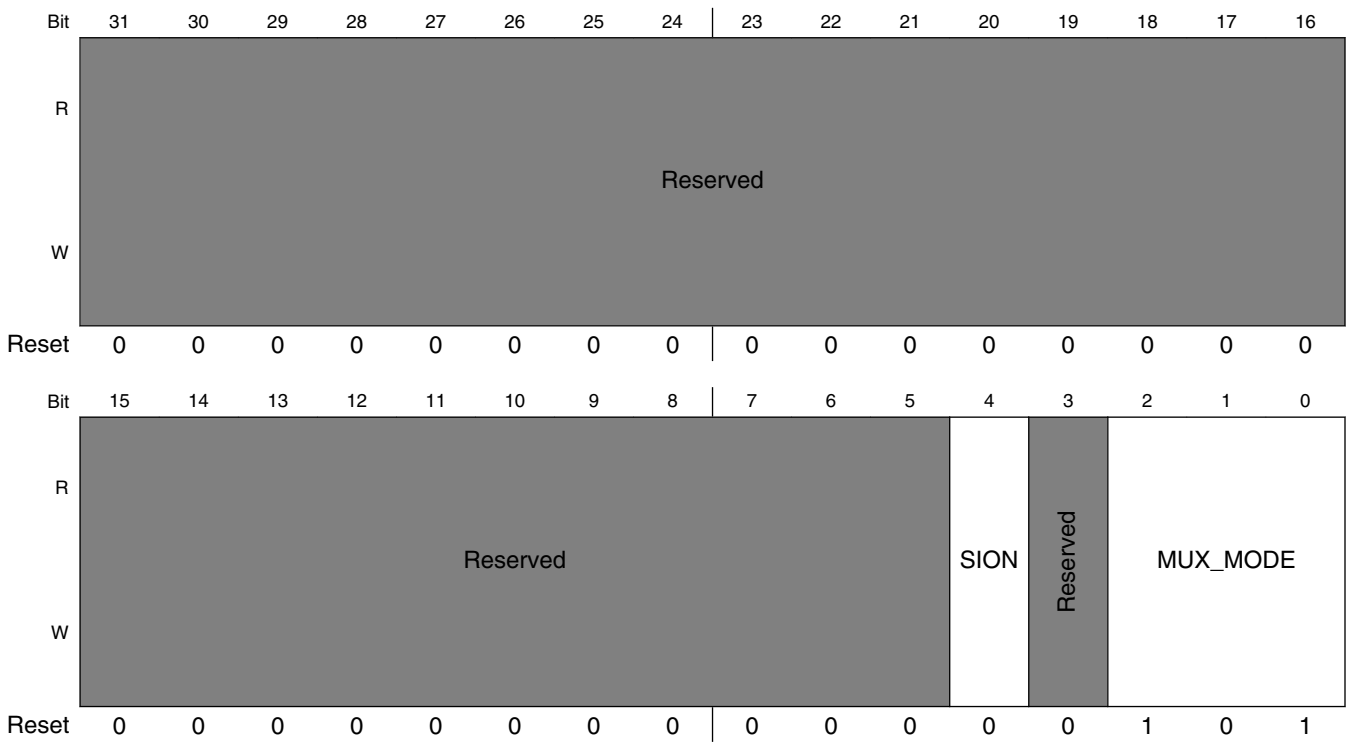


IOMUXC_SW_MUX_CTL_PAD_SAI5_MCLK field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI5_MCLK 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: SAI5_MCLK. **NOTE:** Pad SAI5_MCLK is involved in Daisy Chain. 000 **ALT0** — Select signal SAI5_MCLK - Configure register IOMUXC_SAI5_MCLK_SELECT_INPUT for mode ALT0. 001 **ALT1** — Select signal SAI1_TX_BCLK - Configure register IOMUXC_SAI1_TX_BCLK_SELECT_INPUT for mode ALT1. 101 **ALT5** — Select signal GPIO3_IO25

8.2.5.83 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXFS)

Address: 3033_0000h base + 15Ch offset = 3033_015Ch



IOMUXC_SW_MUX_CTL_PAD_SAI1_RXFS field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_RXFS 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: SAI1_RXFS. **NOTE:** Pad SAI1_RXFS is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_RX_SYNC - Configure register IOMUXC_SAI1_RX_SYNC_SELECT_INPUT for mode ALT0. 001 **ALT1** — Select signal SAI5_RX_SYNC - Configure register IOMUXC_SAI5_RX_SYNC_SELECT_INPUT for mode ALT1.

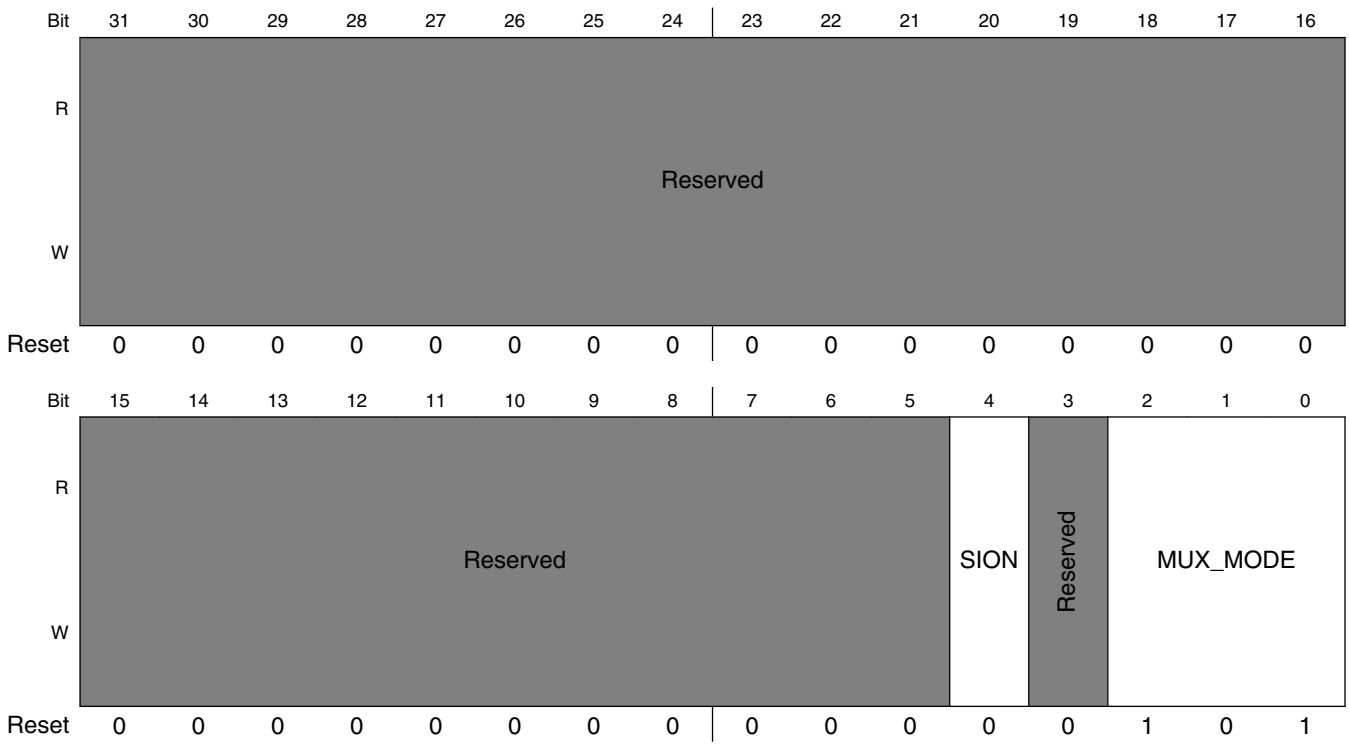
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_RXFS field descriptions (continued)

Field	Description
	100 **ALT4** — Select signal CORESIGHT_TRACE_CLK 101 **ALT5** — Select signal GPIO4_IO00

8.2.5.84 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXC)

Address: 3033_0000h base + 160h offset = 3033_0160h



IOMUXC_SW_MUX_CTL_PAD_SAI1_RXC field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_RXC 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: SAI1_RXC. **NOTE:** Pad SAI1_RXC is involved in Daisy Chain.

Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_RXC field descriptions (continued)

Field	Description
	000 **ALT0** — Select signal SAI1_RX_BCLK 001 **ALT1** — Select signal SAI5_RX_BCLK - Configure register IOMUXC_SAI5_RX_BCLK_SELECT_INPUT for mode ALT1. 100 **ALT4** — Select signal CORESIGHT_TRACE_CTL 101 **ALT5** — Select signal GPIO4_IO01

8.2.5.85 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD0)

Address: 3033_0000h base + 164h offset = 3033_0164h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD0 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_RXD0 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).

Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD0 field descriptions (continued)

Field	Description
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for pad: SAI1_RXD0. **NOTE:** Pad SAI1_RXD0 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_RX_DATA0 001 **ALT1** — Select signal SAI5_RX_DATA0 - Configure register IOMUXC_SAI5_RX_DATA_SELECT_INPUT_0 for mode ALT1. 010 **ALT2** — Select signal SAI1_TX_DATA1 011 **ALT3** — Select signal PDM_BIT_STREAM0 - Configure register IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_0 for mode ALT3. 100 **ALT4** — Select signal CORESIGHT_TRACE0 101 **ALT5** — Select signal GPIO4_IO02 110 **ALT6** — Select signal SRC_BOOT_CFG0

8.2.5.86 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD1)

Address: 3033_0000h base + 168h offset = 3033_0168h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

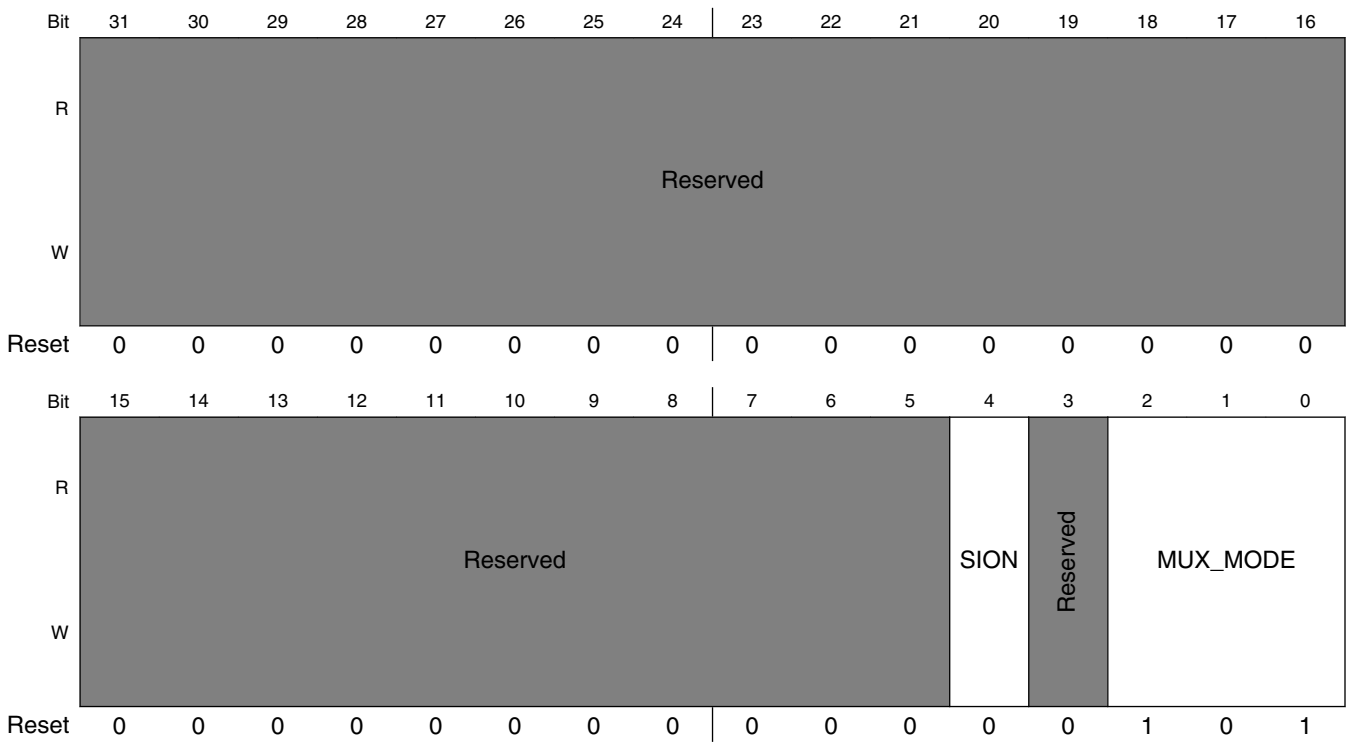
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD1 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_RXD1 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI1_RXD1. **NOTE:** Pad SAI1_RXD1 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_RX_DATA1 001 **ALT1** — Select signal SAI5_RX_DATA1 - Configure register IOMUXC_SAI5_RX_DATA_SELECT_INPUT_1 for mode ALT1. 011 **ALT3** — Select signal PDM_BIT_STREAM1 - Configure register IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_1 for mode ALT3. 100 **ALT4** — Select signal CORESIGHT_TRACE1 101 **ALT5** — Select signal GPIO4_IO03 110 **ALT6** — Select signal SRC_BOOT_CFG1

8.2.5.87 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD2)

Address: 3033_0000h base + 16Ch offset = 3033_016Ch



IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD2 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_RXD2 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI1_RXD2. **NOTE:** Pad SAI1_RXD2 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_RX_DATA2 001 **ALT1** — Select signal SAI5_RX_DATA2 - Configure register IOMUXC_SAI5_RX_DATA_SELECT_INPUT_2 for mode ALT1. 011 **ALT3** — Select signal PDM_BIT_STREAM2

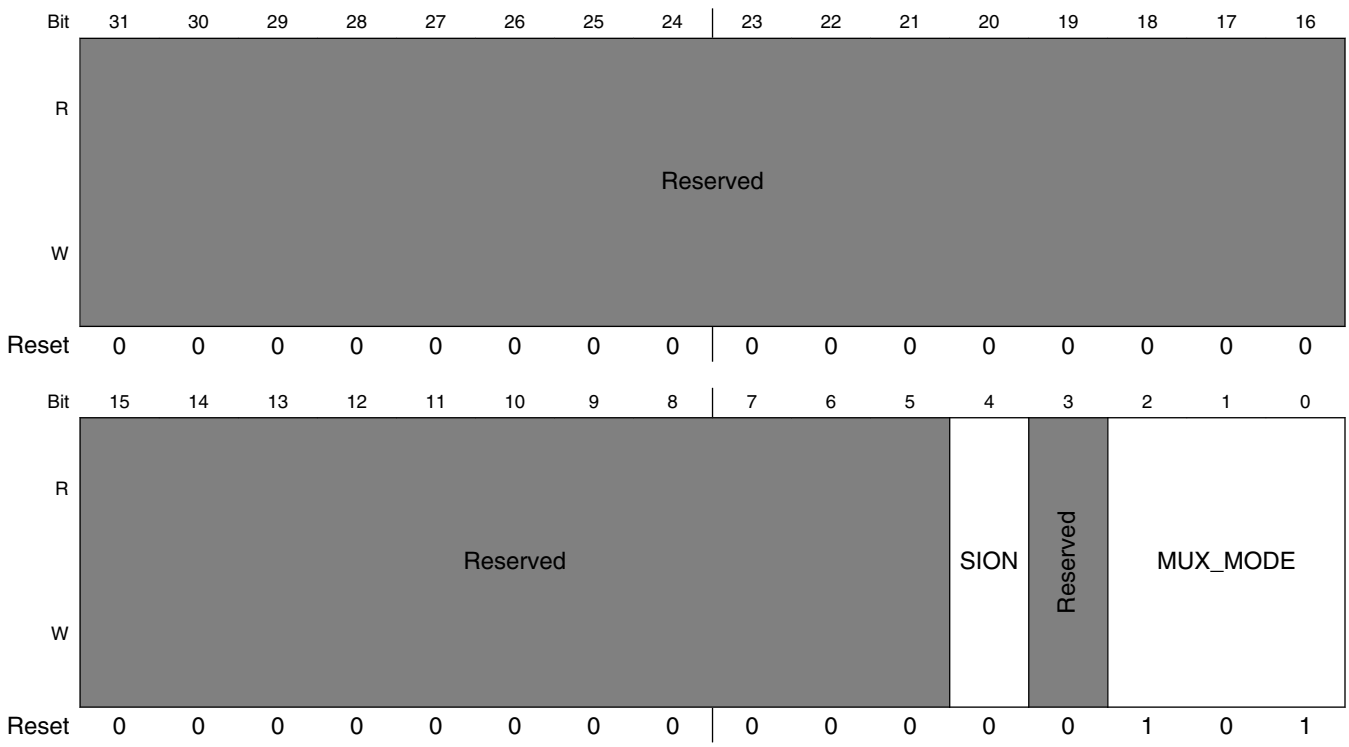
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD2 field descriptions (continued)

Field	Description
	- Configure register IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_2 for mode ALT3. 100 **ALT4** — Select signal CORESIGHT_TRACE2 101 **ALT5** — Select signal GPIO4_IO04 110 **ALT6** — Select signal SRC_BOOT_CFG2

8.2.5.88 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD3)

Address: 3033_0000h base + 170h offset = 3033_0170h



IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD3 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_RXD3 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved

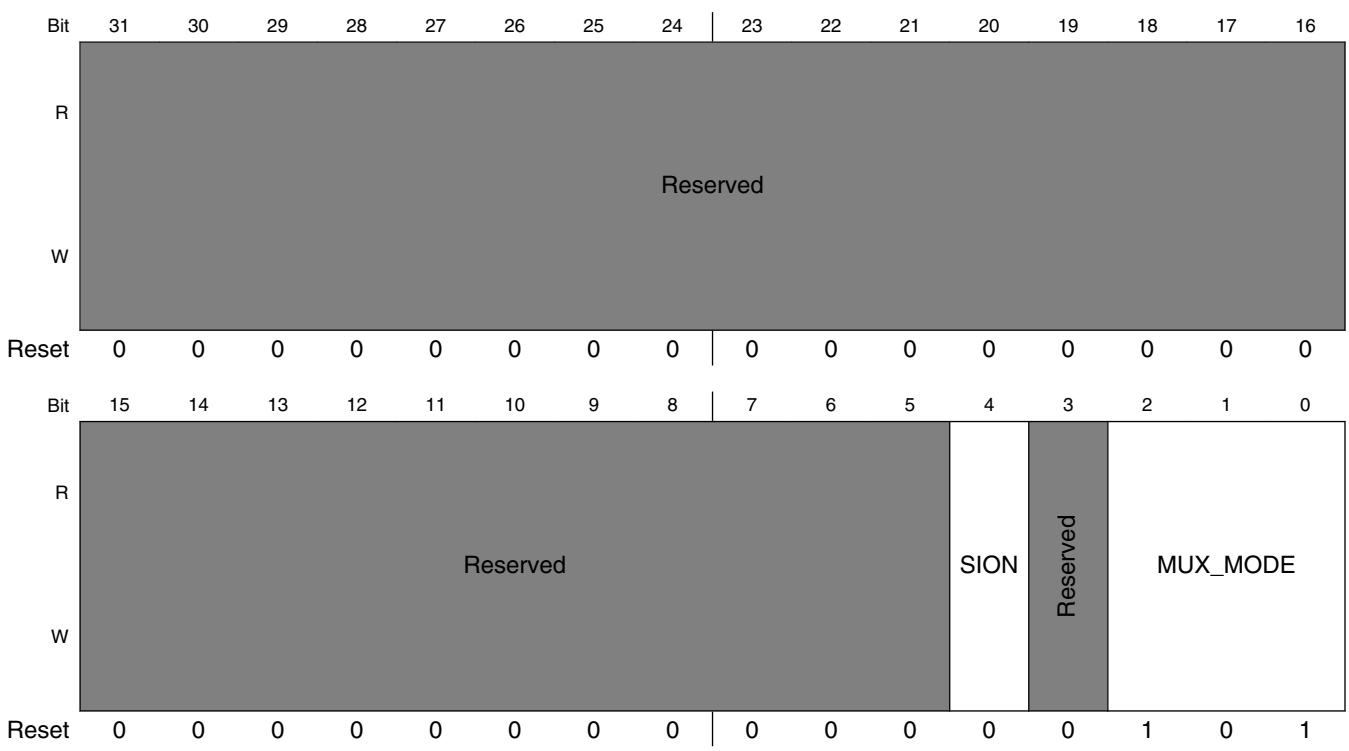
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD3 field descriptions (continued)

Field	Description
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI1_RXD3. **NOTE:** Pad SAI1_RXD3 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_RX_DATA3 001 **ALT1** — Select signal SAI5_RX_DATA3 - Configure register IOMUXC_SAI5_RX_DATA_SELECT_INPUT_3 for mode ALT1. 011 **ALT3** — Select signal PDM_BIT_STREAM3 - Configure register IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_3 for mode ALT3. 100 **ALT4** — Select signal CORESIGHT_TRACE3 101 **ALT5** — Select signal GPIO4_IO05 110 **ALT6** — Select signal SRC_BOOT_CFG3

8.2.5.89 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD4)

Address: 3033_0000h base + 174h offset = 3033_0174h

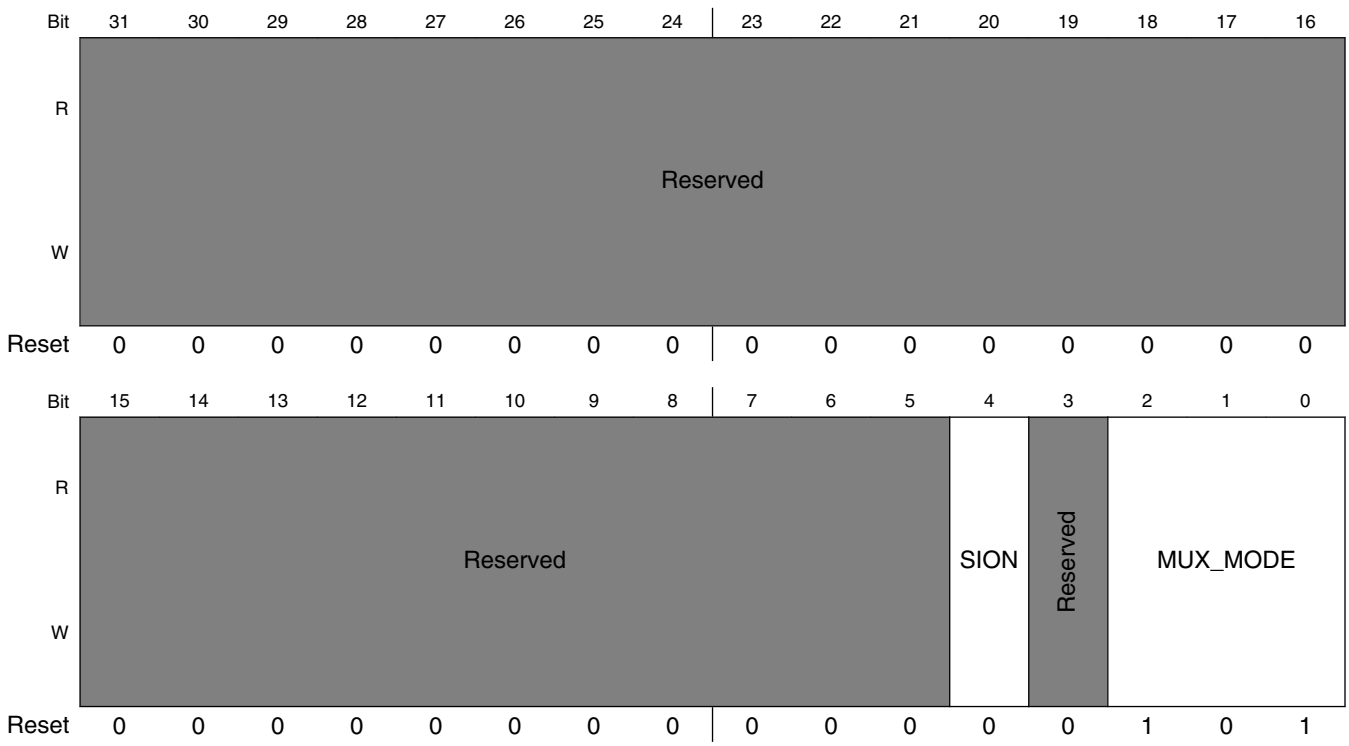


IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD4 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_RXD4 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI1_RXD4. **NOTE:** Pad SAI1_RXD4 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_RX_DATA4 001 **ALT1** — Select signal SAI6_TX_BCLK - Configure register IOMUXC_SAI6_TX_BCLK_SELECT_INPUT for mode ALT1. 010 **ALT2** — Select signal SAI6_RX_BCLK - Configure register IOMUXC_SAI6_RX_BCLK_SELECT_INPUT for mode ALT2. 100 **ALT4** — Select signal CORESIGHT_TRACE4 101 **ALT5** — Select signal GPIO4_IO06 110 **ALT6** — Select signal SRC_BOOT_CFG4

8.2.5.90 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD5)

Address: 3033_0000h base + 178h offset = 3033_0178h



IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD5 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_RXD5 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for pad: SAI1_RXD5. **NOTE:** Pad SAI1_RXD5 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_RX_DATA5 001 **ALT1** — Select signal SAI6_TX_DATA0 010 **ALT2** — Select signal SAI6_RX_DATA0 - Configure register IOMUXC_SAI6_RX_DATA_SELECT_INPUT_0 for mode ALT2.

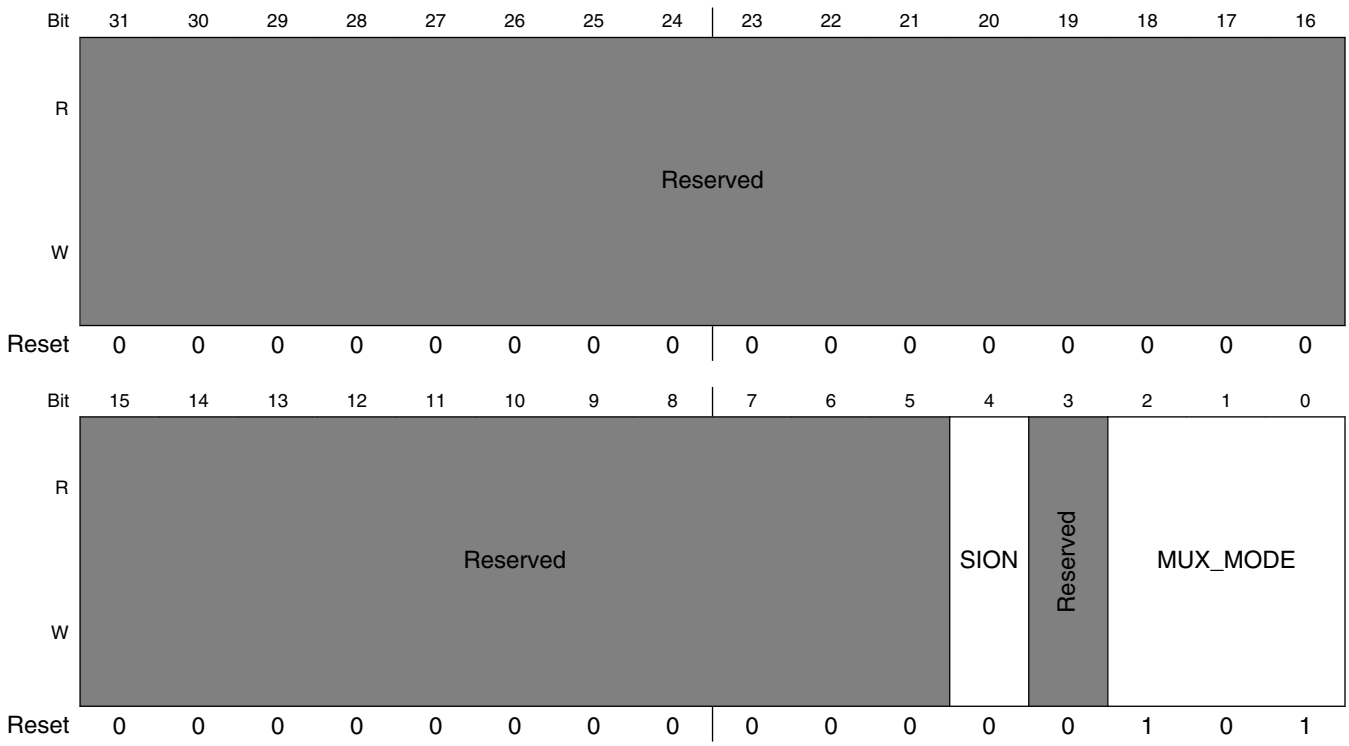
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD5 field descriptions (continued)

Field	Description
	011 **ALT3** — Select signal SAI1_RX_SYNC - Configure register IOMUXC_SAI1_RX_SYNC_SELECT_INPUT for mode ALT3. 100 **ALT4** — Select signal CORESIGHT_TRACE5 101 **ALT5** — Select signal GPIO4_IO07 110 **ALT6** — Select signal SRC_BOOT_CFG5

8.2.5.91 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD6)

Address: 3033_0000h base + 17Ch offset = 3033_017Ch



IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD6 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_RXD6 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).

Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD6 field descriptions (continued)

Field	Description
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI1_RXD6. **NOTE:** Pad SAI1_RXD6 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_RX_DATA6 001 **ALT1** — Select signal SAI6_TX_SYNC - Configure register IOMUXC_SAI6_TX_SYNC_SELECT_INPUT for mode ALT1. 010 **ALT2** — Select signal SAI6_RX_SYNC - Configure register IOMUXC_SAI6_RX_SYNC_SELECT_INPUT for mode ALT2. 100 **ALT4** — Select signal CORESIGHT_TRACE6 101 **ALT5** — Select signal GPIO4_IO08 110 **ALT6** — Select signal SRC_BOOT_CFG6

8.2.5.92 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD7)

Address: 3033_0000h base + 180h offset = 3033_0180h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

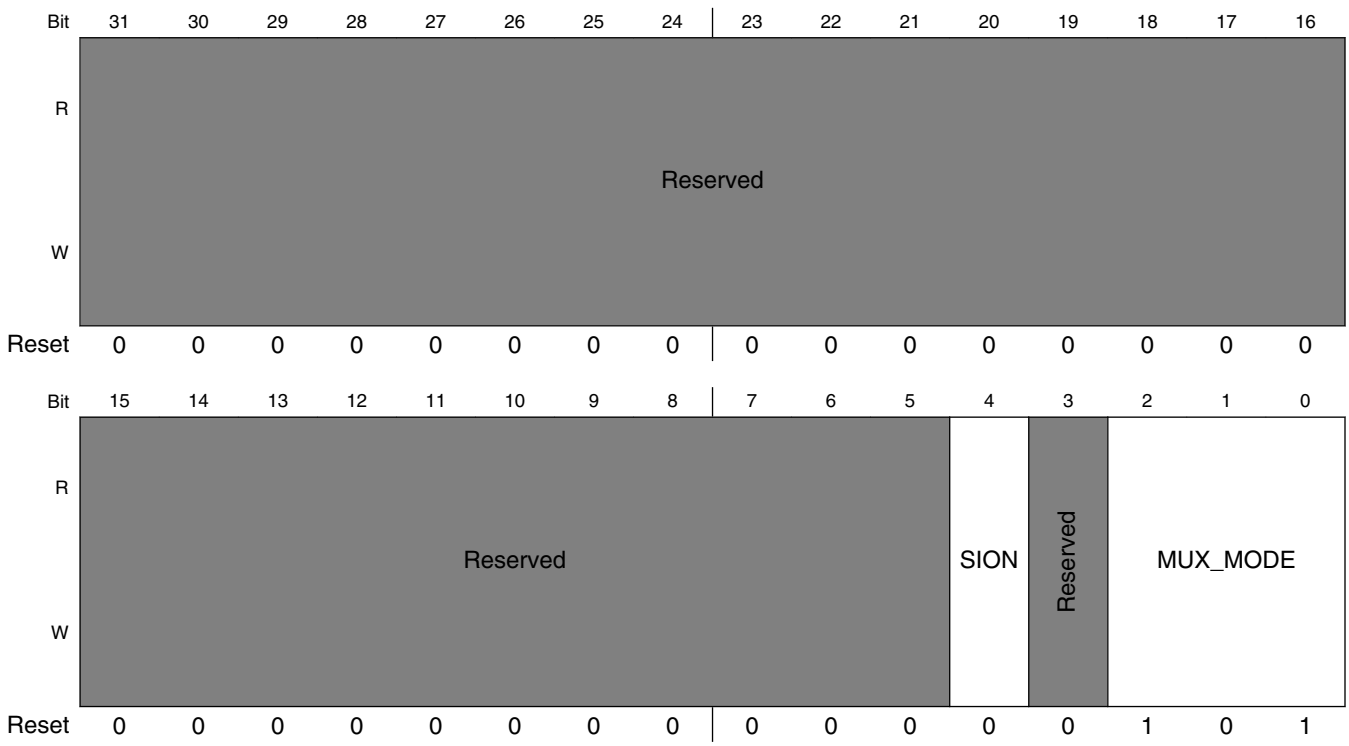
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_SAI1_RXD7 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_RXD7 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for pad: SAI1_RXD7. **NOTE:** Pad SAI1_RXD7 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_RX_DATA7 001 **ALT1** — Select signal SAI6_MCLK - Configure register IOMUXC_SAI6_MCLK_SELECT_INPUT for mode ALT1. 010 **ALT2** — Select signal SAI1_TX_SYNC - Configure register IOMUXC_SAI1_TX_SYNC_SELECT_INPUT for mode ALT2. 011 **ALT3** — Select signal SAI1_TX_DATA4 100 **ALT4** — Select signal CORESIGHT_TRACE7 101 **ALT5** — Select signal GPIO4_IO09 110 **ALT6** — Select signal SRC_BOOT_CFG7

8.2.5.93 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXFS)

Address: 3033_0000h base + 184h offset = 3033_0184h



IOMUXC_SW_MUX_CTL_PAD_SAI1_TXFS field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_TXFS 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: SAI1_TXFS. **NOTE:** Pad SAI1_TXFS is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_TX_SYNC - Configure register IOMUXC_SAI1_TX_SYNC_SELECT_INPUT for mode ALT0. 001 **ALT1** — Select signal SAI5_TX_SYNC - Configure register IOMUXC_SAI5_TX_SYNC_SELECT_INPUT for mode ALT1.

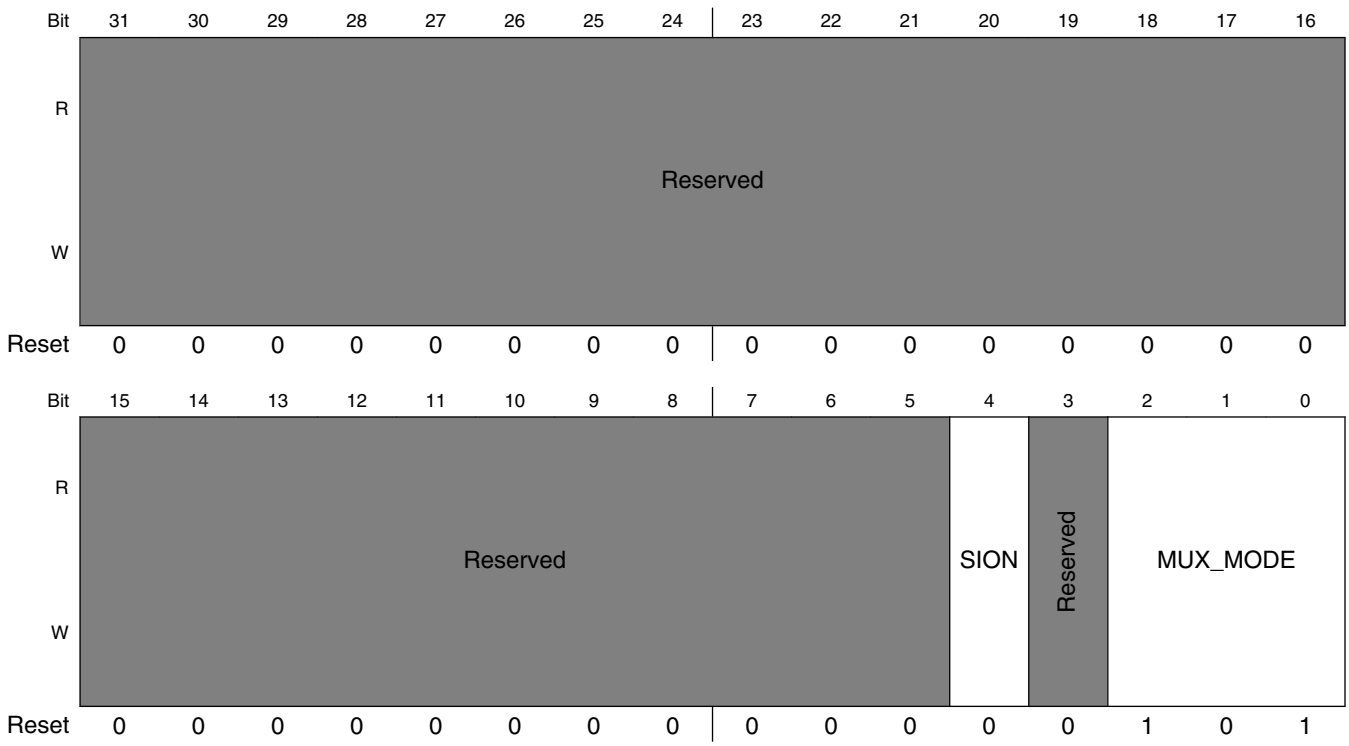
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_TXFS field descriptions (continued)

Field	Description
	100 **ALT4** — Select signal CORESIGHT_EVENTO 101 **ALT5** — Select signal GPIO4_IO10

8.2.5.94 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXC)

Address: 3033_0000h base + 188h offset = 3033_0188h



IOMUXC_SW_MUX_CTL_PAD_SAI1_TXC field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_TXC 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: SAI1_TXC. **NOTE:** Pad SAI1_TXC is involved in Daisy Chain.

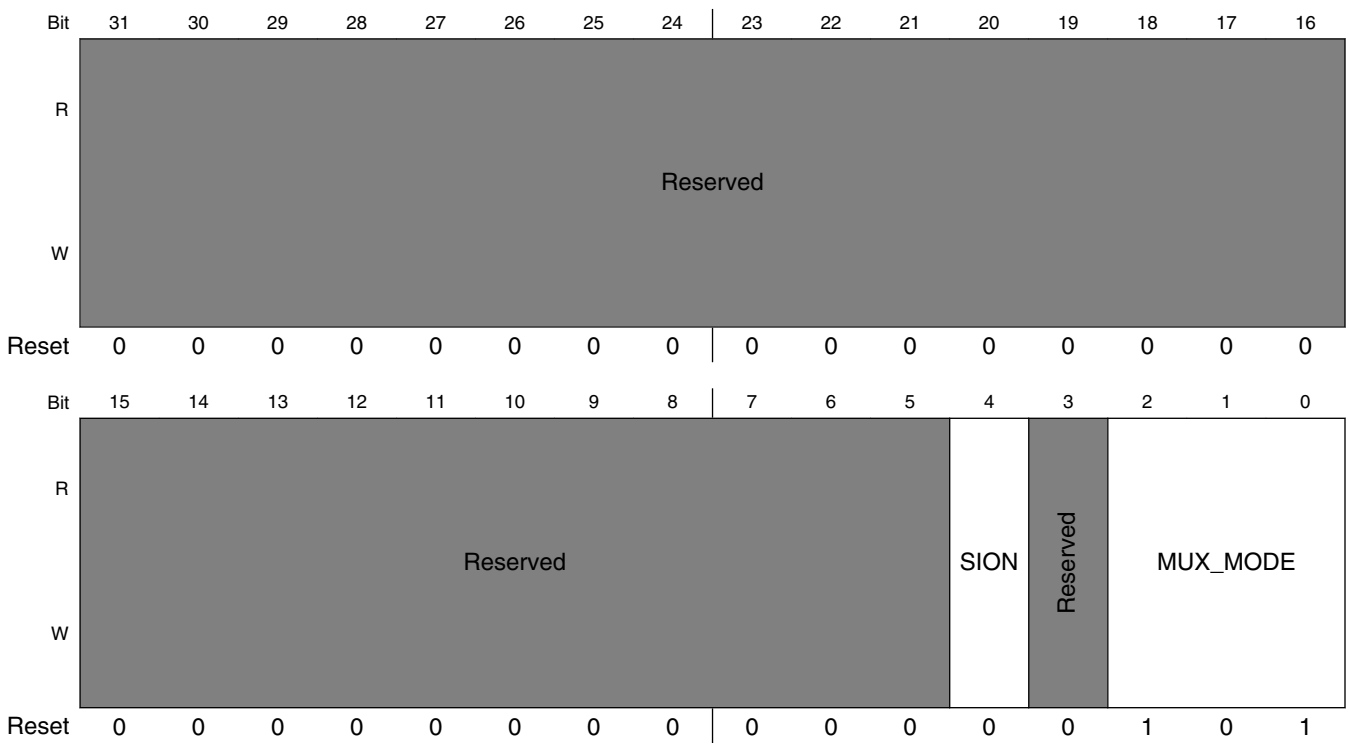
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_TXC field descriptions (continued)

Field	Description
	000 **ALT0** — Select signal SAI1_TX_BCLK - Configure register IOMUXC_SAI1_TX_BCLK_SELECT_INPUT for mode ALT0. 001 **ALT1** — Select signal SAI5_TX_BCLK - Configure register IOMUXC_SAI5_TX_BCLK_SELECT_INPUT for mode ALT1. 100 **ALT4** — Select signal CORESIGHT_EVENTI 101 **ALT5** — Select signal GPIO4_IO11

8.2.5.95 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD0)

Address: 3033_0000h base + 18Ch offset = 3033_018Ch



IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD0 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_TXD0 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).

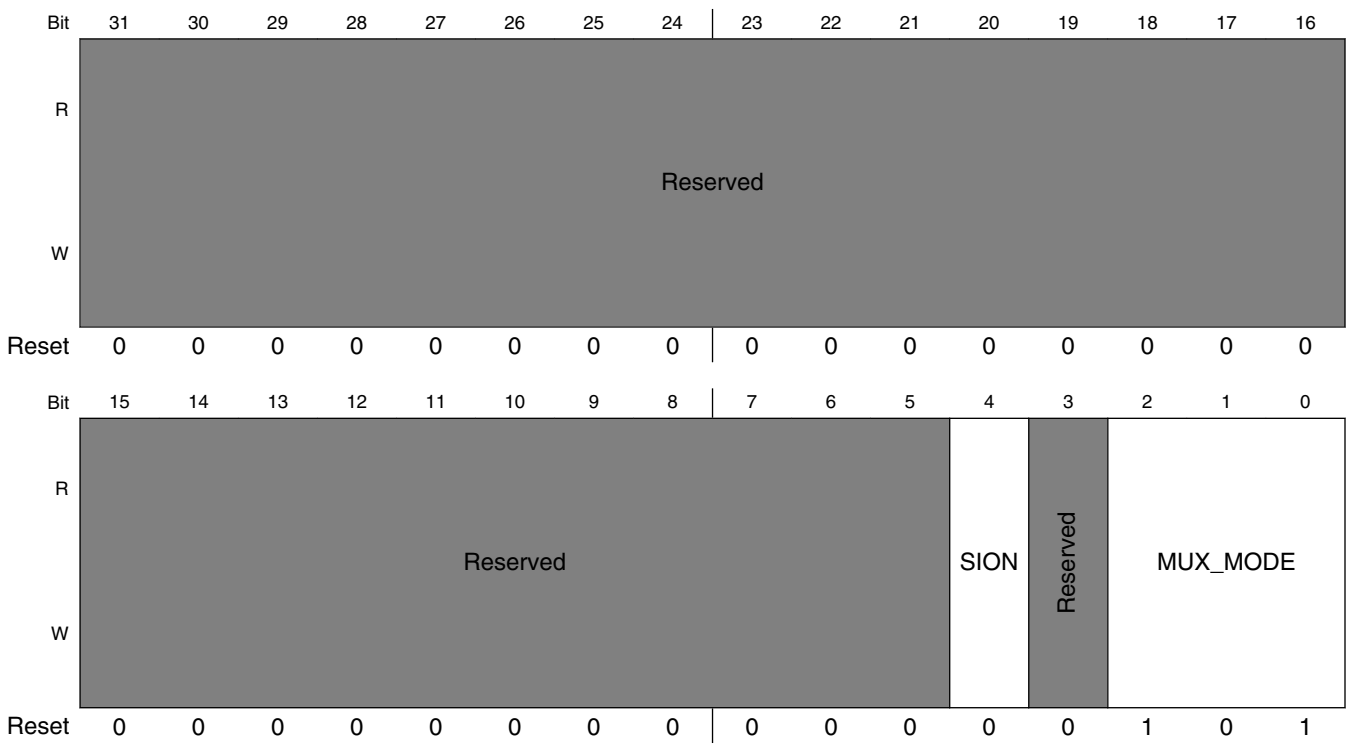
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD0 field descriptions (continued)

Field	Description
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: SAI1_TXD0. 000 **ALT0** — Select signal SAI1_TX_DATA0 001 **ALT1** — Select signal SAI5_TX_DATA0 100 **ALT4** — Select signal CORESIGHT_TRACE8 101 **ALT5** — Select signal GPIO4_IO12 110 **ALT6** — Select signal SRC_BOOT_CFG8

8.2.5.96 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD1)

Address: 3033_0000h base + 190h offset = 3033_0190h



IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD1 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field.

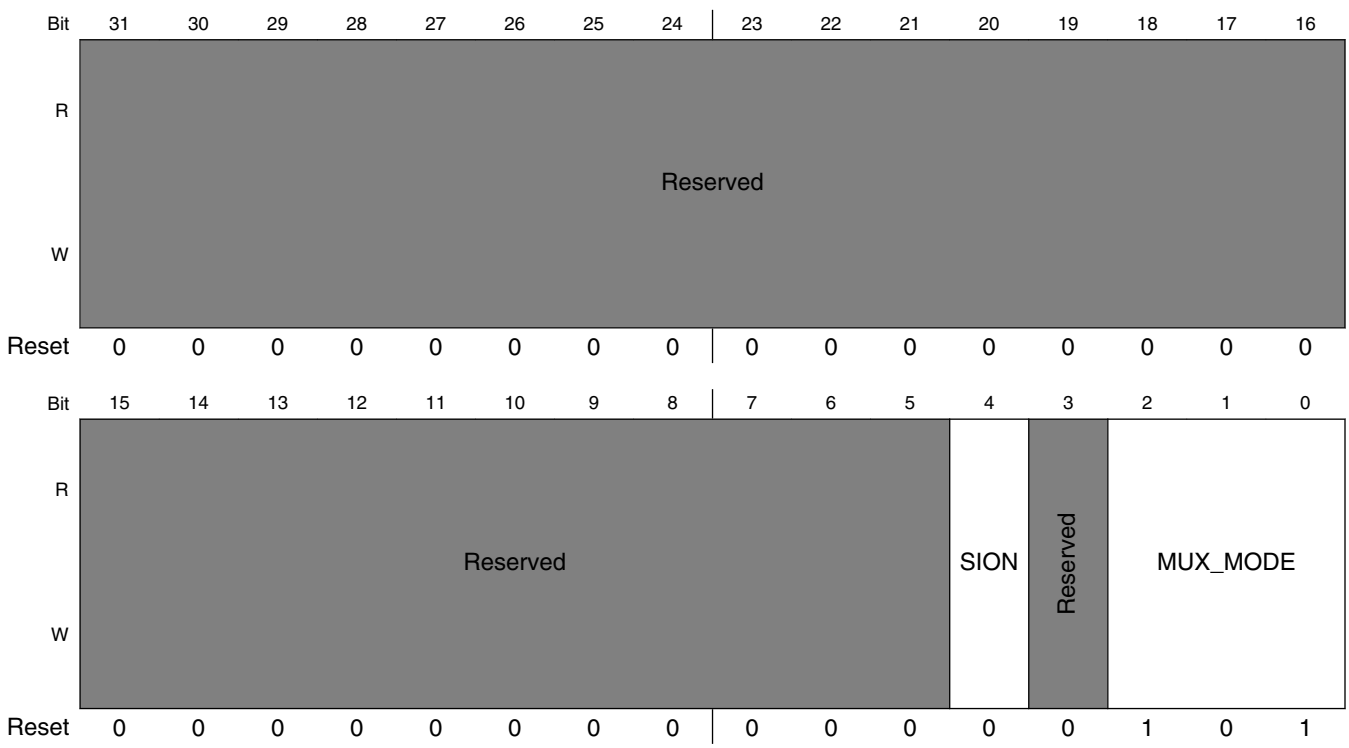
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD1 field descriptions (continued)

Field	Description
	Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_TXD1 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: SAI1_TXD1. 000 **ALT0** — Select signal SAI1_TX_DATA1 001 **ALT1** — Select signal SAI5_TX_DATA1 100 **ALT4** — Select signal CORESIGHT_TRACE9 101 **ALT5** — Select signal GPIO4_IO13 110 **ALT6** — Select signal SRC_BOOT_CFG9

8.2.5.97 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD2)

Address: 3033_0000h base + 194h offset = 3033_0194h

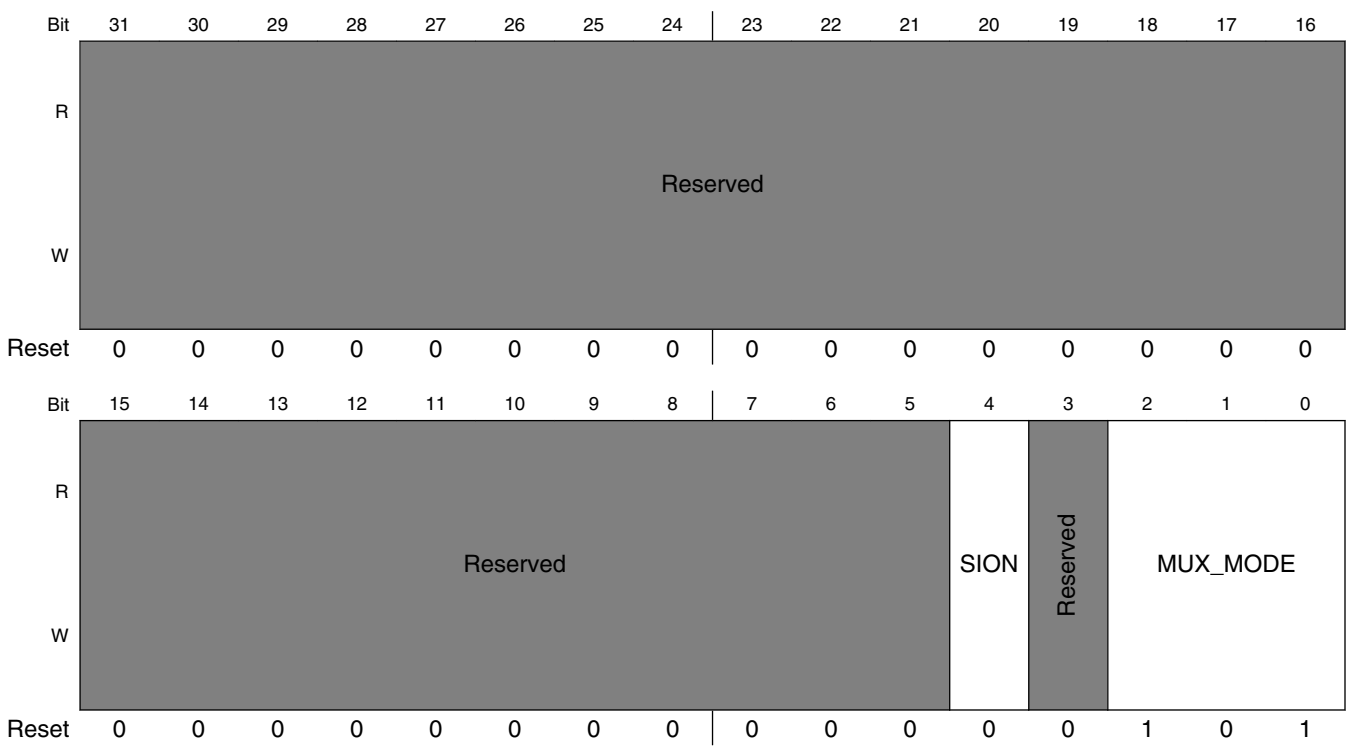


IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD2 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_TXD2 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: SAI1_TXD2. 000 **ALT0** — Select signal SAI1_TX_DATA2 001 **ALT1** — Select signal SAI5_TX_DATA2 100 **ALT4** — Select signal CORESIGHT_TRACE10 101 **ALT5** — Select signal GPIO4_IO14 110 **ALT6** — Select signal SRC_BOOT_CFG10

8.2.5.98 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD3)

Address: 3033_0000h base + 198h offset = 3033_0198h

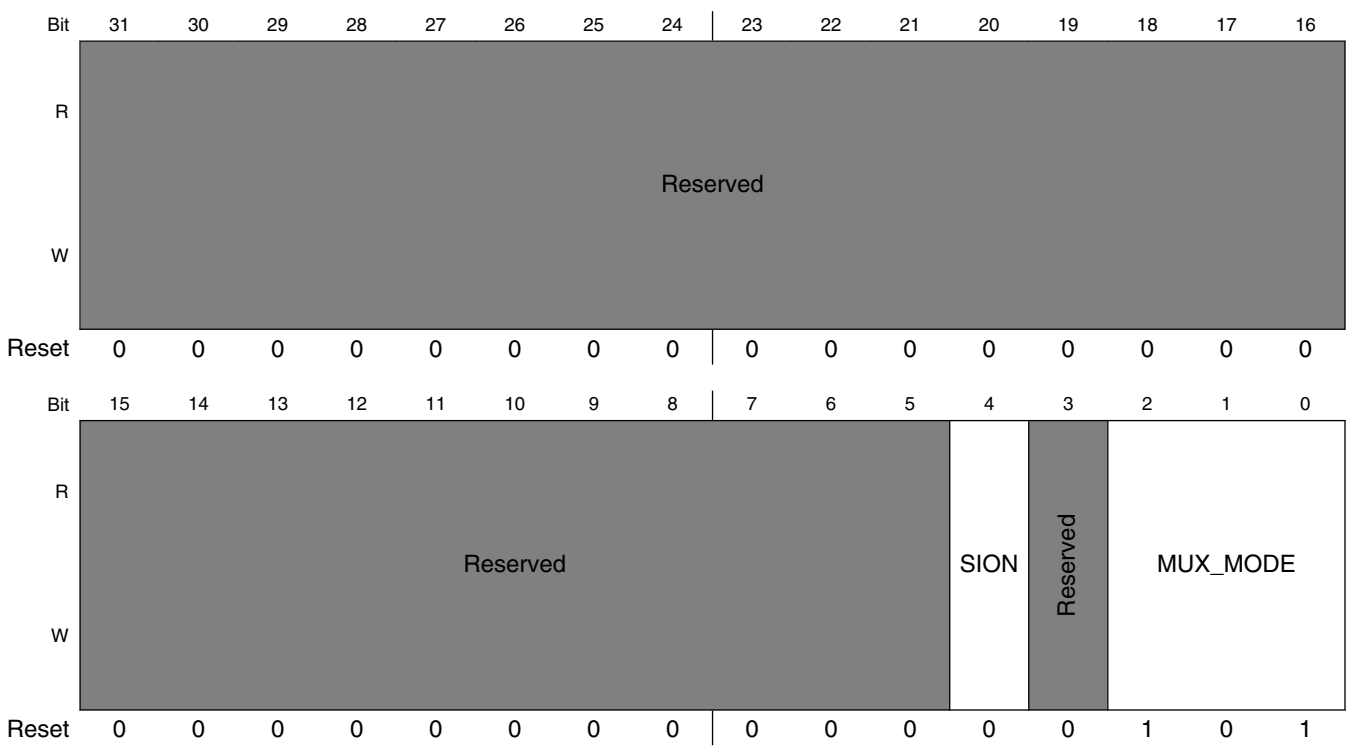


IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD3 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_TXD3 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: SAI1_TXD3. 000 **ALT0** — Select signal SAI1_TX_DATA3 001 **ALT1** — Select signal SAI5_TX_DATA3 100 **ALT4** — Select signal CORESIGHT_TRACE11 101 **ALT5** — Select signal GPIO4_IO15 110 **ALT6** — Select signal SRC_BOOT_CFG11

8.2.5.99 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD4)

Address: 3033_0000h base + 19Ch offset = 3033_019Ch

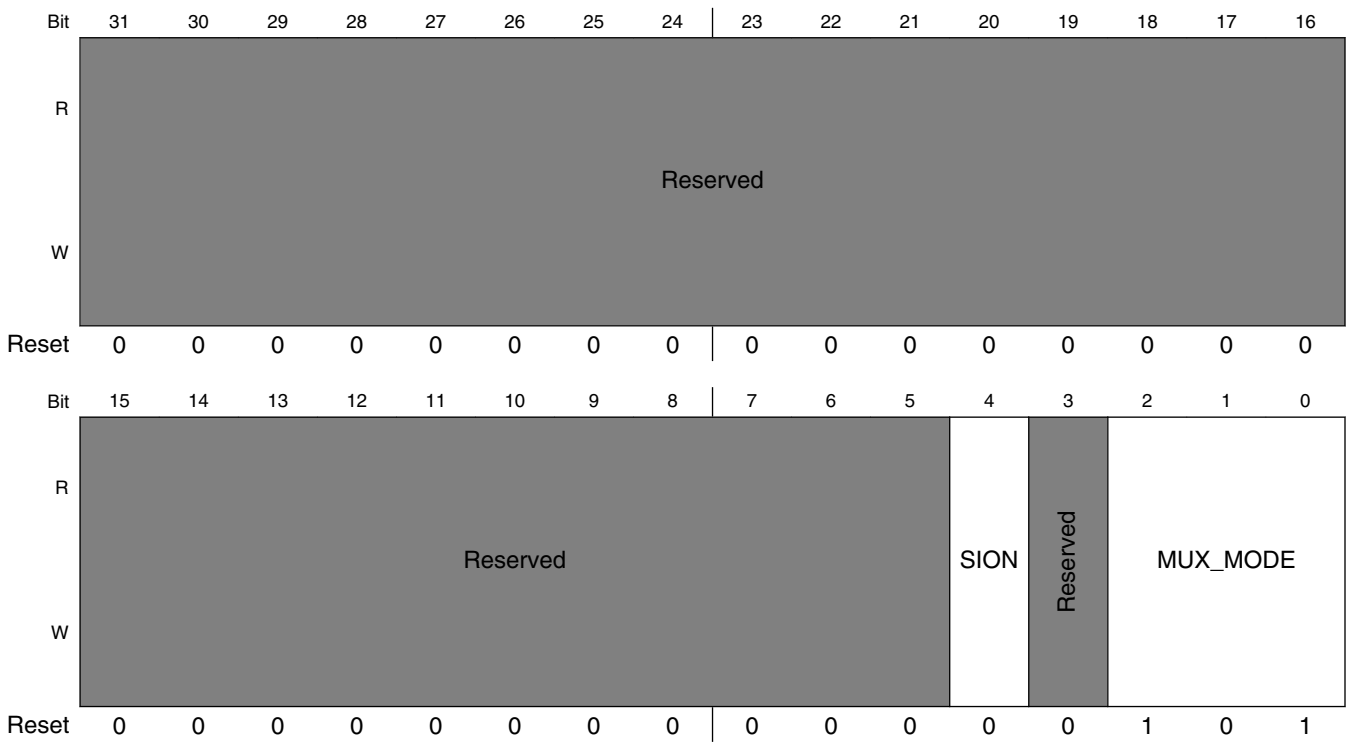


IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD4 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_TXD4 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI1_TXD4. **NOTE:** Pad SAI1_TXD4 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_TX_DATA4 001 **ALT1** — Select signal SAI6_RX_BCLK - Configure register IOMUXC_SAI6_RX_BCLK_SELECT_INPUT for mode ALT1. 010 **ALT2** — Select signal SAI6_TX_BCLK - Configure register IOMUXC_SAI6_TX_BCLK_SELECT_INPUT for mode ALT2. 100 **ALT4** — Select signal CORESIGHT_TRACE12 101 **ALT5** — Select signal GPIO4_IO16 110 **ALT6** — Select signal SRC_BOOT_CFG12

8.2.5.100 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD5)

Address: 3033_0000h base + 1A0h offset = 3033_01A0h



IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD5 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_TXD5 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI1_TXD5. **NOTE:** Pad SAI1_TXD5 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_TX_DATA5 001 **ALT1** — Select signal SAI6_RX_DATA0 - Configure register IOMUXC_SAI6_RX_DATA_SELECT_INPUT_0 for mode ALT1. 010 **ALT2** — Select signal SAI6_TX_DATA0

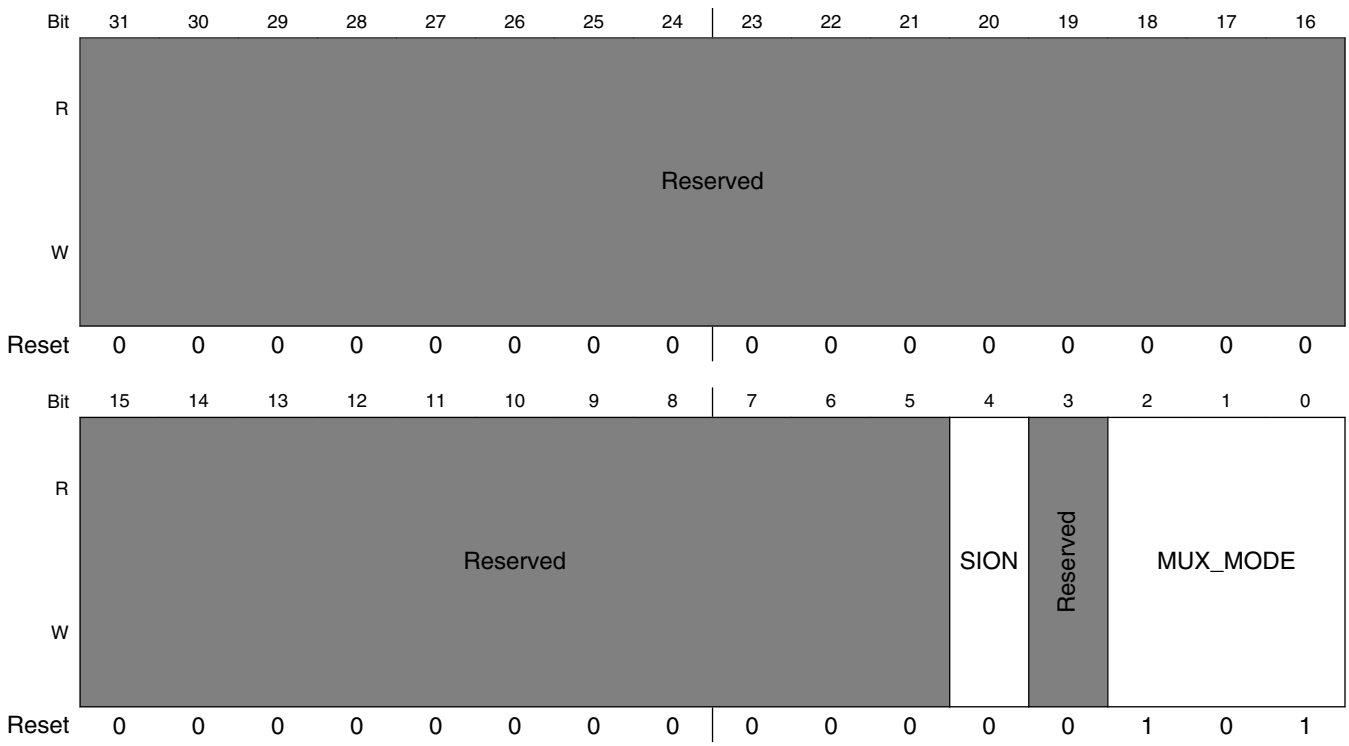
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD5 field descriptions (continued)

Field	Description
	100 **ALT4** — Select signal CORESIGHT_TRACE13 101 **ALT5** — Select signal GPIO4_IO17 110 **ALT6** — Select signal SRC_BOOT_CFG13

8.2.5.101 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD6)

Address: 3033_0000h base + 1A4h offset = 3033_01A4h



IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD6 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_TXD6 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field.

Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD6 field descriptions (continued)

Field	Description
	Select 1 of 6 iomux modes to be used for pad: SAI1_TXD6. **NOTE:** Pad SAI1_TXD6 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_TX_DATA6 001 **ALT1** — Select signal SAI6_RX_SYNC - Configure register IOMUXC_SAI6_RX_SYNC_SELECT_INPUT for mode ALT1. 010 **ALT2** — Select signal SAI6_TX_SYNC - Configure register IOMUXC_SAI6_TX_SYNC_SELECT_INPUT for mode ALT2. 100 **ALT4** — Select signal CORESIGHT_TRACE14 101 **ALT5** — Select signal GPIO4_IO18 110 **ALT6** — Select signal SRC_BOOT_CFG14

8.2.5.102 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD7)

Address: 3033_0000h base + 1A8h offset = 3033_01A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD7 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved

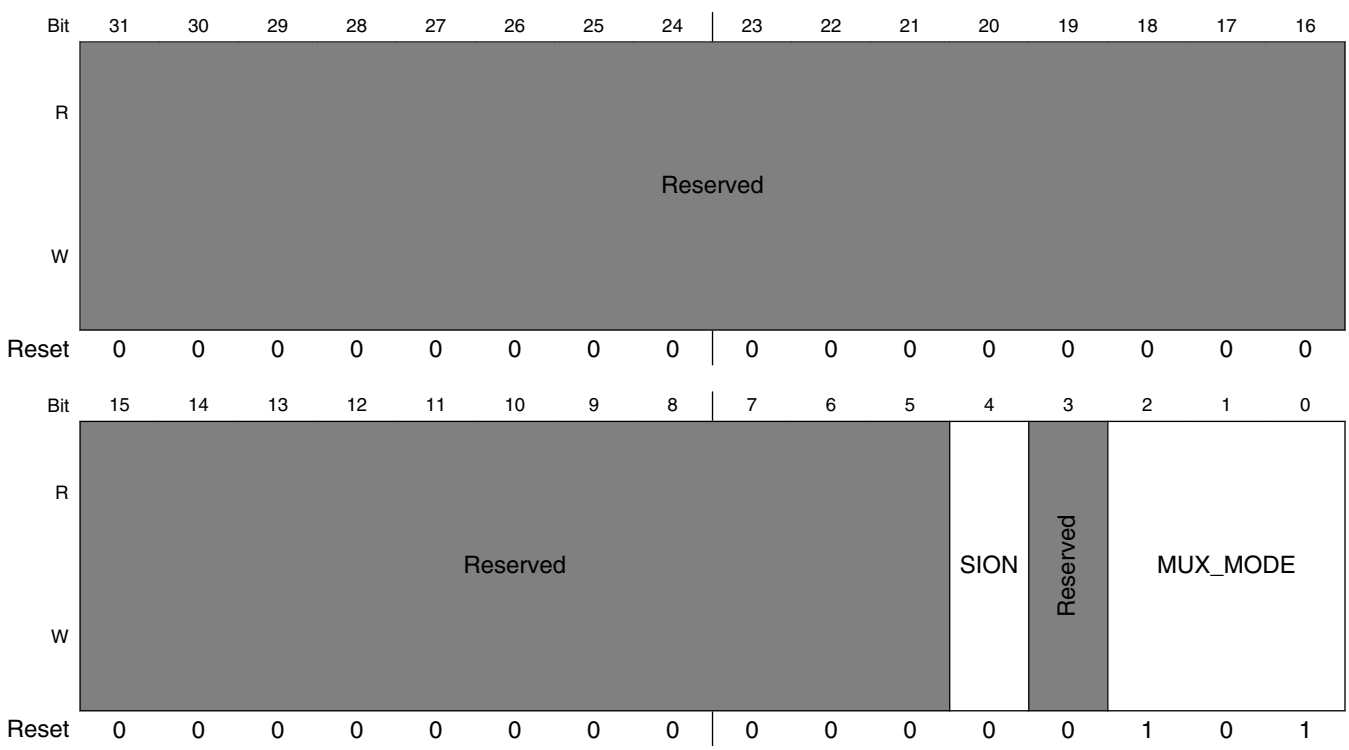
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI1_TXD7 field descriptions (continued)

Field	Description
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_TXD7 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI1_TXD7. **NOTE:** Pad SAI1_TXD7 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_TX_DATA7 001 **ALT1** — Select signal SAI6_MCLK - Configure register IOMUXC_SAI6_MCLK_SELECT_INPUT for mode ALT1. 011 **ALT3** — Select signal PDM_CLK 100 **ALT4** — Select signal CORESIGHT_TRACE15 101 **ALT5** — Select signal GPIO4_IO19 110 **ALT6** — Select signal SRC_BOOT_CFG15

8.2.5.103 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI1_MCLK)

Address: 3033_0000h base + 1ACh offset = 3033_01ACh

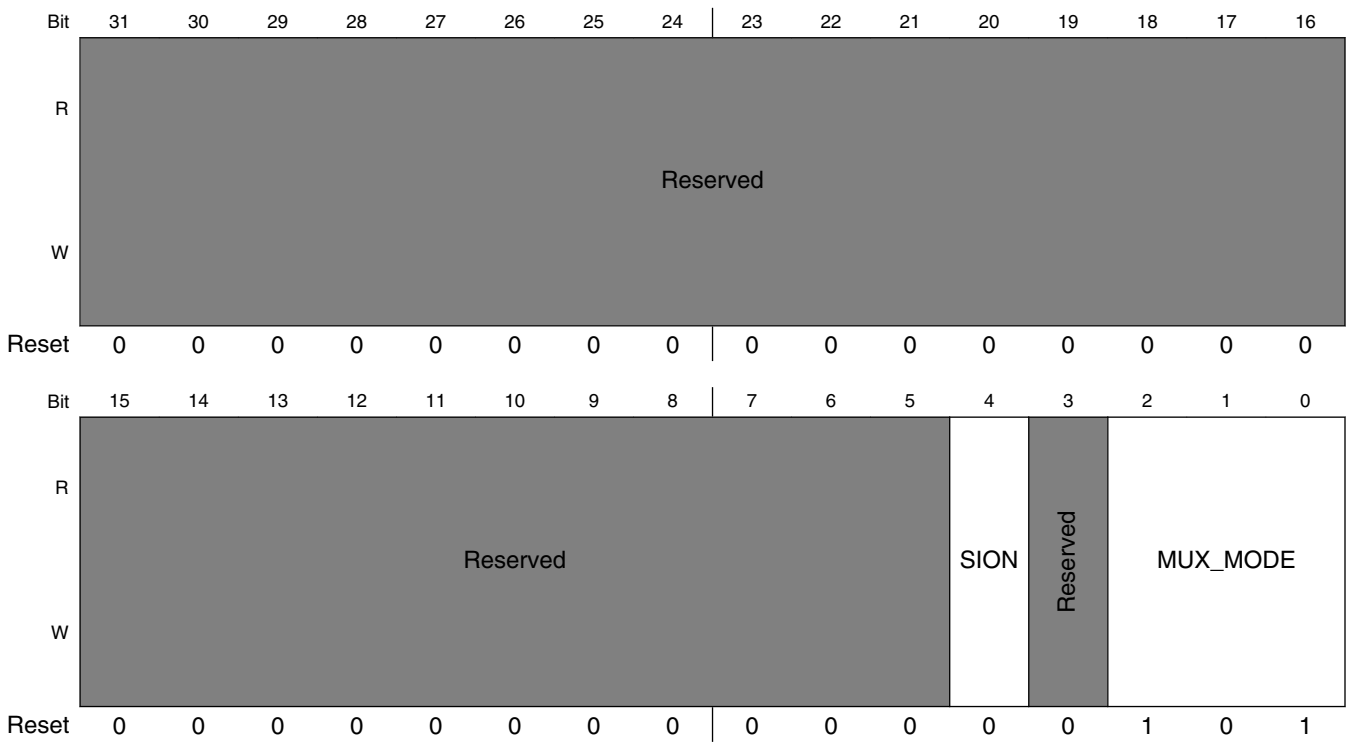


IOMUXC_SW_MUX_CTL_PAD_SAI1_MCLK field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI1_MCLK 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: SAI1_MCLK. **NOTE:** Pad SAI1_MCLK is involved in Daisy Chain. 000 **ALT0** — Select signal SAI1_MCLK 001 **ALT1** — Select signal SAI5_MCLK - Configure register IOMUXC_SAI5_MCLK_SELECT_INPUT for mode ALT1. 010 **ALT2** — Select signal SAI1_TX_BCLK - Configure register IOMUXC_SAI1_TX_BCLK_SELECT_INPUT for mode ALT2. 011 **ALT3** — Select signal PDM_CLK 101 **ALT5** — Select signal GPIO4_IO20

8.2.5.104 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_RXFS)

Address: 3033_0000h base + 1B0h offset = 3033_01B0h



IOMUXC_SW_MUX_CTL_PAD_SAI2_RXFS field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI2_RXFS 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI2_RXFS. **NOTE:** Pad SAI2_RXFS is involved in Daisy Chain. 000 **ALT0** — Select signal SAI2_RX_SYNC 001 **ALT1** — Select signal SAI5_TX_SYNC - Configure register IOMUXC_SAI5_TX_SYNC_SELECT_INPUT for mode ALT1. 010 **ALT2** — Select signal SAI5_TX_DATA1

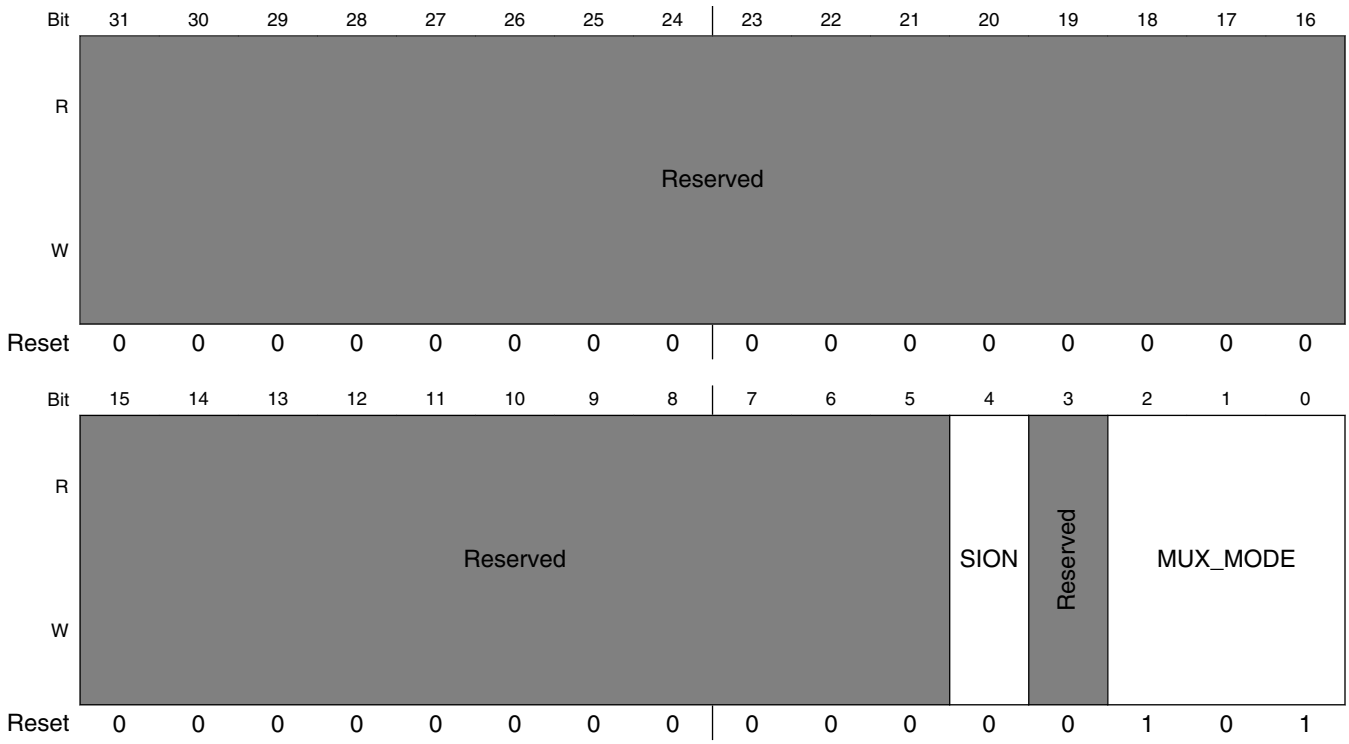
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI2_RXFS field descriptions (continued)

Field	Description
	011 **ALT3** — Select signal SAI2_RX_DATA1 100 **ALT4** — Select signal UART1_TX 101 **ALT5** — Select signal GPIO4_IO21

8.2.5.105 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_RXC)

Address: 3033_0000h base + 1B4h offset = 3033_01B4h



IOMUXC_SW_MUX_CTL_PAD_SAI2_RXC field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI2_RXC 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field.

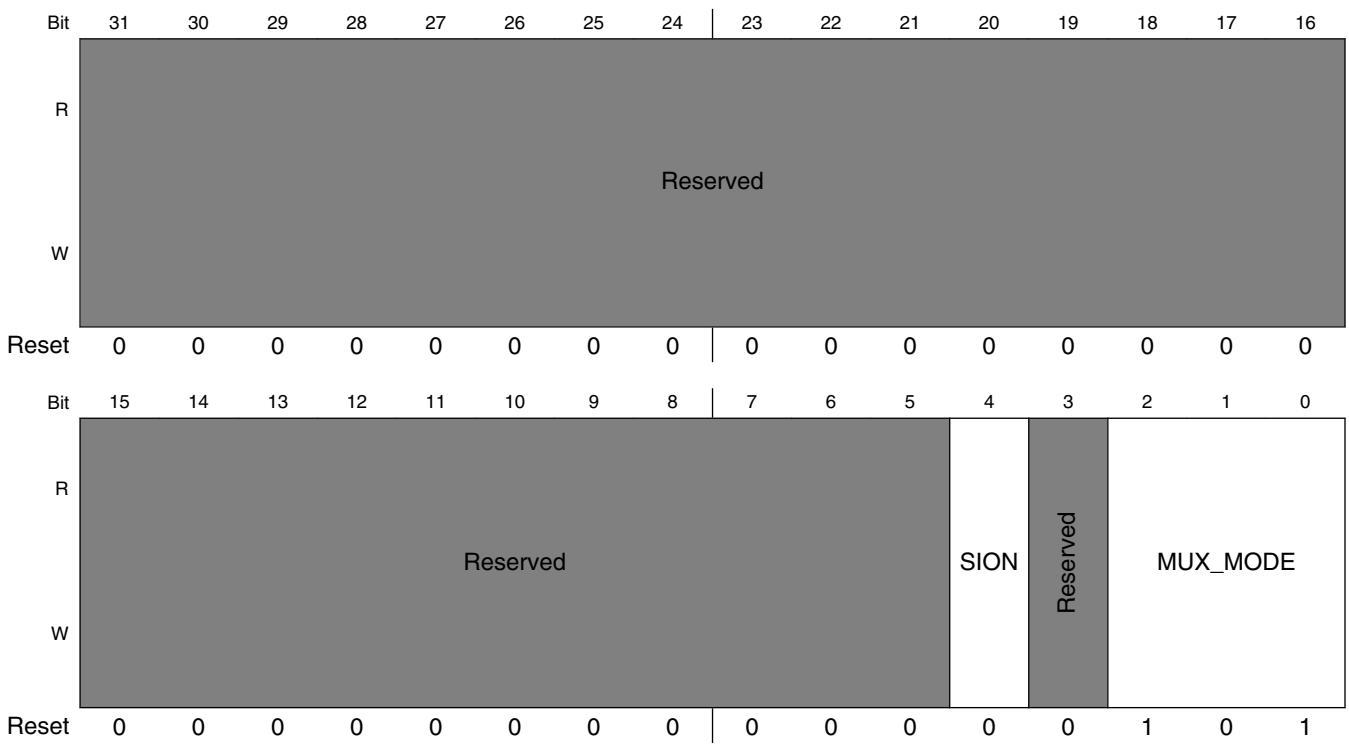
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI2_RXC field descriptions (continued)

Field	Description
	Select 1 of 4 iomux modes to be used for pad: SAI2_RXC. **NOTE:** Pad SAI2_RXC is involved in Daisy Chain. 000 **ALT0** — Select signal SAI2_RX_BCLK 001 **ALT1** — Select signal SAI5_TX_BCLK - Configure register IOMUXC_SAI5_TX_BCLK_SELECT_INPUT for mode ALT1. 100 **ALT4** — Select signal UART1_RX 101 **ALT5** — Select signal GPIO4_IO22

8.2.5.106 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_RXD0)

Address: 3033_0000h base + 1B8h offset = 3033_01B8h



IOMUXC_SW_MUX_CTL_PAD_SAI2_RXD0 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality.

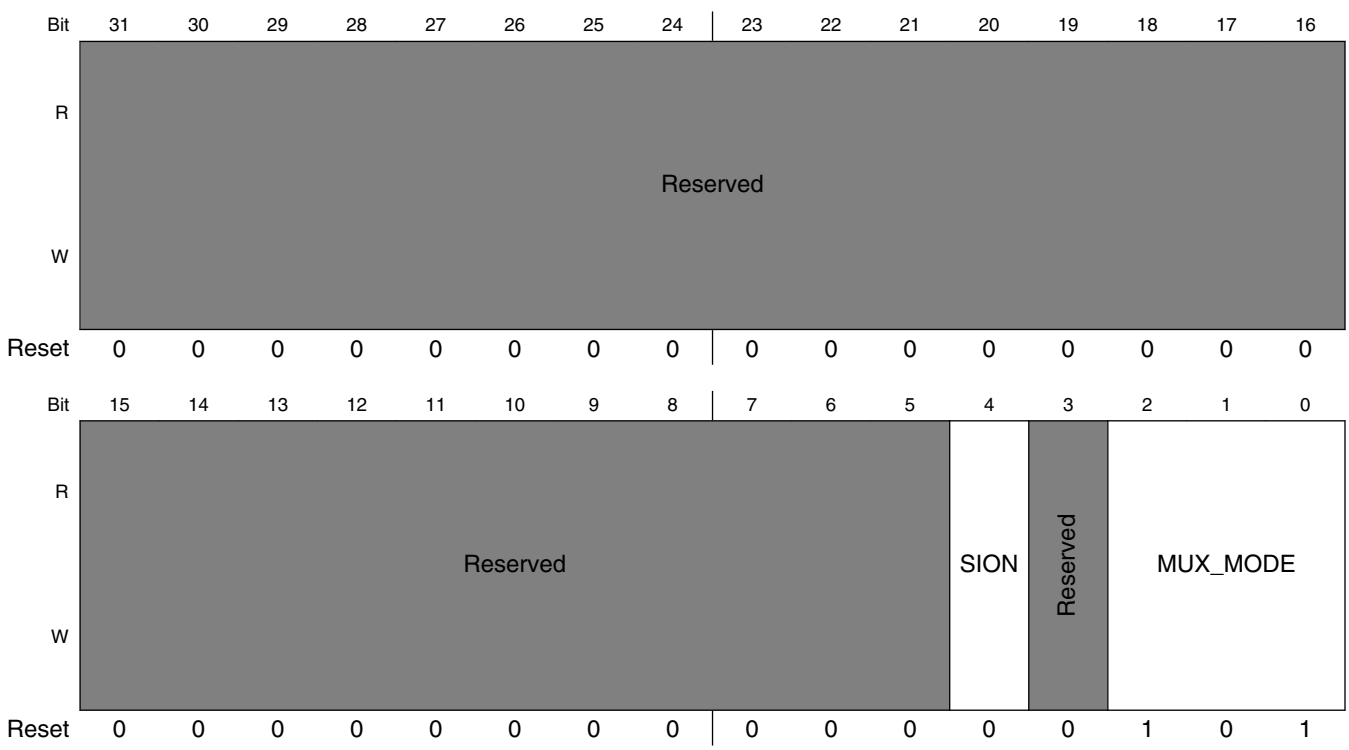
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI2_RXD0 field descriptions (continued)

Field	Description
	1 **ENABLED** — Force input path of pad SAI2_RXD0 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: SAI2_RXD0. **NOTE:** Pad SAI2_RXD0 is involved in Daisy Chain. 000 **ALT0** — Select signal SAI2_RX_DATA0 001 **ALT1** — Select signal SAI5_TX_DATA0 100 **ALT4** — Select signal UART1_RTS_B - Configure register IOMUXC_UART1_RTS_B_SELECT_INPUT for mode ALT4. 101 **ALT5** — Select signal GPIO4_IO23

8.2.5.107 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_TXFS)

Address: 3033_0000h base + 1BCh offset = 3033_01BCh

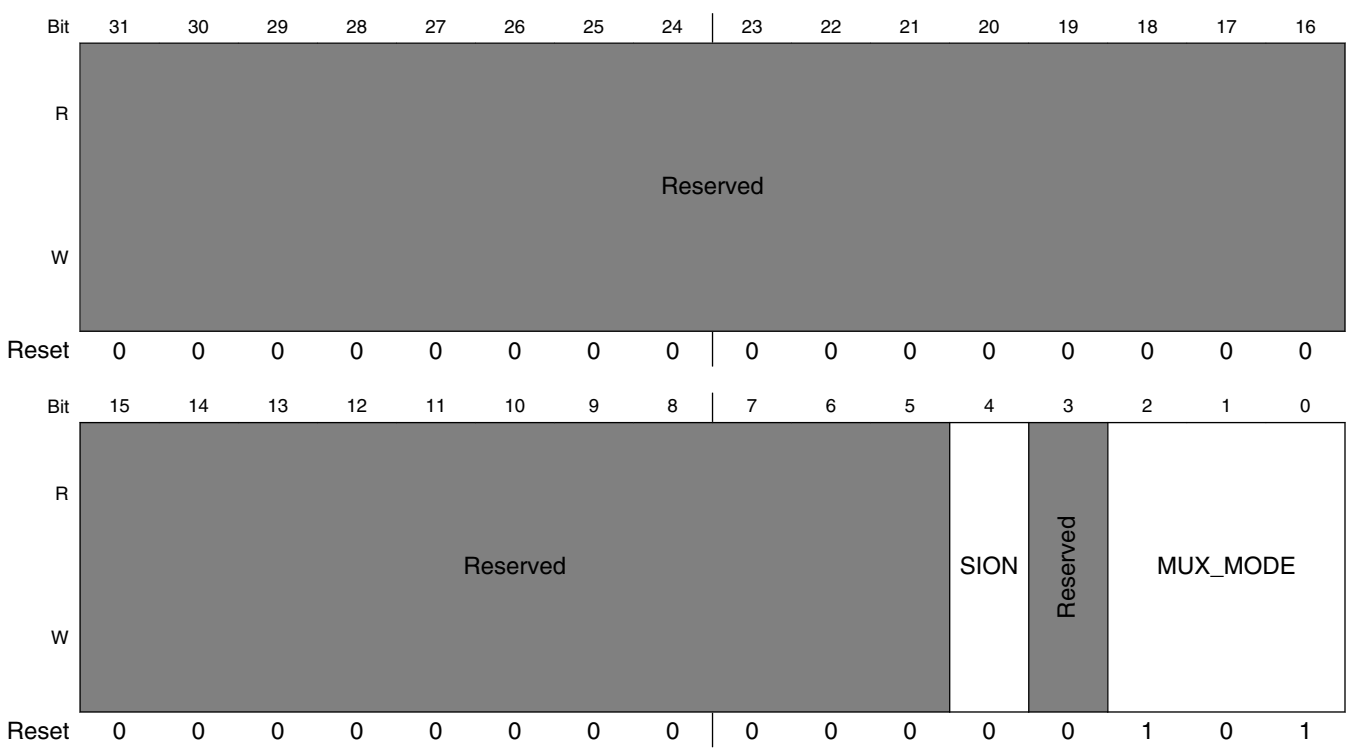


IOMUXC_SW_MUX_CTL_PAD_SAI2_TXFS field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI2_TXFS 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: SAI2_TXFS. 000 **ALT0** — Select signal SAI2_TX_SYNC 001 **ALT1** — Select signal SAI5_TX_DATA1 011 **ALT3** — Select signal SAI2_TX_DATA1 100 **ALT4** — Select signal UART1_CTS_B 101 **ALT5** — Select signal GPIO4_IO24

8.2.5.108 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_TXC)

Address: 3033_0000h base + 1C0h offset = 3033_01C0h

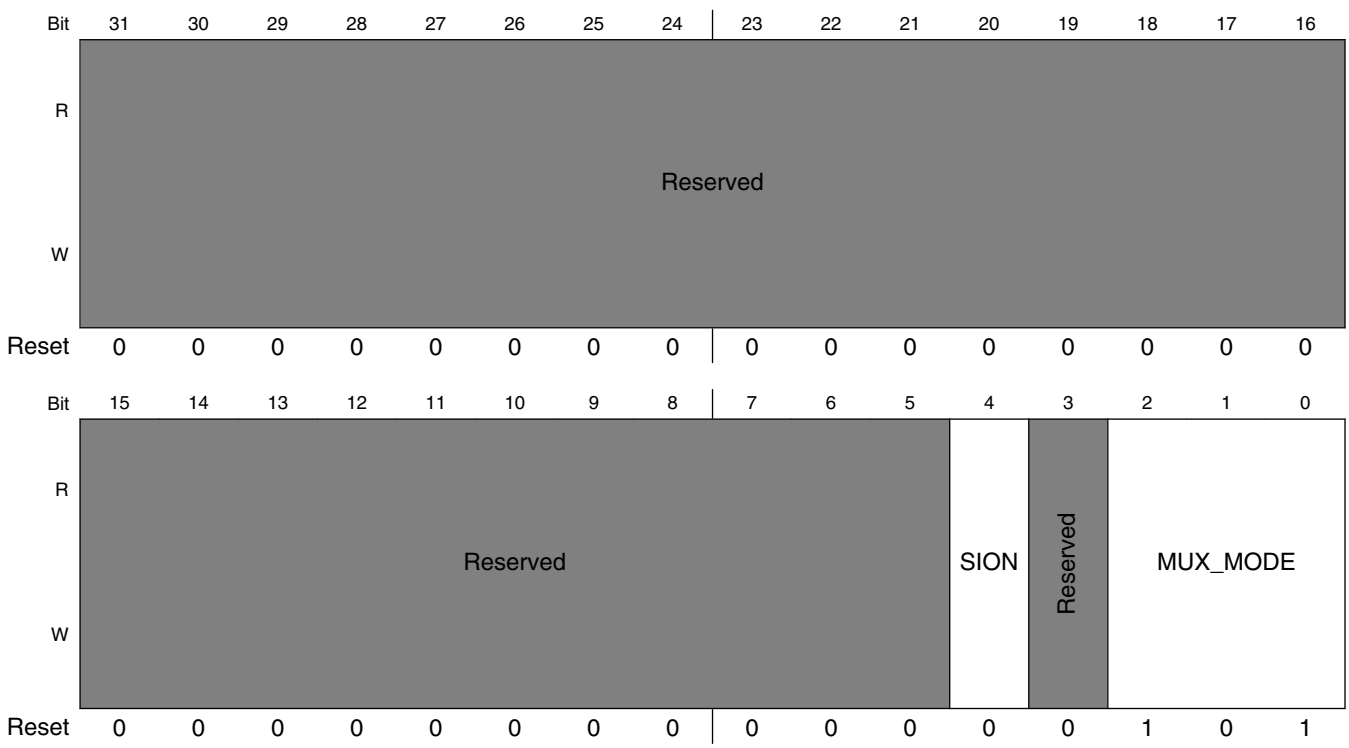


IOMUXC_SW_MUX_CTL_PAD_SAI2_TXC field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI2_TXC 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: SAI2_TXC. 000 **ALT0** — Select signal SAI2_TX_BCLK 001 **ALT1** — Select signal SAI5_TX_DATA2 101 **ALT5** — Select signal GPIO4_IO25

8.2.5.109 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_TXD0)

Address: 3033_0000h base + 1C4h offset = 3033_01C4h



IOMUXC_SW_MUX_CTL_PAD_SAI2_TXD0 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI2_TXD0 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: SAI2_TXD0. 000 **ALT0** — Select signal SAI2_TX_DATA0 001 **ALT1** — Select signal SAI5_TX_DATA3 101 **ALT5** — Select signal GPIO4_IO26

8.2.5.110 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI2_MCLK)

Address: 3033_0000h base + 1C8h offset = 3033_01C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_SAI2_MCLK field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI2_MCLK 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: SAI2_MCLK. **NOTE:** Pad SAI2_MCLK is involved in Daisy Chain. 000 **ALT0** — Select signal SAI2_MCLK 001 **ALT1** — Select signal SAI5_MCLK - Configure register IOMUXC_SAI5_MCLK_SELECT_INPUT for mode ALT1. 101 **ALT5** — Select signal GPIO4_IO27

8.2.5.111 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_RXFS)

Address: 3033_0000h base + 1CCh offset = 3033_01CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

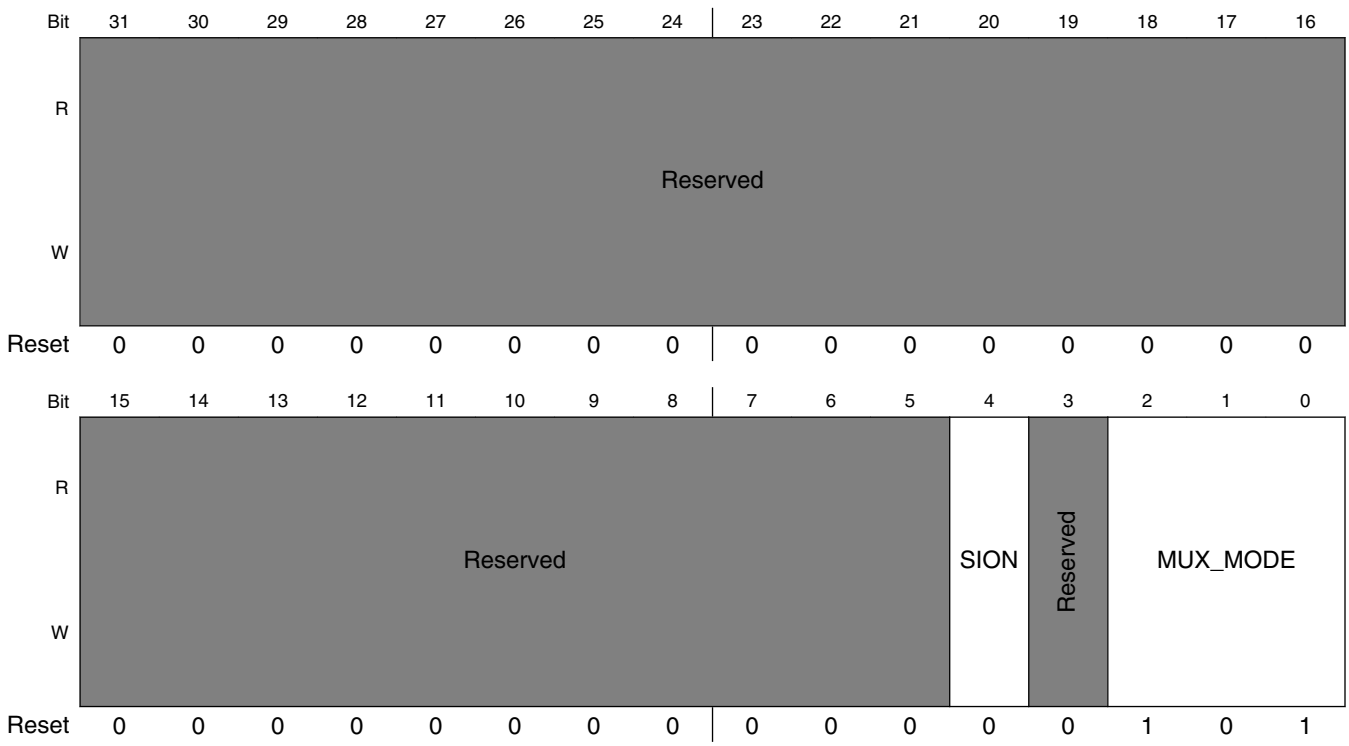
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_SAI3_RXFS field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI3_RXFS 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: SAI3_RXFS. **NOTE:** Pad SAI3_RXFS is involved in Daisy Chain. 000 **ALT0** — Select signal SAI3_RX_SYNC 001 **ALT1** — Select signal GPT1_CAPTURE1 010 **ALT2** — Select signal SAI5_RX_SYNC - Configure register IOMUXC_SAI5_RX_SYNC_SELECT_INPUT for mode ALT2. 011 **ALT3** — Select signal SAI3_RX_DATA1 101 **ALT5** — Select signal GPIO4_IO28

8.2.5.112 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_RXC)

Address: 3033_0000h base + 1D0h offset = 3033_01D0h



IOMUXC_SW_MUX_CTL_PAD_SAI3_RXC field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI3_RXC 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: SAI3_RXC. **NOTE:** Pad SAI3_RXC is involved in Daisy Chain. 000 **ALT0** — Select signal SAI3_RX_BCLK 001 **ALT1** — Select signal GPT1_CLK 010 **ALT2** — Select signal SAI5_RX_BCLK - Configure register IOMUXC_SAI5_RX_BCLK_SELECT_INPUT for mode ALT2.

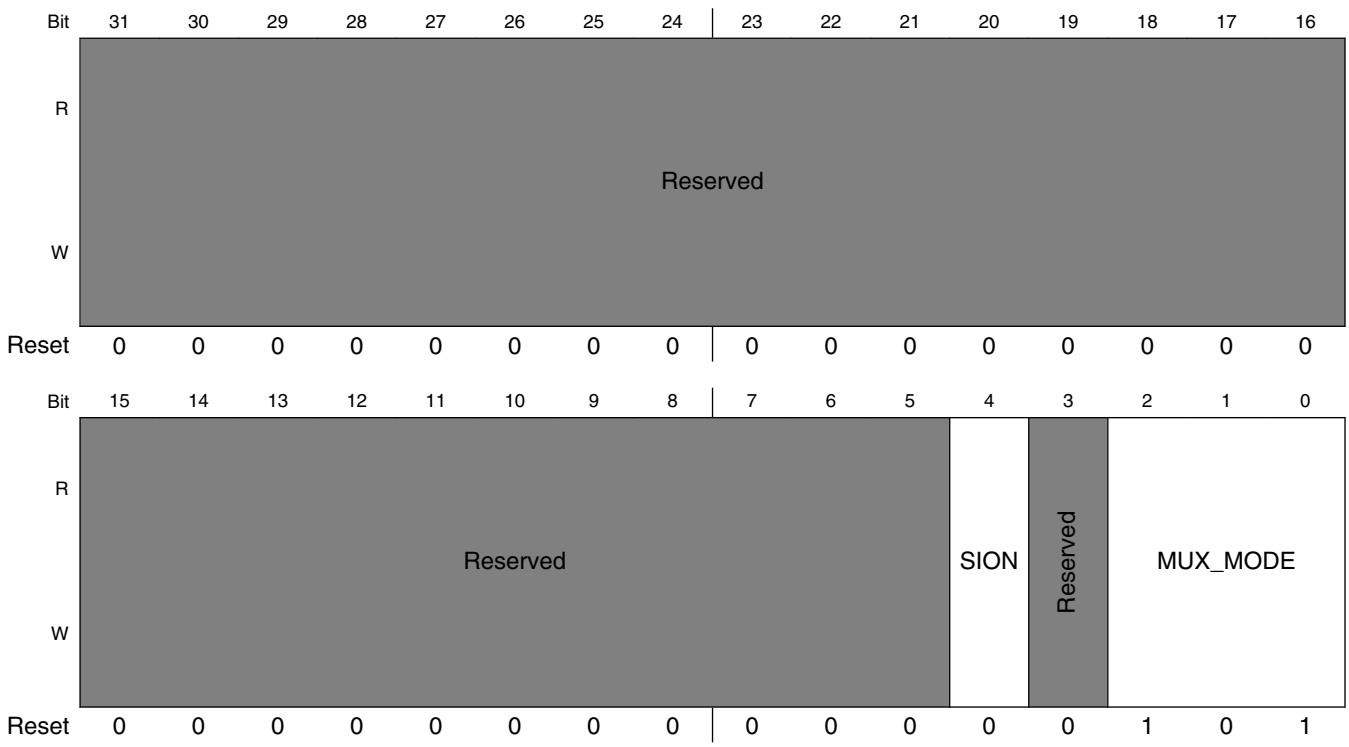
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI3_RXC field descriptions (continued)

Field	Description
	100 **ALT4** — Select signal UART2_CTS_B 101 **ALT5** — Select signal GPIO4_IO29

8.2.5.113 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_RXD)

Address: 3033_0000h base + 1D4h offset = 3033_01D4h



IOMUXC_SW_MUX_CTL_PAD_SAI3_RXD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI3_RXD 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: SAI3_RXD.

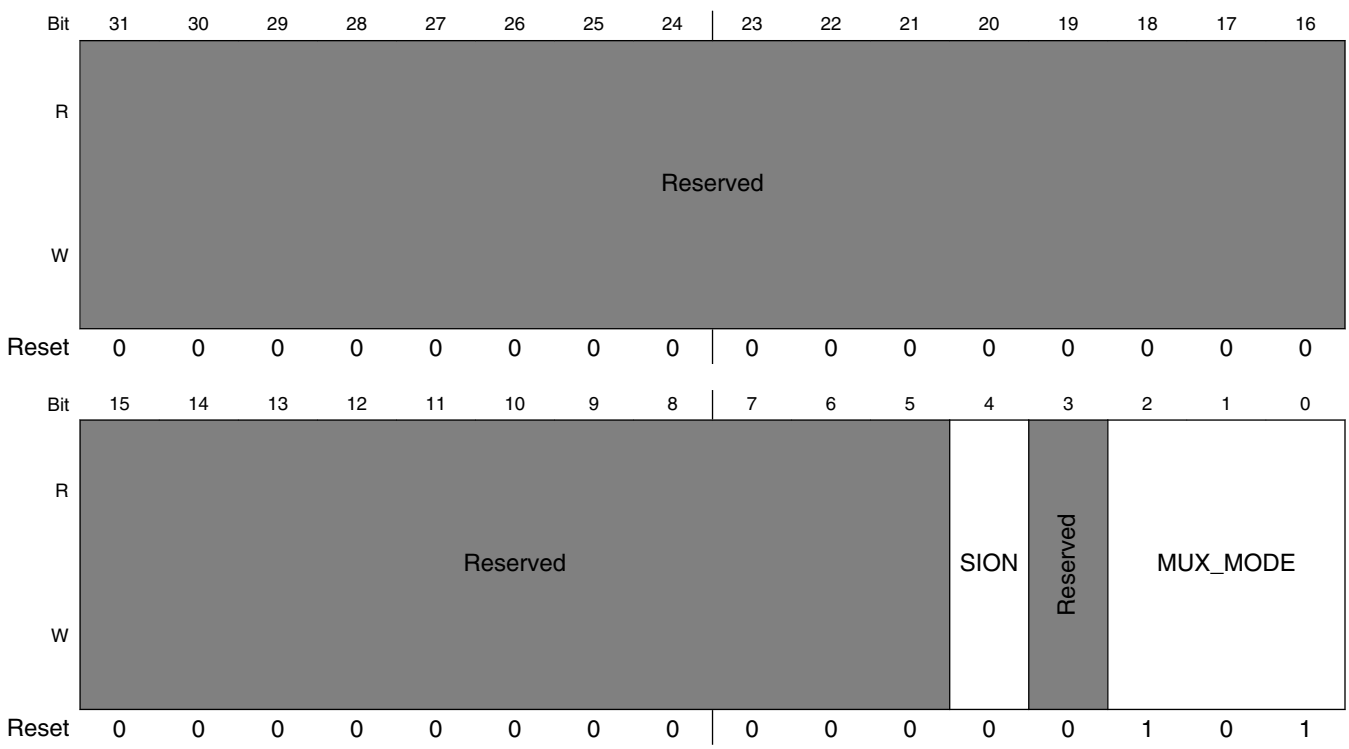
Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI3_RXD field descriptions (continued)

Field	Description
	NOTE: Pad SAI3_RXD is involved in Daisy Chain. 000 **ALT0** — Select signal SAI3_RX_DATA0 001 **ALT1** — Select signal GPT1_COMPARE1 010 **ALT2** — Select signal SAI5_RX_DATA0 - Configure register IOMUXC_SAI5_RX_DATA_SELECT_INPUT_0 for mode ALT2. 100 **ALT4** — Select signal UART2_RTS_B - Configure register IOMUXC_UART2_RTS_B_SELECT_INPUT for mode ALT4. 101 **ALT5** — Select signal GPIO4_IO30

8.2.5.114 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_TXFS)

Address: 3033_0000h base + 1D8h offset = 3033_01D8h



IOMUXC_SW_MUX_CTL_PAD_SAI3_TXFS field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality.

Table continues on the next page...

IOMUXC_SW_MUX_CTL_PAD_SAI3_TXFS field descriptions (continued)

Field	Description
	1 **ENABLED** — Force input path of pad SAI3_TXFS 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: SAI3_TXFS. **NOTE:** Pad SAI3_TXFS is involved in Daisy Chain. 000 **ALT0** — Select signal SAI3_TX_SYNC 001 **ALT1** — Select signal GPT1_CAPTURE2 010 **ALT2** — Select signal SAI5_RX_DATA1 - Configure register IOMUXC_SAI5_RX_DATA_SELECT_INPUT_1 for mode ALT2. 011 **ALT3** — Select signal SAI3_TX_DATA1 100 **ALT4** — Select signal UART2_RX 101 **ALT5** — Select signal GPIO4_IO31

8.2.5.115 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_TXC)

Address: 3033_0000h base + 1DCh offset = 3033_01DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

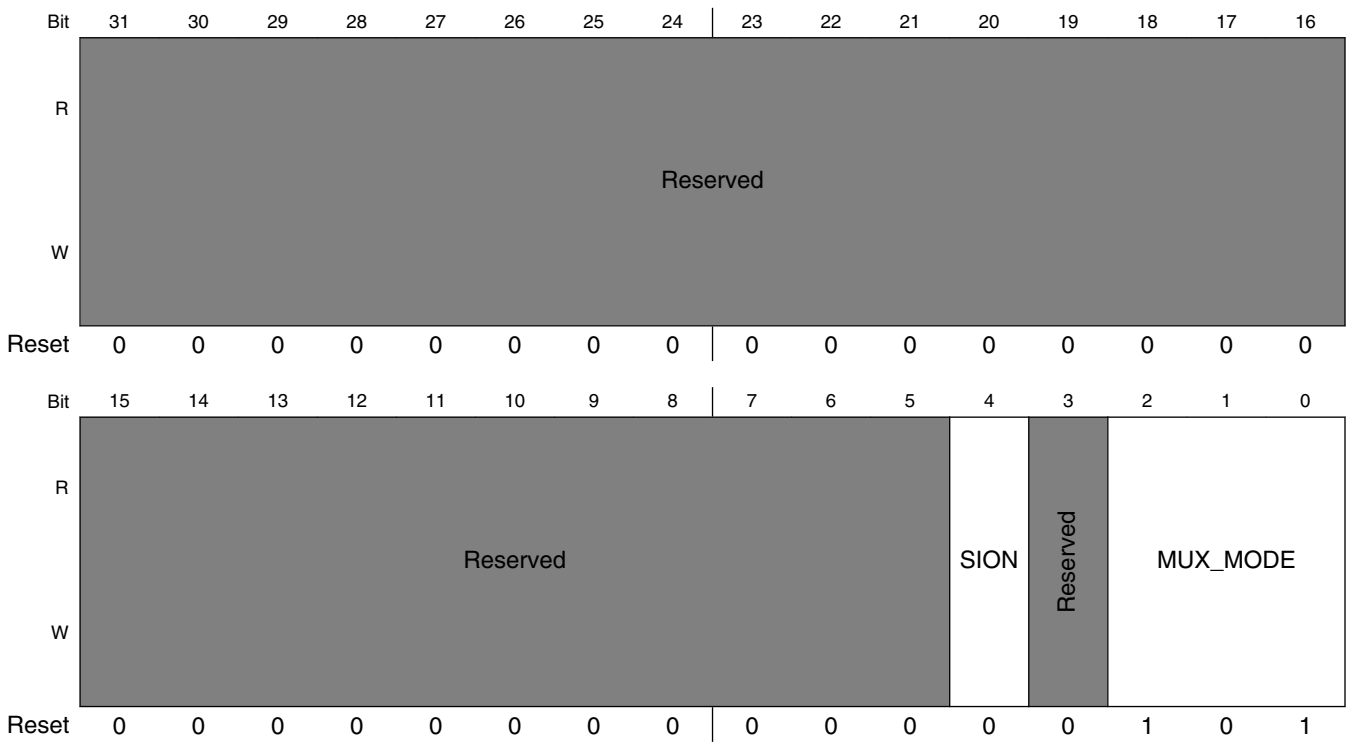
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

IOMUXC_SW_MUX_CTL_PAD_SAI3_TXC field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI3_TXC 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for pad: SAI3_TXC. **NOTE:** Pad SAI3_TXC is involved in Daisy Chain. 000 **ALT0** — Select signal SAI3_TX_BCLK 001 **ALT1** — Select signal GPT1_COMPARE2 010 **ALT2** — Select signal SAI5_RX_DATA2 - Configure register IOMUXC_SAI5_RX_DATA_SELECT_INPUT_2 for mode ALT2. 100 **ALT4** — Select signal UART2_TX 101 **ALT5** — Select signal GPIO5_IO00

8.2.5.116 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_TXD)

Address: 3033_0000h base + 1E0h offset = 3033_01E0h

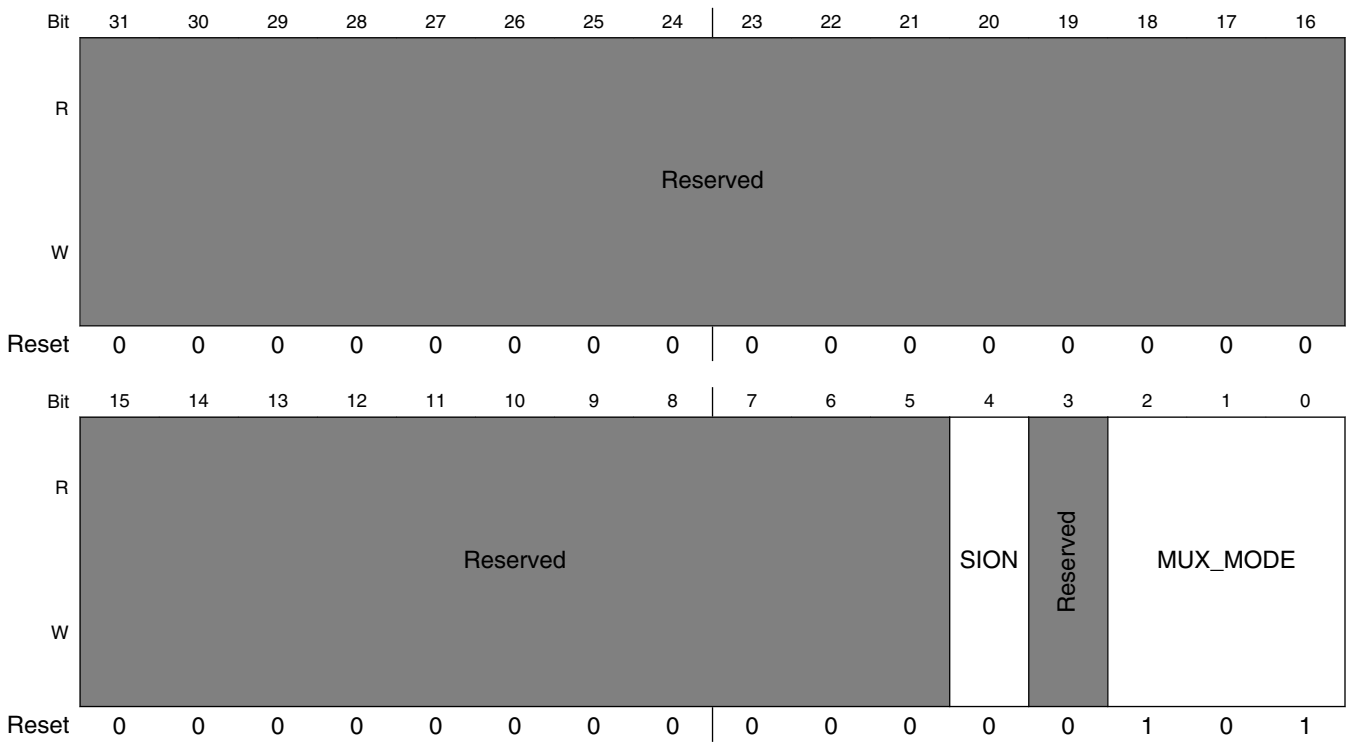


IOMUXC_SW_MUX_CTL_PAD_SAI3_TXD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI3_TXD 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: SAI3_TXD. **NOTE:** Pad SAI3_TXD is involved in Daisy Chain. 000 **ALT0** — Select signal SAI3_TX_DATA0 001 **ALT1** — Select signal GPT1_COMPARE3 010 **ALT2** — Select signal SAI5_RX_DATA3 - Configure register IOMUXC_SAI5_RX_DATA_SELECT_INPUT_3 for mode ALT2. 101 **ALT5** — Select signal GPIO5_IO01

8.2.5.117 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SAI3_MCLK)

Address: 3033_0000h base + 1E4h offset = 3033_01E4h

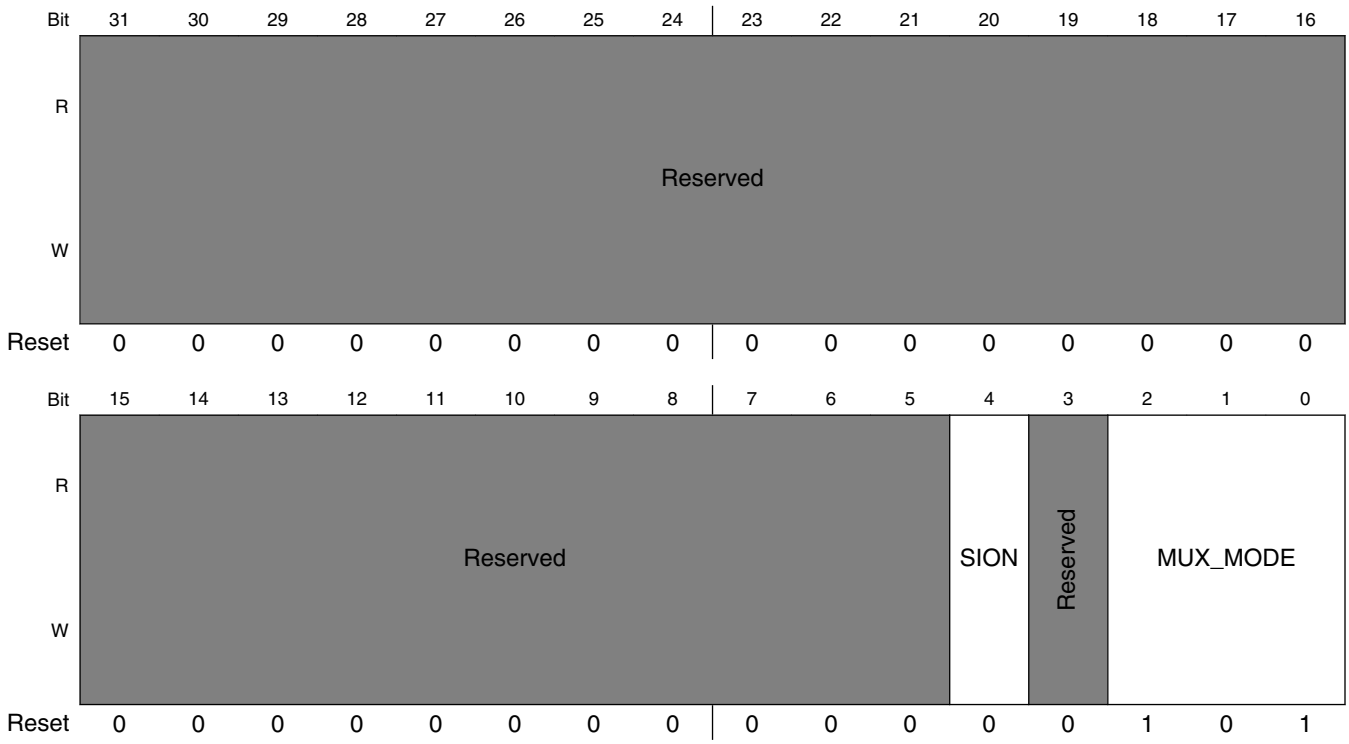


IOMUXC_SW_MUX_CTL_PAD_SAI3_MCLK field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SAI3_MCLK 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: SAI3_MCLK. **NOTE:** Pad SAI3_MCLK is involved in Daisy Chain. 000 **ALT0** — Select signal SAI3_MCLK 001 **ALT1** — Select signal PWM4_OUT 010 **ALT2** — Select signal SAI5_MCLK - Configure register IOMUXC_SAI5_MCLK_SELECT_INPUT for mode ALT2. 101 **ALT5** — Select signal GPIO5_IO02

8.2.5.118 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SPDIF_TX)

Address: 3033_0000h base + 1E8h offset = 3033_01E8h

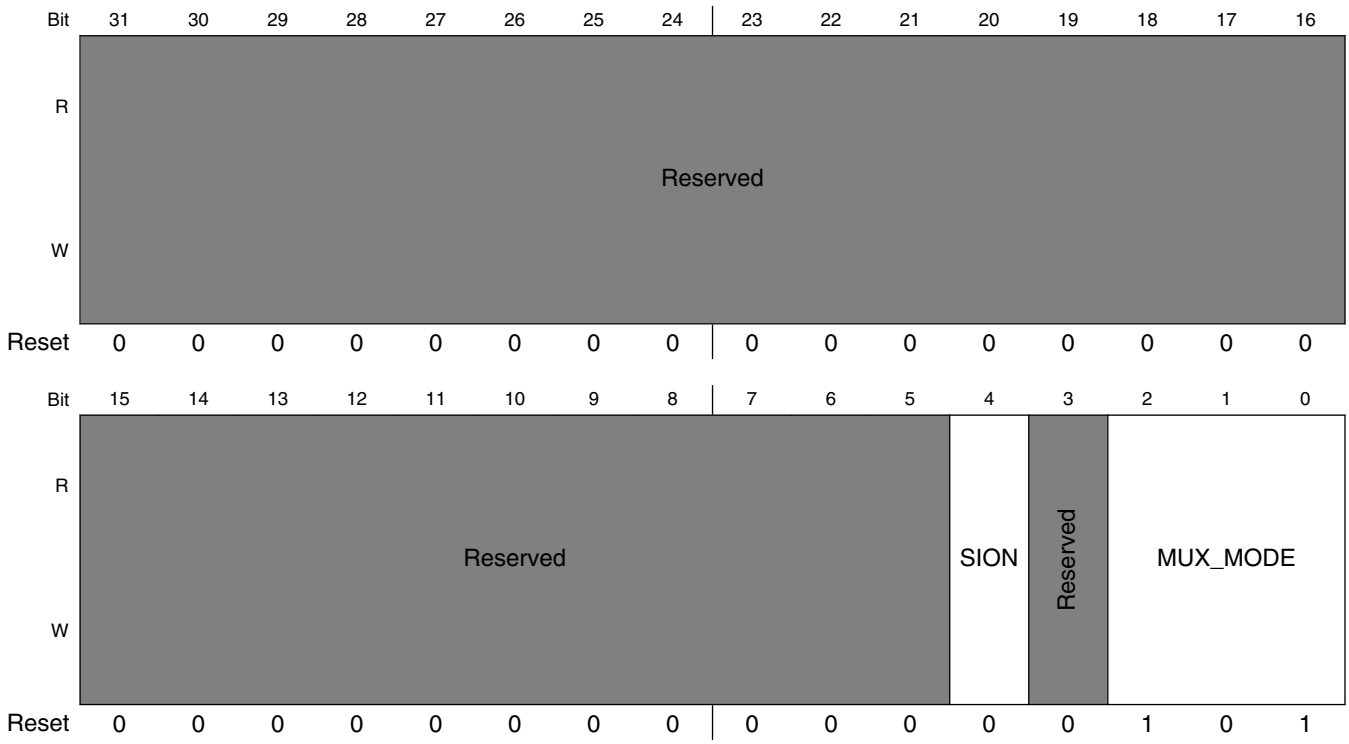


IOMUXC_SW_MUX_CTL_PAD_SPDIF_TX field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SPDIF_TX 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: SPDIF_TX. 000 **ALT0** — Select signal SPDIF1_OUT 001 **ALT1** — Select signal PWM3_OUT 101 **ALT5** — Select signal GPIO5_IO03

8.2.5.119 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SPDIF_RX)

Address: 3033_0000h base + 1ECh offset = 3033_01ECh

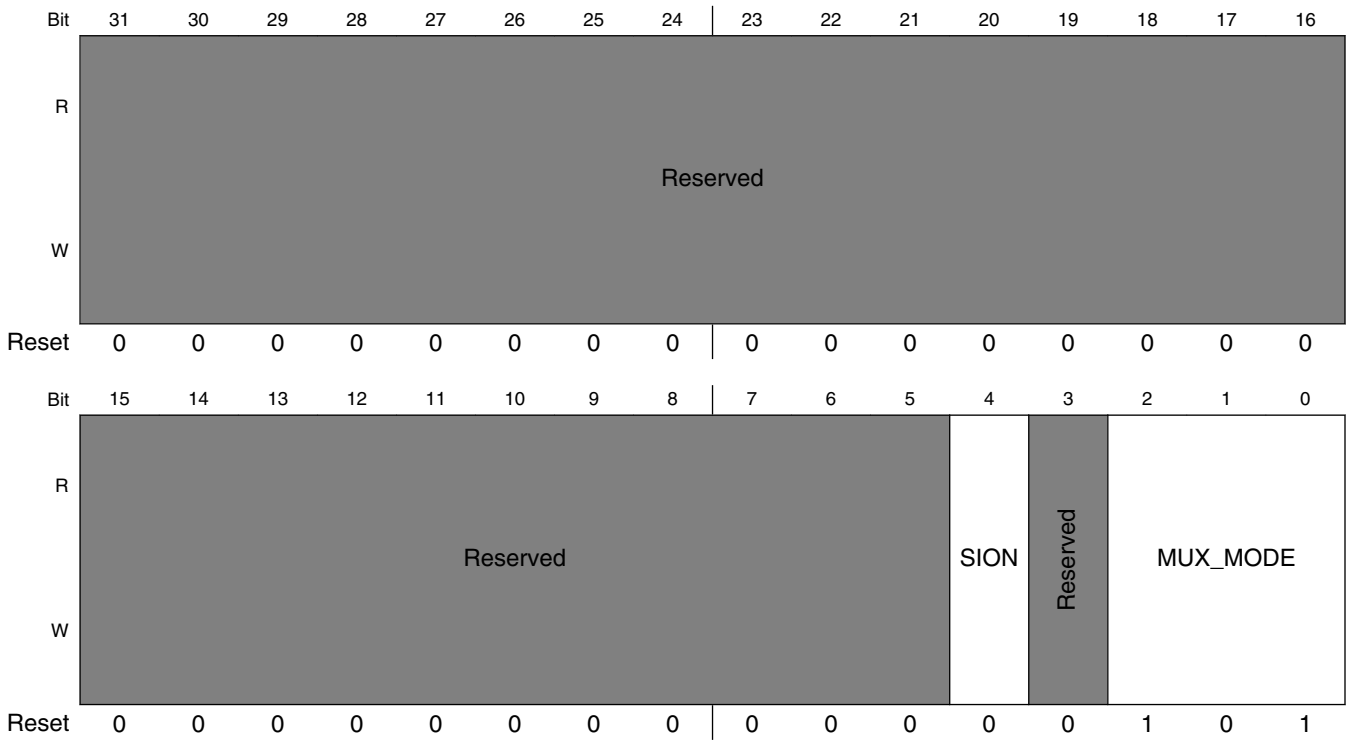


IOMUXC_SW_MUX_CTL_PAD_SPDIF_RX field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SPDIF_RX 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: SPDIF_RX. 000 **ALT0** — Select signal SPDIF1_IN 001 **ALT1** — Select signal PWM2_OUT 101 **ALT5** — Select signal GPIO5_IO04

8.2.5.120 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SPDIF_EXT_CLK)

Address: 3033_0000h base + 1F0h offset = 3033_01F0h

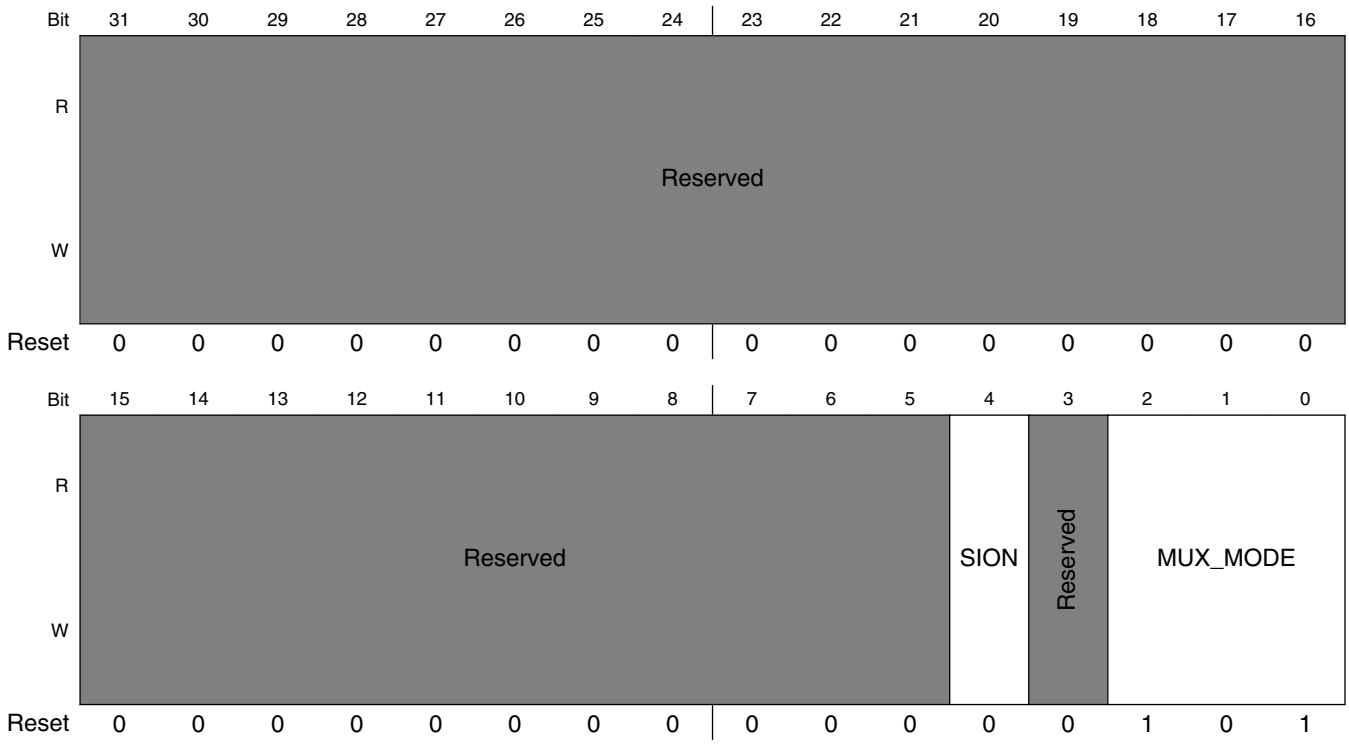


IOMUXC_SW_MUX_CTL_PAD_SPDIF_EXT_CLK field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SPDIF_EXT_CLK 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: SPDIF_EXT_CLK. 000 **ALT0** — Select signal SPDIF1_EXT_CLK 001 **ALT1** — Select signal PWM1_OUT 101 **ALT5** — Select signal GPIO5_IO05

8.2.5.121 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI1_SCLK)

Address: 3033_0000h base + 1F4h offset = 3033_01F4h

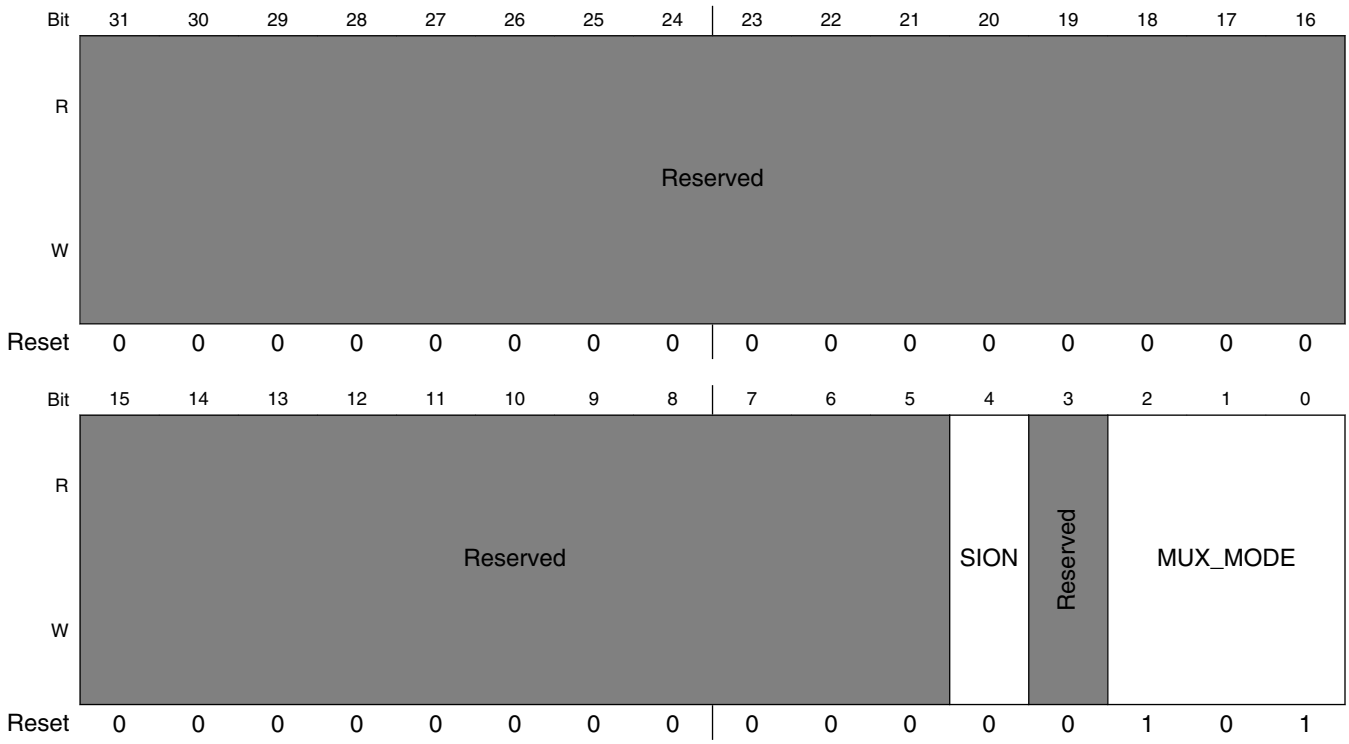


IOMUXC_SW_MUX_CTL_PAD_ECSPI1_SCLK field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ECSPI1_SCLK 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: ECSPI1_SCLK. 000 **ALT0** — Select signal ECSPI1_SCLK 001 **ALT1** — Select signal UART3_RX 101 **ALT5** — Select signal GPIO5_IO06

8.2.5.122 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI1_MOSI)

Address: 3033_0000h base + 1F8h offset = 3033_01F8h

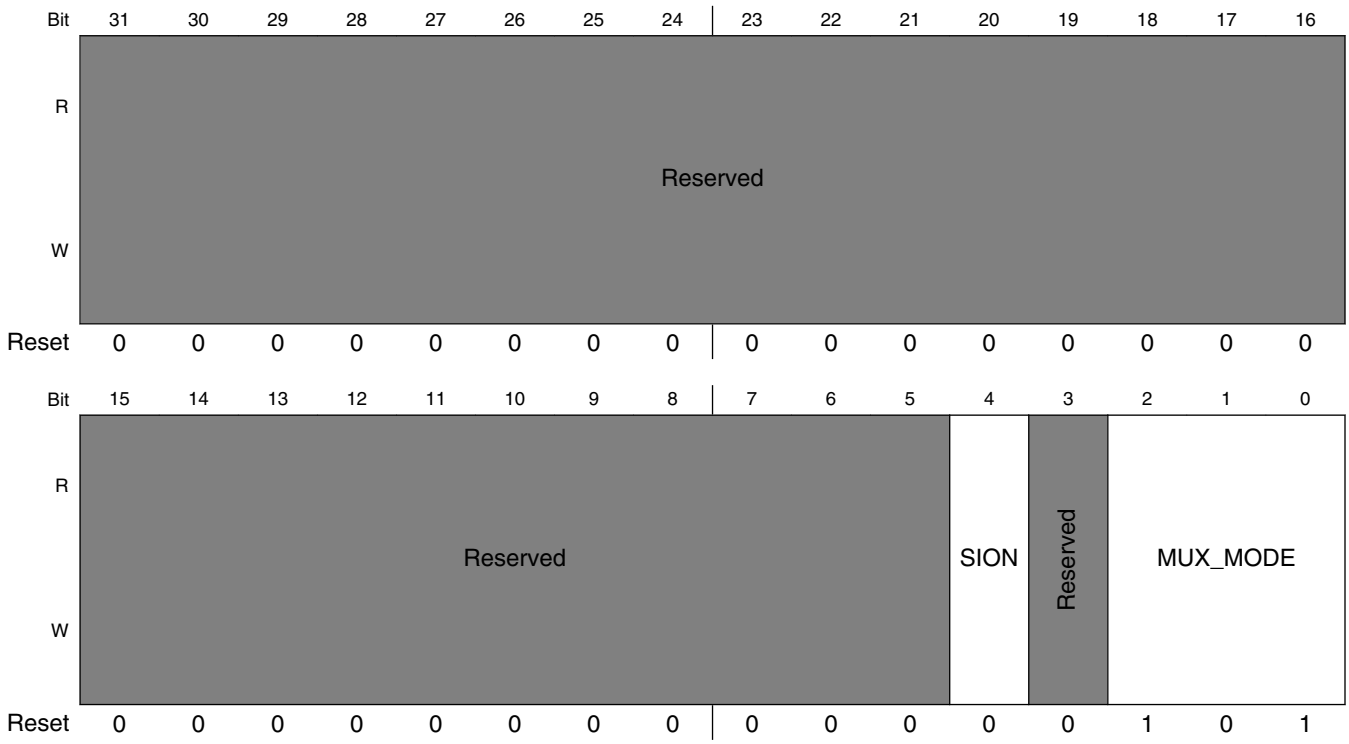


IOMUXC_SW_MUX_CTL_PAD_ECSPI1_MOSI field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ECSPI1_MOSI 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: ECSPI1_MOSI. 000 **ALT0** — Select signal ECSPI1_MOSI 001 **ALT1** — Select signal UART3_TX 101 **ALT5** — Select signal GPIO5_IO07

8.2.5.123 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI1_MISO)

Address: 3033_0000h base + 1FCh offset = 3033_01FCh

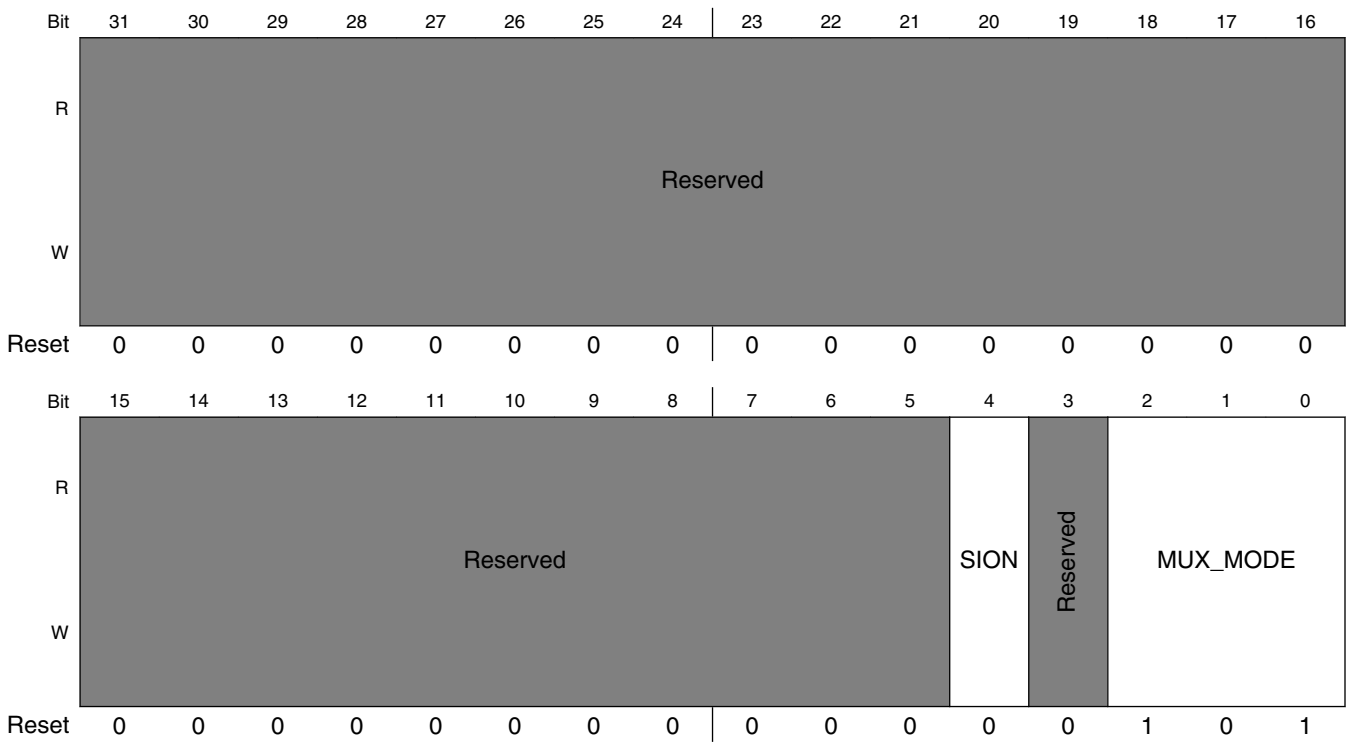


IOMUXC_SW_MUX_CTL_PAD_ECSPI1_MISO field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ECSPI1_MISO 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: ECSPI1_MISO. 000 **ALT0** — Select signal ECSPI1_MISO 001 **ALT1** — Select signal UART3_CTS_B 101 **ALT5** — Select signal GPIO5_IO08

8.2.5.124 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI1_SS0)

Address: 3033_0000h base + 200h offset = 3033_0200h

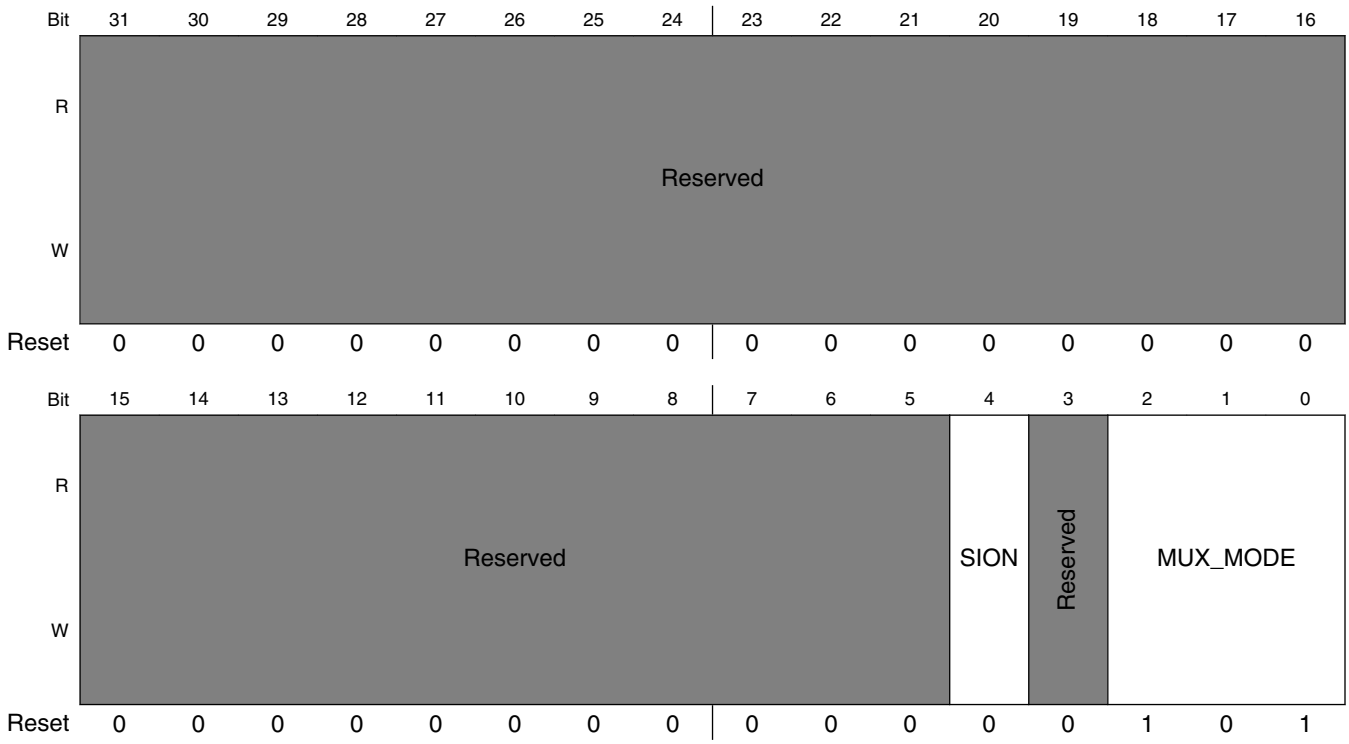


IOMUXC_SW_MUX_CTL_PAD_ECSPI1_SS0 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ECSPI1_SS0 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: ECSPI1_SS0. **NOTE:** Pad ECSPI1_SS0 is involved in Daisy Chain. 000 **ALT0** — Select signal ECSPI1_SS0 001 **ALT1** — Select signal UART3_RTS_B - Configure register IOMUXC_UART3_RTS_B_SELECT_INPUT for mode ALT1. 101 **ALT5** — Select signal GPIO5_IO09

8.2.5.125 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI2_SCLK)

Address: 3033_0000h base + 204h offset = 3033_0204h

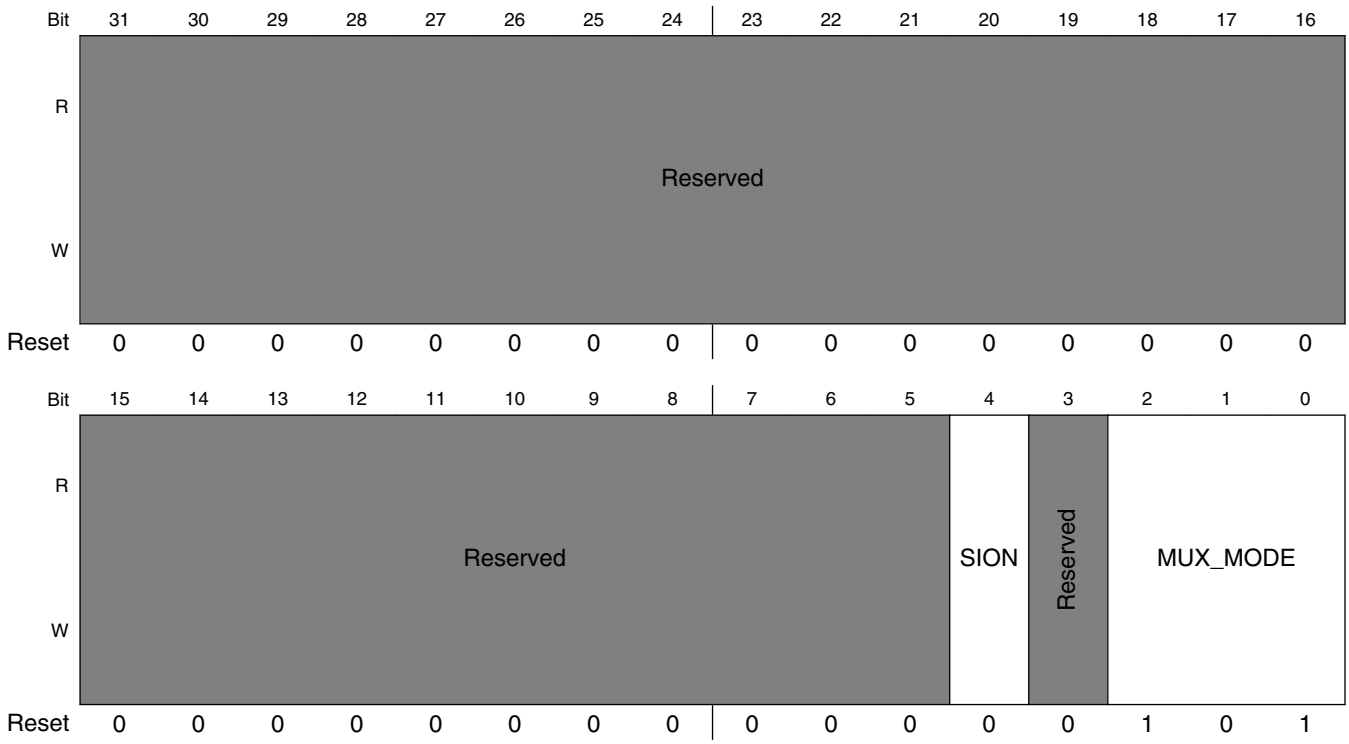


IOMUXC_SW_MUX_CTL_PAD_ECSPI2_SCLK field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ECSPI2_SCLK 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: ECSPI2_SCLK. 000 **ALT0** — Select signal ECSPI2_SCLK 001 **ALT1** — Select signal UART4_RX 101 **ALT5** — Select signal GPIO5_IO10

8.2.5.126 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI2_MOSI)

Address: 3033_0000h base + 208h offset = 3033_0208h

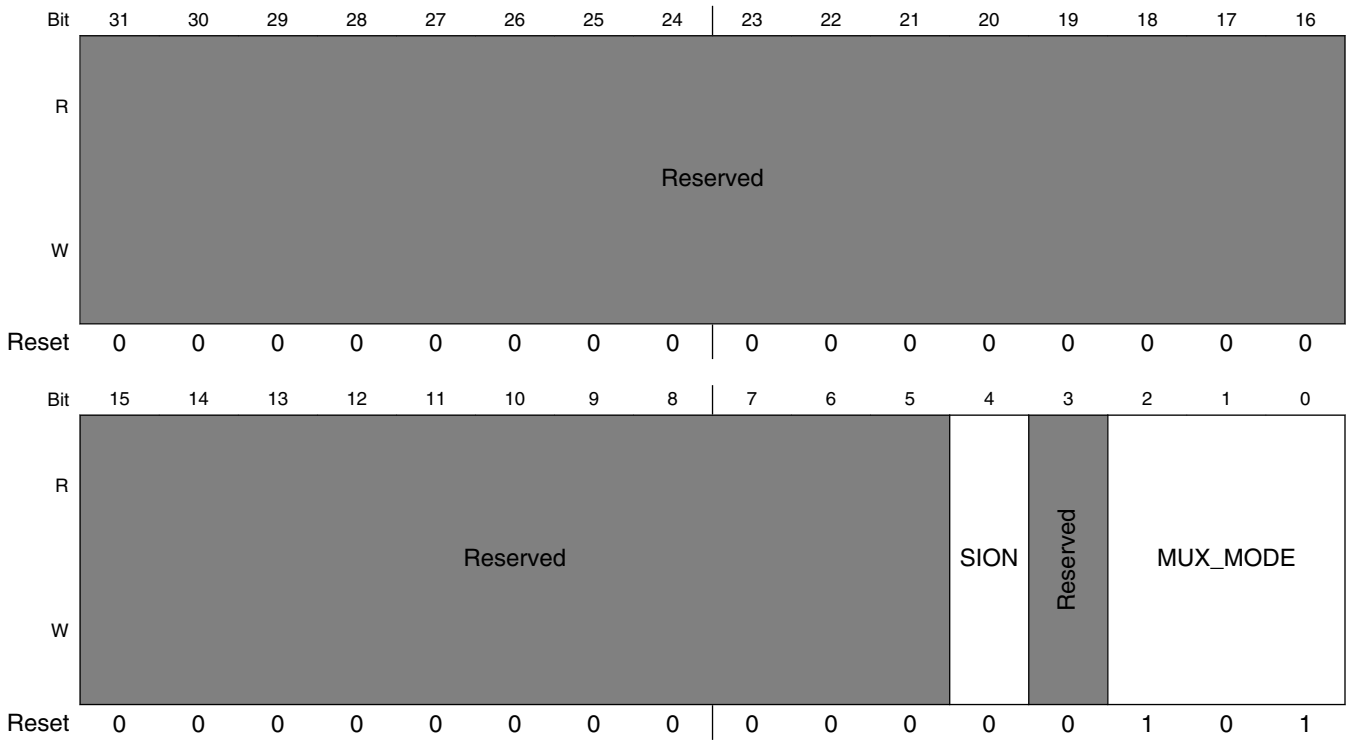


IOMUXC_SW_MUX_CTL_PAD_ECSPI2_MOSI field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ECSPI2_MOSI 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: ECSPI2_MOSI. 000 **ALT0** — Select signal ECSPI2_MOSI 001 **ALT1** — Select signal UART4_TX 101 **ALT5** — Select signal GPIO5_IO11

8.2.5.127 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI2_MISO)

Address: 3033_0000h base + 20Ch offset = 3033_020Ch

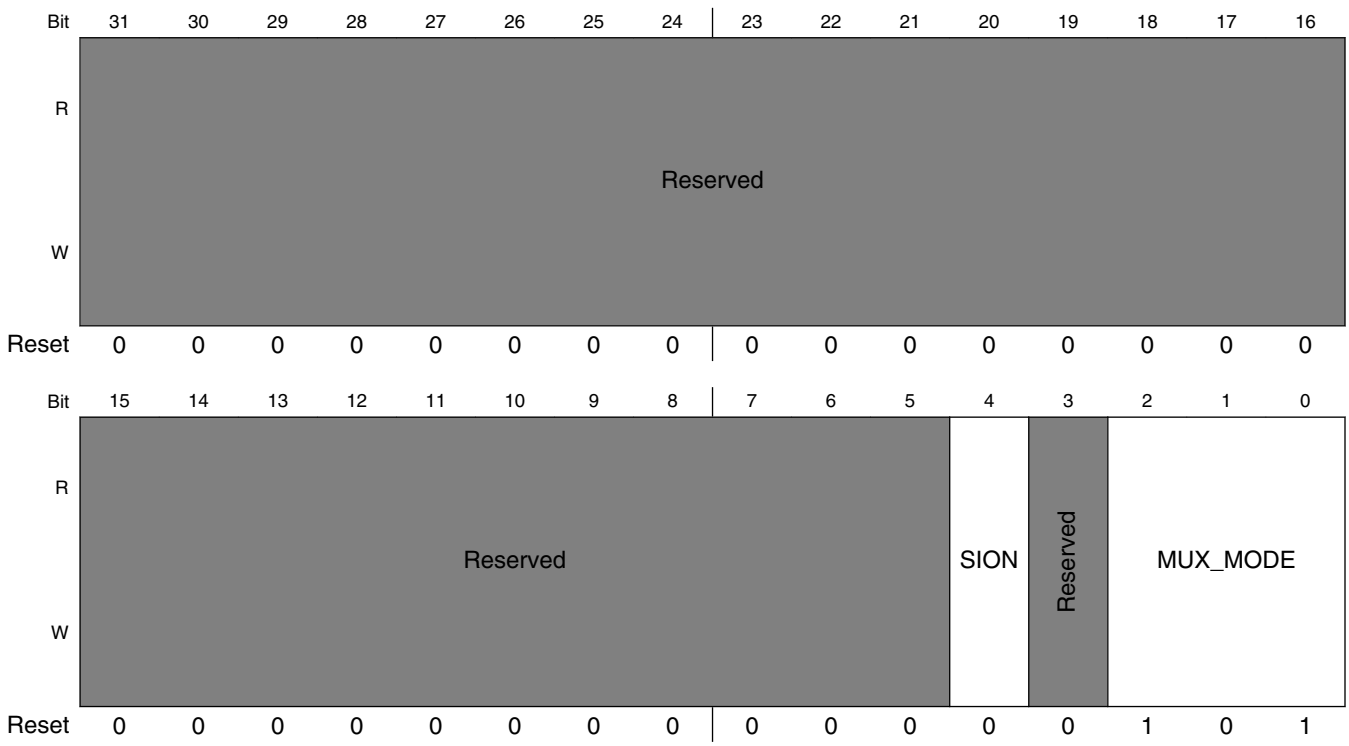


IOMUXC_SW_MUX_CTL_PAD_ECSPI2_MISO field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ECSPI2_MISO 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: ECSPI2_MISO. 000 **ALT0** — Select signal ECSPI2_MISO 001 **ALT1** — Select signal UART4_CTS_B 101 **ALT5** — Select signal GPIO5_IO12

8.2.5.128 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPI2_SS0)

Address: 3033_0000h base + 210h offset = 3033_0210h

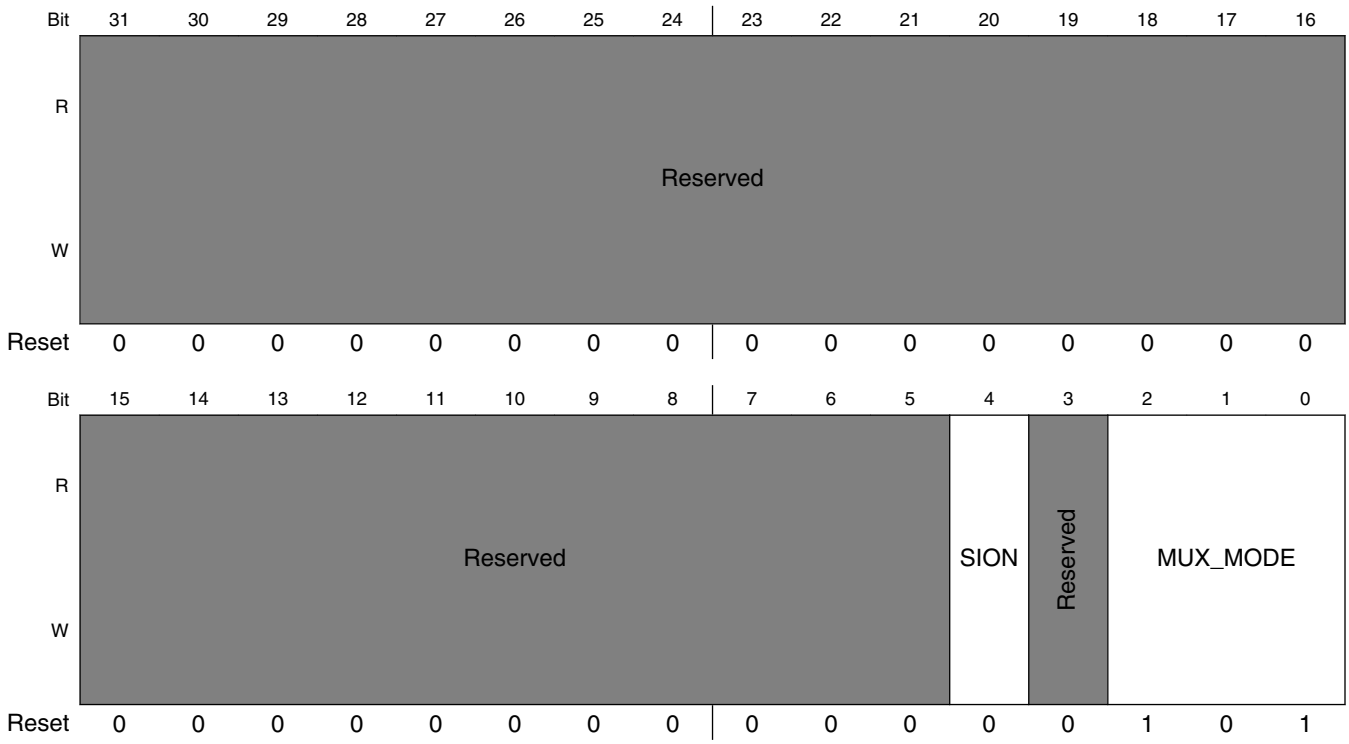


IOMUXC_SW_MUX_CTL_PAD_ECSPI2_SS0 field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad ECSPI2_SS0 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: ECSPI2_SS0. **NOTE:** Pad ECSPI2_SS0 is involved in Daisy Chain. 000 **ALT0** — Select signal ECSPI2_SS0 001 **ALT1** — Select signal UART4_RTS_B - Configure register IOMUXC_UART4_RTS_B_SELECT_INPUT for mode ALT1. 101 **ALT5** — Select signal GPIO5_IO13

8.2.5.129 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C1_SCL)

Address: 3033_0000h base + 214h offset = 3033_0214h

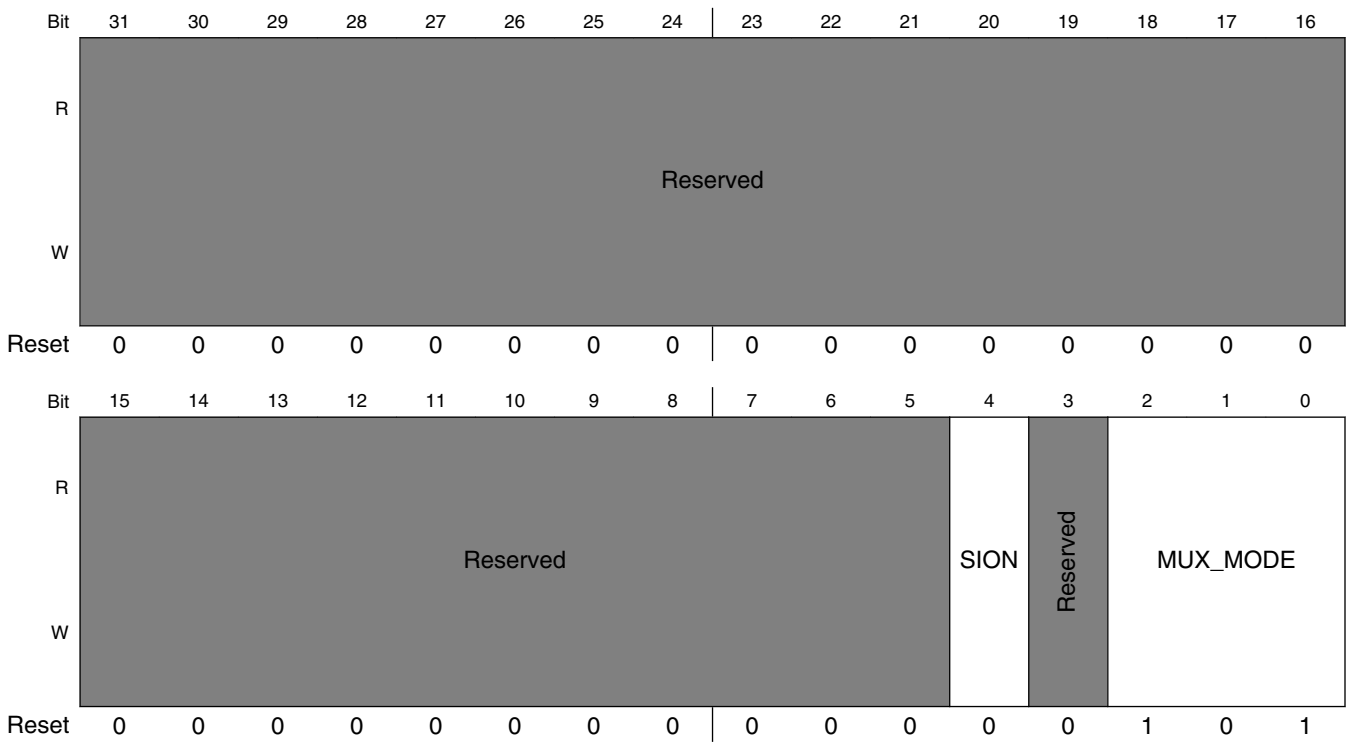


IOMUXC_SW_MUX_CTL_PAD_I2C1_SCL field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad I2C1_SCL 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: I2C1_SCL. 000 **ALT0** — Select signal I2C1_SCL 001 **ALT1** — Select signal ENET1_MDC 101 **ALT5** — Select signal GPIO5_IO14

8.2.5.130 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C1_SDA)

Address: 3033_0000h base + 218h offset = 3033_0218h

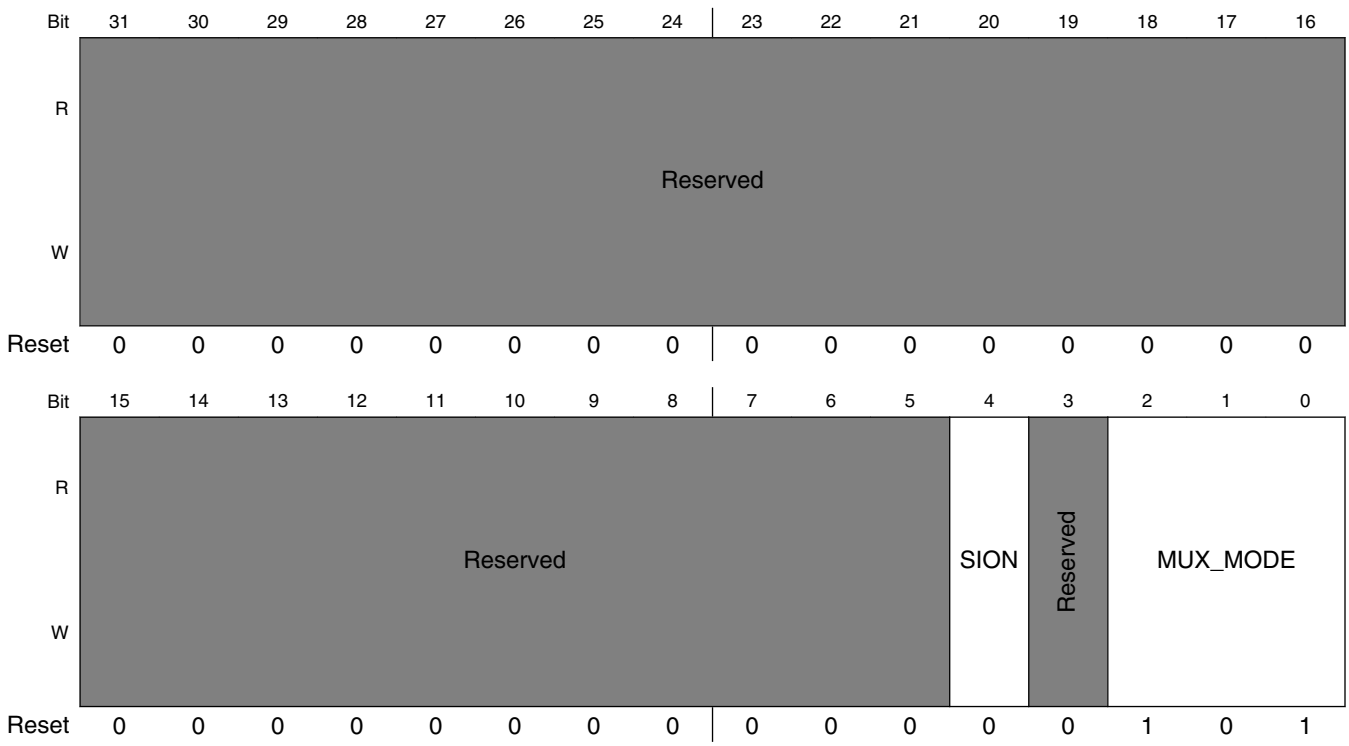


IOMUXC_SW_MUX_CTL_PAD_I2C1_SDA field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad I2C1_SDA 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: I2C1_SDA. **NOTE:** Pad I2C1_SDA is involved in Daisy Chain. 000 **ALT0** — Select signal I2C1_SDA 001 **ALT1** — Select signal ENET1_MDIO - Configure register IOMUXC_ENET1_MDIO_SELECT_INPUT for mode ALT1. 101 **ALT5** — Select signal GPIO5_IO15

8.2.5.131 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C2_SCL)

Address: 3033_0000h base + 21Ch offset = 3033_021Ch

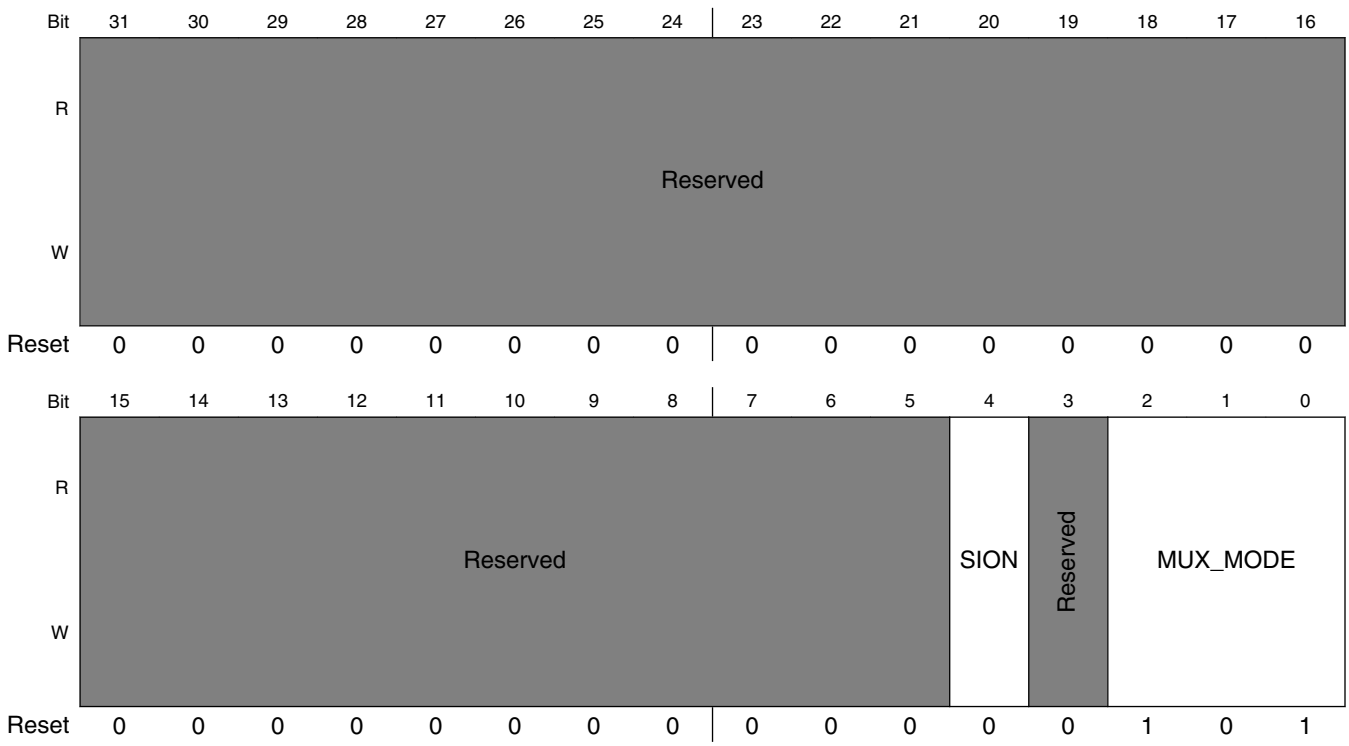


IOMUXC_SW_MUX_CTL_PAD_I2C2_SCL field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad I2C2_SCL 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: I2C2_SCL. **NOTE:** Pad I2C2_SCL is involved in Daisy Chain. 000 **ALT0** — Select signal I2C2_SCL 001 **ALT1** — Select signal ENET1_1588_EVENT1_IN 010 **ALT2** — Select signal USDHC3_CD_B - Configure register IOMUXC_USDHC3_CD_B_SELECT_INPUT for mode ALT2. 101 **ALT5** — Select signal GPIO5_IO16

8.2.5.132 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C2_SDA)

Address: 3033_0000h base + 220h offset = 3033_0220h

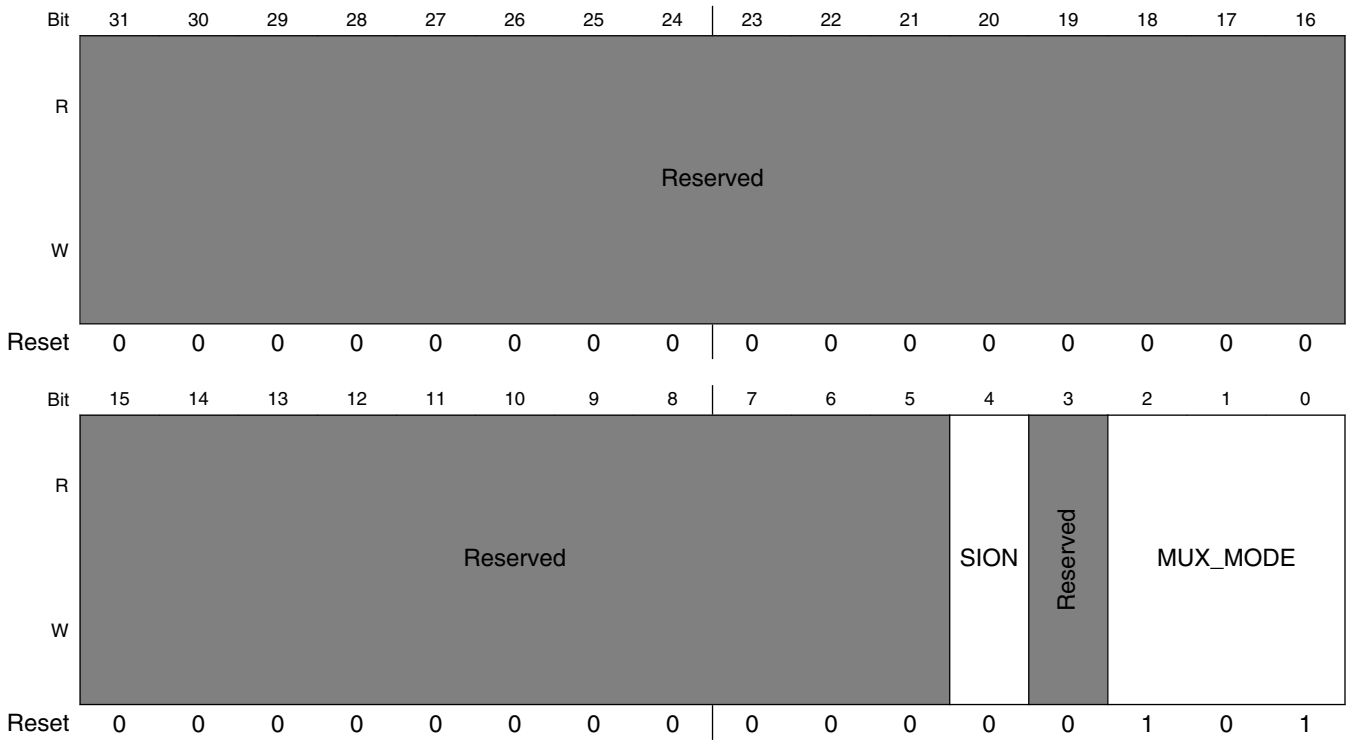


IOMUXC_SW_MUX_CTL_PAD_I2C2_SDA field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad I2C2_SDA 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: I2C2_SDA. **NOTE:** Pad I2C2_SDA is involved in Daisy Chain. 000 **ALT0** — Select signal I2C2_SDA 001 **ALT1** — Select signal ENET1_1588_EVENT1_OUT 010 **ALT2** — Select signal USDHC3_WP - Configure register IOMUXC_USDHC3_WP_SELECT_INPUT for mode ALT2. 101 **ALT5** — Select signal GPIO5_IO17

8.2.5.133 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C3_SCL)

Address: 3033_0000h base + 224h offset = 3033_0224h

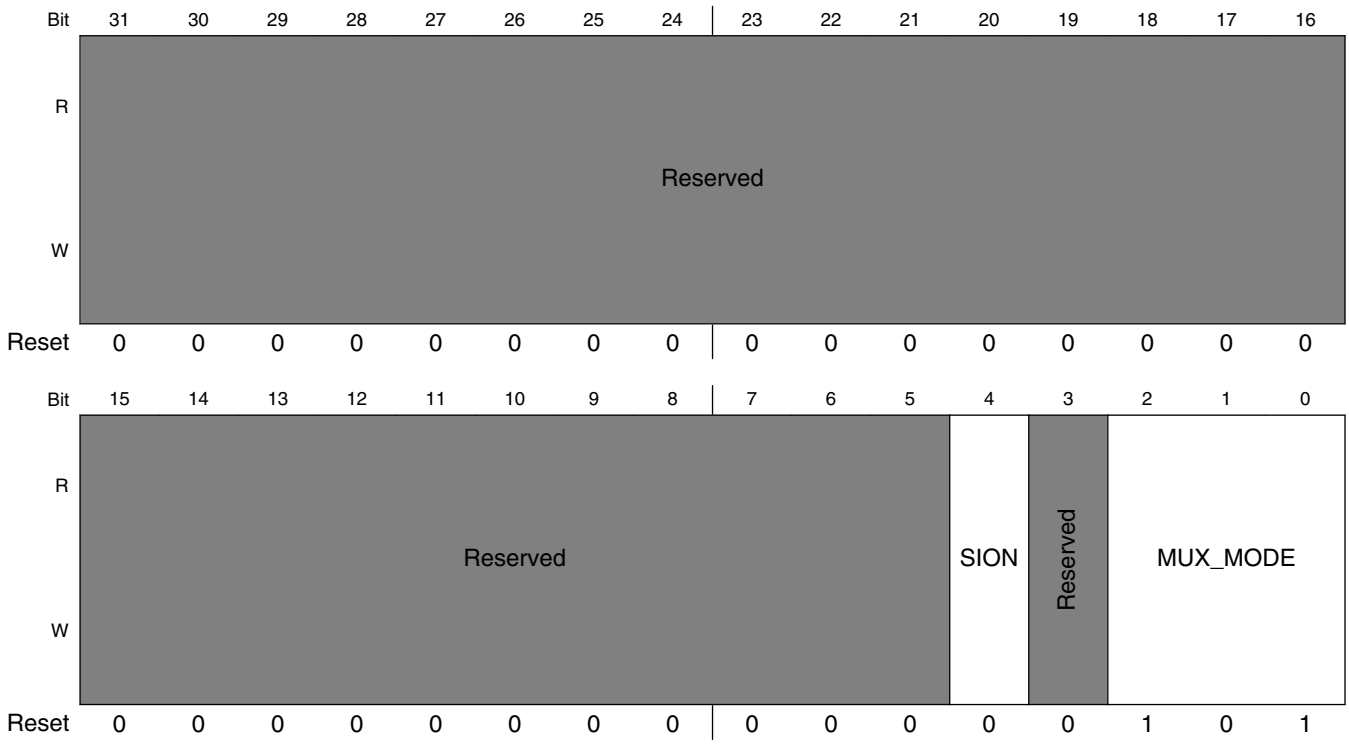


IOMUXC_SW_MUX_CTL_PAD_I2C3_SCL field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad I2C3_SCL 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: I2C3_SCL. 000 **ALT0** — Select signal I2C3_SCL 001 **ALT1** — Select signal PWM4_OUT 010 **ALT2** — Select signal GPT2_CLK 101 **ALT5** — Select signal GPIO5_IO18

8.2.5.134 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C3_SDA)

Address: 3033_0000h base + 228h offset = 3033_0228h

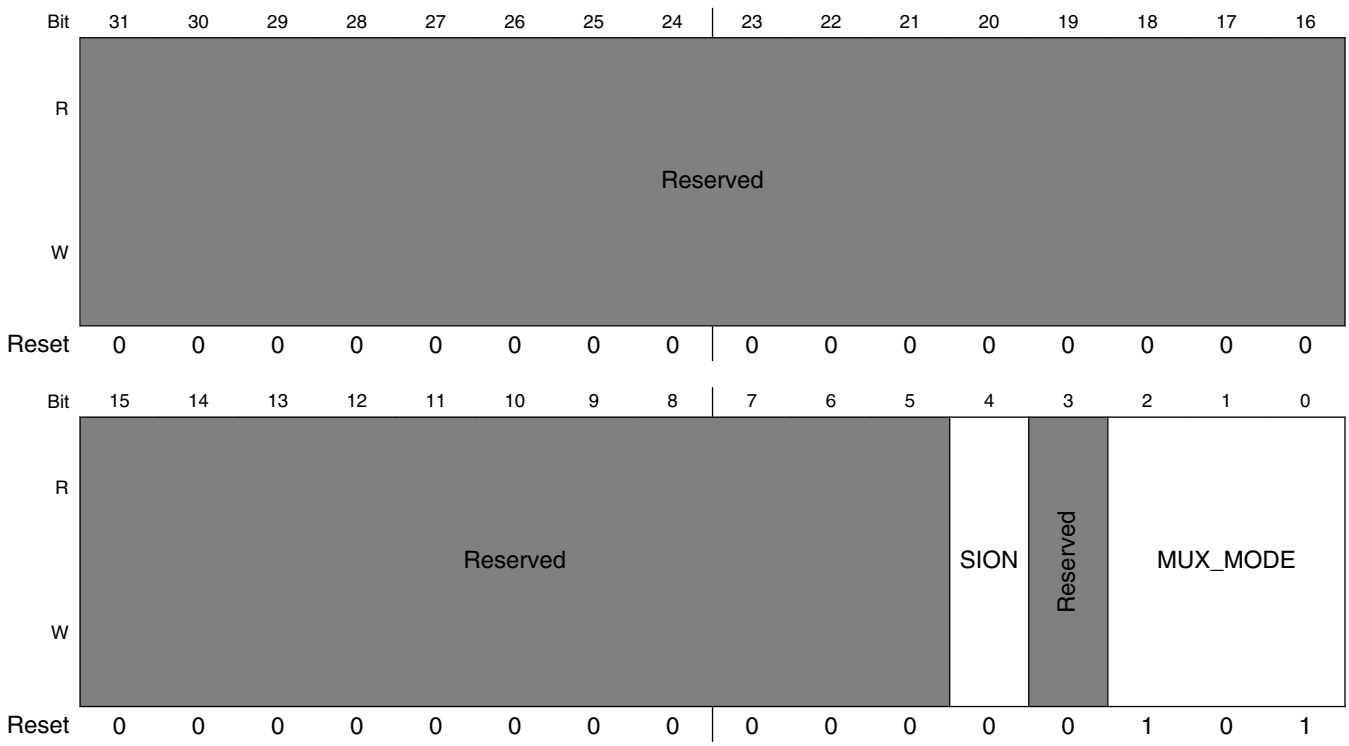


IOMUXC_SW_MUX_CTL_PAD_I2C3_SDA field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad I2C3_SDA 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: I2C3_SDA. 000 **ALT0** — Select signal I2C3_SDA 001 **ALT1** — Select signal PWM3_OUT 010 **ALT2** — Select signal GPT3_CLK 101 **ALT5** — Select signal GPIO5_IO19

8.2.5.135 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C4_SCL)

Address: 3033_0000h base + 22Ch offset = 3033_022Ch

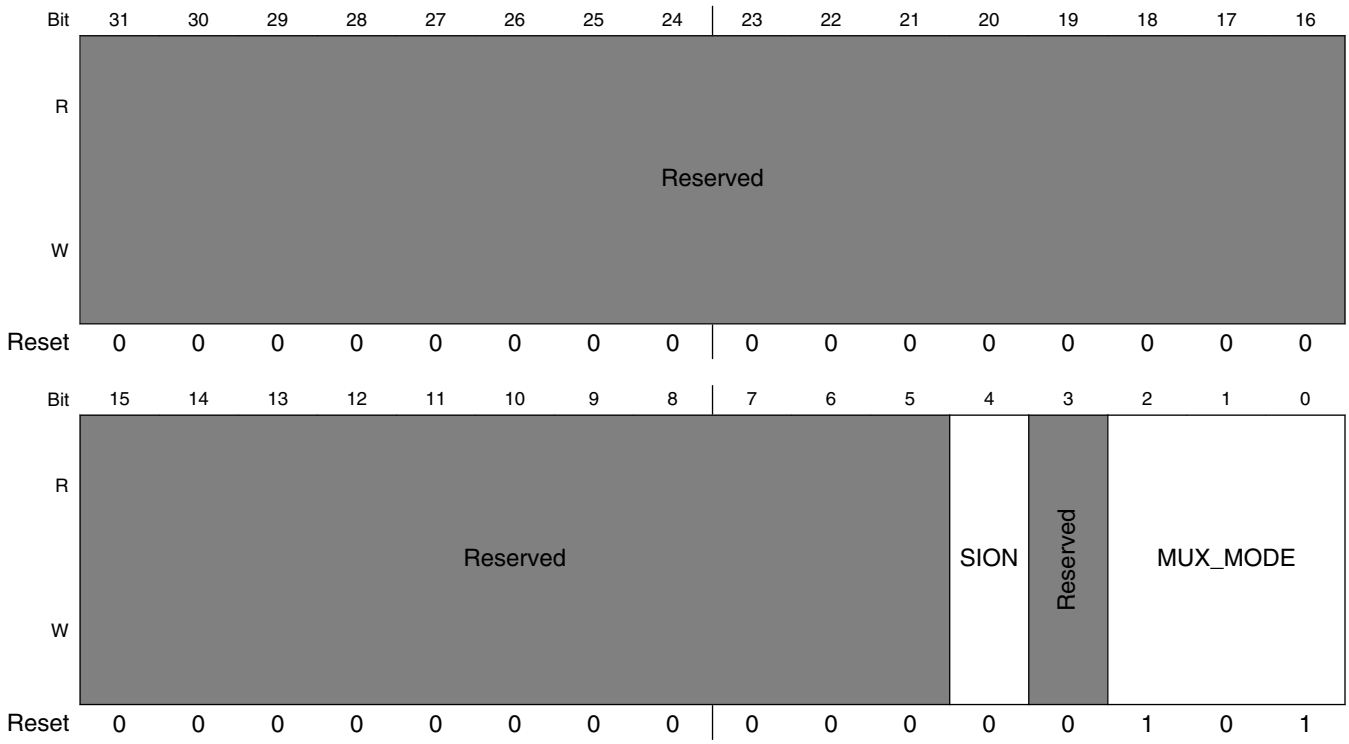


IOMUXC_SW_MUX_CTL_PAD_I2C4_SCL field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad I2C4_SCL 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: I2C4_SCL. **NOTE:** Pad I2C4_SCL is involved in Daisy Chain. 000 **ALT0** — Select signal I2C4_SCL 001 **ALT1** — Select signal PWM2_OUT 010 **ALT2** — Select signal PCIE1_CLKREQ_B - Configure register IOMUXC_PCIE1_CLKREQ_B_SELECT_INPUT for mode ALT2. 101 **ALT5** — Select signal GPIO5_IO20

8.2.5.136 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C4_SDA)

Address: 3033_0000h base + 230h offset = 3033_0230h

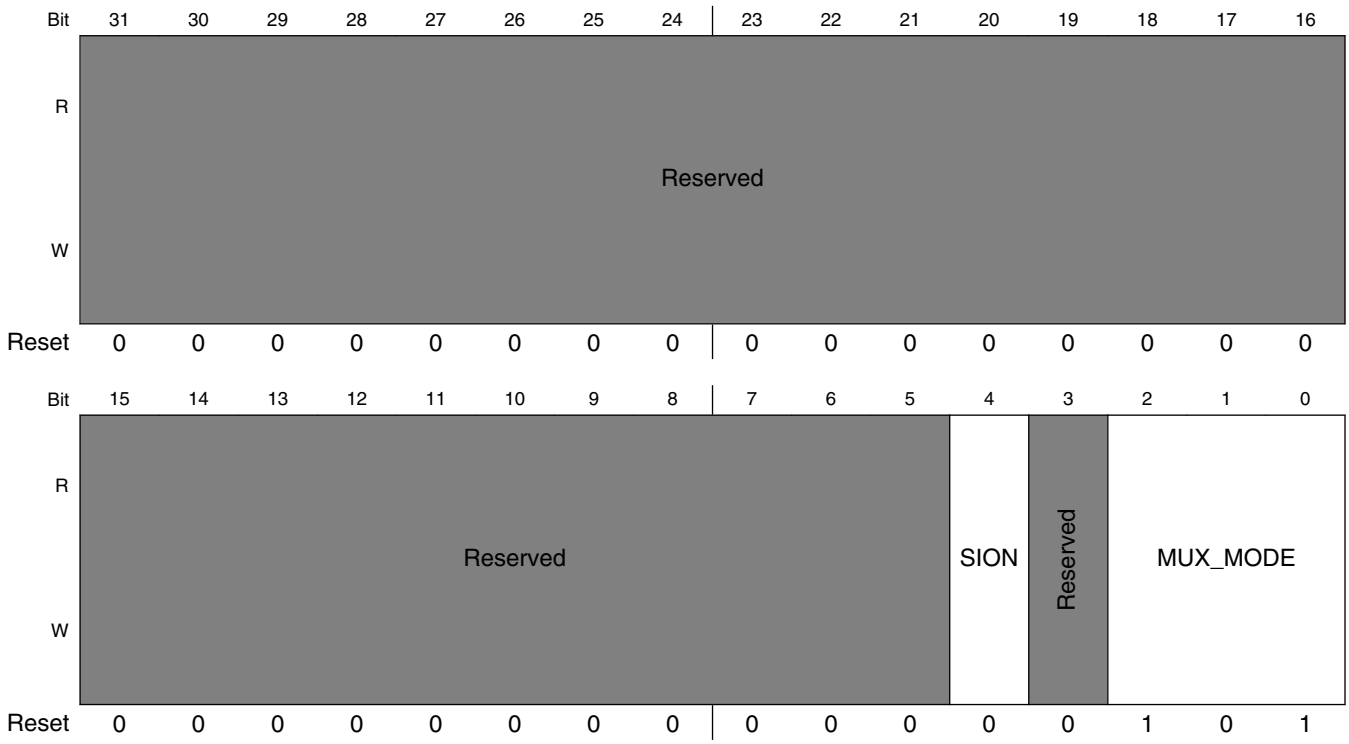


IOMUXC_SW_MUX_CTL_PAD_I2C4_SDA field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad I2C4_SDA 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: I2C4_SDA. 000 **ALT0** — Select signal I2C4_SDA 001 **ALT1** — Select signal PWM1_OUT 101 **ALT5** — Select signal GPIO5_IO21

8.2.5.137 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART1_RXD)

Address: 3033_0000h base + 234h offset = 3033_0234h

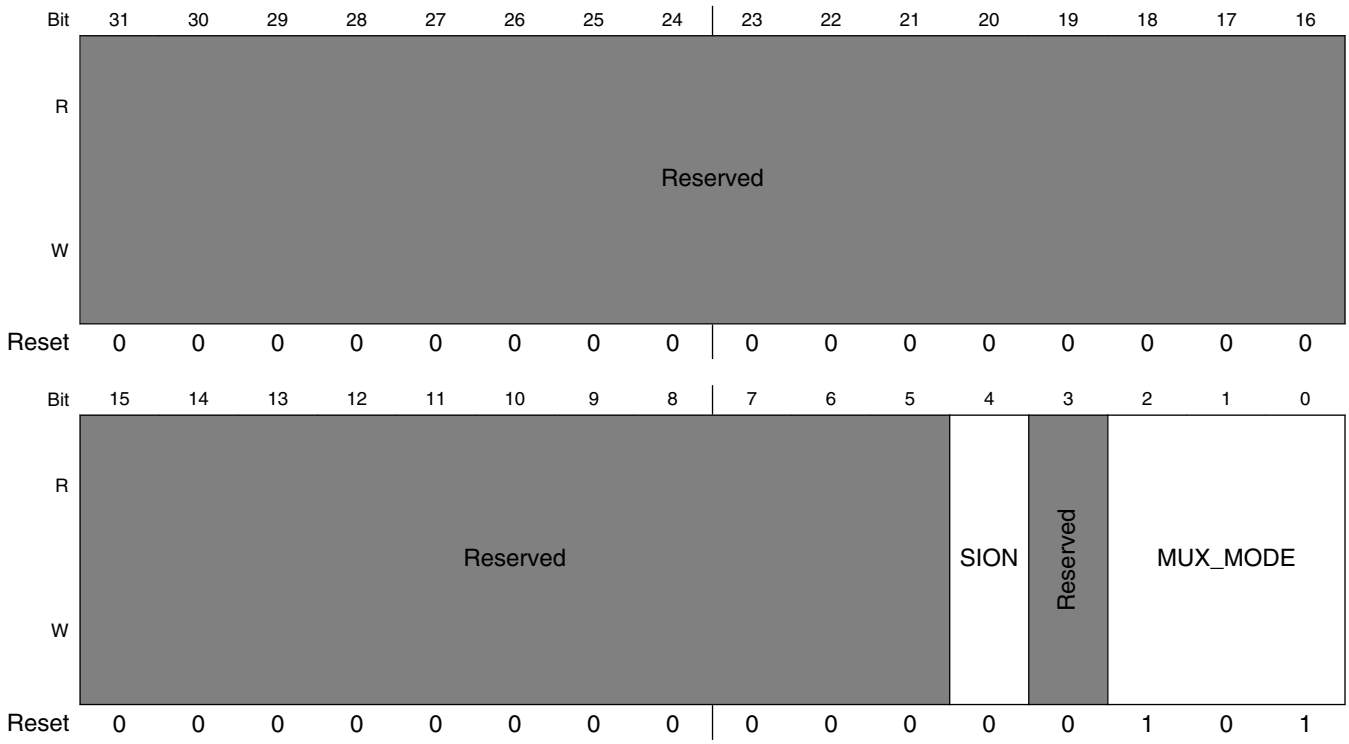


IOMUXC_SW_MUX_CTL_PAD_UART1_RXD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad UART1_RXD 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: UART1_RXD. 000 **ALT0** — Select signal UART1_RX 001 **ALT1** — Select signal ECSPI3_SCLK 101 **ALT5** — Select signal GPIO5_IO22

8.2.5.138 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART1_TXD)

Address: 3033_0000h base + 238h offset = 3033_0238h

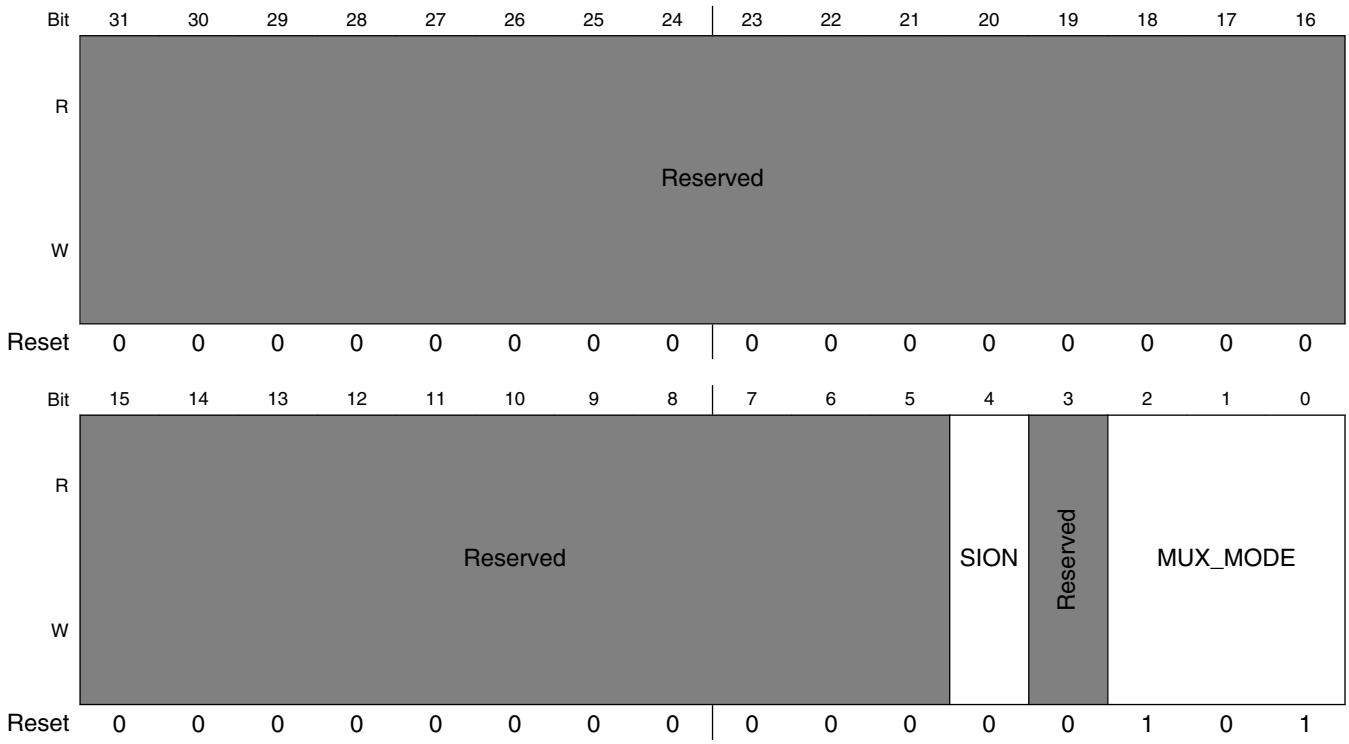


IOMUXC_SW_MUX_CTL_PAD_UART1_TXD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad UART1_TXD 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: UART1_TXD. 000 **ALT0** — Select signal UART1_TX 001 **ALT1** — Select signal ECSPI3_MOSI 101 **ALT5** — Select signal GPIO5_IO23

8.2.5.139 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART2_RXD)

Address: 3033_0000h base + 23Ch offset = 3033_023Ch

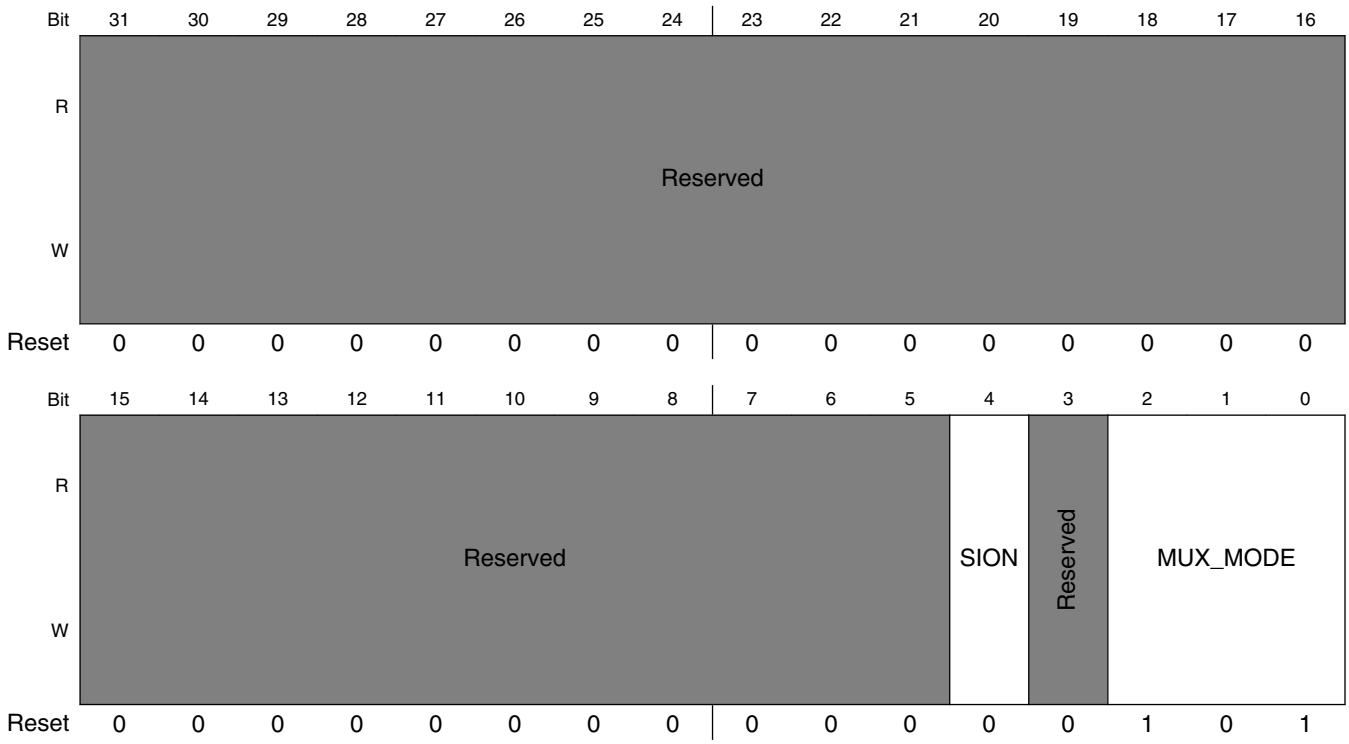


IOMUXC_SW_MUX_CTL_PAD_UART2_RXD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad UART2_RXD 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: UART2_RXD. 000 **ALT0** — Select signal UART2_RX 001 **ALT1** — Select signal ECSPI3_MISO 101 **ALT5** — Select signal GPIO5_IO24

8.2.5.140 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART2_TXD)

Address: 3033_0000h base + 240h offset = 3033_0240h

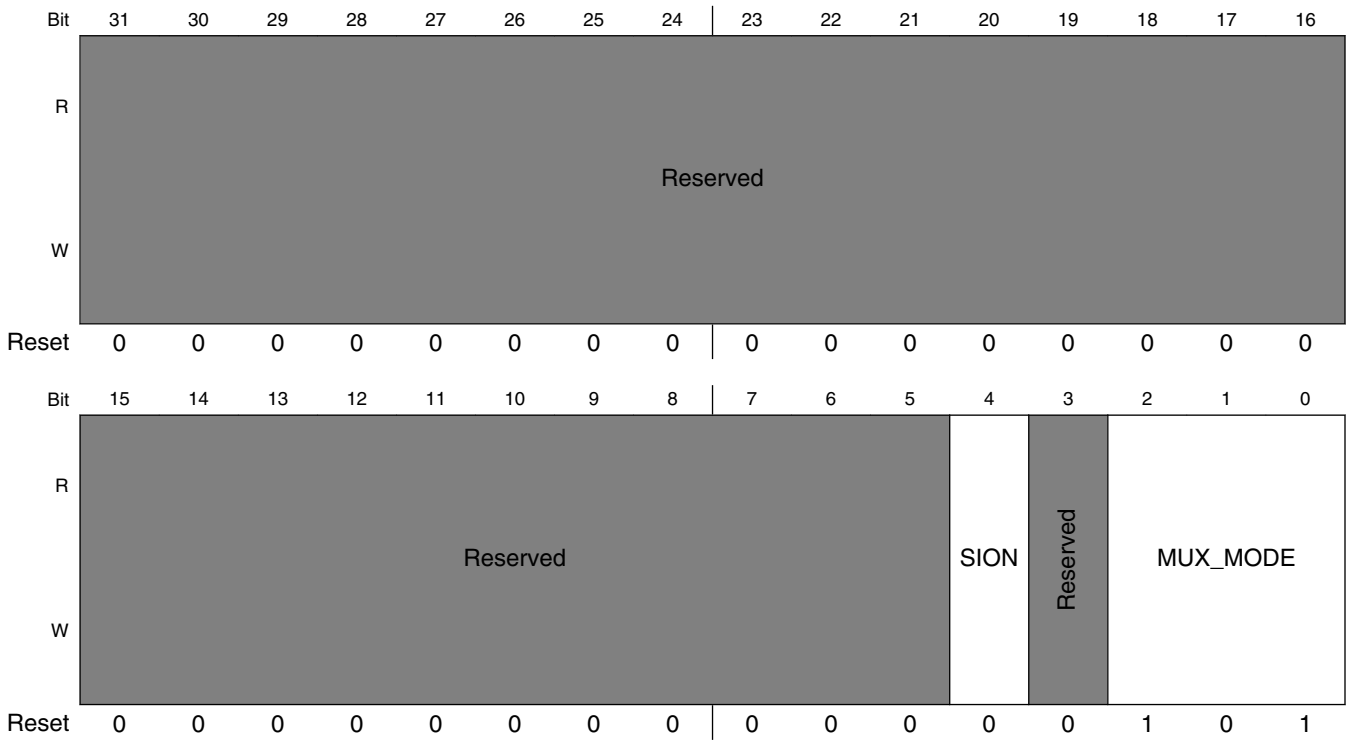


IOMUXC_SW_MUX_CTL_PAD_UART2_TXD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad UART2_TXD 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: UART2_TXD. 000 **ALT0** — Select signal UART2_TX 001 **ALT1** — Select signal ECSPI3_SS0 101 **ALT5** — Select signal GPIO5_IO25

8.2.5.141 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART3_RXD)

Address: 3033_0000h base + 244h offset = 3033_0244h

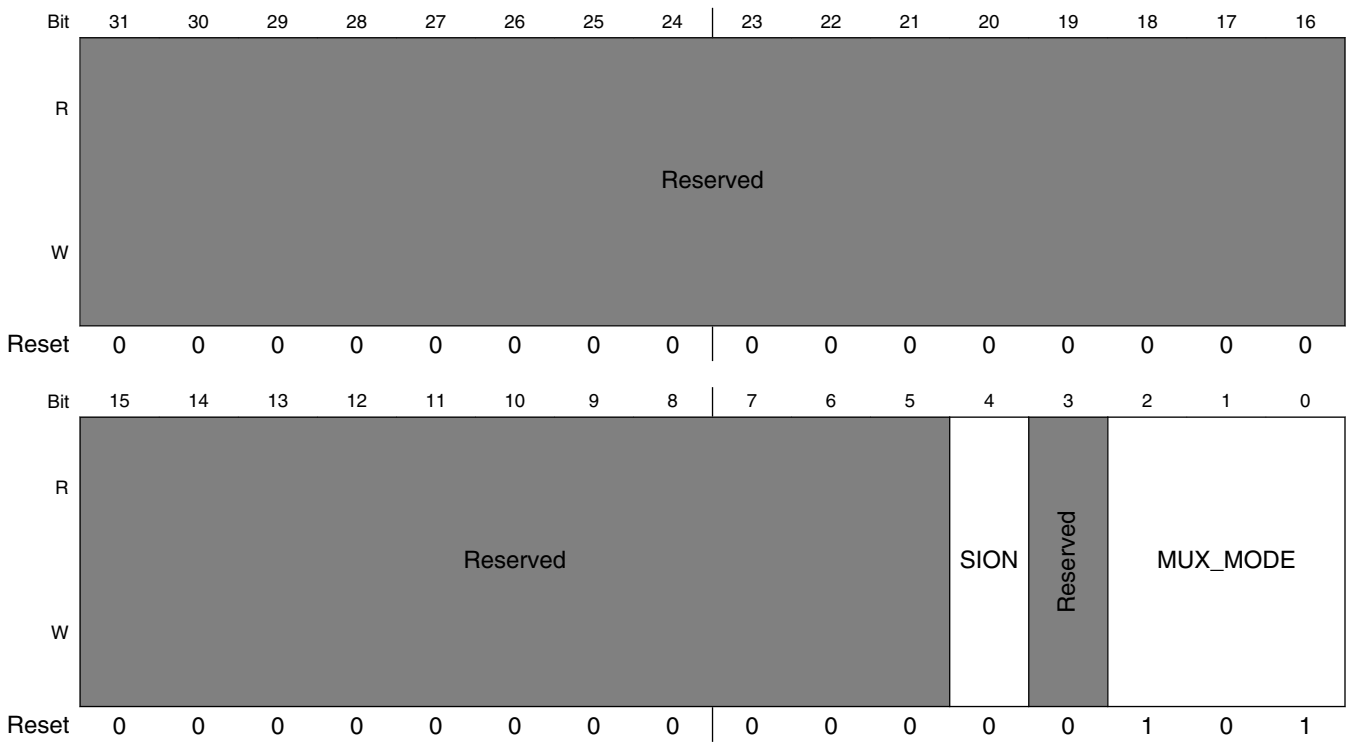


IOMUXC_SW_MUX_CTL_PAD_UART3_RXD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad UART3_RXD 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: UART3_RXD. 000 **ALT0** — Select signal UART3_RX 001 **ALT1** — Select signal UART1_CTS_B 010 **ALT2** — Select signal USDHC3_RESET_B 101 **ALT5** — Select signal GPIO5_IO26

8.2.5.142 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART3_TXD)

Address: 3033_0000h base + 248h offset = 3033_0248h

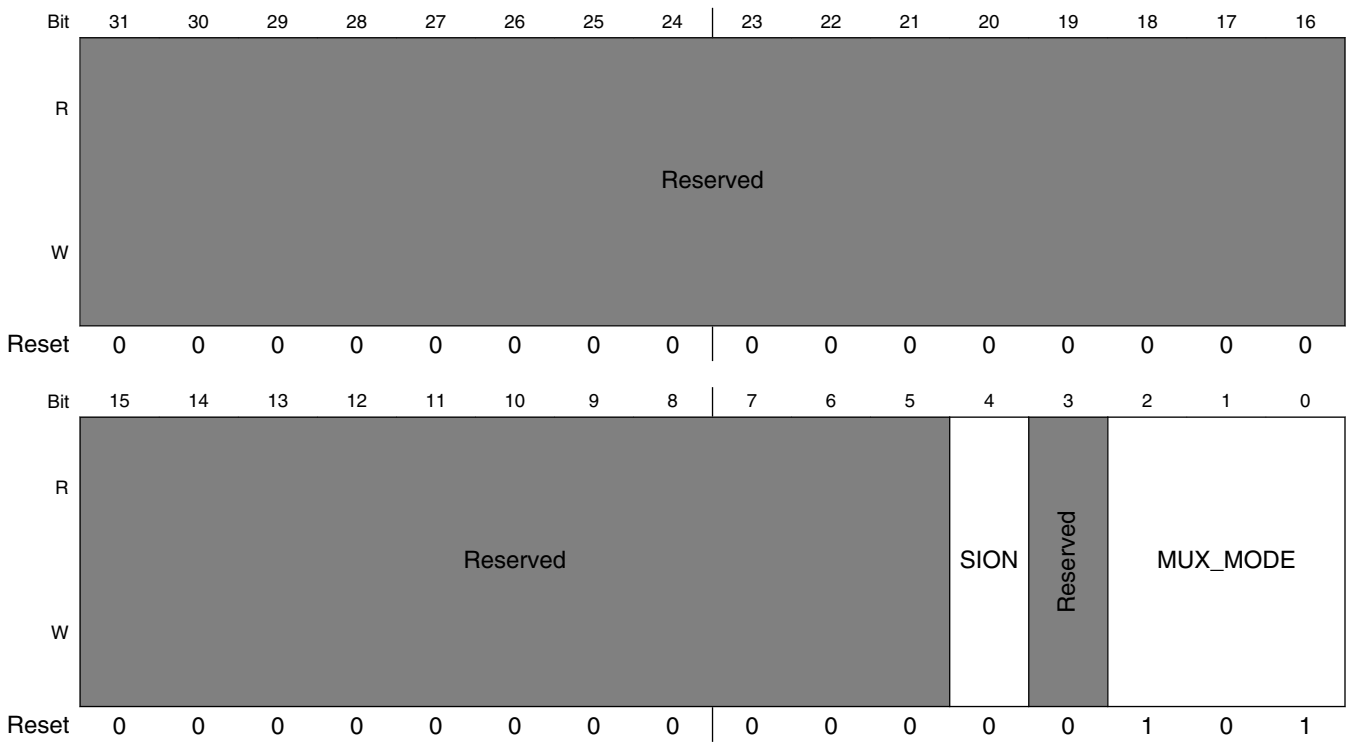


IOMUXC_SW_MUX_CTL_PAD_UART3_TXD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad UART3_TXD 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: UART3_TXD. **NOTE:** Pad UART3_TXD is involved in Daisy Chain. 000 **ALT0** — Select signal UART3_TX 001 **ALT1** — Select signal UART1_RTS_B - Configure register IOMUXC_UART1_RTS_B_SELECT_INPUT for mode ALT1. 010 **ALT2** — Select signal USDHC3_VSELECT 101 **ALT5** — Select signal GPIO5_IO27

8.2.5.143 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART4_RXD)

Address: 3033_0000h base + 24Ch offset = 3033_024Ch

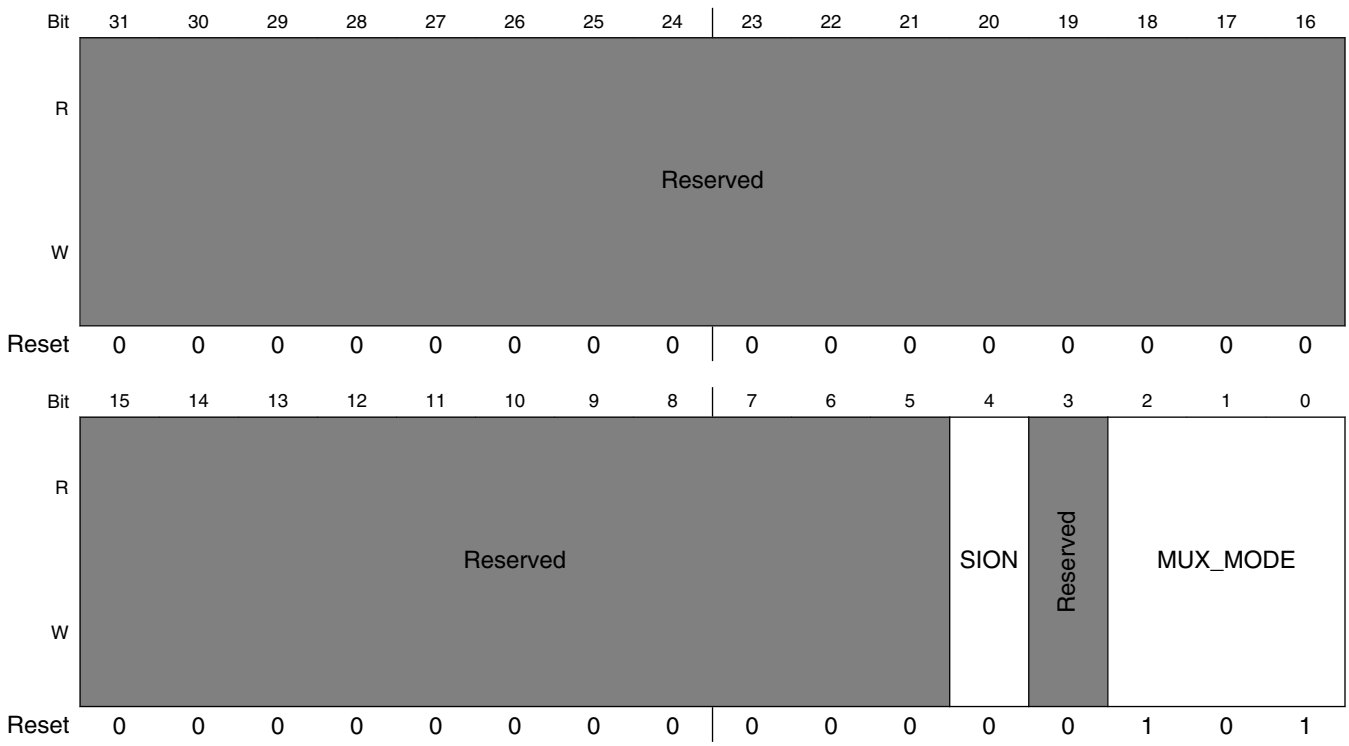


IOMUXC_SW_MUX_CTL_PAD_UART4_RXD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad UART4_RXD 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for pad: UART4_RXD. **NOTE:** Pad UART4_RXD is involved in Daisy Chain. 000 **ALT0** — Select signal UART4_RX 001 **ALT1** — Select signal UART2_CTS_B 010 **ALT2** — Select signal PCIE1_CLKREQ_B - Configure register IOMUXC_PCIE1_CLKREQ_B_SELECT_INPUT for mode ALT2. 101 **ALT5** — Select signal GPIO5_IO28

8.2.5.144 Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART4_TXD)

Address: 3033_0000h base + 250h offset = 3033_0250h

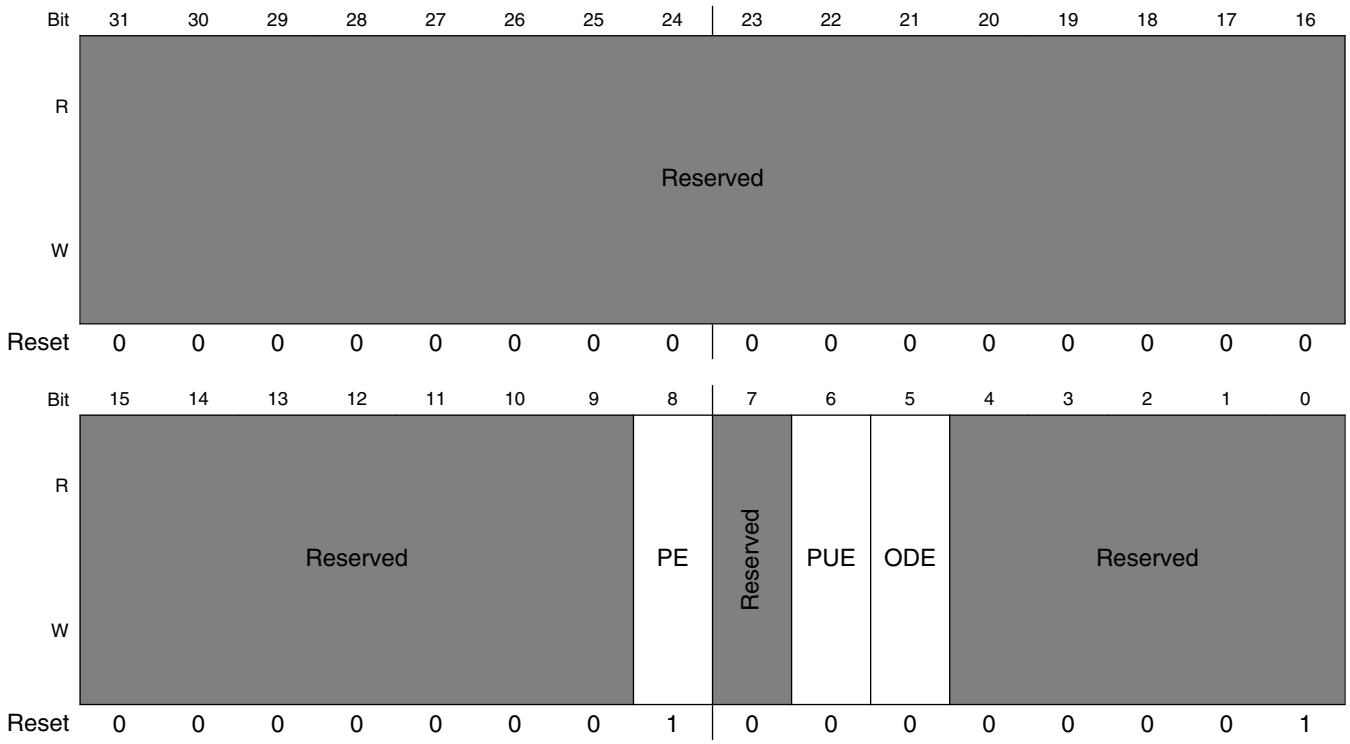


IOMUXC_SW_MUX_CTL_PAD_UART4_TXD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad UART4_TXD 0 **DISABLED** — Input Path is determined by functionality of the selected mux mode (regular).
3 -	This field is reserved. Reserved
MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: UART4_TXD. **NOTE:** Pad UART4_TXD is involved in Daisy Chain. 000 **ALT0** — Select signal UART4_TX 001 **ALT1** — Select signal UART2_RTS_B - Configure register IOMUXC_UART2_RTS_B_SELECT_INPUT for mode ALT1. 101 **ALT5** — Select signal GPIO5_IO29

8.2.5.145 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_TEST_MODE)

Address: 3033_0000h base + 254h offset = 3033_0254h



IOMUXC_SW_PAD_CTL_PAD_TEST_MODE field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 -	This field is reserved. Reserved
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Not used for this IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.146 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_BOOT_MODE0)

Address: 3033_0000h base + 258h offset = 3033_0258h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	Reserved	PUE	ODE	Reserved				
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1

IOMUXC_SW_PAD_CTL_PAD_BOOT_MODE0 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 -	This field is reserved. Reserved
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Not used for this IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.147 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_BOOT_MODE1)

Address: 3033_0000h base + 25Ch offset = 3033_025Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	Reserved	PUE	ODE	Reserved				
Reset	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1

IOMUXC_SW_PAD_CTL_PAD_BOOT_MODE1 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 -	This field is reserved. Reserved
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Not used for this IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.148 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_MOD)

Address: 3033_0000h base + 260h offset = 3033_0260h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	Reserved				
Reset	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	1

IOMUXC_SW_PAD_CTL_PAD_JTAG_MOD field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Not used for this IO control, can be used by SW for general purpose
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Not used for this IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.149 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TRST_B)

Address: 3033_0000h base + 264h offset = 3033_0264h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	Reserved				
Reset	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1

IOMUXC_SW_PAD_CTL_PAD_JTAG_TRST_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Not used for this IO control, can be used by SW for general purpose
7 HYS	Hysteresis Enable Field Not used for this IO control, can be used by SW for general purpose
6 PUE	Not used for this IO control, can be used by SW for general purpose
5 ODE	Open Drain Enable Field Not used for this IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.150 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TDI)

Address: 3033_0000h base + 268h offset = 3033_0268h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	Reserved				
Reset	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1

IOMUXC_SW_PAD_CTL_PAD_JTAG_TDI field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Not used for this IO control, can be used by SW for general purpose
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Not used for this IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.151 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TMS)

Address: 3033_0000h base + 26Ch offset = 3033_026Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	Reserved				
Reset	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1

IOMUXC_SW_PAD_CTL_PAD_JTAG_TMS field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_JTAG_TMS field descriptions (continued)

Field	Description
7 HYS	Hysteresis Enable Field Not used for this IO control, can be used by SW for general purpose
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Not used for this IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.152 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TCK)

Address: 3033_0000h base + 270h offset = 3033_0270h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	Reserved				
Reset	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1

IOMUXC_SW_PAD_CTL_PAD_JTAG_TCK field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Not used for this IO control, can be used by SW for general purpose
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_JTAG_TCK field descriptions (continued)

Field	Description
	Not used for this IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.153 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TDO)

Address: 3033_0000h base + 274h offset = 3033_0274h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	Reserved				
Reset	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1

IOMUXC_SW_PAD_CTL_PAD_JTAG_TDO field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Not used for this IO control, can be used by SW for general purpose
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Not used for this IO control, can be used by SW for general purpose
-	This field is reserved. Reserved

8.2.5.154 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_RTC)

Address: 3033_0000h base + 278h offset = 3033_0278h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SW_PAD_CTL_PAD_RTC field descriptions

Field	Description
- Reserved	This field is reserved.

8.2.5.155 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_PMIC_STBY_REQ)

Address: 3033_0000h base + 27Ch offset = 3033_027Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0

IOMUXC_SW_PAD_CTL_PAD_PMIC_STBY_REQ field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Not used for this IO control, can be used by SW for general purpose
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_PMIC_STBY_REQ field descriptions (continued)

Field	Description
	Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Not used for this IO control, can be used by SW for general purpose

8.2.5.156 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_PMIC_ON_REQ)

Address: 3033_0000h base + 280h offset = 3033_0280h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	1	1	0	1	1	0	0

IOMUXC_SW_PAD_CTL_PAD_PMIC_ON_REQ field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Not used for this IO control, can be used by SW for general purpose

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_PMIC_ON_REQ field descriptions (continued)

Field	Description
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Not used for this IO control, can be used by SW for general purpose

8.2.5.157 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ONOFF)

Address: 3033_0000h base + 284h offset = 3033_0284h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	1	1	0	0	1	1	0	0

IOMUXC_SW_PAD_CTL_PAD_ONOFF field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Not used for this IO control, can be used by SW for general purpose
7 HYS	Hysteresis Enable Field Not used for this IO control, can be used by SW for general purpose
6 PUE	Not used for this IO control, can be used by SW for general purpose

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ONOFF field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Not used for this IO control, can be used by SW for general purpose
4–3 FSEL	Slew Rate Field Not used for this IO control, can be used by SW for general purpose
DSE	Drive Strength Field Not used for this IO control, can be used by SW for general purpose

8.2.5.158 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_POR_B)

Address: 3033_0000h base + 288h offset = 3033_0288h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	1	1	0	0	1	1	0	0

IOMUXC_SW_PAD_CTL_PAD_POR_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Not used for this IO control, can be used by SW for general purpose
7 HYS	Hysteresis Enable Field Not used for this IO control, can be used by SW for general purpose
6 PUE	Not used for this IO control, can be used by SW for general purpose
5 ODE	Open Drain Enable Field Not used for this IO control, can be used by SW for general purpose
4–3 FSEL	Slew Rate Field Not used for this IO control, can be used by SW for general purpose
DSE	Drive Strength Field Not used for this IO control, can be used by SW for general purpose

8.2.5.159 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_RTC_RESET_B)

Address: 3033_0000h base + 28Ch offset = 3033_028Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	1	1	0	0	1	1	0	0

IOMUXC_SW_PAD_CTL_PAD_RTC_RESET_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Not used for this IO control, can be used by SW for general purpose
7 HYS	Hysteresis Enable Field Not used for this IO control, can be used by SW for general purpose
6 PUE	Not used for this IO control, can be used by SW for general purpose
5 ODE	Open Drain Enable Field Not used for this IO control, can be used by SW for general purpose
4–3 FSEL	Slew Rate Field Not used for this IO control, can be used by SW for general purpose
DSE	Drive Strength Field Not used for this IO control, can be used by SW for general purpose

8.2.5.160 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO00)

Address: 3033_0000h base + 290h offset = 3033_0290h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO00 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO00 field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.161 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO01)

Address: 3033_0000h base + 294h offset = 3033_0294h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO01 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO01 field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.162 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO02)

Address: 3033_0000h base + 298h offset = 3033_0298h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	1	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO02 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO02 field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.163 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO03)

Address: 3033_0000h base + 29Ch offset = 3033_029Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO03 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO03 field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.164 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO04)

Address: 3033_0000h base + 2A0h offset = 3033_02A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO04 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO04 field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.165 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO05)

Address: 3033_0000h base + 2A4h offset = 3033_02A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	1	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO05 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.166 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO06)

Address: 3033_0000h base + 2A8h offset = 3033_02A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO06 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO06 field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.167 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO07)

Address: 3033_0000h base + 2ACh offset = 3033_02ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO07 field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO07 field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.168 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO08)

Address: 3033_0000h base + 2B0h offset = 3033_02B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO08 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO08 field descriptions (continued)

Field	Description
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.169 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO09)

Address: 3033_0000h base + 2B4h offset = 3033_02B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO09 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO09 field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.170 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO10)

Address: 3033_0000h base + 2B8h offset = 3033_02B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO10 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.171 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO11)

Address: 3033_0000h base + 2BCh offset = 3033_02BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO11 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO11 field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.172 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO12)

Address: 3033_0000h base + 2C0h offset = 3033_02C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO12 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO12 field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.173 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO13)

Address: 3033_0000h base + 2C4h offset = 3033_02C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO13 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO13 field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.174 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO14)

Address: 3033_0000h base + 2C8h offset = 3033_02C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO14 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO14 field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.175 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO15)

Address: 3033_0000h base + 2CCh offset = 3033_02CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO15 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO15 field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.176 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_MDC)

Address: 3033_0000h base + 2D0h offset = 3033_02D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_MDC field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.177 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_MDIO)

Address: 3033_0000h base + 2D4h offset = 3033_02D4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_MDIO field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ENET_MDIO field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.178 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_TD3)

Address: 3033_0000h base + 2D8h offset = 3033_02D8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_TD3 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ENET_TD3 field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.179 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_TD2)

Address: 3033_0000h base + 2DCh offset = 3033_02DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_TD2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ENET_TD2 field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.180 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_TD1)

Address: 3033_0000h base + 2E0h offset = 3033_02E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_TD1 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ENET_TD1 field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.181 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_TD0)

Address: 3033_0000h base + 2E4h offset = 3033_02E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_TD0 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ENET_TD0 field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.182 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_TX_CTL)

Address: 3033_0000h base + 2E8h offset = 3033_02E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_TX_CTL field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.183 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_TXC)

Address: 3033_0000h base + 2ECh offset = 3033_02ECh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_TXC field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ENET_TXC field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.184 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_RX_CTL)

Address: 3033_0000h base + 2F0h offset = 3033_02F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_RX_CTL field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ENET_RX_CTL field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.185 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_RXC)

Address: 3033_0000h base + 2F4h offset = 3033_02F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_RXC field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ENET_RXC field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.186 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_RD0)

Address: 3033_0000h base + 2F8h offset = 3033_02F8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_RD0 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ENET_RD0 field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.187 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_RD1)

Address: 3033_0000h base + 2FCh offset = 3033_02FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_RD1 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ENET_RD1 field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.188 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_RD2)

Address: 3033_0000h base + 300h offset = 3033_0300h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_RD2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.189 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ENET_RD3)

Address: 3033_0000h base + 304h offset = 3033_0304h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ENET_RD3 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ENET_RD3 field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.190 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_CLK)

Address: 3033_0000h base + 308h offset = 3033_0308h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD1_CLK field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD1_CLK field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.191 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_CMD)

Address: 3033_0000h base + 30Ch offset = 3033_030Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD1_CMD field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD1_CMD field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.192 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA0)

Address: 3033_0000h base + 310h offset = 3033_0310h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA0 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA0 field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.193 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA1)

Address: 3033_0000h base + 314h offset = 3033_0314h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA1 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA1 field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.194 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA2)

Address: 3033_0000h base + 318h offset = 3033_0318h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.195 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA3)

Address: 3033_0000h base + 31Ch offset = 3033_031Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA3 field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA3 field descriptions (continued)

Field	Description
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.196 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA4)

Address: 3033_0000h base + 320h offset = 3033_0320h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA4 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA4 field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.197 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA5)

Address: 3033_0000h base + 324h offset = 3033_0324h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA5 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA5 field descriptions (continued)

Field	Description
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.198 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA6)

Address: 3033_0000h base + 328h offset = 3033_0328h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA6 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA6 field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.199 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA7)

Address: 3033_0000h base + 32Ch offset = 3033_032Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD1_DATA7 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.200 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_RESET_B)

Address: 3033_0000h base + 330h offset = 3033_0330h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD1_RESET_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD1_RESET_B field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.201 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_STROBE)

Address: 3033_0000h base + 334h offset = 3033_0334h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD1_STROBE field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD1_STROBE field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.202 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_CD_B)

Address: 3033_0000h base + 338h offset = 3033_0338h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD2_CD_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD2_CD_B field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.203 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_CLK)

Address: 3033_0000h base + 33Ch offset = 3033_033Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD2_CLK field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD2_CLK field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.204 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_CMD)

Address: 3033_0000h base + 340h offset = 3033_0340h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD2_CMD field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD2_CMD field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.205 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA0)

Address: 3033_0000h base + 344h offset = 3033_0344h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD2_DATA0 field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.206 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA1)

Address: 3033_0000h base + 348h offset = 3033_0348h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD2_DATA1 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD2_DATA1 field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.207 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA2)

Address: 3033_0000h base + 34Ch offset = 3033_034Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD2_DATA2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD2_DATA2 field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.208 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA3)

Address: 3033_0000h base + 350h offset = 3033_0350h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD2_DATA3 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD2_DATA3 field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.209 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_RESET_B)

Address: 3033_0000h base + 354h offset = 3033_0354h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD2_RESET_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD2_RESET_B field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.210 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_WP)

Address: 3033_0000h base + 358h offset = 3033_0358h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SD2_WP field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SD2_WP field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.211 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_ALE)

Address: 3033_0000h base + 35Ch offset = 3033_035Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_ALE field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.212 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_CE0_B)

Address: 3033_0000h base + 360h offset = 3033_0360h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_CE0_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_CE0_B field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.213 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_CE1_B)

Address: 3033_0000h base + 364h offset = 3033_0364h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_CE1_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_CE1_B field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.214 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_CE2_B)

Address: 3033_0000h base + 368h offset = 3033_0368h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_CE2_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_CE2_B field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.215 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_CE3_B)

Address: 3033_0000h base + 36Ch offset = 3033_036Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_CE3_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_CE3_B field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.216 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_CLE)

Address: 3033_0000h base + 370h offset = 3033_0370h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_CLE field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_CLE field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.217 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA00)

Address: 3033_0000h base + 374h offset = 3033_0374h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA00 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.218 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA01)

Address: 3033_0000h base + 378h offset = 3033_0378h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA01 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA01 field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.219 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA02)

Address: 3033_0000h base + 37Ch offset = 3033_037Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA02 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA02 field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.220 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA03)

Address: 3033_0000h base + 380h offset = 3033_0380h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA03 field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA03 field descriptions (continued)

Field	Description
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.221 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA04)

Address: 3033_0000h base + 384h offset = 3033_0384h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA04 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA04 field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.222 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA05)

Address: 3033_0000h base + 388h offset = 3033_0388h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA05 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.223 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA06)

Address: 3033_0000h base + 38Ch offset = 3033_038Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA06 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA06 field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.224 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DATA07)

Address: 3033_0000h base + 390h offset = 3033_0390h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA07 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_DATA07 field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.225 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_DQS)

Address: 3033_0000h base + 394h offset = 3033_0394h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_DQS field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_DQS field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.226 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_RE_B)

Address: 3033_0000h base + 398h offset = 3033_0398h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_RE_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_RE_B field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.227 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_READY_B)

Address: 3033_0000h base + 39Ch offset = 3033_039Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_READY_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_READY_B field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.228 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_WE_B)

Address: 3033_0000h base + 3A0h offset = 3033_03A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_WE_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.229 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_NAND_WP_B)

Address: 3033_0000h base + 3A4h offset = 3033_03A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_NAND_WP_B field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_NAND_WP_B field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.230 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_RXFS)

Address: 3033_0000h base + 3A8h offset = 3033_03A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI5_RXFS field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI5_RXFS field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.231 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_RXC)

Address: 3033_0000h base + 3ACh offset = 3033_03ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI5_RXC field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI5_RXC field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.232 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD0)

Address: 3033_0000h base + 3B0h offset = 3033_03B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD0 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD0 field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.233 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD1)

Address: 3033_0000h base + 3B4h offset = 3033_03B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD1 field descriptions

Field	Description
31–14 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD1 field descriptions (continued)

Field	Description
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.234 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD2)

Address: 3033_0000h base + 3B8h offset = 3033_03B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD2 field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.235 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD3)

Address: 3033_0000h base + 3BCh offset = 3033_03BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD3 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI5_RXD3 field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.236 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI5_MCLK)

Address: 3033_0000h base + 3C0h offset = 3033_03C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI5_MCLK field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI5_MCLK field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.237 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXFS)

Address: 3033_0000h base + 3C4h offset = 3033_03C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXFS field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXFS field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.238 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXC)

Address: 3033_0000h base + 3C8h offset = 3033_03C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXC field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXC field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.239 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD0)

Address: 3033_0000h base + 3CCh offset = 3033_03CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD0 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.240 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD1)

Address: 3033_0000h base + 3D0h offset = 3033_03D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD1 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD1 field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.241 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD2)

Address: 3033_0000h base + 3D4h offset = 3033_03D4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD2 field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.242 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD3)

Address: 3033_0000h base + 3D8h offset = 3033_03D8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD3 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD3 field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.243 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD4)

Address: 3033_0000h base + 3DCh offset = 3033_03DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD4 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD4 field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.244 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD5)

Address: 3033_0000h base + 3E0h offset = 3033_03E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD5 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD5 field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.245 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD6)

Address: 3033_0000h base + 3E4h offset = 3033_03E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD6 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.246 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD7)

Address: 3033_0000h base + 3E8h offset = 3033_03E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD7 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_RXD7 field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.247 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXFS)

Address: 3033_0000h base + 3ECh offset = 3033_03ECh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXFS field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXFS field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.248 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXC)

Address: 3033_0000h base + 3F0h offset = 3033_03F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXC field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXC field descriptions (continued)

Field	Description
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.249 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD0)

Address: 3033_0000h base + 3F4h offset = 3033_03F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD0 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD0 field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.250 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD1)

Address: 3033_0000h base + 3F8h offset = 3033_03F8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD1 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.251 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD2)

Address: 3033_0000h base + 3FCh offset = 3033_03FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD2 field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.252 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD3)

Address: 3033_0000h base + 400h offset = 3033_0400h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD3 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD3 field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.253 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD4)

Address: 3033_0000h base + 404h offset = 3033_0404h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD4 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD4 field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.254 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD5)

Address: 3033_0000h base + 408h offset = 3033_0408h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD5 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD5 field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.255 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD6)

Address: 3033_0000h base + 40Ch offset = 3033_040Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD6 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD6 field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.256 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD7)

Address: 3033_0000h base + 410h offset = 3033_0410h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_TXD7 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.257 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI1_MCLK)

Address: 3033_0000h base + 414h offset = 3033_0414h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI1_MCLK field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI1_MCLK field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.258 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_RXFS)

Address: 3033_0000h base + 418h offset = 3033_0418h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI2_RXFS field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI2_RXFS field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.259 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_RXC)

Address: 3033_0000h base + 41Ch offset = 3033_041Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI2_RXC field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI2_RXC field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.260 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_RXD0)

Address: 3033_0000h base + 420h offset = 3033_0420h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI2_RXD0 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI2_RXD0 field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.261 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_TXFS)

Address: 3033_0000h base + 424h offset = 3033_0424h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI2_TXFS field descriptions

Field	Description
31–14 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI2_TXFS field descriptions (continued)

Field	Description
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.262 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_TXC)

Address: 3033_0000h base + 428h offset = 3033_0428h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI2_TXC field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI2_TXC field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.263 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_TXD0)

Address: 3033_0000h base + 42Ch offset = 3033_042Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI2_TXD0 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI2_TXD0 field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.264 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI2_MCLK)

Address: 3033_0000h base + 430h offset = 3033_0430h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI2_MCLK field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI2_MCLK field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.265 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_RXFS)

Address: 3033_0000h base + 434h offset = 3033_0434h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI3_RXFS field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI3_RXFS field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.266 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_RXC)

Address: 3033_0000h base + 438h offset = 3033_0438h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI3_RXC field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

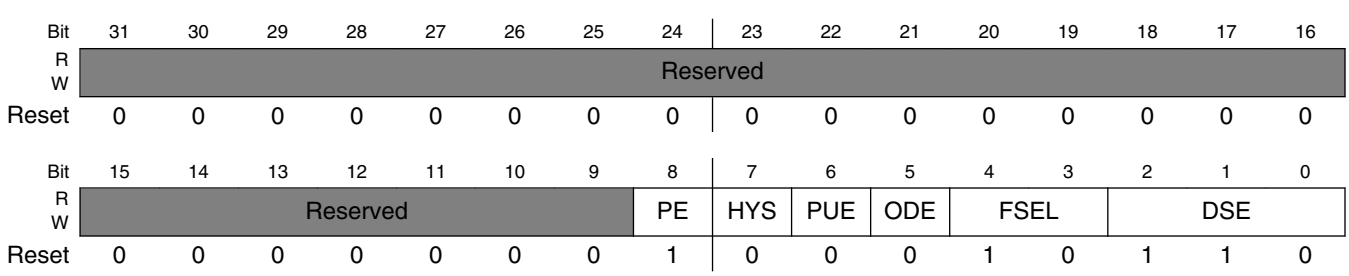
Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI3_RXC field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.267 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_RXD)

Address: 3033_0000h base + 43Ch offset = 3033_043Ch



IOMUXC_SW_PAD_CTL_PAD_SAI3_RXD field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.268 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_TXFS)

Address: 3033_0000h base + 440h offset = 3033_0440h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI3_TXFS field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI3_TXFS field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.269 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_TXC)

Address: 3033_0000h base + 444h offset = 3033_0444h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI3_TXC field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI3_TXC field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.270 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_TXD)

Address: 3033_0000h base + 448h offset = 3033_0448h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI3_TXD field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI3_TXD field descriptions (continued)

Field	Description
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.271 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SAI3_MCLK)

Address: 3033_0000h base + 44Ch offset = 3033_044Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SAI3_MCLK field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SAI3_MCLK field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.272 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SPDIF_TX)

Address: 3033_0000h base + 450h offset = 3033_0450h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SPDIF_TX field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.273 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SPDIF_RX)

Address: 3033_0000h base + 454h offset = 3033_0454h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SPDIF_RX field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SPDIF_RX field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.274 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SPDIF_EXT_CLK)

Address: 3033_0000h base + 458h offset = 3033_0458h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_SPDIF_EXT_CLK field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_SPDIF_EXT_CLK field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.275 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI1_SCLK)

Address: 3033_0000h base + 45Ch offset = 3033_045Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ECSPI1_SCLK field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ECSPI1_SCLK field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.276 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI1_MOSI)

Address: 3033_0000h base + 460h offset = 3033_0460h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ECSPI1_MOSI field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ECSPI1_MOSI field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.277 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI1_MISO)

Address: 3033_0000h base + 464h offset = 3033_0464h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ECSPI1_MISO field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ECSPI1_MISO field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.278 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI1_SS0)

Address: 3033_0000h base + 468h offset = 3033_0468h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ECSPI1_SS0 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.279 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI2_SCLK)

Address: 3033_0000h base + 46Ch offset = 3033_046Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ECSPI2_SCLK field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ECSPI2_SCLK field descriptions (continued)

Field	Description
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.280 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI2_MOSI)

Address: 3033_0000h base + 470h offset = 3033_0470h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ECSPI2_MOSI field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ECSPI2_MOSI field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.281 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI2_MISO)

Address: 3033_0000h base + 474h offset = 3033_0474h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ECSPI2_MISO field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ECSPI2_MISO field descriptions (continued)

Field	Description
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.282 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSPI2_SS0)

Address: 3033_0000h base + 478h offset = 3033_0478h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_ECSPI2_SS0 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_ECSPI2_SS0 field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.283 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C1_SCL)

Address: 3033_0000h base + 47Ch offset = 3033_047Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_I2C1_SCL field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.284 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C1_SDA)

Address: 3033_0000h base + 480h offset = 3033_0480h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_I2C1_SDA field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

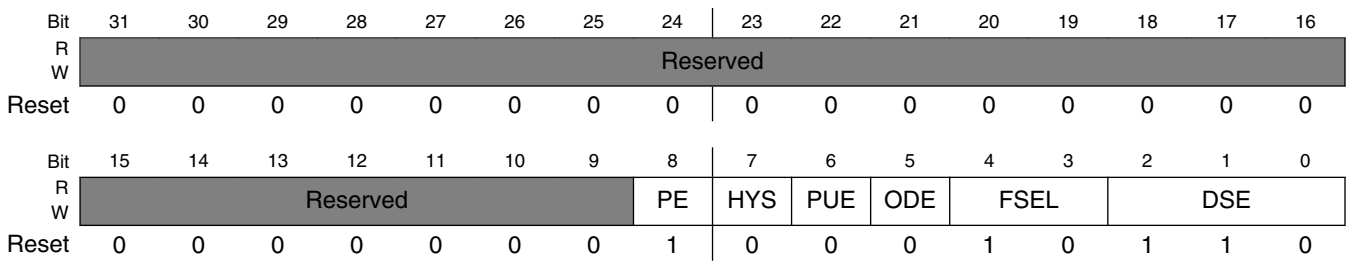
Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_I2C1_SDA field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.285 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C2_SCL)

Address: 3033_0000h base + 484h offset = 3033_0484h



IOMUXC_SW_PAD_CTL_PAD_I2C2_SCL field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_I2C2_SCL field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.286 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C2_SDA)

Address: 3033_0000h base + 488h offset = 3033_0488h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_I2C2_SDA field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_I2C2_SDA field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.287 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C3_SCL)

Address: 3033_0000h base + 48Ch offset = 3033_048Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_I2C3_SCL field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_I2C3_SCL field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.288 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C3_SDA)

Address: 3033_0000h base + 490h offset = 3033_0490h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_I2C3_SDA field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.289 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C4_SCL)

Address: 3033_0000h base + 494h offset = 3033_0494h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_I2C4_SCL field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_I2C4_SCL field descriptions (continued)

Field	Description
	10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.290 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C4_SDA)

Address: 3033_0000h base + 498h offset = 3033_0498h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_I2C4_SDA field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used):

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_I2C4_SDA field descriptions (continued)

Field	Description
	0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.291 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART1_RXD)

Address: 3033_0000h base + 49Ch offset = 3033_049Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_UART1_RXD field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_UART1_RXD field descriptions (continued)

Field	Description
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.292 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART1_TXD)

Address: 3033_0000h base + 4A0h offset = 3033_04A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_UART1_TXD field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_UART1_TXD field descriptions (continued)

Field	Description
	Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.293 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART2_RXD)

Address: 3033_0000h base + 4A4h offset = 3033_04A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_UART2_RXD field descriptions

Field	Description
31–9 -	This field is reserved. Reserved

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_UART2_RXD field descriptions (continued)

Field	Description
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.294 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART2_TXD)

Address: 3033_0000h base + 4A8h offset = 3033_04A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_UART2_TXD field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.295 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART3_RXD)

Address: 3033_0000h base + 4ACh offset = 3033_04ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved			Reserved		PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_UART3_RXD field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–11 -	This field is reserved. Reserved
10–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_UART3_RXD field descriptions (continued)

Field	Description
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.296 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART3_TXD)

Address: 3033_0000h base + 4B0h offset = 3033_04B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_UART3_TXD field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_UART3_TXD field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.297 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART4_RXD)

Address: 3033_0000h base + 4B4h offset = 3033_04B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_UART4_RXD field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE: 0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_UART4_RXD field descriptions (continued)

Field	Description
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.298 Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART4_TXD)

Address: 3033_0000h base + 4B8h offset = 3033_04B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PE	HYS	PUE	ODE	FSEL		DSE		
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0

IOMUXC_SW_PAD_CTL_PAD_UART4_TXD field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 PE	Pull Resistors Enable Field Control IO ports PE:

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_UART4_TXD field descriptions (continued)

Field	Description
	0 **DISABLED** — Disable pull resistors 1 **ENABLED** — Enable pull resistors
7 HYS	Hysteresis Enable Field Control the IO ports IS: 0 **DISABLED** — Select CMOS input 1 **ENABLED** — Select Schmitt input
6 PUE	Control IO ports PS: 0 **DISABLED** — Select pull-down resistors 1 **ENABLED** — Select pull-up resistors
5 ODE	Open Drain Enable Field Control the IO ports ODE: 0 **DISABLED** — Disable open-drain mode 1 **ENABLED** — Enable open-drain mode
4–3 FSEL	Slew Rate Field Control the IO ports SR with inverted value (lsb of this field is not used): 0X **SLOW** — Select slow slew rate (SR=1) 1X **FAST** — Select fast slew rate (SR=0)
DSE	Drive Strength Field Control the IO ports DS1/0 by dse[2:1] (lsb of this field is not used): 00X **X1** — Drive strength X1 10X **X2** — Drive strength X2 01X **X4** — Drive strength X4 11X **X6** — Drive strength X6

8.2.5.299 Select Input Register (IOMUXC_CCM_PMIC_READY_SELECT_INPUT)

Address: 3033_0000h base + 4BCh offset = 3033_04BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															DAISY
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_CCM_PMIC_READY_SELECT_INPUT field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **GPIO1_IO05_ALT5** — Selecting ALT5 mode of pad GPIO1_IO05 for CCM_PMIC_READY. 1 **GPIO1_IO11_ALT5** — Selecting ALT5 mode of pad GPIO1_IO11 for CCM_PMIC_READY.

8.2.5.300 Select Input Register (IOMUXC_ENET1_MDIO_SELECT_INPUT)

Address: 3033_0000h base + 4C0h offset = 3033_04C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_ENET1_MDIO_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **GPIO1_IO07_ALT1** — Selecting ALT1 mode of pad GPIO1_IO07 for ENET1_MDIO. 01 **ENET_MDIO_ALT0** — Selecting ALT0 mode of pad ENET_MDIO for ENET1_MDIO. 10 **I2C1_SDA_ALT1** — Selecting ALT1 mode of pad I2C1_SDA for ENET1_MDIO. 11 **GPIO1_IO07_ALT1** — Same as 00

8.2.5.301 Select Input Register (IOMUXC_SAI1_RX_SYNC_SELECT_INPUT)

Address: 3033_0000h base + 4C4h offset = 3033_04C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															DAISY
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SAI1_RX_SYNC_SELECT_INPUT field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **SAI1_RXFS_ALT0** — Selecting ALT0 mode of pad SAI1_RXFS for SAI1_RX_SYNC. 1 **SAI1_RXD5_ALT3** — Selecting ALT3 mode of pad SAI1_RXD5 for SAI1_RX_SYNC.

8.2.5.302 Select Input Register (IOMUXC_SAI1_TX_BCLK_SELECT_INPUT)

Address: 3033_0000h base + 4C8h offset = 3033_04C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SAI1_TX_BCLK_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **SAI5_MCLK_ALT1** — Selecting ALT1 mode of pad SAI5_MCLK for SAI1_TX_BCLK. 01 **SAI1_TXC_ALT0** — Selecting ALT0 mode of pad SAI1_TXC for SAI1_TX_BCLK. 10 **SAI1_MCLK_ALT2** — Selecting ALT2 mode of pad SAI1_MCLK for SAI1_TX_BCLK. 11 **SAI5_MCLK_ALT1** — Same as 00

8.2.5.303 Select Input Register (IOMUXC_SAI1_TX_SYNC_SELECT_INPUT)

Address: 3033_0000h base + 4CCh offset = 3033_04CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SAI1_TX_SYNC_SELECT_INPUT field descriptions

Field	Description
31–3 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 000 **SAI5_RXD1_ALT2** — Selecting ALT2 mode of pad SAI5_RXD1 for SAI1_TX_SYNC. 001 **SAI5_RXD2_ALT2** — Selecting ALT2 mode of pad SAI5_RXD2 for SAI1_TX_SYNC.

Table continues on the next page...

IOMUXC_SAI1_TX_SYNC_SELECT_INPUT field descriptions (continued)

Field	Description
	010 **SAI5_RXD3_ALT2** — Selecting ALT2 mode of pad SAI5_RXD3 for SAI1_TX_SYNC. 011 **SAI1_TXFS_ALT0** — Selecting ALT0 mode of pad SAI1_TXFS for SAI1_TX_SYNC. 100 **SAI1_RXD7_ALT2** — Selecting ALT2 mode of pad SAI1_RXD7 for SAI1_TX_SYNC. 101 **SAI5_RXD1_ALT2** — Same as 000 110 **SAI5_RXD1_ALT2** — Same as 000 111 **SAI5_RXD1_ALT2** — Same as 000

8.2.5.304 Select Input Register (IOMUXC_SAI5_RX_BCLK_SELECT_INPUT)

Address: 3033_0000h base + 4D0h offset = 3033_04D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SAI5_RX_BCLK_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **SAI5_RXC_ALT0** — Selecting ALT0 mode of pad SAI5_RXC for SAI5_RX_BCLK. 01 **SAI1_RXC_ALT1** — Selecting ALT1 mode of pad SAI1_RXC for SAI5_RX_BCLK. 10 **SAI3_RXC_ALT2** — Selecting ALT2 mode of pad SAI3_RXC for SAI5_RX_BCLK. 11 **SAI5_RXC_ALT0** — Same as 00

8.2.5.305 Select Input Register (IOMUXC_SAI5_RX_DATA_SELECT_INPUT_0)

Address: 3033_0000h base + 4D4h offset = 3033_04D4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SAI5_RX_DATA_SELECT_INPUT_0 field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **SAI5_RXD0_ALT0** — Selecting ALT0 mode of pad SAI5_RXD0 for SAI5_RX_DATA_0. 01 **SAI1_RXD0_ALT1** — Selecting ALT1 mode of pad SAI1_RXD0 for SAI5_RX_DATA_0. 10 **SAI3_RXD_ALT2** — Selecting ALT2 mode of pad SAI3_RXD for SAI5_RX_DATA_0. 11 **SAI5_RXD0_ALT0** — Same as 00

8.2.5.306 Select Input Register (IOMUXC_SAI5_RX_DATA_SELECT_INPUT_1)

Address: 3033_0000h base + 4D8h offset = 3033_04D8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SAI5_RX_DATA_SELECT_INPUT_1 field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field

Table continues on the next page...

IOMUXC_SAI5_RX_DATA_SELECT_INPUT_1 field descriptions (continued)

Field	Description
	Selecting Pads Involved in Daisy Chain. 00 **SAI5_RXD1_ALT0** — Selecting ALT0 mode of pad SAI5_RXD1 for SAI5_RX_DATA_1. 01 **SAI1_RXD1_ALT1** — Selecting ALT1 mode of pad SAI1_RXD1 for SAI5_RX_DATA_1. 10 **SAI3_TXFS_ALT1** — Selecting ALT1 mode of pad SAI3_TXFS for SAI5_RX_DATA_1. 11 **SAI5_RXD1_ALT0** — Same as 00

8.2.5.307 Select Input Register (IOMUXC_SAI5_RX_DATA_SELECT_INPUT_2)

Address: 3033_0000h base + 4DCh offset = 3033_04DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SAI5_RX_DATA_SELECT_INPUT_2 field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **SAI5_RXD2_ALT0** — Selecting ALT0 mode of pad SAI5_RXD2 for SAI5_RX_DATA_2. 01 **SAI1_RXD2_ALT1** — Selecting ALT1 mode of pad SAI1_RXD2 for SAI5_RX_DATA_2. 10 **SAI3_TXC_ALT2** — Selecting ALT2 mode of pad SAI3_TXC for SAI5_RX_DATA_2. 11 **SAI5_RXD2_ALT0** — Same as 00

8.2.5.308 Select Input Register (IOMUXC_SAI5_RX_DATA_SELECT_INPUT_3)

Address: 3033_0000h base + 4E0h offset = 3033_04E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SAI5_RX_DATA_SELECT_INPUT_3 field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **SAI5_RXD3_ALT0** — Selecting ALT0 mode of pad SAI5_RXD3 for SAI5_RX_DATA_3. 01 **SAI1_RXD3_ALT1** — Selecting ALT1 mode of pad SAI1_RXD3 for SAI5_RX_DATA_3. 10 **SAI3_TXD_ALT2** — Selecting ALT2 mode of pad SAI3_TXD for SAI5_RX_DATA_3. 11 **SAI5_RXD3_ALT0** — Same as 00

8.2.5.309 Select Input Register (IOMUXC_SAI5_RX_SYNC_SELECT_INPUT)

Address: 3033_0000h base + 4E4h offset = 3033_04E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SAI5_RX_SYNC_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field

Table continues on the next page...

IOMUXC_SAI5_RX_SYNC_SELECT_INPUT field descriptions (continued)

Field	Description
	Selecting Pads Involved in Daisy Chain. 00 **SAI5_RXFS_ALT0** — Selecting ALT0 mode of pad SAI5_RXFS for SAI5_RX_SYNC. 01 **SAI1_RXFS_ALT1** — Selecting ALT1 mode of pad SAI1_RXFS for SAI5_RX_SYNC. 10 **SAI3_RXFS_ALT2** — Selecting ALT2 mode of pad SAI3_RXFS for SAI5_RX_SYNC. 11 **SAI5_RXFS_ALT0** — Same as 00

8.2.5.310 Select Input Register (IOMUXC_SAI5_TX_BCLK_SELECT_INPUT)

Address: 3033_0000h base + 4E8h offset = 3033_04E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SAI5_TX_BCLK_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **SAI5_RXD2_ALT3** — Selecting ALT3 mode of pad SAI5_RXD2 for SAI5_TX_BCLK. 01 **SAI1_TXC_ALT1** — Selecting ALT1 mode of pad SAI1_TXC for SAI5_TX_BCLK. 10 **SAI2_RXC_ALT1** — Selecting ALT1 mode of pad SAI2_RXC for SAI5_TX_BCLK. 11 **SAI5_RXD2_ALT3** — Same as 00

8.2.5.311 Select Input Register (IOMUXC_SAI5_TX_SYNC_SELECT_INPUT)

Address: 3033_0000h base + 4ECh offset = 3033_04ECh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SAI5_TX_SYNC_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **SAI5_RXD1_ALT3** — Selecting ALT3 mode of pad SAI5_RXD1 for SAI5_TX_SYNC. 01 **SAI1_TXFS_ALT1** — Selecting ALT1 mode of pad SAI1_TXFS for SAI5_TX_SYNC. 10 **SAI2_RXFS_ALT1** — Selecting ALT1 mode of pad SAI2_RXFS for SAI5_TX_SYNC. 11 **SAI5_RXD1_ALT3** — Same as 00

8.2.5.312 Select Input Register (IOMUXC_UART1_RTS_B_SELECT_INPUT)

Address: 3033_0000h base + 4F0h offset = 3033_04F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_UART1_RTS_B_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field

Table continues on the next page...

IOMUXC_UART1_RTS_B_SELECT_INPUT field descriptions (continued)

Field	Description
	Selecting Pads Involved in Daisy Chain. 00 **UART3_RXD_ALT1** — Selecting ALT1 mode of pad UART3_RXD for UART1_RTS_B. 01 **UART3_TXD_ALT1** — Selecting ALT1 mode of pad UART3_TXD for UART1_RTS_B. 10 **SAI2_RXD0_ALT4** — Selecting ALT4 mode of pad SAI2_RXD0 for UART1_RTS_B. 11 **SAI2_TXFS_ALT4** — Selecting ALT4 mode of pad SAI2_TXFS for UART1_RTS_B.

8.2.5.313 Select Input Register (IOMUXC_UART1_RXD_SELECT_INPUT)

Address: 3033_0000h base + 4F4h offset = 3033_04F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_UART1_RXD_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **UART1_RXD_ALT0** — Selecting ALT0 mode of pad UART1_RXD for UART1_RXD. 01 **UART1_TXD_ALT0** — Selecting ALT0 mode of pad UART1_TXD for UART1_RXD. 10 **SAI2_RXFS_ALT4** — Selecting ALT4 mode of pad SAI2_RXFS for UART1_RXD. 11 **SAI2_RXC_ALT4** — Selecting ALT4 mode of pad SAI2_RXC for UART1_RXD.

8.2.5.314 Select Input Register (IOMUXC_UART2_RTS_B_SELECT_INPUT)

Address: 3033_0000h base + 4F8h offset = 3033_04F8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_UART2_RTS_B_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **UART4_RXD_ALT1** — Selecting ALT1 mode of pad UART4_RXD for UART2_RTS_B. 01 **UART4_TXD_ALT1** — Selecting ALT1 mode of pad UART4_TXD for UART2_RTS_B. 10 **SAI3_RXC_ALT4** — Selecting ALT4 mode of pad SAI3_RXC for UART2_RTS_B. 11 **SAI3_RXD_ALT4** — Selecting ALT4 mode of pad SAI3_RXD for UART2_RTS_B.

8.2.5.315 Select Input Register (IOMUXC_UART2_RXD_SELECT_INPUT)

Address: 3033_0000h base + 4FCh offset = 3033_04FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_UART2_RXD_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field

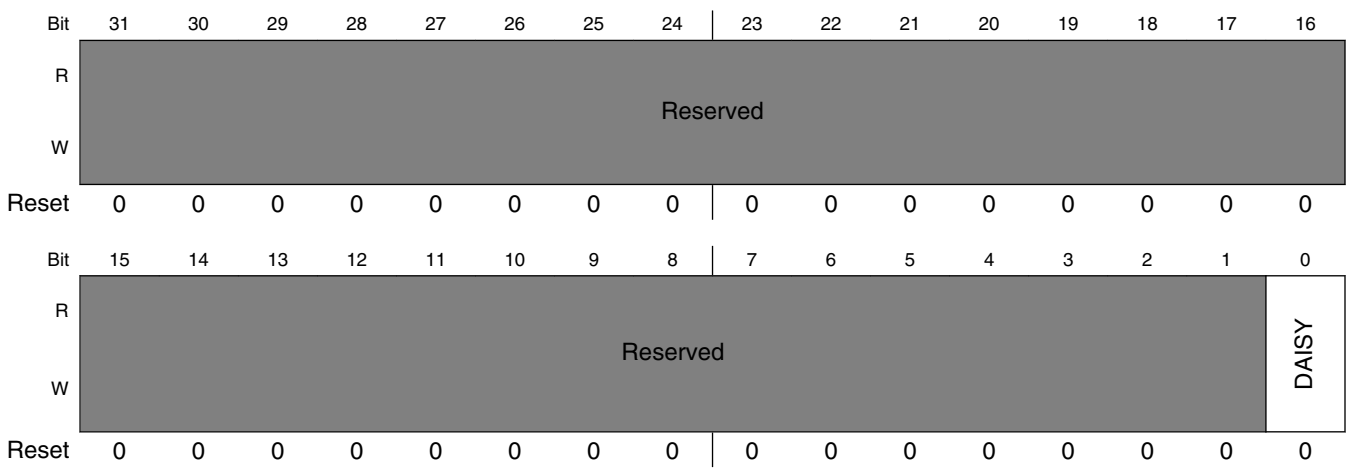
Table continues on the next page...

IOMUXC_UART2_RXD_SELECT_INPUT field descriptions (continued)

Field	Description
	Selecting Pads Involved in Daisy Chain. 00 **UART2_RXD_ALT0** — Selecting ALT0 mode of pad UART2_RXD for UART2_RXD. 01 **UART2_TXD_ALT0** — Selecting ALT0 mode of pad UART2_TXD for UART2_RXD. 10 **SAI3_TXFS_ALT4** — Selecting ALT4 mode of pad SAI3_TXFS for UART2_RXD. 11 **SAI3_TXC_ALT4** — Selecting ALT4 mode of pad SAI3_TXC for UART2_RXD.

8.2.5.316 Select Input Register (IOMUXC_UART3_RTS_B_SELECT_INPUT)

Address: 3033_0000h base + 500h offset = 3033_0500h



IOMUXC_UART3_RTS_B_SELECT_INPUT field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **ECSPI1_MISO_ALT1** — Selecting ALT1 mode of pad ECSPI1_MISO for UART3_RTS_B. 1 **ECSPI1_SS0_ALT1** — Selecting ALT1 mode of pad ECSPI1_SS0 for UART3_RTS_B.

8.2.5.317 Select Input Register (IOMUXC_UART3_RXD_SELECT_INPUT)

Address: 3033_0000h base + 504h offset = 3033_0504h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_UART3_RXD_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **ECSPI1_SCLK_ALT1** — Selecting ALT1 mode of pad ECSPI1_SCLK for UART3_RXD. 01 **ECSPI1_MOSI_ALT1** — Selecting ALT1 mode of pad ECSPI1_MOSI for UART3_RXD. 10 **UART3_RXD_ALT0** — Selecting ALT0 mode of pad UART3_RXD for UART3_RXD. 11 **UART3_TXD_ALT0** — Selecting ALT0 mode of pad UART3_TXD for UART3_RXD.

8.2.5.318 Select Input Register (IOMUXC_UART4_RTS_B_SELECT_INPUT)

Address: 3033_0000h base + 508h offset = 3033_0508h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															DAISY
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_UART4_RTS_B_SELECT_INPUT field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **ECSPI2_MISO_ALT1** — Selecting ALT1 mode of pad ECSPI2_MISO for UART4_RTS_B. 1 **ECSPI2_SS0_ALT1** — Selecting ALT1 mode of pad ECSPI2_SS0 for UART4_RTS_B.

8.2.5.319 Select Input Register (IOMUXC_UART4_RXD_SELECT_INPUT)

Address: 3033_0000h base + 50Ch offset = 3033_050Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

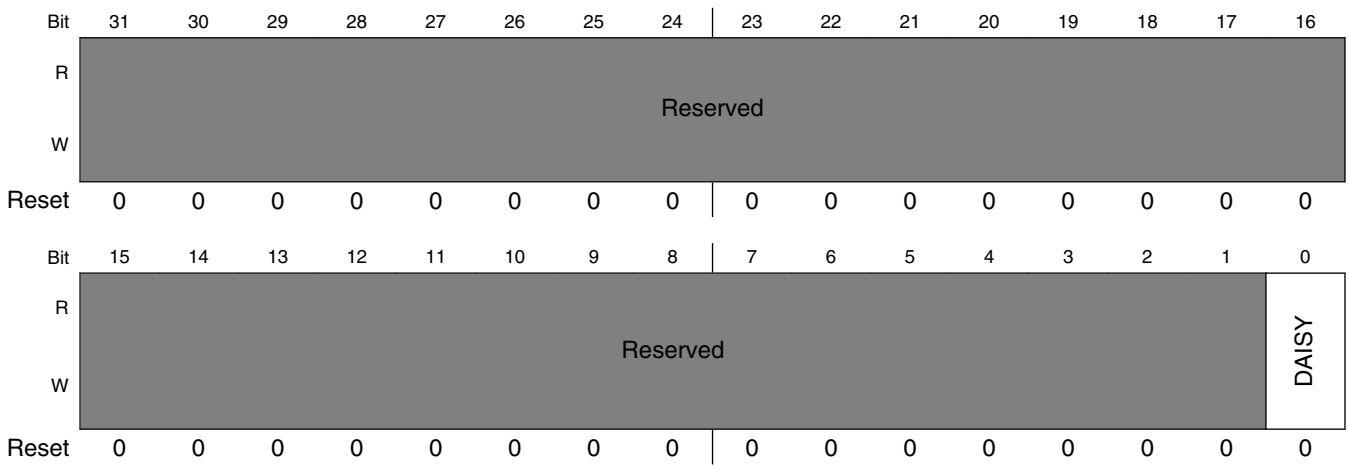
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_UART4_RXD_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **ECSPI2_SCLK_ALT1** — Selecting ALT1 mode of pad ECSPI2_SCLK for UART4_RXD. 01 **ECSPI2_MOSI_ALT1** — Selecting ALT1 mode of pad ECSPI2_MOSI for UART4_RXD. 10 **UART4_RXD_ALT0** — Selecting ALT0 mode of pad UART4_RXD for UART4_RXD. 11 **UART4_TXD_ALT0** — Selecting ALT0 mode of pad UART4_TXD for UART4_RXD.

8.2.5.320 Select Input Register (IOMUXC_SAI6_RX_BCLK_SELECT_INPUT)

Address: 3033_0000h base + 510h offset = 3033_0510h

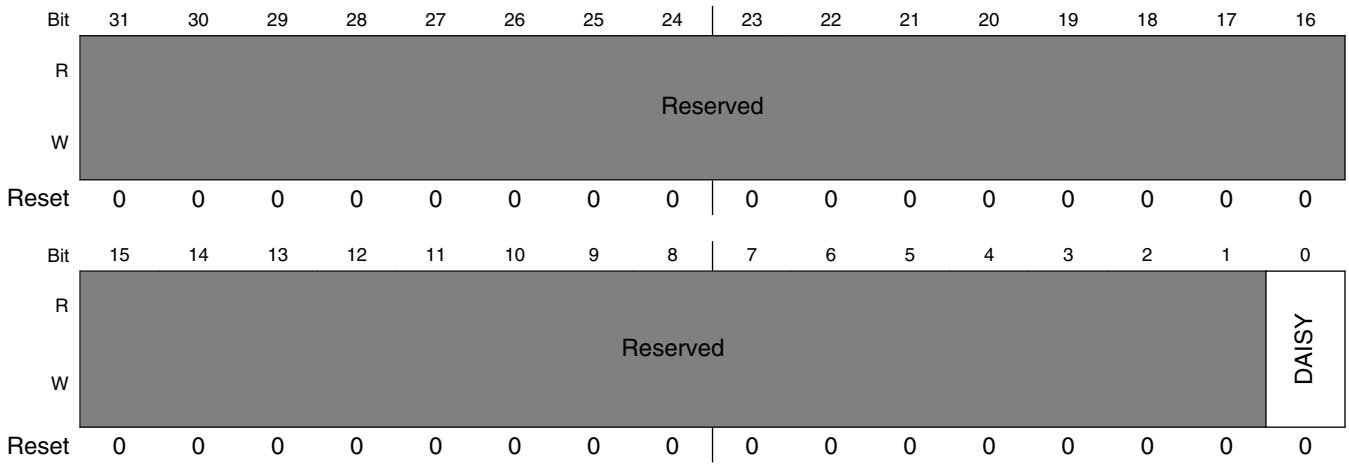


IOMUXC_SAI6_RX_BCLK_SELECT_INPUT field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **SAI1_RXD4_ALT2** — Selecting ALT2 mode of pad SAI1_RXD4 for SAI6_RX_BCLK. 1 **SAI1_TXD4_ALT1** — Selecting ALT1 mode of pad SAI1_TXD4 for SAI6_RX_BCLK.

8.2.5.321 Select Input Register (IOMUXC_SAI6_RX_DATA_SELECT_INPUT_0)

Address: 3033_0000h base + 514h offset = 3033_0514h

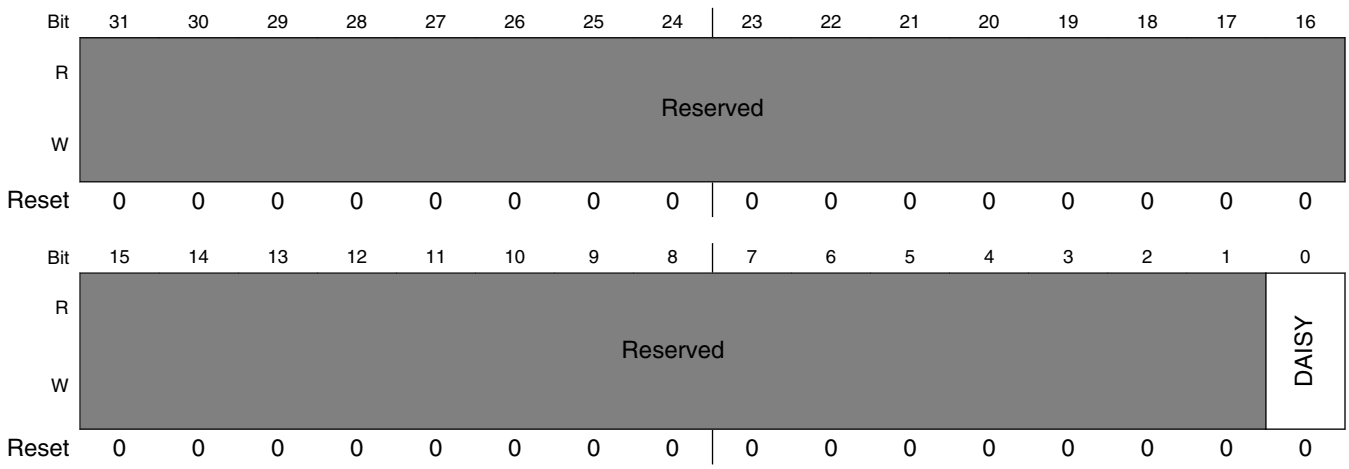


IOMUXC_SAI6_RX_DATA_SELECT_INPUT_0 field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **SAI1_RXD5_ALT2** — Selecting ALT2 mode of pad SAI1_RXD5 for SAI6_RX_DATA_0. 1 **SAI1_TXD5_ALT1** — Selecting ALT1 mode of pad SAI1_TXD5 for SAI6_RX_DATA_0.

8.2.5.322 Select Input Register (IOMUXC_SAI6_RX_SYNC_SELECT_INPUT)

Address: 3033_0000h base + 518h offset = 3033_0518h

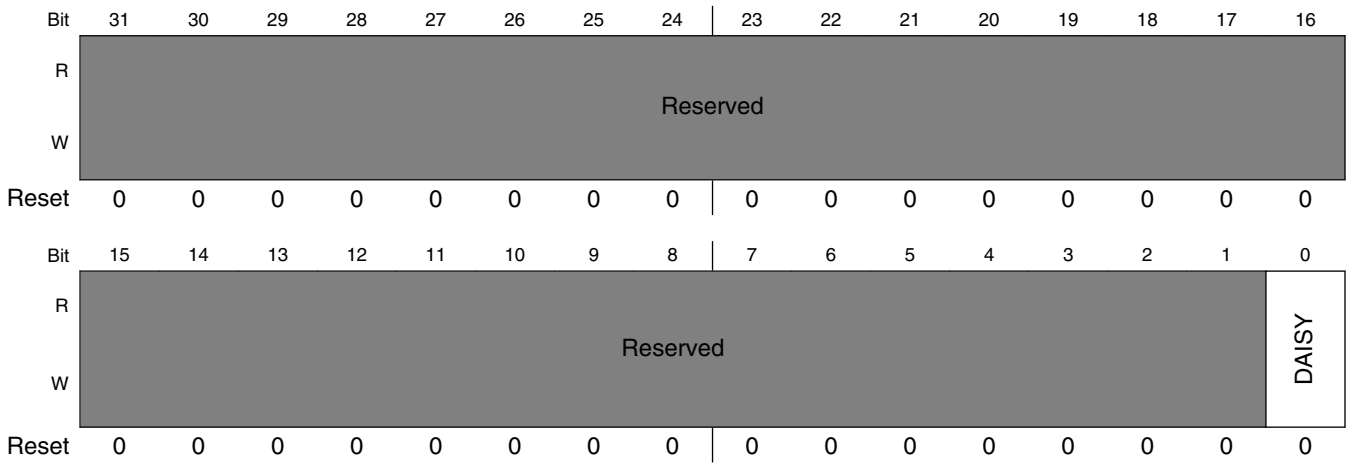


IOMUXC_SAI6_RX_SYNC_SELECT_INPUT field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **SAI1_RXD6_ALT2** — Selecting ALT2 mode of pad SAI1_RXD6 for SAI6_RX_SYNC. 1 **SAI1_TXD6_ALT1** — Selecting ALT1 mode of pad SAI1_TXD6 for SAI6_RX_SYNC.

8.2.5.323 Select Input Register (IOMUXC_SAI6_TX_BCLK_SELECT_INPUT)

Address: 3033_0000h base + 51Ch offset = 3033_051Ch

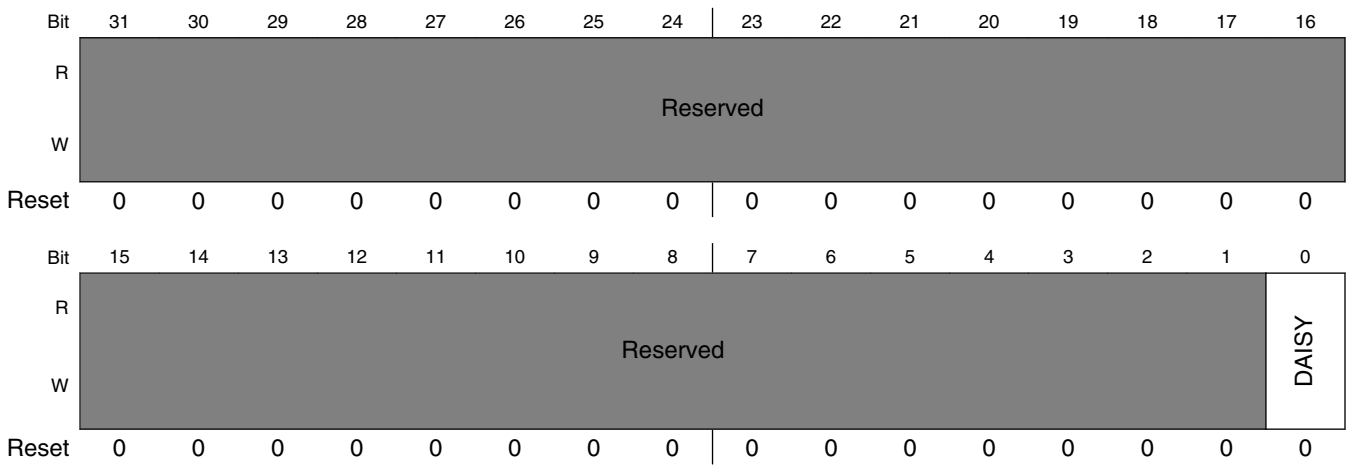


IOMUXC_SAI6_TX_BCLK_SELECT_INPUT field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **SAI1_RXD4_ALT1** — Selecting ALT1 mode of pad SAI1_RXD4 for SAI6_TX_BCLK. 1 **SAI1_TXD4_ALT2** — Selecting ALT2 mode of pad SAI1_TXD4 for SAI6_TX_BCLK.

8.2.5.324 Select Input Register (IOMUXC_SAI6_TX_SYNC_SELECT_INPUT)

Address: 3033_0000h base + 520h offset = 3033_0520h

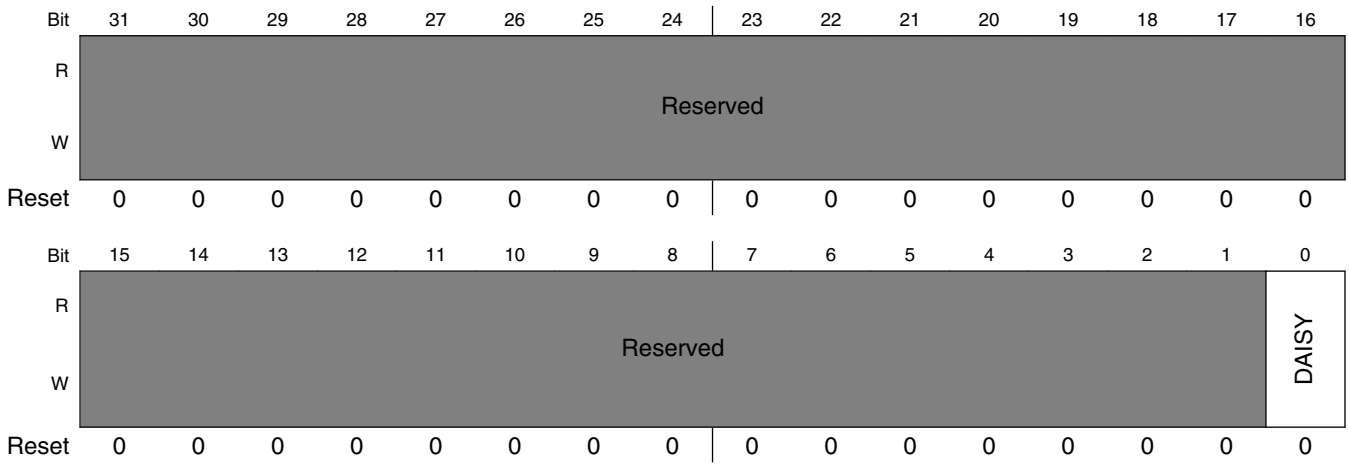


IOMUXC_SAI6_TX_SYNC_SELECT_INPUT field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **SAI1_RXD6_ALT1** — Selecting ALT1 mode of pad SAI1_RXD6 for SAI6_TX_SYNC. 1 **SAI1_TXD6_ALT2** — Selecting ALT2 mode of pad SAI1_TXD6 for SAI6_TX_SYNC.

8.2.5.325 Select Input Register (IOMUXC_PCIE1_CLKREQ_B_SELECT_INPUT)

Address: 3033_0000h base + 524h offset = 3033_0524h

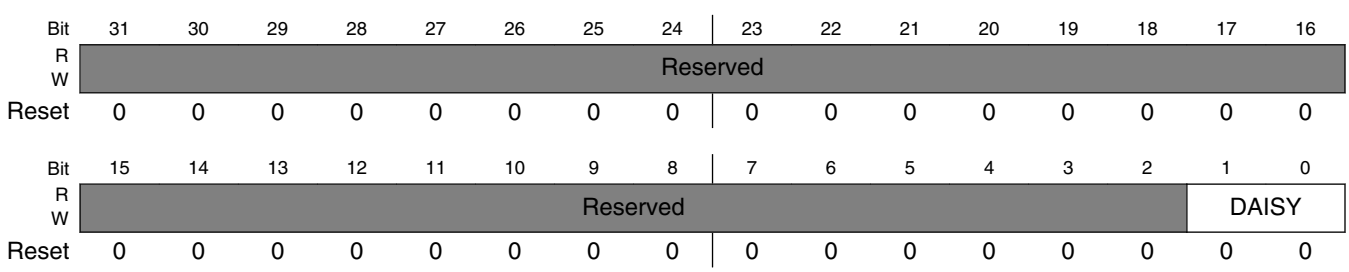


IOMUXC_PCIE1_CLKREQ_B_SELECT_INPUT field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **I2C4_SCL_ALT2** — Selecting ALT2 mode of pad I2C4_SCL for PCIE1_CLKREQ_B. 1 **UART4_RXD_ALT2** — Selecting ALT2 mode of pad UART4_RXD for PCIE1_CLKREQ_B.

8.2.5.326 Select Input Register (IOMUXC_SAI5_MCLK_SELECT_INPUT)

Address: 3033_0000h base + 52Ch offset = 3033_052Ch



IOMUXC_SAI5_MCLK_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **SAI5_MCLK_ALT0** — Selecting ALT0 mode of pad SAI5_MCLK for SAI5_MCLK. 01 **SAI1_MCLK_ALT1** — Selecting ALT1 mode of pad SAI1_MCLK for SAI5_MCLK. 10 **SAI2_MCLK_ALT1** — Selecting ALT1 mode of pad SAI2_MCLK for SAI5_MCLK. 11 **SAI3_MCLK_ALT2** — Selecting ALT2 mode of pad SAI3_MCLK for SAI5_MCLK.

8.2.5.327 Select Input Register (IOMUXC_SAI6_MCLK_SELECT_INPUT)

Address: 3033_0000h base + 530h offset = 3033_0530h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

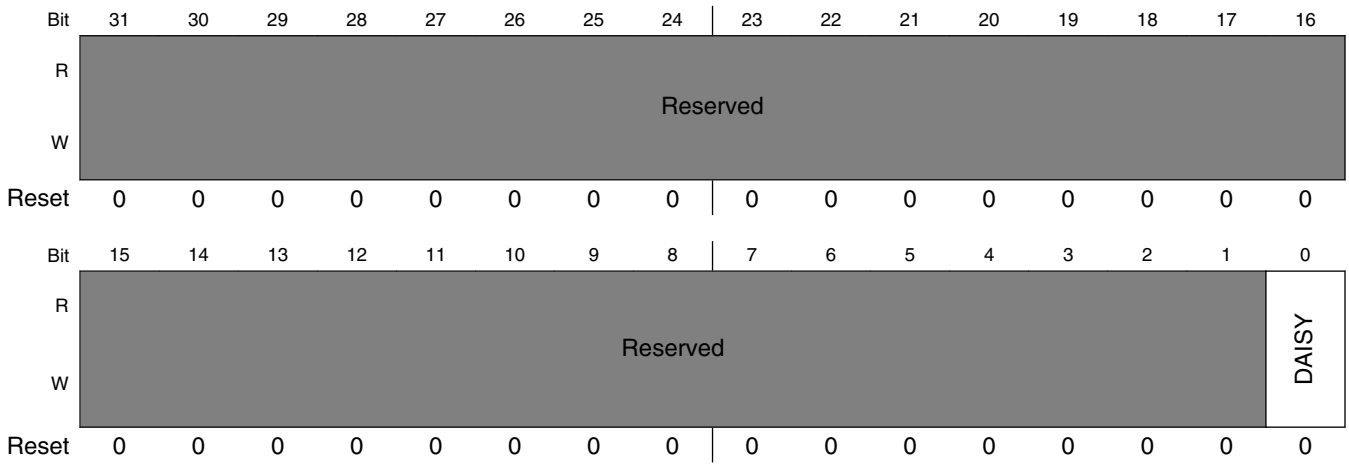
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															DAISY
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SAI6_MCLK_SELECT_INPUT field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **SAI1_RXD7_ALT1** — Selecting ALT1 mode of pad SAI1_RXD7 for SAI6_MCLK. 1 **SAI1_TXD7_ALT1** — Selecting ALT1 mode of pad SAI1_TXD7 for SAI6_MCLK.

8.2.5.328 Select Input Register (IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_0)

Address: 3033_0000h base + 534h offset = 3033_0534h

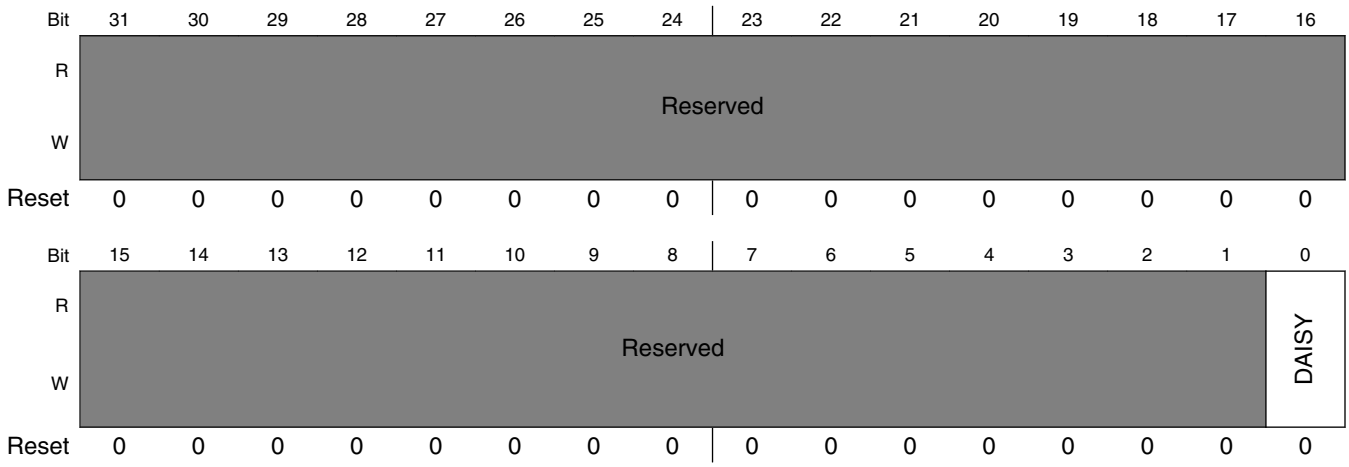


IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_0 field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **SAI5_RXD0_ALT4** — Selecting ALT4 mode of pad SAI5_RXD0 for PDM_BIT_STREAM_0. 1 **SAI1_RXD0_ALT3** — Selecting ALT3 mode of pad SAI1_RXD0 for PDM_BIT_STREAM_0.

8.2.5.329 Select Input Register (IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_1)

Address: 3033_0000h base + 538h offset = 3033_0538h

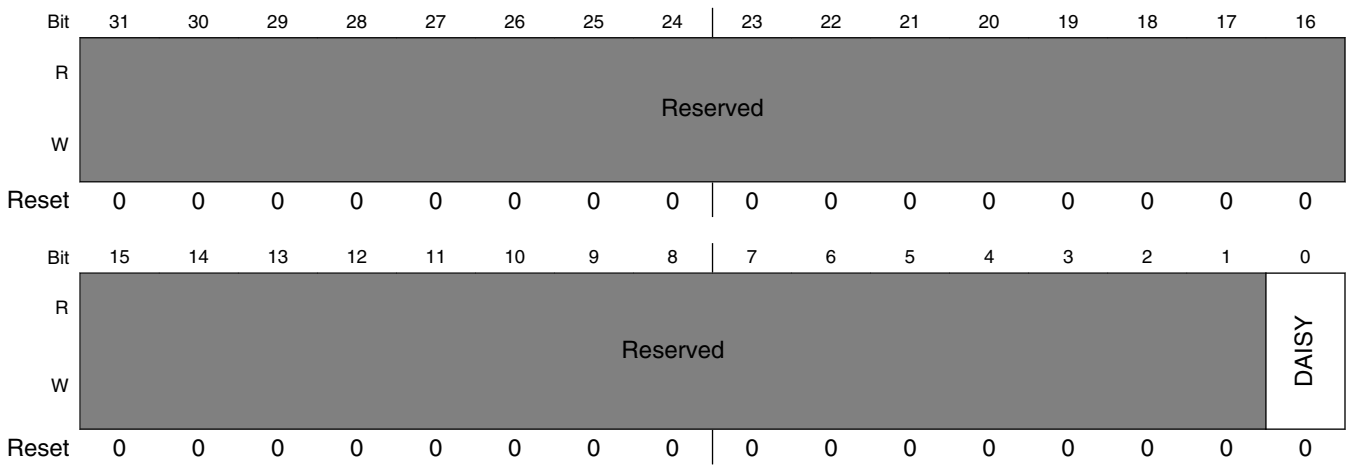


IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_1 field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **SAI5_RXD1_ALT4** — Selecting ALT4 mode of pad SAI5_RXD1 for PDM_BIT_STREAM_1. 1 **SAI1_RXD1_ALT3** — Selecting ALT3 mode of pad SAI1_RXD1 for PDM_BIT_STREAM_1.

8.2.5.330 Select Input Register (IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_2)

Address: 3033_0000h base + 53Ch offset = 3033_053Ch

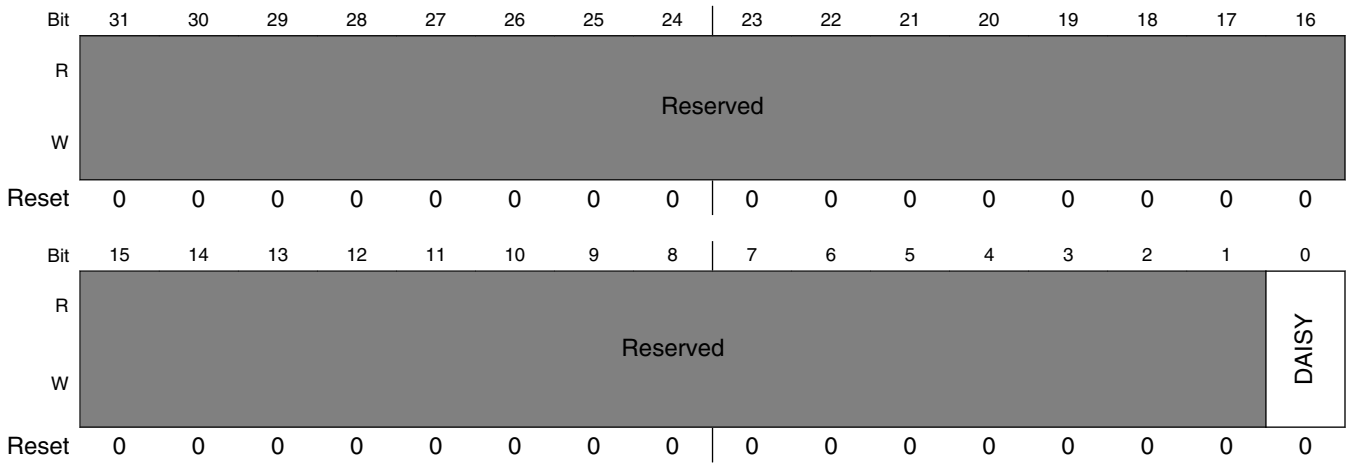


IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_2 field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **SAI5_RXD2_ALT4** — Selecting ALT4 mode of pad SAI5_RXD2 for PDM_BIT_STREAM_2. 1 **SAI1_RXD2_ALT3** — Selecting ALT3 mode of pad SAI1_RXD2 for PDM_BIT_STREAM_2.

8.2.5.331 Select Input Register (IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_3)

Address: 3033_0000h base + 540h offset = 3033_0540h

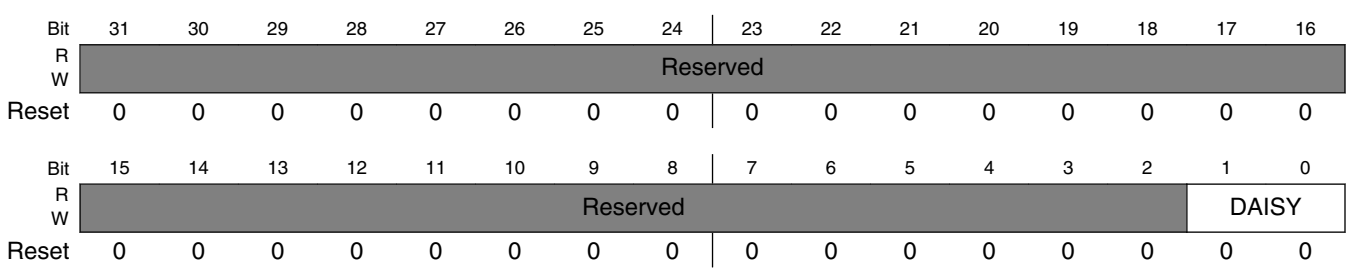


IOMUXC_PDM_BIT_STREAM_SELECT_INPUT_3 field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **SAI5_RXD3_ALT4** — Selecting ALT4 mode of pad SAI5_RXD3 for PDM_BIT_STREAM_3. 1 **SAI1_RXD3_ALT3** — Selecting ALT3 mode of pad SAI1_RXD3 for PDM_BIT_STREAM_3.

8.2.5.332 Select Input Register (IOMUXC_USDHC3_CD_B_SELECT_INPUT)

Address: 3033_0000h base + 544h offset = 3033_0544h



IOMUXC_USDHC3_CD_B_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **NAND_DATA02_ALT2** — Selecting ALT2 mode of pad NAND_DATA02 for USDHC3_CD_B. 01 **I2C2_SCL_ALT2** — Selecting ALT2 mode of pad I2C2_SCL for USDHC3_CD_B. 10 **GPIO1_IO14_ALT4** — Selecting ALT4 mode of pad GPIO1_IO14 for USDHC3_CD_B. 11 **NAND_DATA02_ALT2** — Same as 00

8.2.5.333 Select Input Register (IOMUXC_USDHC3_WP_SELECT_INPUT)

Address: 3033_0000h base + 548h offset = 3033_0548h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														DAISY	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_USDHC3_WP_SELECT_INPUT field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 00 **NAND_DATA03_ALT2** — Selecting ALT2 mode of pad NAND_DATA03 for USDHC3_WP. 01 **I2C2_SDA_ALT2** — Selecting ALT2 mode of pad I2C2_SDA for USDHC3_WP. 10 **GPIO1_IO15_ALT4** — Selecting ALT4 mode of pad GPIO1_IO15 for USDHC3_WP. 11 **NAND_DATA03_ALT2** — Same as 00

8.3 General Purpose Input/Output (GPIO)

8.3.1 Overview

The GPIO general-purpose input/output peripheral provides dedicated general-purpose pins that can be configured as either inputs or outputs.

When configured as an output, it is possible to write to an internal register to control the state driven on the output pin. When configured as an input, it is possible to detect the state of the input by reading the state of an internal register. In addition, the GPIO peripheral can produce CORE interrupts.

The GPIO is one of the blocks controlling the IOMUX of the chip.

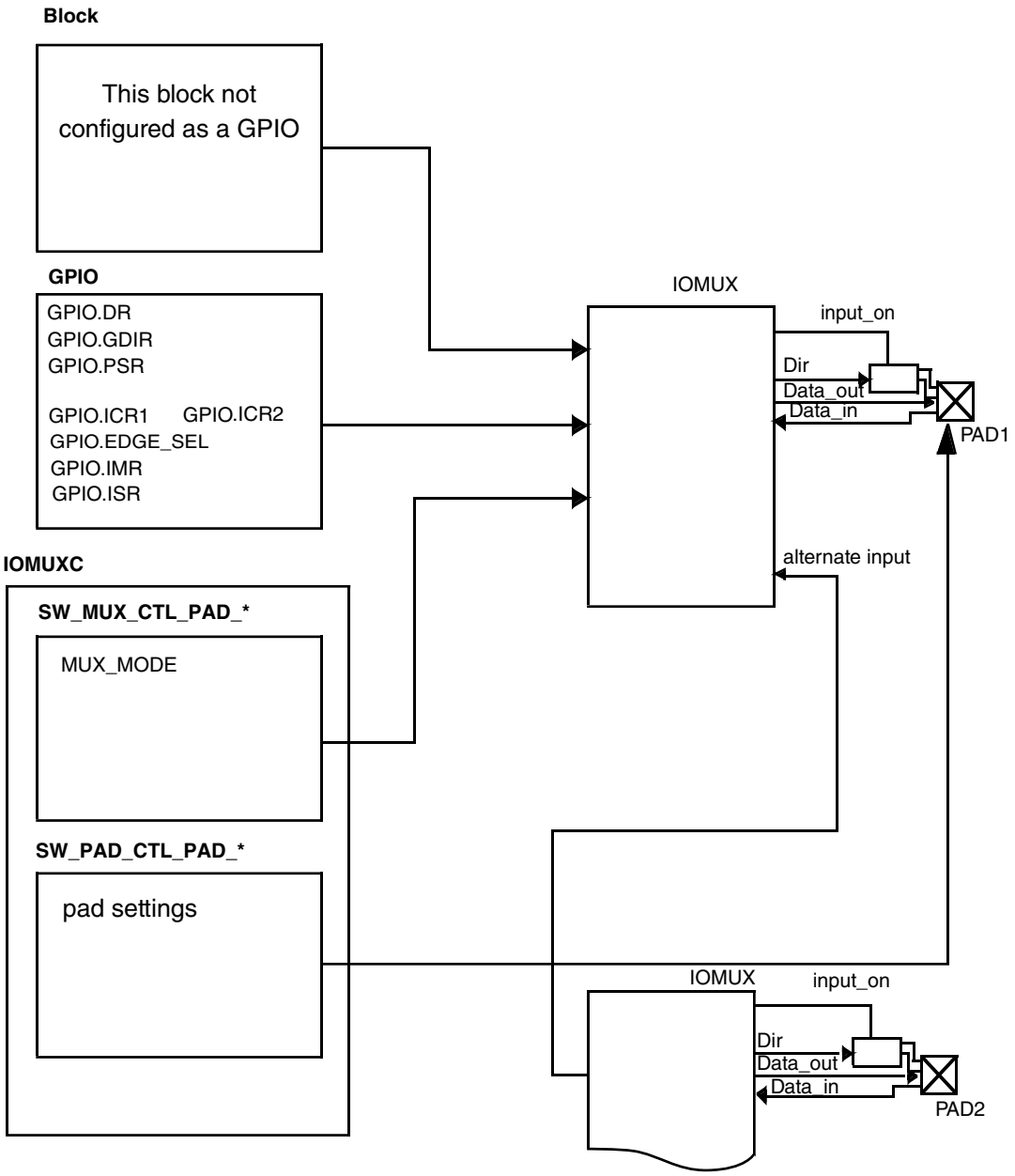


Figure 8-4. Chip IOMUX Scheme

The GPIO functionality is provided through eight registers, an edge-detect circuit, and interrupt generation logic.

The eight registers are:

- Data register (GPIO_DR)
- GPIO direction register (GPIO_GDIR)
- Pad sample register (GPIO_PSR)
- Interrupt control registers (GPIO_ICR1, GPIO_ICR2)

- Edge select register (GPIO_EDGE_SEL)
- Interrupt mask register (GPIO_IMR)
- Interrupt status register (GPIO_ISR)

These registers are described in detail in GPIO Memory Map/Register Definition.

Each GPIO input has a dedicated edge-detect circuit which can be configured through software to detect rising edges, falling edges, logic low-levels or logic high-levels on the input signals. The outputs of the edge detect circuits are optionally masked by setting the corresponding bit in the interrupt mask register (GPIO_IMR). These qualified outputs are OR'ed together to generate two one-bit interrupt lines:

- Combined interrupt indication for GPIOx signals 0 - 15
- Combined interrupt indication for GPIOx signals 16 - 31

In addition, GPIO1 provides visibility to each of its 8 low order interrupt sources (i.e. GPIO1 interrupt n, for n = 0 – 7). However, individual interrupt indications from other GPIOx are not available.

The GPIO edge detection is described further in Interrupt Control Unit.

The GPIO's overall functionality is described further in GPIO Functional Description.

8.3.1.1 Block Diagram

The GPIO subsystem contains multiple GPIO blocks, which can generate and control up to 32 signals for general purpose.

A block diagram of the GPIO is shown in Figure 8-5

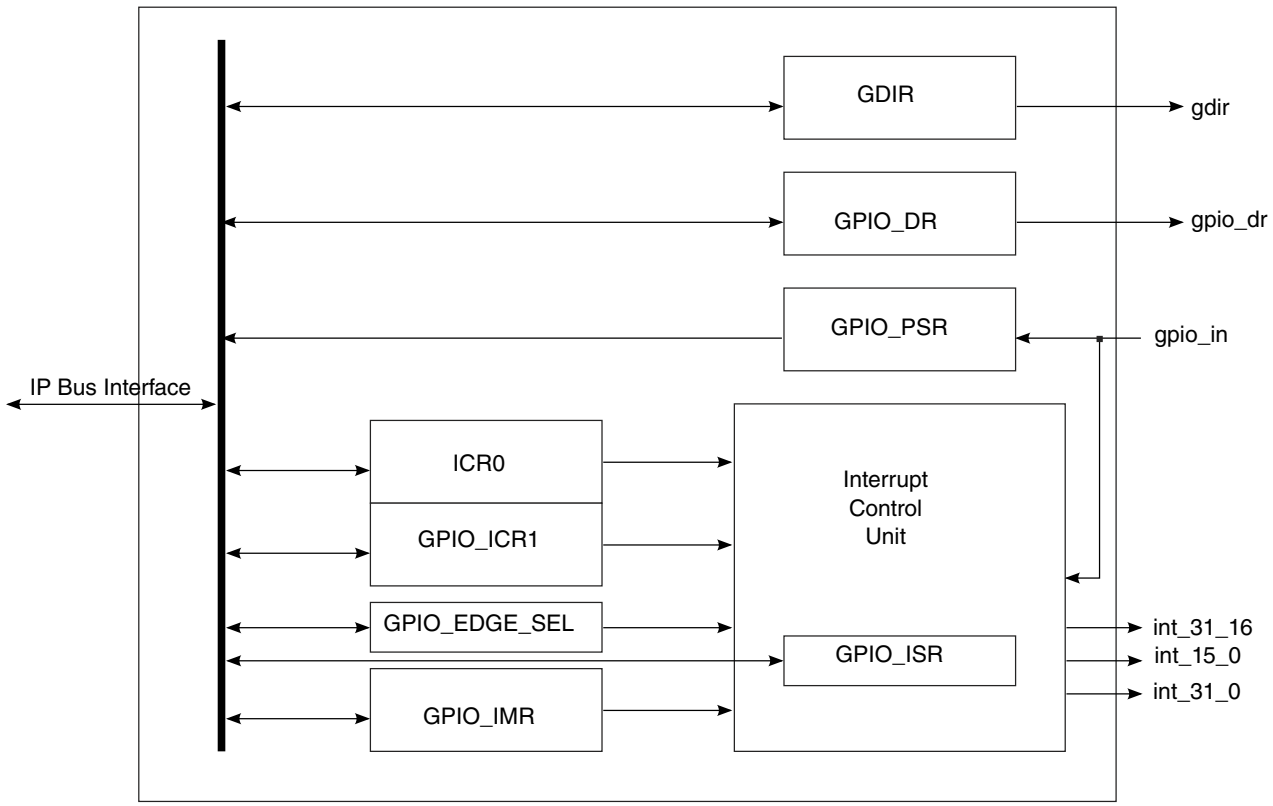


Figure 8-5. GPIO Block Diagram

8.3.1.2 Features

The GPIO includes the following features:

- General purpose input/output logic capabilities:
  - Drives specific data to output using the data register (GPIO_DR)
  - Controls the direction of the signal using the GPIO direction register (GPIO_GDIR)
  - Enables the core to sample the status of the corresponding inputs by reading the pad sample register (GPIO_PSR).
- GPIO interrupt capabilities:
  - Supports up to 32 interrupts
  - Identifies interrupt edges
  - Generates three active-high interrupts to the SoC interrupt controller

8.3.2 GPIO Functional Description

This section provides a complete functional description of the block.

8.3.2.1 GPIO Function

A GPIO signal can operate as a general-purpose input/output when the IOMUX is set to GPIO mode. Each GPIO signal may be independently configured as either an input or an output using the GPIO direction register (GPIO_GDIR).

When configured as an output (GPIO_GDIR bit = 1), the value in the data bit in the GPIO data register (GPIO_DR) is driven on the corresponding GPIO line. When a signal is configured as an input (GPIO_GDIR bit = 0), the state of the input can be read from the corresponding GPIO_PSR bit.

8.3.2.2 GPIO Programming

8.3.2.2.1 GPIO Read Mode

The programming sequence for reading input signals should be as follows:

1. Configure IOMUX to select GPIO mode (Via IOMUX Controller (IOMUXC)).
2. Configure GPIO direction register to input (GPIO_GDIR[GDIR] set to 0b).
3. Read value from data reg*ister*/pad status register.

A pseudocode description to read [input3:input0] values is as follows:

```
// SET INPUTS TO GPIO MODE.
write sw_mux_ctl_<input0>_<input1>_<input2>_<input3>, 32'h00000000
// SET GDIR TO INPUT.
write GDIR[31:4,input3_bit, input2_bit, input1_bit, input0_bit,] 32'hxxxxxxx0
// READ INPUT VALUE FROM DR.
read DR
// READ INPUT VALUE FROM PSR.
read PSR
```

NOTE

While the GPIO direction is set to input (GPIO_GDIR = 0), a read access to GPIO_DR does not return GPIO_DR data. Instead, it returns the GPIO_PSR data, which is the corresponding input signal value.

8.3.2.2.2 GPIO Write Mode

The programming sequence for driving output signals should be as follows:

1. Configure IOMUX to select GPIO mode (Via IOMUXC), also enable SION if need to read loopback pad value through PSR
2. Configure GPIO direction register to output (GPIO_GDIR[GDIR] set to 1b).

3. Write value to data register (GPIO_DR).

A pseudocode description to drive 4'b0101 on [output3:output0] is as follows:

```
// SET PADS TO GPIO MODE VIA IOMUX.
write sw_mux_ctl_pad_<output[0-3]>.mux_mode, <GPIO_MUX_MODE>
// Enable loopback so we can capture pad value into PSR in output mode
write sw_mux_ctl_pad_<output[0-3]>.sion, 1
// SET GDIR=1 TO OUTPUT BITS.
write GDIR[31:4,output3_bit,output2_bit, output1_bit, output0_bit,] 32'hxxxxxxxF
// WRITE OUTPUT VALUE=4'b0101 TO DR.
write DR, 32'hxxxxxxx5
// READ OUTPUT VALUE FROM PSR ONLY.
read_cmp PSR, 32'hxxxxxxx5
```

8.3.2.3 Interrupt Control Unit

In addition to the general-purpose input/output function, the edge-detect logic in the GPIO peripheral reflects whether a transition has occurred on a given GPIO signal that is configured as an input (GDIR bit = 0). The interrupt control registers (GPIO_ICR1 and GPIO_ICR2) may be used to independently configure the interrupt condition of each input signal (low-to-high transition, high-to-low transition, low, or high). For information about GPIO_ICR1 and GPIO_ICR2 settings, see GPIO Memory Map/Register Definition.

The interrupt control unit is built of 32 interrupt control subunits, where each subunit handles a single interrupt line.

8.3.3 GPIO Memory Map/Register Definition

There are eight 32-bit GPIO registers. All registers are accessible from the IP interface. Only 32-bit access is supported.

The GPIO memory map is shown in the following table.

GPIO memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3020_0000	GPIO data register (GPIO1_DR)	32	R/W	0000_0000h	8.3.3.1/ 1660
3020_0004	GPIO direction register (GPIO1_GDIR)	32	R/W	0000_0000h	8.3.3.2/ 1661

Table continues on the next page...

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3020_0008	GPIO pad status register (GPIO1_PSR)	32	R	0000_0000h	8.3.3.3/ 1662
3020_000C	GPIO interrupt configuration register1 (GPIO1_ICR1)	32	R/W	0000_0000h	8.3.3.4/ 1662
3020_0010	GPIO interrupt configuration register2 (GPIO1_ICR2)	32	R/W	0000_0000h	8.3.3.5/ 1666
3020_0014	GPIO interrupt mask register (GPIO1_IMR)	32	R/W	0000_0000h	8.3.3.6/ 1670
3020_0018	GPIO interrupt status register (GPIO1_ISR)	32	w1c	0000_0000h	8.3.3.7/ 1670
3020_001C	GPIO edge select register (GPIO1_EDGE_SEL)	32	R/W	0000_0000h	8.3.3.8/ 1671
3021_0000	GPIO data register (GPIO2_DR)	32	R/W	0000_0000h	8.3.3.1/ 1660
3021_0004	GPIO direction register (GPIO2_GDIR)	32	R/W	0000_0000h	8.3.3.2/ 1661
3021_0008	GPIO pad status register (GPIO2_PSR)	32	R	0000_0000h	8.3.3.3/ 1662
3021_000C	GPIO interrupt configuration register1 (GPIO2_ICR1)	32	R/W	0000_0000h	8.3.3.4/ 1662
3021_0010	GPIO interrupt configuration register2 (GPIO2_ICR2)	32	R/W	0000_0000h	8.3.3.5/ 1666
3021_0014	GPIO interrupt mask register (GPIO2_IMR)	32	R/W	0000_0000h	8.3.3.6/ 1670
3021_0018	GPIO interrupt status register (GPIO2_ISR)	32	w1c	0000_0000h	8.3.3.7/ 1670
3021_001C	GPIO edge select register (GPIO2_EDGE_SEL)	32	R/W	0000_0000h	8.3.3.8/ 1671
3022_0000	GPIO data register (GPIO3_DR)	32	R/W	0000_0000h	8.3.3.1/ 1660
3022_0004	GPIO direction register (GPIO3_GDIR)	32	R/W	0000_0000h	8.3.3.2/ 1661
3022_0008	GPIO pad status register (GPIO3_PSR)	32	R	0000_0000h	8.3.3.3/ 1662
3022_000C	GPIO interrupt configuration register1 (GPIO3_ICR1)	32	R/W	0000_0000h	8.3.3.4/ 1662
3022_0010	GPIO interrupt configuration register2 (GPIO3_ICR2)	32	R/W	0000_0000h	8.3.3.5/ 1666
3022_0014	GPIO interrupt mask register (GPIO3_IMR)	32	R/W	0000_0000h	8.3.3.6/ 1670
3022_0018	GPIO interrupt status register (GPIO3_ISR)	32	w1c	0000_0000h	8.3.3.7/ 1670
3022_001C	GPIO edge select register (GPIO3_EDGE_SEL)	32	R/W	0000_0000h	8.3.3.8/ 1671

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3023_0000	GPIO data register (GPIO4_DR)	32	R/W	0000_0000h	8.3.3.1/ 1660
3023_0004	GPIO direction register (GPIO4_GDIR)	32	R/W	0000_0000h	8.3.3.2/ 1661
3023_0008	GPIO pad status register (GPIO4_PSR)	32	R	0000_0000h	8.3.3.3/ 1662
3023_000C	GPIO interrupt configuration register1 (GPIO4_ICR1)	32	R/W	0000_0000h	8.3.3.4/ 1662
3023_0010	GPIO interrupt configuration register2 (GPIO4_ICR2)	32	R/W	0000_0000h	8.3.3.5/ 1666
3023_0014	GPIO interrupt mask register (GPIO4_IMR)	32	R/W	0000_0000h	8.3.3.6/ 1670
3023_0018	GPIO interrupt status register (GPIO4_ISR)	32	w1c	0000_0000h	8.3.3.7/ 1670
3023_001C	GPIO edge select register (GPIO4_EDGE_SEL)	32	R/W	0000_0000h	8.3.3.8/ 1671
3024_0000	GPIO data register (GPIO5_DR)	32	R/W	0000_0000h	8.3.3.1/ 1660
3024_0004	GPIO direction register (GPIO5_GDIR)	32	R/W	0000_0000h	8.3.3.2/ 1661
3024_0008	GPIO pad status register (GPIO5_PSR)	32	R	0000_0000h	8.3.3.3/ 1662
3024_000C	GPIO interrupt configuration register1 (GPIO5_ICR1)	32	R/W	0000_0000h	8.3.3.4/ 1662
3024_0010	GPIO interrupt configuration register2 (GPIO5_ICR2)	32	R/W	0000_0000h	8.3.3.5/ 1666
3024_0014	GPIO interrupt mask register (GPIO5_IMR)	32	R/W	0000_0000h	8.3.3.6/ 1670
3024_0018	GPIO interrupt status register (GPIO5_ISR)	32	w1c	0000_0000h	8.3.3.7/ 1670
3024_001C	GPIO edge select register (GPIO5_EDGE_SEL)	32	R/W	0000_0000h	8.3.3.8/ 1671

8.3.3.1 GPIO data register (GPIOx_DR)

The 32-bit GPIO_DR register stores data that is ready to be driven to the output lines. If the IOMUXC is in GPIO mode and a given GPIO direction bit is set, then the corresponding DR bit is driven to the output. If a given GPIO direction bit is cleared, then a read of GPIO_DR reflects the value of the corresponding signal.Two wait states are required in read access for synchronization.

The results of a read of a DR bit depends on the IOMUXC input mode settings and the corresponding GDIR bit as follows:

- If GDIR[*n*] is set and IOMUXC input mode is GPIO, then reading DR[*n*] returns the contents of DR[*n*].
- If GDIR[*n*] is cleared and IOMUXC input mode is GPIO, then reading DR[*n*] returns the corresponding input signal's value.
- If GDIR[*n*] is set and IOMUXC input mode is not GPIO, then reading DR[*n*] returns the contents of DR[*n*].
- If GDIR[*n*] is cleared and IOMUXC input mode is not GPIO, then reading DR[*n*] always returns zero.

Address: Base address + 0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPIOx_DR field descriptions

Field	Description
DR	Data bits. This register defines the value of the GPIO output when the signal is configured as an output (GDIR[n]=1). Writes to this register are stored in a register. Reading GPIO_DR returns the value stored in the register if the signal is configured as an output (GDIR[n]=1), or the input signal's value if configured as an input (GDIR[n]=0). **NOTE:** The I/O multiplexer must be configured to GPIO mode for the GPIO_DR value to connect with the signal. Reading the data register with the input path disabled always returns a zero value.

8.3.3.2 GPIO direction register (GPIOx_GDIR)

GPIO_GDIR functions as direction control when the IOMUXC is in GPIO mode. Each bit specifies the direction of a one-bit signal. The mapping of each DIR bit to a corresponding SoC signal is determined by the SoC's pin assignment and the IOMUX table. For more details consult the IOMUXC chapter.

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GDIR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPIOx_GDIR field descriptions

Field	Description
GDIR	GPIO direction bits. Bit n of this register defines the direction of the GPIO[n] signal. **NOTE:** GPIO_GDIR affects only the direction of the I/O signal when the corresponding bit in the I/O MUX is configured for GPIO.

GPIOx_GDIR field descriptions (continued)

Field	Description
	0 **INPUT** — GPIO is configured as input. 1 **OUTPUT** — GPIO is configured as output.

8.3.3.3 GPIO pad status register (GPIOx_PSR)

GPIO_PSR is a read-only register. Each bit stores the value of the corresponding input signal (as configured in the IOMUX). This register is clocked with the ipg_clk_s clock, meaning that the input signal is sampled only when accessing this location. Two wait states are required any time this register is accessed for synchronization.

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PSR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPIOx_PSR field descriptions

Field	Description
PSR	GPIO pad status bits (status bits). Reading GPIO_PSR returns the state of the corresponding input signal. Settings: **NOTE:** The IOMUXC must be configured to GPIO mode for GPIO_PSR to reflect the state of the corresponding signal.

8.3.3.4 GPIO interrupt configuration register1 (GPIOx_ICR1)

GPIO_ICR1 contains 16 two-bit fields, where each field specifies the interrupt configuration for a different input signal.

Address: Base address + Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ICR15		ICR14		ICR13		ICR12		ICR11		ICR10		ICR9		ICR8	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ICR7		ICR6		ICR5		ICR4		ICR3		ICR2		ICR1		ICR0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPIOx_ICR1 field descriptions

Field	Description
31–30 ICR15	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 15. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
29–28 ICR14	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 14. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
27–26 ICR13	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 13. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
25–24 ICR12	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 12. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
23–22 ICR11	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 11. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.

Table continues on the next page...

GPIOx_ICR1 field descriptions (continued)

Field	Description
21–20 ICR10	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 10. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
19–18 ICR9	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 9. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
17–16 ICR8	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 8. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
15–14 ICR7	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 7. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
13–12 ICR6	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 6. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.

Table continues on the next page...

GPIOx_ICR1 field descriptions (continued)

Field	Description
11–10 ICR5	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 5. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
9–8 ICR4	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 4. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
7–6 ICR3	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 3. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
5–4 ICR2	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 2. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
3–2 ICR1	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 1. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.

Table continues on the next page...

GPIOx_ICR1 field descriptions (continued)

Field	Description
ICR0	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 0. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.

8.3.3.5 GPIO interrupt configuration register2 (GPIOx_ICR2)

GPIO_ICR2 contains 16 two-bit fields, where each field specifies the interrupt configuration for a different input signal.

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ICR31		ICR30		ICR29		ICR28		ICR27		ICR26		ICR25		ICR24	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ICR23		ICR22		ICR21		ICR20		ICR19		ICR18		ICR17		ICR16	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPIOx_ICR2 field descriptions

Field	Description
31–30 ICR31	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 31. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
29–28 ICR30	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 30. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive.

Table continues on the next page...

GPIOx_ICR2 field descriptions (continued)

Field	Description
	01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
27–26 ICR29	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 29. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
25–24 ICR28	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 28. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
23–22 ICR27	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 27. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
21–20 ICR26	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 26. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
19–18 ICR25	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 25. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive.

Table continues on the next page...

GPIOx_ICR2 field descriptions (continued)

Field	Description
	10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
17–16 ICR24	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 24. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
15–14 ICR23	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 23. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
13–12 ICR22	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 22. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
11–10 ICR21	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 21. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
9–8 ICR20	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 20. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive.

Table continues on the next page...

GPIOx_ICR2 field descriptions (continued)

Field	Description
	10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
7–6 ICR19	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 19. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
5–4 ICR18	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 18. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
3–2 ICR17	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 17. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.
ICR16	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 16. Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows: 00 **LOW_LEVEL** — Interrupt n is low-level sensitive. 01 **HIGH_LEVEL** — Interrupt n is high-level sensitive. 10 **RISING_EDGE** — Interrupt n is rising-edge sensitive. 11 **FALLING_EDGE** — Interrupt n is falling-edge sensitive.

8.3.3.6 GPIO interrupt mask register (GPIOx_IMR)

GPIO_IMR contains masking bits for each interrupt line.

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPIOx_IMR field descriptions

Field	Description
IMR	Interrupt Mask bits. This register is used to enable or disable the interrupt function on each of the 32 GPIO signals. Settings: Bit IMR[n] (n=0...31) controls interrupt n as follows: 0 **MASKED** — Interrupt n is disabled. 1 **UNMASKED** — Interrupt n is enabled.

8.3.3.7 GPIO interrupt status register (GPIOx_ISR)

The GPIO_ISR functions as an interrupt status indicator. Each bit indicates whether an interrupt condition has been met for the corresponding input signal. When an interrupt condition is met (as determined by the corresponding interrupt condition register field), the corresponding bit in this register is set. Two wait states are required in read access for synchronization. One wait state is required for reset.

Address: Base address + 18h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPIOx_ISR field descriptions

Field	Description
ISR	Interrupt status bits - Bit n of this register is asserted (active high) when the active condition (as determined by the corresponding ICR bit) is detected on the GPIO input and is waiting for service. The value of this register is independent of the value in GPIO_IMR.

GPIOx_ISR field descriptions (continued)

Field	Description
	When the active condition has been detected, the corresponding bit remains set until cleared by software. Status flags are cleared by writing a 1 to the corresponding bit position.

8.3.3.8 GPIO edge select register (GPIOx_EDGE_SEL)

GPIO_EDGE_SEL may be used to override the ICR registers' configuration. If the GPIO_EDGE_SEL bit is set, then a rising edge or falling edge in the corresponding signal generates an interrupt. This register provides backward compatibility. On reset all bits are cleared (ICR is not overridden).

Address: Base address + 1Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPIO_EDGE_SEL																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPIOx_EDGE_SEL field descriptions

Field	Description
GPIO_EDGE_SEL	Edge select. When GPIO_EDGE_SEL[*n*] is set, the GPIO disregards the ICR[*n*] setting, and detects any edge on the corresponding input signal.

Chapter 9
External Memory

9.1 External Memory Overview

9.1.1 External Memory Overview

9.1.1.1 DRAM Interface

The chip has a single 16/32-bit DRAM controller.

The DRAM controller and PHY are licensed from SNPS. The key features of the DRAM controller and PHY include:

- LPDDR4-3000, DDR4-2400, and DDR3L-1600 in Non-POP BGA package
- 32/16-bit DRAM interface
- 750MHz 128-bit AXI bus
- Up to 8GB of memory capacity
- Support various low power modes, clock and power gated operation, that are defined for i.MX 8M Mini. In addition, it shall be able to place the external DRAM into and out of its self-refresh mode as requested by different low-power operating modes

9.1.1.2 eSD/eMMC/SDIO Support

The chip has three Ultra Secured Digital Host Controller (uSDHC) modules for the SD/eMMC interface. It provides the interface between the host system and SD/SDIO/MMC cards. The key features include:

- Support SD/SDIO standard, up to version 3.0
- Support MMC standard, up to version 5.0
- Support 3.3V and 1.8V operation, but do not support 1.2V operation
- Supports 1-bit / 4-bit SD and SDIO modes, 1-bit / 4-bit / 8-bit MMC modes

- Up to 400 Mbps of data transfer for SDIO cards using 4 parallel data lines in SDR mode
- Up to 800 Mbps of data transfer for SDIO card using 4 parallel data lines in DDR mode
- Up to 400 Mbps of data transfer for SDXC cards using 4 parallel data lines in SDR mode
- Up to 800 Mbps of data transfer for SDXC card using 4 parallel data lines in DDR mode
- Up to 1600 Mbps of data transfer for MMC cards using 8 parallel data lines in SDR mode
- Up to 3200 Mbps of data transfer for MMC cards using 8 parallel data lines in DDR mode

- Two uSDHC controllers (SD1 and SD3) can support up to 8-bit interface, the other controller (SD2) can only support up to 4-bit interface.

9.1.1.3 Raw NAND Support

The Raw NAND Flash controller consists of three components:

- GPMI as the NAND controller pin interface
- APBH_DMA as the DMA engine that drives the GPMI module
- BCH as the 62-bit error correction hardware engine with an AXI bus master and a private connection to GPMI.

The Raw NAND Flash controller support following key features:

- 8-bit NAND FLASH, up to 4 devices supported by 4 chip-selects and 1 ganged ready/busy
- ONFI 2.x complaint, synchronous clock rate of up to 100 MHz with data rate of up to 200 MB/s
- Support ONFI NAND for Micron and Hynix and Toggle NAND for Toshiba and Samsung
- BCH62 for ECC, up to 200MB/s

9.1.1.4 Quad SPI Interface

There is one Quad Serial Peripheral Interface (QuadSPI) block that acts as an interface to one or two external serial flash devices, each with up to four bidirectional data lines. The key features include:

- Each channel can be configured as 1/2/4-bit operation
- Support both dual-channel or single-channel operation
- Support both SDR mode and DDR mode

- Support up to 166MHz SDR Mode and 166MHz DDR Mode (with external Flash device DQS input)
- Support up to 133MHz SDR Mode and 66MHz DDR Mode (with internal DQS loopback mode)

9.2 DDR Controller (DDRC)

9.2.1 Overview

The DDRC (DDR Controller) is a very low power, high efficiency, low-latency and high performance memory controller for interfacing DDR based memories.

The following types of SDRAMs are supported by the DDRC:

- DDR4
- DDR3-L
- LP-DDR4

9.2.1.1 Block diagram

The following shows the block diagram for the DDR Controller.

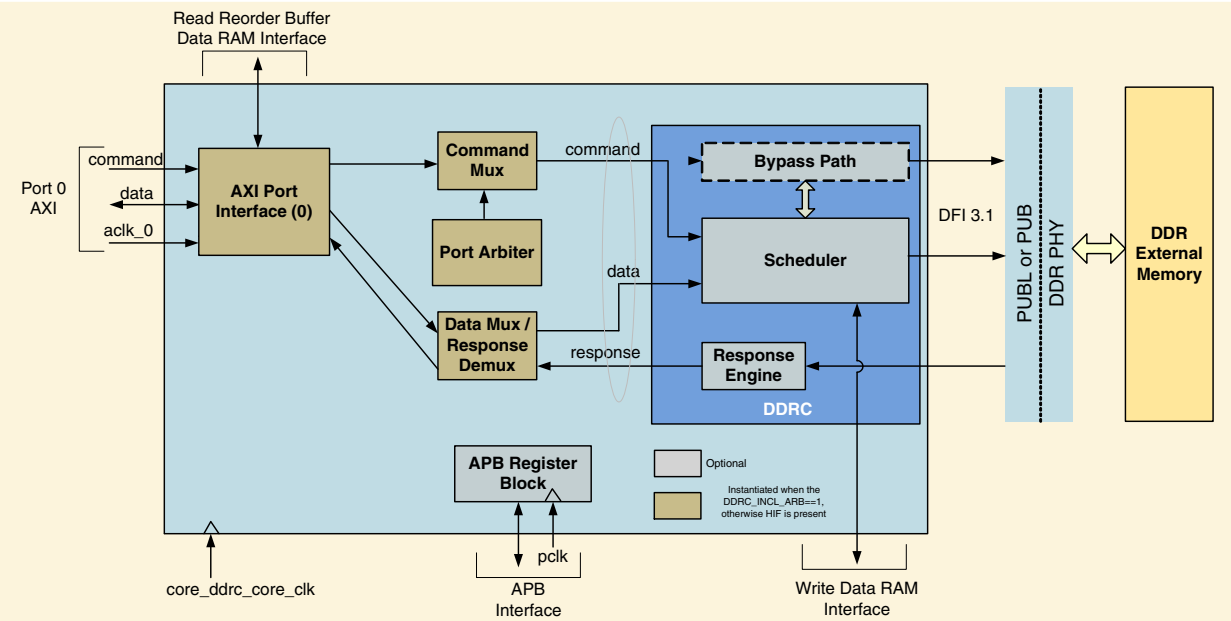


Figure 9-1. DDRC block diagram

The PUB /PUBL is the PHY Utility Block is an adaption layer between the controller and the PHY handling control function.

9.2.1.2 Features

The DDRC includes the following features:

- Direct software request control or programmable internal control for ZQ short calibration cycles
- Support for ZQ long calibration after self-refresh exit
- Direct software request control or programmable internal control for ZQ calibration cycles
- Support for ZQ Reset feature through software
- Dynamic scheduling to optimize bandwidth and latency
- Read and write buffers in fully associative CAMs, 32 each
- Delayed writes for optimum performance on SDRAM data bus
- Out of order execution of commands maximises the SDRAM efficiency
- Hardware configurable and software programmable Quality of Service (QoS) support:

    - Support for three traffic classes on read commands—high priority reads, variable priority reads and low priority reads
    - Support for two traffic classes on write commands—normal priority writes and variable priority writes
    - Support for port urgent and port throttling control
- Programmable SDRAM parameters
- Supports BL16 burst length
- Supports 1,2 memory ranks
- Control options to avoid starvation of lower priorities
- Guaranteed coherency for write-after-read (WAR) and read-after-write (RAW) hazards
- Write combine to allow multiple writes to the same address to be combined into a single write to SDRAM; supported for same starting address
- Paging policy selectable by configuration registers as any of the following:
    - Leave pages open after accesses, or
    - Close page when there are no further accesses available in the controller for that page, or
    - Auto-precharge with each access, with an optimization for page-close mode which leaves the page open after a flush for read-write and write-read collision cases
- Supports automatic SDRAM power-down entry and exit caused by lack of transaction arrival for programmable time
- Supports automatic Clock Stop entry and exit caused by lack of transaction arrival
- Supports automatic DDRC low power mode operation caused by lack of transaction arrival for programmable time via Hardware Low Power Interface
- Supports self-refresh entry and exit as follows:
    - Supports automatic self-refresh entry and exit caused by lack of transaction arrival for programmable time
    - Support for self-refresh entry and exit under software control
    - Support for self-refresh entry and exit using dedicated DDRC hardware low power interface control (similar to the AMBA 3 AXI protocol low power control interface)
- Support for dynamically changing clock frequency while in self-refresh:
    - DDR4 DLL-off mode supported
    - Shadow timing registers provided to allow fast frequency changing
- Support for explicit SDRAM mode register updates under software control
- Flexible address mapper logic to allow application specific mapping of row, column, bank, and rank bits
- Programmable support for 1T or 2T timing
- User-selectable refresh control options:
    - Controller-generated auto-refreshes at programmable average intervals

    - In multi-rank designs, an offset can be applied to each rank's refresh timer to allow rank refreshes to expire at different times (this can increase efficiency by allowing traffic to continue to other ranks while a given rank is being refreshed)
    - Ability to group up to 8 controller-generated refreshes together to be issued consecutively (this reduces the frequency of page closings, increasing overall efficiency)
    - When controller-generated refreshes are grouped, some refreshes can be issued speculatively when the controller is idle for a programmable period of time
    - Ability to disable controller-generated auto-refreshes
    - Ability to issue a refresh through direct software request
    - When LP-DDR4 is used, user-selectable ability to perform per-bank refreshes rather than all-banks refreshes
    - When DDR4 is used, fine granularity refresh can be selected
- DDR4 protocol up to DDR4-3200 speed grade (1:2 frequency ratio mode) and DDR4-2133 speed grade (1:1 frequency ratio mode) are supported by the DDRC. The following features are supported:
    - Data Bus Inversion (DBI)
    - Geardown mode may be programmed where allowed by memory and PHY specification (not supported for 1:1 frequency ratio mode)
    - Maximum power saving mode
    - Multi-purpose register (MPR) reads and writes
    - Per DRAM addressability
    - Fine granularity refresh
    - Cyclic redundancy check (CRC) with retry (not supported for 1:1 frequency ratio mode)
    - Command/address latency
    - Programmable Preamble (not supported for 1:1 frequency ratio mode)
    - 3DS with up to 8 logical ranks/DDR4-2666
- Support for and DDR3L
- Supports Dual Data Channel, also called "Shared AC", with a common address and command bus
- DDR3-L support with ECC

9.2.2 Functional description

The DDRC converts system bus transactions into memory commands on the DFI interface that are compliant with the DDR protocols. The DDRC supports both memory components and DIMMs(Dual in-line Memory Module).

Please see the block diagram at Block diagram.

The main components of the block are:

- **AXI Port Interface (XPI) block**: This block provides the interface to the application ports. It provides bus protocol handling, data buffering and reordering for read data, data bus size conversion (upsizing or downsizing), and memory burst address alignment. Read data is stored in a SRAM, read re-order buffer and returned in order, to the AXI ports. The SRAM may be instantiated as embedded memory external to the DDRC or be implemented as flops within the DDRC.
- **Port Arbiter (PA) block**: This block provides latency sensitive, priority based arbitration between the addresses issued by the XPIs (by the ports).
- **DDR Controller (DDRC) block**: This block contains a logical CAM (Content Addressable Memory), which can be synthesized using standard cells. This holds information on the commands, which is used by the scheduling algorithms to optimally schedule commands to be sent to the PHY, based on priority, bank/rank status and DDR timing constraints. A bypass path is also provided (optionally).
- **APB Register Block**: This block contains the software accessible registers.

The details on DDRC are as follows:

- Write data is stored in DDRC until its associated command is issued to the PHY.
- Read data is handled by the response engine in the DDRC and is returned in the order of scheduled read commands on the HIF.
- ECC handling is an optional function which is handled by logic modules within the DDRC in the write data path and in the response engine.

1:1 and 1:2 frequency ratios are terms used for the frequency ratio between the DDRC clock (core_ddrc_core_clk) and the memory clock.

When MEMC_PROG_FREQ_RATIO = 1 the parameter MEMC_FREQ_RATIO is tied to 2 and the HIF data bus is four times the memory data width.

The register MSTR.frequency_ratio controls whether the controller operates in 1:1 frequency ratio mode or 1:2 frequency ratio mode. When MSTR.frequency_ratio is programmed to 1 the DDRC clock rate is the same as the memory clock rate (referred to as SDR, single data rate) i.e. 1:1 frequency ratio mode. When MSTR.frequency_ratio is programmed to 0 the DDRC clock rate is one half the memory clock rate (referred to as HDR, half data rate) i.e. 1:2 frequency ratio mode.

When MEMC_PROG_FREQ_RATIO = 0 the parameter MEMC_FREQ_RATIO controls whether the controller operates in 1:1 frequency ratio mode or 1:2 frequency ratio mode. When MEMC_FREQ_RATIO=1 the DDRC clock rate is the same as the memory clock rate (referred to as SDR, single data rate) i.e. 1:1 frequency ratio mode.

The HIF data bus is twice the memory data width. When MEMC_FREQ_RATIO=2 the DDRC clock rate is one half the memory clock rate (referred to as HDR, half data rate), that is, 1:2 frequency ratio mode. The HIF data bus is four times the memory data width

In the case of 1:1 frequency ratio mode, the DDRC and PHY operate at the same clock frequency and are referred to as a ‘matched frequency system’ in the DFI Specification.

For example if MEMC_PROG_FREQ_RATIO=1, in 1:1 frequency ratio mode, if the memory data width is 32 bits and the memory clock rate is 400 MHz or 800 Mbps, the HIF data bus would be 128 bits wide and only the lower half of the dfi bus is utilized. The DDRC clock rate would be 400 MHz. If the frequency ratio mode is changed to 1:2, the HIF data bus would be 128 bits wide and all of the dfi data bus is utilized. The DDRC clock rate would be 200 MHz.

For example if MEMC_PROG_FREQ_RATIO=0, in 1:1 frequency mode, if the memory data width is 32 bits and the memory clock rate is 400 MHz or 800 Mbps, the HIF data bus would be 64 bits wide and all of the dfi data bus is utilized. The DDRC clock rate would be 400 MHz. If the frequency mode is changed to 1:2, the HIF data bus would be 128 bits wide and all of the dfi data bus is utilized. The DDRC clock rate would be 200 MHz.

The terms “DFI clock” and “DFI PHY clock” are used in this document in accordance with the DFI specification:

- The DFI clock is the clock on which the DFI interface runs. It is the core_ddrc_core_clk.
- The DFI PHY clock is the SDRAM clock. In 1:1 frequency ratio, this has the same frequency as the DFI clock. In 1:2 frequency ratio, this has twice the frequency of the DFI clock.

9.2.2.1 AXI Port Interface (XPI)

The AXI protocol is burst-based with read and write request channels that specify the host ID for the request, start byte address, burst length, burst size, and burst type. This information is processed by the interface and is used subsequently by the DDRC.

The XPI interfaces the AXI application port to the DDRC and performs the following main functions:

- Read address generation
- Write address generation
- Read data generation

- Read data and response generation
- Write response generation

The XPI converts AXI bursts into read and write requests, which are forwarded to the Port Arbiter (PA). In the opposite direction, the XPI converts the responses from the DDRC into appropriate AXI responses.

The interface between XPI and PA is same as HIF (but with independent channels for read and write commands). Each AXI port can be independently configured to operate with a clock that is synchronous or asynchronous to the memory controller clock.

The AXI Specification requires that transactions must not cross a 4K address boundary. AXI compliant masters must meet this requirement. However, if a master does not comply with this AXI protocol requirement, the user can relax this restriction in the DDRC and set the boundary between 4K and 4G using the hardware parameter DDRC_AXI_ADDR_BOUNDARY.

9.2.2.1.1 Read Address Channel

The AXI read address channel has the following features:

- The read transaction can be of any length up to 256 for incremental bursts, 16 for wrapping bursts.
- The burst types supported are incremental and wrapping.
- The burst start address can be unaligned to the AXI data width boundaries
- The size of the burst can be less than the full width of the AXI data bus (also known as sub-sized transfers).

The signals on the read address channel are registered into the XPI in accordance with the AXI valid/ready handshaking protocol and are synchronous with the AXI clock (aclk).

Read requests are accepted if the request can be written into the Read Address Queue (RAQ). This is used to store all the read address requests. If DDRC_XPI_USE2RAQ_n is set to 0, there is single address queue on a given port. If DDRC_XPI_USE2RAQ_n is set to 1, there are two address queues on a given port, named Blue and Red address queues. Clock domain crossing from aclk to DDRC clock (core_ddrc_core_clk) is performed in the RAQ block. The depth of the RAQ can be configured to suit the system requirements of the user.

After registering the read address channel by the RAQ, the following operations are performed:

- Generation of new read requests is based on alignment (derived from AXI address and size), burst lengths (derived from AXI length and memory burst length), and

burst type (derived from AXI incremental or wrapping). Each AXI burst is divided into packets of length equal to the memory burst length (BL2, BL4, BL8, BL16).
- Generation of new HIF address in the case of unaligned burst and burst expansion. A burst is aligned to the memory burst boundary when:
  - A(R|W)ADDR[X]=0 if BL2
  - A(R|W)ADDR[1+X:X]=0 if BL4
  - A(R|W)ADDR[2+X:X]=0 if BL8 or
  - A(R|W)ADDR[3+X:X]=0 if BL16, Where, X is log2 of the number of data bytes in the SDRAM interface. Therefore,
  - If MEMC_DRAM_DATA_WIDTH = 8, X = 0
  - If MEMC_DRAM_DATA_WIDTH = 16, X = 1
  - If MEMC_DRAM_DATA_WIDTH = 32, X = 2 and so on

In case of an unaligned burst, the first read request is unaligned and the remaining read requests are aligned.

In general, realignment to a memory burst boundary potentially causes some data beats to be discarded (affecting bandwidth) and potentially introduces additional latency on the read data and response channel.

The arready output is defaulted to logic one and remains in logic one unless the RAQ becomes full or a WRAP burst is requested (arready is low on the next cycle after a WRAP burst is accepted). The read transaction is popped from the RAQ when it is not empty.

The XPI handles the generation of the token which is used by the DDRC for identifying the read command and corresponding data.

9.2.2.1.2 Write Address Channel

Write Address Queue (WAQ) is used to store all the addresses for write requests from a given port. There is a single queue for all AXI IDs from a given port. The write address channel behavior is similar to the read address channel.

The depth of the WAQ can be configured to suit the system requirements of the user. Write address and read address channels are independent and the ordering between the write and read requests may not be preserved.

To preserve the sequence, a higher-level protocol needs to wait for read/write response before sending the next transaction.

Transactions across the ports are independent and can be issued in any order.

The write command is not forwarded to the DDRC until the write data is collected and the strobes are evaluated.

9.2.2.1.3 Wrap Burst Expansion

A WRAP burst may be expanded by the XPI into multiple SDRAM transactions. This is also true for cases where WRAP bursts, which could be accommodated by a single transaction on the SDRAM interface. A WRAP burst is expanded into a single SDRAM transaction only in particular cases as highlighted in the tables below.

NOTE

SDRAM burst length 16 is not supported by XPI in this case. If the burst length is programmed to 16, XPI uses the value 8.

Table 9-1. MEMC_BURST_LENGTH 16 Wrap Expansion

Direction	Port Data Width	(SDRAM BurstLength8&&Quarter Burst) || (SDRAM Burst Length4&&Half Burst)|| (SDRAM BurstLength4&&Quarter Burst)	(SDRAM Burst Length16&&Quarter Burst)|| (SDRAM Burst Length8&&Half Burst)|| (SDRAM Burst Length4&& Full Burst)	(SDRAM Burst Length16&&Half Burst)|| (SDRAM Burst Length8&&Full Burst)	SDRAM Burst Length16&&Full Burst
Read	Native-Sized	N/A	Single	Single	Single
	Up-Sized	Single(onlyif araddris alignedtotheHIF burst)	Single	Single	Single
	Down-Sized	N/A	N/A	Single	Single
Write	Native-Sized	N/A	Multiple	Multiple	Single
	Up-Sized	Multiple	Multiple	Multiple	Single(onlyif awaddrisalignedtotheHIF burst)
	Down-Sized	N/A	N/A	Multiple	Single

In the tables, full burst is when:

DDR_bytes = AXI_bytes, half burst is when DDR_bytes = 2*AXI_bytes and quarter burst is when DDR_bytes = 4*AXI_bytes.

Where, DDR_bytes refers to the number of bytes transferred in a DDR burst: MSTR.burst_rdwr*2*MEMC_DRAM_DATA_WIDTH/8

AXI_bytes refers to the number of bytes transferred in an AXI burst: 2(ALEN +1)*(DDRC_PORT_DW_n/8)

As an example, in a natural-size port, with MEMC_FREQ_RATIO=1 and MEMC_BURST_LENGTH=16, when the SDRAM burst length is programmed to 8 (Burst length of 16), ALEN 7 results in a full burst.

9.2.2.1.4 Read Data and Response Channel

The XPI handles the common response interface to the DDRC to process the read data from the memory.

AXI read data and response channel has a single data storage queue (RDQ - Read Data Queue) with two clocks—AXI clock for reading and DDRC clock for writing.

Data from different IDs are stored in the same queue and are returned in the order of read address acceptance. You can configure the depth of the queue independently for each port.

The FSM handling the AXI handshake as well as the FIFO push and pop has the following features:

- Based on the information stored, filter the dummy beats
- Based on the signal hif_rdata_last, generate the rlast

The controller provides an OKAY response for the each read, except for exclusive read transactions. The DDRC provides an EXOKAY response.

SLVERR respinse may be returned for the read transactions (both normal and exclusive) for the following cases:

- ECC uncorrected error detected at the DFI
- On-chip parity address or data error
- InvalidLPDDR4 row address
- Transaction has been poisoned

NOTE

SLVERR has the highest priority.

9.2.2.1.4.1 Read Reorder Buffer

The read data can be returned from the DDRC in a different order from which the read commands are forwarded from the XPI. (due to the re-ordering of read commands in the DDRC to maximize SDRAM bandwidth).A read reorder buffer is implemented in each port to reorder the read data for that port to the same order as the order of the AXI read commands.

The read reorder buffer SRAM holds the same number of entries as the read CAM and each entry holds the read data corresponding to a DDR command. Storage for the reorder buffer can be implemented internally in the DDRC or implemented externally to the DDRC as embedded SRAM and accessed through an External RAM Interface. In either case, the control circuits for the read reorder SRAM are clocked by the DDRC clock (and not AXI clock). If implemented as embedded SRAM, a two-port SRAM is required.

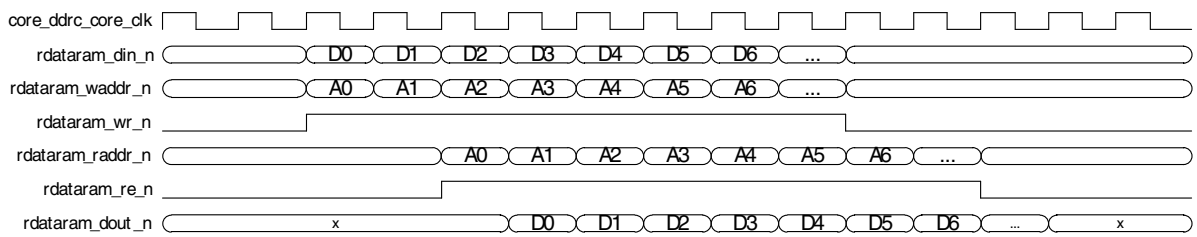


Figure 9-2. read Reorder Buffer Data RAM Interface Timing

The AXI protocol allows the read data for transactions of different IDs to be interleaved. To reduce potential delays where read data for one ID is blocked waiting for data associated with another ID, the read reorder buffer is organized as number of virtual channels (See the figure below). The Read Reorder Virtual Channel is a mechanism to allow independent read data reordering between multiple groups of AXI IDs.

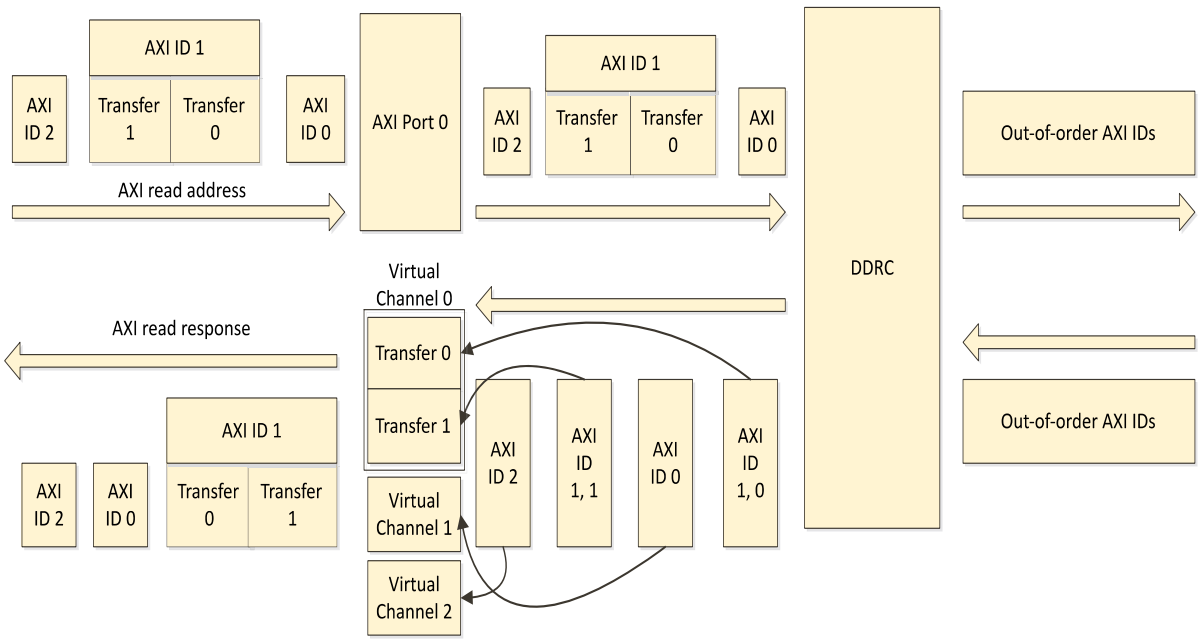


Figure 9-3. RRB Configured with Three Virtual Channels

9.2.2.1.5 Write data Channel

The AXI write data channel has a data storage queue (WDQ - Write Data Queue) with the following clocks:

- AXI clock (aclk_n) for writing and
- DDRC clock (core_ddrc_core_clk) for reading

This queue is used for the clock domain crossing between the AXI clock domain and the controller clock domain and acts as a registering layer in the case of a synchronous interface.

At the output of WDQ, some beats of a write data packet can be masked depending on alignment, burst size and burst length.

Write data from each port is forwarded to the DDRC, which forwards the data to a common data storage.

9.2.2.1.6 Data Width Conversion

The XPI performs the data width conversion between the data width at the AXI interface and internal HIF data width. The AXI data width can be set by the hardware parameter DDRC_PORT_DW.

Both data width down-conversion (where the AXI data bus is wider than the internal HIF data-bus width) and data width up-conversion (where the AXI data bus is narrower than the internal HIF data-bus width) are supported.

If no data width conversion is performed, port is referred as Native-sized:

DDRC_PORT_DW_n = 2 * MEMC_DRAM_DATA_WIDTH if MEMC_FREQ_RATIO = 1

DDRC_PORT_DW_n = 4 * MEMC_DRAM_DATA_WIDTH if MEMC_FREQ_RATIO = 2

When data width up conversion is performed, port is referred as Up-sized:

DDRC_PORT_DW_n < 2 * MEMC_DRAM_DATA_WIDTH if MEMC_FREQ_RATIO = 1

DDRC_PORT_DW_n < 4 * MEMC_DRAM_DATA_WIDTH if MEMC_FREQ_RATIO = 2

When data width down conversion is performed, port is referred as Down-sized:

DDRC_PORT_DW_n > 2 * MEMC_DRAM_DATA_WIDTH if MEMC_FREQ_RATIO = 1

DDRC_PORT_DW_n > 4 * MEMC_DRAM_DATA_WIDTH if MEMC_FREQ_RATIO = 2

The only exception is when Shared-AC with data channel interleaving is used, where Native-sized port is equivalent to a Down-sized port (1:2 ratio) in Single Channel configuration.

The width of the queues, Write Data Queue (WDQ) and Read Data Queue (RDQ) are always set to the wider data width. In the case of data width down-conversion, these queues are set to the width of the AXI data bus. In the case of data width up-conversion, these queues are set to the width of the internal HIF data-bus width

9.2.2.1.7 Write Response Channel

The write response is generated once the last beat of write data, for a given AXI burst, is accepted by the DDRC. The write response queue can be instantiated with configurable depth which is set by the hardware parameter DDRC_AXI_WRQD_n.

The write response generation makes use of the result of the exclusive access monitor. For a write transaction, the response is always returned as OKAY. For an exclusive write transaction, the response can be returned as OKAY or EXOKAY.

SLVERR response may be returned in the following cases:

- Invalid LPDDR4 row address
- On-chip parity address or data error
- Transaction has been poisoned

NOTE

SLVERR has the highest priority

9.2.2.1.8 Exclusive Access

The DDRC supports the AXI Exclusive Access feature. This feature is enabled using the hardware parameter DDRC_EXCL_ACCESS. This parameter defines the number of addresses the DDRC can monitor.

All exclusive read transactions have an EXOKAY response (except in the case of an ECC uncorrectable error). Successful exclusive write accesses have an EXOKAY response, unsuccessful exclusive write accesses return an OKAY response. If an exclusive write fails, the data mask for the exclusive write is forced low so the data is not written.

One address range per AXI ID is monitored for exclusivity. Therefore, if a master does not complete the write portion of an exclusive operation, a subsequent exclusive read to the same ID changes the address that is being monitored for exclusivity.

Once an exclusive access monitor for a given address is enabled, all write transactions are monitored for violation, regardless of the originating port. In other words, the violation check operates across the ports.

The exclusive access monitor compares the exclusive write transaction address, size, length, ID and port number against the exclusive read transaction address, size, length, ID and port number, and only accepts an exclusive write when these parameters match. Otherwise, the exclusive write is considered as fail.

If the EXCL_ACCESS parameter is configured to support no (0) exclusive accesses, the DDRC behaves like an AXI slave that does not support exclusive accesses. Therefore, all exclusive accesses return OKAY responses.

In some cases SLVERR response may be returned for exclusive access transactions. For more information about this, see **“Read Data and Response Channel”** and **“Write Response Channel”**.

NOTE

- The read response SLVERR is generated for an exclusive read transaction for ECC configurations, when ECC is enabled and an uncorrectable error is detected. The master, upon receiving a SLVERR to an exclusive read command should not complete the exclusive operation by sending an exclusive write command. If it does, the DDRC monitors return EXOKAY if there is no violation.
- The maximum number of bytes that are monitored as a region per address cannot exceed 128 bytes. The AXI exclusive transaction length must always be a power of two.
- When all the monitors are active, further exclusive read should not be issued. If this happens, one of the active monitors is evicted using a round robin control and new exclusive read is accepted. The selection of the monitor to be evicted is based on a pointer which rotates across all the locations when a new exclusive read is received.

9.2.2.1.8.1 Dual Channel considerations

In dual channel configurations, Shared AC and LPDDR4 dual channel, a second set of monitors is instantiated for the second channel. This means that the total number of exclusive monitors is doubled.

This has the following implications:

- Maximum number of monitored locations is equal to DDRC_EXCL_ACCESS*2 (DDRC_EXCL_ACCESS locations per data channel).
- If a master issues two Exclusive Read accesses to the same ARID to different channels (for example, channel 0 first, followed by channel 1, without issuing Exclusive Write access in between), channel 0 monitor for this ID is not evicted immediately and channel 1 monitor starts monitoring for the same ID.

According to AXI protocol, the maximum transfer size for exclusive accesses is 128 bytes. For correct functionality, an exclusive access transfer must not be interleaved across two data channels, that is, one EXA transfer goes to one channel only.

The following are the possible ways to achieve this requirement:

- The channel select bit of the address map (ADDRMAP0.addrmap_dch_bit0) can be programmed at or above the 128 byte boundary. For

MEMC_DRAM_DATA_WIDTH=16, this boundary corresponds to HIF[6] or addrmap_dch_bit0=4. For MEMC_DRAM_DATA_WIDTH=32, this boundary corresponds to HIF[5] or addrmap_dch_bit0=3.

- If the required channel interleaving boundary is less than 128 bytes, then the size of all EXA transfers must be constrained to be less or equal to the interleaving boundary.

9.2.2.1.9 Software Coherency for AXI Ports

The section “Address Collision Handling” describes how the DDRC handles in-order execution of commands to the same address.

For the commands issued at the AXI port, the logic in the DDRC protects against all types of software coherency hazards when the AXI master waits for write (read) response before sending the next same address read (same address write) or vice versa.

Some special AXI masters expect an ordering relationship between the reads and writes when the opposite direction transactions are sent without waiting for the previous response. For these special cases, additional logic should be instantiated in the XPI to enforce the order of acceptance within the reads and writes between the AXI and HIF interfaces at the DDRC. This logic is included by setting of the hardware parameter DDRC_RDWR_ORDERED_n and enabled by the register PCFGR_n.rdwr_ordered_en. When this feature is enabled and read and write transactions are presented at the same cycle, the pre-arbiter gives precedence to read transactions, incrementing the write transaction order token by 1 with respect to the read transaction order token. For example, if the last order token issued was equal to 0, then the read order token will be equal to 1 and the write order token will be equal to 2.

9.2.2.1.10 Transaction Poisoning

Sideband signals arpoison/awpoison, render an AXI transaction (read or write) invalid and the DDRC signals that AXI transaction poisoning has occurred. The input signal must be asserted coinciding with the associated AXI transaction. If a write is poisoned, all of its strobes are de-asserted, making the write effectively transparent to the memory. If a read is poisoned, the command is issued to the DDR memory and all the read data beats are overwritten and returned as zeros.

The DDRC may be programmed to signal that an AXI transaction poisoning has occurred by setting an interrupt or by driving the AXI SLVERR response for that transaction response. The AXI Poison Configuration Register POISONCFG (see DDRC_REGS Registers) is used to define whether an interrupt and/or AXI SLVERR is generated when an AXI transaction poisoning has occurred. There is separate control for read and write

transactions. The port specific AXI Poison status register is asserted whenever an AXI transaction is poisoned on that port. There is a separate status for read and write transactions. The interrupt and status register is cleared by the register POISONCFG.rd_poison_intr_clr/POISONCFG.wr_poison_intr_clr. Interrupt status is stored in the AXI Poison Status register POISONSTAT (see DDRC_REGS Registers onpage 464). This register is a mirror in the APB clock domain of the actual interrupts, so status is updated with 2 or 3 pclk cycles of delay depending on the CDC logic.

An external block is expected to drive these new sideband signals depending on the access security requirements on transaction by transaction basis. Therefore, this feature replaces the need to implement ARM Trustzone.

9.2.2.1.11 AXI Data Channel Interleaving

A given AXI port (XPI) can access both data channels and low granularity interleaving between the two data channels is possible by mapping a suitable LSB address bit which is at the memory burst boundary. In that case, for example, a long AXI burst can be expanded to HIF commands so that back to back accesses are to alternating data channels. This setting achieves the highest efficiency possible from a single AXI port.

On the read data path, data order - based on AXI ordering rules - has to be preserved within each channel and between both channels. The read reorder buffer reorders the data coming from a given channel. The data ordered from both channels is then reordered by the data channel reorder logic. Thus two separate data SRAMs are instantiated, one for each channel, whose outputs are combined to provide a single wide AXI read data channel.

By setting the correct system mapping, one port can be set to access one data channel and another port can be set to access the other data channel. This configuration saves logic area in XPI block.

9.2.2.2 AHB Port Interface

The AHB slave port interface consists of an A2X block and XPI block as described in “AXI Port Interface (XPI)”. The A2X block converts the AHB slave port transactions into AXI equivalent transactions before forwarding them to the XPI, where the transactions are finally broken down into HIF packets and sent to the Port Arbiter (PA).

9.2.2.2.1 A2X

The A2X controls the following features of the AHB Slave port interface.

9.2.2.2.1.1 Split Mode

The split mode of the AHB slave port can be set by the parameter DDRC_AHB_SPLIT_MODE_n. Split response is used to control the AHB bus for efficiency and coherency on a multi-master split capable port. This is especially for AHB read transactions where the master is split once the NSEQ address is captured. When the DDRC fetches the read data for split transaction, the AHB interface slave port is free to receive a transaction for another master. The maximum number of splits (simultaneous) is equal to the number of AHB masters configured for the port. Once data is retrieved for a particular master, the master that is split is recalled.

For AHB write, the transaction may be split depending on the write response mode. For more information about this, see **"Write Response Mode"**.

In non-split mode, a single master controls the AHB interface slave port until its transaction completes, unless it is early burst terminated before completion. In the case of a read, the port output hready_resp is deasserted after the NSEQ phase of the transaction and asserted again during the data phase of each beat once the read data is available. When operated in this mode, every read transaction holds the bus for the amount of data beats requested plus at least the read idle latency time of the DDRC. For this reason, nonsplit AHB configuration has a negative impact on the performance of the port when compared with the split configurations.

9.2.2.2.1.2 Write Response mode

There are three AHB write response modes namely non-bufferable, bufferable, or dynamic write response modes. These modes can be programmed (per port) by the parameter DDRC_AHB_WRITE_RESP_MODE_n.

In non-bufferable mode, the hready_resp for the data phase of the last write beat for a defined length AHB burst is not asserted until the A2X logic receives confirmation from XPI (through bresp_n) that the last data beat is accepted by the DDRC. When the port operates in non-split mode, hready_resp remains low until this event occurs. When the port operates in split mode, the master whose write transaction is on the bus is split on either the last write data beat or if hready_resp has been driven low for DDRC_AHB_HREADY_LOW_PERIOD_n cycles (whichever occurs latter), and then recalled when that beat is accepted by the DDRC. For an undefined length INCRs, hready_resp for the data phase of each beat is not asserted until the A2X logic receives

confirmation that each beat is accepted by the DDRC. This indicates performance degradation for long undefined INCR writes when compared with the bufferable equivalent.

In bufferable mode, the logic does not wait for this last beat of write data to be accepted by the DDRC. Instead, hready_resp on the AHB port interface is asserted for the data phase of this beat once the AHB port of the DDRC can accept it. If operating in split mode, a split is generated if the hready_resp is driven low for greater than DDRC_AHB_HREADY_LOW_PERIOD_n cycles.

In dynamic mode, the transactions can be either bufferable or non-bufferable depending on the value of hprot according to the AMBA AHB Specification for the individual AHB transactions, where hprot[2]=1'b0 is non-bufferable and hprot[2]=1'b1 is bufferable.

9.2.2.2.1.3 Error Response

The AHB port only returns an error response on hresp_n for a data beat that has an uncorrectable ECC error. For this to occur, the DDRC should be configured to enable ECC logic. For more information about this, see "Read-Modify-Write (RMW) Generation".

9.2.2.2.1.4 Early Burst Termination (EBT)

A defined length write transaction that is early burst terminated by another transaction has its remaining write data beats masked during memory access. When the early burst terminated master returns, only the remaining write data beats that are masked after the initial EBT are sent to the memory. This EBT of defined length write transactions causes multiple memory accesses. The non-bufferable response for the single defined length transaction that is split into multiple memory access is given only after all memory accesses are completed.

Any defined length read transaction must return to retrieve all its data if early burst terminated. Any undefined length INCR read transaction has all remaining data beats that are not retrieved by a master, up to the read prefetch beats set by the pin hincr_arlen_n, flushed internally in the DDRC.

9.2.2.2.1.5 AHB Lite mode

The parameter DDRC_AHB_LITE_MODE_n is used to set an AHB port to operate in AHB Lite mode. This means that the port can be connected only to a single master and responds to AHB read and nonbufferable write transactions by driving hready_resp low (as opposed to split responses). This simplifies the AHB master/arbiter system connected to an AHB Lite port.

9.2.2.2.1.6 Endian conversion

The A2X provides endian mapping from the AHB port to the XPI with respect to the transaction size. This section outlines the key features of the A2X endian mapping.

The key features of endian conversion are as follows:

- Register: The functionality is enabled by PCFGC_n.ahb_endianness (see DDRC_MP Registers)
- Endian transformations supported:
  - Little Endian (LE)
  - Big Endian-32 (BE-32) invariant
  - Big Endian-A (BE-A) address invariant

9.2.2.2.2 Write Streaming

For AHB write streaming, the write data buffer parameter DDRC_AHB_WDQD should be set to a size that can support enough write data beats from the first valid AHB write beat. So, the transaction on the AHB slave interface port to time the first beat is accepted by the DDRC, signified by asserting the DDRCs hif_wdata_valid and hif_wdata_stall outputs.

9.2.2.2.3 Read Streaming

For non-split configurations, it is not possible to stream different transaction due to the nature of the AHB protocol, as we cannot get the address of the next AHB transaction to access memory until the read data for the current transaction has completed. We can stream within a transaction from the time the first read beat is available until the last read beat of the same transaction.

For split capable configurations, depending on the number of masters and arbitration scheme used accessing a port and the type of AHB transaction used, it is possible to stream reads after the initial read latency delay from the first AHB transaction.

9.2.2.2.4 XPI

The XPI used in conjunction with the A2X provides features to complete the overall AHB slave port solution. These features include data width conversion from the AHB port data width to the internal data width used by the PA and DDRC, which is twice the width of the SDRAM data interface in case of 1:1 clock frequency ratio and four times the width of the SDRAM data interface in case of 1:2 clock frequency ratio.

Each AHB port can be independently configured to operate with a clock that is synchronous or asynchronous to the memory controller clock.This clock crossing logic is done within the XPI.

ECC is supported for configurations with AHB port interfaces.

9.2.2.2.5 Software Coherency For AHB Ports

Coherency of read after write (RAW) and write after read (WAR) to the same address is always guaranteed. For more information about this, see “Software Coherency for AXI Ports” and “Address Collision Handling”.

9.2.2.3 Host Interface (HIF)

HIF is an internal interface when the DDRC is configured to include a multi port arbiter (the hardware parameter DDRC_INCL_ARB is set to 1). In this case, the AXI Port Interface and/or Port Arbiter handle all signals on this interface, effectively acting as the SoC core.

Read, write, as well as write data are received through the HIF. Following the receipt and acceptance of a write the DDRC requests write data from the interface. The interface subsequently provides the independently timed write data in the requested order (which matches the order in which requests are made to the DDRC). See “Dual HIF Functionality” for further details on Dual HIF functionality. Both read and write requests can be reordered later, before being output on the DFI interface.

Requests to the DDRC are throttled in two ways:

- A credit mechanism that ensures that buffer space is available for any request the SoC core makes to the DDRC, prior to the request being made.
- An independent stall mechanism that throttles requests when address collisions take place or if entering self-refresh through the software self-refresh or hardware low power self-refresh.

Data is handled separately. The DDRC throttles data by potentially waiting after it receives a write before requesting the associated data. If there is collision, then the throttling lasts until the collision is cleared in the DDRC. Once write data is requested, it is always accepted by the DDRC.

Read data is sent to the host via a common response interface. For more information about this, see “Common Response”.

This section describes the following about the HIF:

- “Dual HIF Functionality”
- “Credit Mechanism”
- “Read Requests”
- “Write Requests and Data”
- “Read-Modify-Write Requests”
- “Common Response”

9.2.2.3.1 Dual HIF functionality

Dual HIF functionality is as follows:

- Converts HIF single command channel into separate HIF command channel for Read and Write commands.
- Read/Write Arbitration is not performed by the PA
- Dual HIF functionality is supported only if the DDRC is configured to support a multi port arbiter (DDRC_INCL_ARB=1).
- Dual HIF functionality is supported only if the DDRC is configured to include the logic to optimize timing over scheduling efficiency (MEMC_OPT_TIMING=1).
- Dual HIF functionality continues to guarantee coherency for write-after-read (WAR) and read-afterwrite (RAW) hazards. Also, if a Read and Write/ to the same address are taken at the same time (both hif_rcmd_stall=0 and hif_wcmd_stall=0), the Write/ is performed first.

The table provides the details of the signals modified by the DDRC_DUAL_HIF parameter.

Table 9-2. Comparison of Signals Modified by DDRC_DUAL_HIF Parameter

DDRC_DUAL_HIF=0 Single HIFCommand Channel	DDRC_DUAL_HIF=1 [1] Separate HIF Command Channels One for Reads (*_rd) One for Writes/ (*_wr)
hif_cmd_valid	hif_rcmd_valid hif_wcmd_valid
hif_cmd_type	hif_rcmd_type hif_wcmd_type

Table continues on the next page...

Table 9-2. Comparison of Signals Modified by DDRC_DUAL_HIF Parameter (continued)

hif_cmd_addr	hif_rcmd_addr hif_wcmd_addr
hif_cmd_pri	hif_rcmd_prihif_wcmd_pri
hif_cmd_token	hif_rcmd_token hif_wcmd_token
hif_cmd_token	hif_rcmd_length hif_wcmd_length
hif_cmd_wdata_ptr	hif_rcmd_wdata_ptr hif_wcmd_wdata_ptr
hif_cmd_autopre	hif_rcmd_autopre hif_wcmd_autopre
hif_cmd_stall	hif_rcmd_stall hif_wcmd_stall

NOTE

1. As DDRC_DUAL_HIF=1 is only possible if DDRC_INCL_ARB=1, the right hand column refers to internal signals within the DDRC. It is provided for debug purposes only.

NOTE

The databook uses DDRC_DUAL_HIF=0 notation.

If you are using DDRC_DUAL_HIF=1, the internal signals should be interpreted in the following way:

- hif_cmd_valid refers to hif_rcmd_valid and/or hif_wcmd_valid.
- hif_cmd_stall refers to hif_rcmd_stall and/or hif_wcmd_stall and so on.

9.2.2.3.2 Credit Mechanism

The DDRC employs a credit mechanism to ensure that buffers do not overflow. The interface making the request to the DDRC, can only request commands for which it has been granted “credits” to issue.

Credits are tracked seperately for the following three command types:

- High Priority read(HPR)
- Low Prioroity rad(LPR)
- Write

Credits are counted for each command type independently according to the following rules:

- Initially the interface has zero credits.

- The interface logic must include counters to track the number of credits granted by the DDRC and decremented due to commands issued by the interface logic to the DDRC. The interface logic must have separate credit counters for each individual command type.
- Following the de-assertion of reset to the DDRC, credits are issued to the interface for each command type. A given credit count increments every time the DDRC issues a credit, indicated by the assertion of the appropriate *_credit signal on the rising edge of core_ddrc_core_clk. The credit counting logic implemented externally to the DDRC must run on the same clock.
- When the credit count is greater than zero, the interface can issue requests of that type to the DDRC. Each time a request is issued to the DDRC, the associated credit count is decremented.
- When DDRC_VPRW_EN is set to 1, the arbiter or the host can send Variable Priority Read (VPR) and Variable Priority Write (VPW) commands.
- VPR commands do not have a pre-allocated storage resource in the Read CAM (unlike HPR commands). VPR commands share the same resource with the LPR commands. As far as the credit mechanism is concerned, the VPR commands are counted in the LPR credit bucket.
- Similarly, VPW commands do not have a per-allocated storage resource in the Write CAM. VPW commands share the same resource with the Normal Write commands. As far as credit mechanism is concerned, the VPW commands are counted in the WR credit bucket.

The signals related to the credit request interface are:

- hif_cmd_valid
- hif_cmd_type
- hif_cmd_pri
- hif_lpr_credit
- hif_hpr_credit
- hif_wr_credit
- hif_cmd_stall

9.2.2.3.3 Clear Requests

When the LPR or HPR credit count is greater than zero, the interface can issue read requests to the DDRC accordingly.

A command is issued to the DDRC by asserting hif_cmd_valid on the rising edge of the clock. All read request fields must be driven to the appropriate values at the same time. These fields are:

- **hif_cmd_type**: specifies the request type as follows:

  - '00' indicates a write request
  - '01' indicates a read request
  - '11' is not supported
- **hif_cmd_pri**:
  - In HIF-only configurations (DDRC_INCL_ARB=0) where DDRC_VPRW_EN is set to 0, this signal is 1-bit wide. A '1' indicates the read request is high priority. '0' indicates the read request is low priority. This field is meaningless for write commands.
  - In Arbiter configurations (DDRC_INCL_ARB=1) or in HIF-only configurations (DDRC_INCL_ARB=0) where DDRC_VPRW_EN is set to 1, this signal is 2-bit wide.

  The encoding for Read is:

  - 2'b00 - LPR
  - 2'b01 - VPR
  - 2'b10 - HPR
  - 2'b11 - Reserved

  VPR commands are sent only in configurations with DDRC_VPRW_EN=1

- **hif_cmd_addr**: indicates the address for the read. Each word on the HIF (equivalent to 2 DDR words for 1:1 mode configurations (MEMC_FREQ_RATIO=1), or 4 DDR words for 1:2 mode configurations (MEMC_FREQ_RATIO=2)) is uniquely addressable.
- **hif_cmd_length**, is 2 bits if MEMC_BURST_LENGTH=16, 1 bit otherwise.

  Note the following:

  - For MEMC_BURST_LENGTH = 16, a normal read is 16 SDRAM words

  If MEMC_BURST_LENGTH=16:

  - 2'b11 -> Partial (Quarter) Read
  - 2'b10 -> Partial (Half) Read
  - 2'b00 -> Full Read

  Otherwise:

  - 1'b1 -> Partial (Half) Read
  - 1'b0 -> Full Read

In each case, a partial (half) read is half the length of a normal full read (where a word is the amount of data transferred on one edge of the DQS to the SDRAM in a fully configured system.) If MEMC_BURST_LENGTH=16, Partial (Quarter) Read is possible and is quarter the length of a normal full read. This field is meaningless for write or commands.

- **hif_cmd_token**: a bit field that is presented with the read command and is returned with the read response. As the responses may be presented out-of-order, the token is the identifier that indicates the read for which data is being returned on the response side. Therefore, multiple reads with the same token must never be pending in the DDRC simultaneously, as the responses would be indecipherable. Common uses for this field include identifying the requestor and serializing the response data. This field is meaningless for write or commands.

9.2.2.3.3.1 Valid And Stall

the requests by asserting hif_cmd_stall. For any cycle in which hif_cmd_stall is asserted on the rising edge of the clock, the hif_cmd_valid and all other associated request signals are ignored by the DDRC. When this happens, the interface must hold the request on the interface for another cycle and must not yet decrement the associated credit counter. When the DDRC de-asserts hif_cmd_stall on the rising edge of the clock, the request is accepted and the interface can then de-assert hif_cmd_valid or it can issue the next read, write or request (credits allowing).

The DDRC asserts hif_cmd_stall for any of the following conditions:

1. Software driven self-refresh entry requests.This is to ensure that DDRC is empty when memories are in self-refresh. For more information about this, see “Power Saving Features”.
2. Successful DDRC hardware low power entry requests to self-refresh. This is to ensure that DDRC is empty when memories are in self-refresh. For more information about this, see “Power Saving Features”.
3. Address collisions in the DDRC, where flow control is applied to prevent further commands from entering the DDRC. For more information about this, see “Address Collision Handling".
4. In the event of the Write CAM being full and no corresponding data for any of the write command has yet arrived.
5. Setting DBG1.dis_hif=1, causes hif_cmd_stall to be asserted.

NOTE

If Dual HIF functionality is enabled (DDRC_DUAL_HIF=1), hif_rcmd_stall/hif_wcmd_stall is asserted as follows:

- Conditions (1), (2), (3), and (5) cause both hif_rcmd_stall and hif_wcmd_stall to assert at the same time.
- Condition (4) only affects hif_wcmd_stall.

9.2.2.3.4 Write Requests And Data

When the WR credit count is greater than zero, the interface can issue write requests to the DDRC accordingly. A command is issued to the DDRC by asserting hif_cmd_valid on the rising edge of the clock.

All write request fields must be driven to the appropriate values at the same time. These fields are:

- **hif_cmd_type**: specifies the request type as follows:
  - '00' indicates a write request
  - '01' indicates a read request
  - '11' is not supported
- **hif_cmd_pri**: specifies the priority of the write request
  - In HIF-only configurations (DDRC_INCL_ARB=0) where DDRC_VPRW_EN is set to 0, this signal is 1-bit wide. This field is meaningless for write commands.
  - In Arbiter configurations (DDRC_INCL_ARB=1) or in HIF-only configurations (DDRC_INCL_ARB=0) where DDRC_VPRW_EN is set to 1, this signal is 2-bit wide.

  The encoding for Write is:

  - 2'b00 - NPW (Normal Priority Writes)
  - 2'b01 - VPW (Variable Priority Writes)
  - 2'b10 - Reserved
  - 2'b11 - Reserved

  VPW commands are sent only in configurations where DDRC_VPRW_EN=1
- **hif_cmd_addr**: indicates the address for the write or . Each word on the HIF (hif_wdata/hif_rdata_data) is uniquely addressable. For 1:1 frequency ratio mode configurations each 2 DDR words on the DQ bus are uniquely addressable via hif_cmd_addr. In 1:2 frequency ratio mode configurations every 4 DDR words are uniquely addressable.
- **hif_cmd_wdata_ptr**: a pointer into the requestor's own buffers that can be used to subsequently retrieve the write data. Once a write or request is accepted, this pointer is returned to the interface to retrieve the associated write data. (This field is not required, as requests are always accepted in order; the interface can choose to FIFO these pointers internally and ignore this field.)

The stall mechanism for write or requests is identical to that for read requests as explained in the section “Valid and Stall”.

Writes have three phases of operation: a write command, a write data pointer return, and finally write data provided to the DDRC. The write command phase indicates to the DDRC that a write transfer is required. The write data pointer return phase indicates to the SoC core that the DDRC is ready to receive the write data. The write data phase provides the write data to the DDRC.

In all these three phases, the transactions must follow the same order. The data pointer is returned in the same order in which the command is sent. The write data must also be sent in the same order. The DDRC can perform re-ordering to the commands later, before sending them on the DFI interface, to make efficient use of the SDRAM bandwidth.

9.2.2.3.4.1 Write Data Pointer Return

Following the acceptance of a write or request, the DDRC then returns the write data pointer to the interface logic to retrieve the associated write data. This is done by asserting hif_wdata_ptr_valid on the rising edge of clock. In the same cycle, hif_wdata_ptr indicates the value of the write data pointer. This is same as a pointer that is presented to the DDRC as hif_cmd_wdata_ptr during the write command phase.

There is no mechanism for stalling the acceptance of the write data pointer from the DDRC; the interface logic must present write data for every cycle in which hif_wdata_ptr_valid is asserted by the DDRC. The mechanism to throttle hif_wdata_ptr_valid assertions is for the interface to throttle write or requests presented to the DDRC.

There is no required timing between a write or command presented to the DDRC, and the same pointer is returned as hif_wdata_ptr. Multiple pointers can also be presented to the DDRC with multiple requests before the first is returned to the interface. The order of the pointers returned matches the order of the requests presented to the DDRC.

9.2.2.3.4.2 Write Data

After a write data pointer is returned to the interface logic, the interface logic must retrieve the associated data and present it to the DDRC. Data presented to the DDRC has no required timing relationship to the transfer of the write pointer on the interface. The data, however, must be returned in the same order that the pointers are presented to the interface (same order in which the original requests are made).

The write data can be throttled using hif_wdata_stall signal. The write data is accepted by the DDRC when hif_wdata_valid is detected high and hif_wdata_stall is detected low at the same positive edge of the clock. The signal hif_wdata_stall is asserted by the DDRC during a RMW operation. When the DDRC writes the read data for the RMW operation into the write data SRAM, it blocks the SoC core interface from sending any write data during those cycles by asserting this signal.

The write data SRAM can be implemented internally as synthesized flip-flops or implemented externally to the DDRC as embedded SRAM using DDRC_WDATA_EXTRAM parameter.

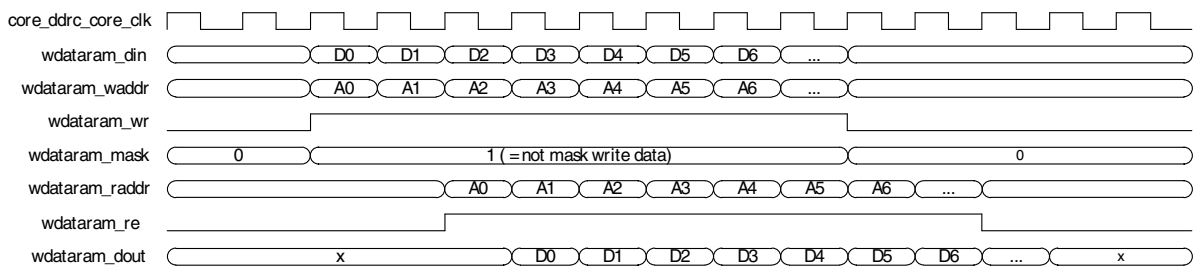


Figure 9-4. Write Data RAM Interface Timing

9.2.2.3.4.2.1 Write Data for the DDRC in BL16 Mode (MEMC_BURST_LENGTH = 16)

Write data is presented to the DDRC with the assertion of hif_wdata_valid. Each request can have one to eight data phases. hif_wdata_valid must be asserted for each data phase and hif_wdata_end must be asserted on the last data phase. hif_wdata is the data. Both hif_wdata_end and hif_wdata are valid only when hif_wdata_valid is '1'.

A normal write or RMW command has eight clocks of data associated with it. In this case, hif_wdata_end must be asserted on the eighth data phase. If the SoC core has less than 8 cycles of write data for a write command, it can choose to send 1, 2, 3, 4, 5, 6 or 7 cycles of data, and hif_wdata_end must be asserted in the appropriate last data phase - these are referred to as Partial Writes. In this case, the DDRC has all the data that it requires 7, 6, 5, 4, 3, 2 or 1 clock cycles earlier than in case of a full write. For these partial writes, the DDRC's behavior is dependent on whether DDRC_PARTIAL_WR is enabled or not:

- DDRC_PARTIAL_WR=0: DDRC still issues the number of SDRAM bursts on the DDR interface that it would issue for a normal (full) write, but masks the unused data phases.
- DDRC_PARTIAL_WR=1: DDRC issues the minimum number of SDRAM bursts on the DDRC interface, depending on the number of HIF write data beats and the HIF address alignment with respect to the SDRAM column address, because SDRAM writes must be sent aligned with BL. This may be smaller than that for a full write.

This analysis assumes 1:1 frequency ratio mode. For 1:2 frequency ratio mode, the number of data beats of the write data on the HIF is halved, so each request has one to four data phases. A Partial Write will have less than four data phases.

9.2.2.3.5 Common Response

The Common Response Interface returns data from the DDRC to the SoC core in response to read requests. The following signals are associated with this functionality:

- hif_rdata_valid
- hif_rdata_end
- hif_rdata_token
- hif_rdata_data

9.2.2.3.5.1 Response Data Ordering

Read data is returned on the HIF in the same order in which it is received from the PHY on the DFI interface. It is independent of whether the read address is aligned or unaligned. For information on how unaligned addresses are handled, see “Sequential/ Interleaved Operations”. For data lane mapping examples where the addresses are aligned, see “Data Lane Mapping Examples”.

9.2.2.3.5.2 Common Response Interface Example

The figure below shows a pair of read requests. The first request is to address A0 with token T0 and length 0. The second request is to A1 with token T1 and length 1. The second read data is returned before the first read data. The DDRC can determine that this sequence is possible due to the state of the open pages and other actions going on in the memories. This improves overall memory bandwidth utilization. The response token indicates which read's data is being provided. Once started, read data for a given token always completes for that token before the next token's read data is returned. The

hif_rdata_valid signal indicates valid data is on the bus and the hif_rdata_end signal indicates that the last data packet is being transmitted. The hif_rdata_valid can deassert in the middle of returning read data for one or more cycles, so hif_rdata_valid must be monitored to validate each cycle of returning read data.

NOTE

Block A: If hif_cmd_length is 1’b1 for this request, 1 cycle of read data comes back and the END signal is asserted in the same cycle.

Block B: If hif_cmd_length is 1’b0 for this request, 2 cycles of read data comes back and the END signal is asserted in the second cycle.

For MEMC_BURST_LENGTH=16, DFI 1:1 configurations, the following is applicable for Block A and Block B in the above figure:

Block A: If hif_cmd_length is 2’b10 for this request, 4 cycle of read data comes back and the END signal is asserted in the fourth cycle.

Block B: If hif_cmd_length is 2’b00 for this request, 8 cycles of read data comes back and the END signal is asserted in the 8th cycle.

9.2.2.3.5.3 LPDDR4 6Gb/12Gb and hif_rdata_addr_err

For LPDDR4 configurations, there is an extra output for DDRC, named hif_rdata_addr_err. This is asserted only in cases where LPDDR4 6Gb/12Gb (ADDRMAP6.lpddr4_6gb_12gb_24gb == 2'b01 or 2'b10) device density is used and the requested address for a read is not valid (when both row address bits MSB and MSB-1 are high). In these cases, DDRC performs a dummy read by changing the physical address to a valid one (row[MSB:MSB-1] changes from 2'b11 to 2'b10). The returned data is masked to 0 and hif_rdata_addr_err is asserted along with the relevant hif_rdata_valid/hif_rdata_token.

9.2.2.4 DFI Interface

The DDRC contains a DFI MC (Memory Controller) interface, which is used to connect to a DFIcompliant PHY, and to transfer address, control and data to the PHY. The DDRC interfaces to a PUB, PUBL, or PUBM2 block which acts as the DFI PHY interface. You can get more information about the DFI interface from the following specifications:

- DDR PHY Interface (DFI) Specification, Revision 3.1 (Preliminary), 21 March 2014 and
- DDR PHY Interface (DFI) Specification, Preliminary DFI 4.0 Specification - Addendum to DFI 3.1, 2 April 2014

The DFI interface is sub-divided into the following interface groups:

- Control Interface
- Write Data Interface
- Read Data Interface
- Update Interface
- Status Interface
- Training Interface (PHY-independent mode supported by the DDRC)
- Low Power Control Interface

9.2.2.4.1 1:1 Frequency Ratio Mode Considerations

The DDRC, including its DFI interface, runs at the single data rate (SDR) clock (referred to as 1:1 frequency ratio mode in the DFI Specification), so all DFI command and address signals are equal in width to the equivalent DDR SDRAM signals. DFI data signals are twice the width of the equivalent DDR SDRAM signals. The DDRC runs with an SDR clock and the PHY operates the SDR to DDR rate conversion on the data buses. In 1:1 frequency ratio mode, core_ddrc_core_clk and CK/CK# run at the same SDR clock frequency.

9.2.2.4.2 1:2 Frequency Ratio Mode Considerations

The DDRC, including its DFI interface runs at the half data rate (HDR) clock (referred to as 1:2 frequency ratio mode in the DFI Specification), so all DFI command and address signals are twice the width of the equivalent DDR SDRAM signals. DFI data signals are four times the width of the equivalent DDR SDRAM signals. The DDRC operates with an HDR clock. The PHY handles HDR-to-SDR conversion on the address bus to memory and HDR/DDR conversion on the data buses. In 1:2 frequency ratio mode, core_ddrc_core_clk runs at half the frequency of CK/CK#.

For DFI command and address signals in DFI 1:2, the DFI protocol refers to phase 0 (_p0)and phase 1 (_p1) but there are no such *_p0 and *_p1 signals in the DDRC. Instead, the lower half of the bits corresponds to phase 0 and the upper half of the bits corresponds to phase 1. For example, in a 4-rank system in DFI 1:2, dfi_cs[7:0] is the case. dfi_cs[3:0] corresponds to dfi_cs_p0 and dfi_cs[7:4] corresponds to dfi_cs_p1.

Similarly, for DFI write data signals in DFI 1:2, the DFI protocol refers to phase 0 (_p0) and phase 1 (_p1). But, there are no such *_p0 and *_p1 signals in the DDRC. Instead, the lower half of the bits correspond to phase 0 and the upper half of the bits correspond to phase 1.

Consider the following examples:

- In system that has 16 bits of SDRAM in DFI 1:2, dfi_wrdata[63:0] is the case. dfi_wrdata[31:0] corresponds to dfi_wrdata_p0 and dfi_wrdata[63:32] corresponds to dfi_wrdata_p1. Similarly, dfi_wrdata_en[3:0] is the case. dfi_wrdata_en[1:0] corresponds to dfi_wrdata_en_p0 and dfi_wrdata_en[3:2] corresponds to dfi_wrdata_en_p1. Similarly, dfi_wrdata_mask[7:0] is the case. dfi_wrdata_mask[3:0] corresponds to dfi_wrdata_mask_p0 and dfi_wrdata_mask[7:4] corresponds to dfi_wrdata_mask_p1.
- In system that has 32 bits of SDRAM in DFI 1:2, dfi_wrdata[127:0] is the case. dfi_wrdata[63:0] corresponds to dfi_wrdata_p0 and dfi_wrdata[127:64] corresponds to dfi_wrdata_p1. Similarly, dfi_wrdata_en[7:0] is the case. dfi_wrdata_en[3:0] corresponds to dfi_wrdata_en_p0 and dfi_wrdata_en[7:4] corresponds to dfi_wrdata_en_p1. Similarly, dfi_wrdata_mask[15:0] is the case. dfi_wrdata_mask[7:0] corresponds to dfi_wrdata_mask_p0 and dfi_wrdata_mask[15:8] corresponds to dfi_wrdata_mask_p1.

Similarly, for dfi_rddata_en in DFI 1:2, the DFI protocol refers to phase 0 (_p0) and phase 1 (_p1). But, there are no such *_p0 and *_p1 signals in the DDRC. Instead, the lower half of the bits correspond to phase 0 and the upper half of the bits correspond to phase 1.

Consider the following examples:

- In system that has 16 bits of SDRAM in DFI 1:2, dfi_rddata_en[3:0] is the case. dfi_rddata_en[1:0] corresponds to dfi_rddata_en_p0 and dfi_rddata_en[3:2] corresponds to dfi_rddata_en_p1.
- In system that has 32 bits of SDRAM in DFI 1:2, dfi_rddata_en[7:0] is the case. dfi_rddata_en[3:0] corresponds to dfi_rddata_en_p0 and dfi_rddata_en[7:4] corresponds to dfi_rddata_en_p1.

For dfi_rddata_valid/dfi_rddata/dfi_rddata_dbi signals in DFI 1:2, the DFI protocol refers to word 0 (_w0) and word 1 (_w1). But, there are no such *_w0 and *_w1 signals in the DDRC. Instead, the lower half of the bits correspond to word 0 and the upper half of the bits correspond to word 1.

Consider the following examples:

- In system that has 16 bits of SDRAM in DFI 1:2, dfi_rddata[63:0] is the case. dfi_rddata[31:0] corresponds to dfi_rddata_w0 and dfi_rddata[63:32] corresponds to dfi_rddata_w1. Similarly, dfi_rddata_valid[3:0] is the case. dfi_rddata_valid[1:0] corresponds to dfi_rddata_valid_w0 and dfi_rddata_valid[3:2] corresponds to dfi_rddata_valid_w1. Similarly, dfi_rddata_dbi[7:0] is the case. dfi_rddata_dbi[3:0] corresponds to dfi_rddata_dbi_w0 and dfi_rddata_dbi[7:4] corresponds to dfi_rddata_dbi_w1.
- In system that has 32 bits of SDRAM in DFI 1:2, dfi_rddata[127:0] is the case. dfi_rddata[63:0] corresponds to dfi_rddata_w0 and dfi_rddata[127:64] corresponds to dfi_rddata_w1. Similarly, dfi_rddata_valid[7:0] is the case. dfi_rddata_valid[3:0] corresponds to dfi_rddata_valid_w0 and dfi_rddata_valid[7:4] corresponds to dfi_rddata_valid_w1. Similarly, dfi_rddata_dbi[15:0] is the case. dfi_rddata_dbi[7:0] corresponds to dfi_rddata_dbi_w0 and dfi_rddata_dbi[15:8] corresponds to dfi_rddata_dbi_w1.

9.2.2.4.3 DFI Write/Read Data to SDRAM Conversion

This conversion is performed in the PHY, and therefore PHY-Specific. The following details are applicable only to Synopsys DDR PHYs; other PHYs may differ from this.

The DFI data (both read and write) should contain data for multiple byte lanes and for multiple data beats (2 in 1:1 mode, 4 in 1:2 mode). The DFI Specification does not explicitly state how this data should be packed within dfi_rddata and dfi_wrdata. The Synopsys DDR PHYs and the DDRC DFI interfaces are designed with the assumption that the data is ordered by beat and then (within each beat) by byte.

For example, for a 2-byte configuration (16 bits of SDRAM) in 1:2 mode, the DFI data order is as follows (from MSB to LSB):

[byte1-beat3, byte0-beat3, byte1-beat2, byte0-beat2, byte1-beat1, byte0-beat1, byte1-beat0, byte0-beat0]

Therefore, the PHY sends the data it receives out to the memory in the following way:

- In beat 0, the data from dfi_wrdata[15:0] are sent on DQ[15:0]
- In beat 1, the data from dfi_wrdata[31:16] are sent on DQ[15:0]

- In beat 2, the data from dfi_wrdata[47:32] are sent on DQ[15:0]
- In beat 3, the data from dfi_wrdata[63:48] are sent on DQ[15:0]

Similarly, for a 2-byte configuration (16 bits of SDRAM) in 1:1 mode, the DFI data order is as follows (from MSB to LSB):

[byte1-beat1, byte0-beat1, byte1-beat0, byte0-beat0]

Therefore, the PHY sends the data it receives out to the memory in the following way:

- In beat 0, the data from dfi_wrdata[15:0] are sent on DQ[15:0]
- In beat 1, the data from dfi_wrdata[31:16] are sent on DQ[15:0]

Similarly, for a 4-byte configuration with 32 bits of SDRAM in 1:2 mode, the DFI data order is as follows (from MSB to LSB):

[byte3-beat3, byte2-beat3, byte1-beat3, byte0-beat3, byte3-beat2, byte2-beat2, byte1-beat2, byte0- beat2, byte3-beat1, byte2-beat1, byte1-beat1, byte0-beat1, byte3-beat0, byte2-beat0, byte1-beat0, byte0-beat0]

Therefore, the PHY sends the data it receives out to the memory in the following way:

- In beat 0, the data from dfi_wrdata[31:0] are sent on DQ[31:0]
- In beat 1, the data from dfi_wrdata[63:32] are sent on DQ[31:0]
- In beat 2, the data from dfi_wrdata[95:64] are sent on DQ[31:0]
- In beat 3, the data from dfi_wrdata[127:96] are sent on DQ[31:0]

Similarly, for a 4-byte configuration (32 bits of SDRAM) in 1:1 mode, the DFI data order is as follows (from MSB to LSB):

[byte3-beat1, byte2-beat1, byte1-beat1, byte0-beat1, byte3-beat0, byte2-beat0, byte1-beat0, byte0-beat0]

Therefore, the PHY sends the data it receives out to the memory in the following way:

- In beat 0, the data from dfi_wrdata[31:0] are sent on DQ[31:0]
- In beat 1, the data from dfi_wrdata[63:32] are sent on DQ[31:0]

For more information, see the addressing example in Figure A-1. Similar ordering is true for dfi_wrdata_en/dfi_wrdata_mask and dfi_rddata_en/dfi_rddata/dfi_rddata_valid/ dfi_rddata_dbi.

NOTE

- If dfi_rddata data slice independence is enabled (DDRC_DFI_RDDATA_PER_BYTE=1), the logic assumes the DDR PHY ordering on bytes/beats for dfi_rddata_valid/dfi_rddata/dfi_rddata_dbi.

9.2.2.4.4 Control Interface

The control interface is a reflection of the SDRAM control interface including address, bank, chip select, row strobe, column strobe, write enable, clock enable and ODT control, as applicable for the memory technology. This interface consists of the following signals:

- dfi_address (width of this signal depends on configuration - 20 bits if 18 bits if MEMC_DDR4 = 1; otherwise 16 bits)
- dfi_bank
- dfi_cas_n
- dfi_cke
- dfi_cs
- dfi_odt
- dfi_ras_n
- dfi_reset_n (this signal only exists if MEMC_DDR4 = 1)
- dfi_we_n
- dfi_bg (this signal only exists if MEMC_DDR4=1)
- dfi_act_n (this signal only exists if MEMC_DDR4 = 1)

For more information about the above signals, refer to “Signal Descriptions”.

9.2.2.4.5 Write Data Interface

The write data interface handles the transmission of write data across the DFI interface. The DFI Specification defines signals and timing relationships. This interface consists of the following signals:

- dfi_wrdata
- dfi_wrdata_en
- dfi_wrdata_mask

For more information about the above signals, refer to “Signal Descriptions”. The timing of this interface is user-configurable, to support all DFI-compliant PHYs through the DFITMG0.dfi_tphy_wrlat and DFITMG0.dfi_tphy_wrdata register (see DDRC_REGS Registers). For DFI 1:2, DFITMG0.dfi_wrdata_use_dfi_phy_clk determines whether DFITMG0.dfi_tphy_wrlat and DFITMG0.dfi_tphy_wrdata are in terms of SDR or HDR clock cycles. If HDR, timing for dfi_wrdata* generation for write commands on phase 0/ phase 1 are the same and dfi_wrdata* signals are aligned to HDR clock. If SDR, timing for dfi_wrdata* generation for write commands on phase 0/phase 1 are treated separately and dfi_wrdata* signals are not guaranteed to be aligned to HDR clock. Check your PHY requirements for correct programming.

In DDR4 configurations (when MEMC_DDR4 is set) or LPDDR4 configurations (when MEMC_LPDDR4 is set), the signal dfi_wrdata_mask can act as Write DBI signal depending on the user programming of the SDRAM mode register. For more information on DBI, see “Data Bus Inversion (DBI)”.

Also, as per JEDEC Specification, the polarity of dfi_wrdata_mask is inverted when using DDR4 memories (if MSTR.ddr4 = 1). Therefore, for DDR4, if a bit of dfi_wrdata_mask is ‘0’, the corresponding byte is masked.

9.2.2.4.5.1 Support for dfi_wrdata_cs/dfi_rddata_cs Signals (DFI 3.1)

There is an optional support for dfi_wrdata_cs and dfi_rddata_cs signals by setting parameter DDRC_DFI_DATA_CS_EN=1. This is a new feature introduced in DFI 3.0, which also applies to DFI 3.1. Enable this feature only if your PHY supports it. These signals are only supported for configurations with MEMC_NUM_RANKS>1. The polarity of these signals is defined by DFIMISC.dfi_data_cs_polarity.

The signal dfi_wrdata_cs is driven as follows:

- dfi_wrdata_cs is driven according to the relevant chip select tphy_wrcslat DFI PHY clock cycles after any command that generates dfi_wrdata. This includes a write, and in DDR4, a PDA (write). It is guaranteed to remain at this value for a minimum of tphy_wrcsgap plus the time for the write data (usually MSTR.burst_rdwr).

  Similarly, for dfi_rddata_c:

- dfi_rddata_cs is driven according to the relevant chip select tphy_rdcslat DFI PHY clock cycles after any command that generates dfi_rddata. This includes a Read, in LPDDR4, a MRR and in DDR4, a MPR (read). It is guaranteed to remain at this value for a minimum of tphy_rdcsgap plus the time for the read data (usually MSTR.burst_rdwr).

The table outlines the DFI timing value and the equivalent register in the DDRC.

Table 9-3. DFI timing values for dfi_wrdata_cs/dfi_rddata_cs

DFI Timing Value	DDRC Register
$t_{phy\_wrcslat}$	DFITMG2.dfi_tphy_wrcslat
$t_{phy\_wrcsgap}$	RANKCTL.diff_rank_wr_gap [1] [2]
$t_{phy\_rdcslat}$	DFITMG2.dfi_tphy_rdcslat
$t_{phy\_rdcsgap}$	RANKCTL.diff_rank_rd_gap[1] [2]

NOTE

1. For configurations with MEMC_FREQ_RATIO=2, this register is in terms of DFI clocks (controller clocks), while tphy_wrcsgap is in terms of DFI PHY clocks.
2. If using DDR4-LRDIMM, refer to TWRWR and TRDRD timing requirements in JEDEC DDR4 Data Buffer (DDR4DB01) Specification.

9.2.2.4.6 Read Data Interface

The read data interface handles the return of read data across the DFI interface. The DFI Specification defines signals and timing relationships. This interface consists of the following signals:

- dfi_rddata
- dfi_rddata_en
- dfi_rddata_valid
- dfi_rddata_dbi (this input signal exists only if MEMC_DDR4=1)

For more information about the above signals, refer to “Signal Descriptions”. The timing of this interface is user-configurable, to support all DFI-compliant PHYs via the DFITMG0.t_rddata_en register.

For DFI 1:2, DFITMG0.dfi_rddata_use_dfi_phy_clk determines whether DFITMG0.dfi_t_rddata_en is in terms of SDR or HDR clock cycles. If HDR, timing for dfi_rddata_en generation for read commands on phase 0/phase 1 is the same and dfi_rddata* signals are aligned to HDR clock. If SDR, timing for dfi_rddata* generation for read commands on phase 0/phase 1 are treated separately and dfi_rddata* signals are not guaranteed to be aligned to HDR clock. Check your PHY requirements for correct programming.

In DDR4 configurations (when MEMC_DDR4 is set), the signal dfi_rddata_dbi is the read DBI signal. For more information on DBI, see “Data Bus Inversion (DBI)”.

The DDRC has been verified with the Synopsys PHYs. It assumes that the DFI parameter tphy_rdlat does not exceed 48 cycles. For mDDR using BL2, it assumes that the DFI parameter tphy_rdlat does not exceed 26 cycles. For both cases, it also assumes that the DFI parameter trddata_en does not exceed the SDRAM read latency (RL).

9.2.2.4.7 Update Interface

During system operation, the system may require updates to internal settings to compensate for environmental conditions. To ensure that updates do not interfere with signals on the SDRAM interface, the DFI interface supports update modes where the DFI read, write, and control interface are suspended from normal activity. The DFI Specification defines both MC-initiated and PHY-initiated updates.

9.2.2.4.7.1 MC-initiated Updates

This interface consists of the following signals:

- dfi_ctrlupd_req
- dfi_ctrlupd_ack
- dfi_ctrlupd_ack2

For more information about the above signals, refer to “Signal Descriptions”.

MC-initiated updates can be acknowledged or ignored by the PHY. DFI MC-initiated updates are performed periodically by DDRC.

9.2.2.4.7.2 PHY-initiated Updates

This interface consists of the following signals:

- dfi_phyupd_req
- dfi_phyupd_type
- dfi_phyupd_ack

For more information about the above signals, see “Signal Descriptions”.

For more details, see “DFI PHY-initiated Update Request”.

9.2.2.4.8 Status Interface

The DFI interface requires status information for initialization and clock control to the SDRAM devices. These signals are used to convey information between the MC and PHY. This interface consists of the following signals:

- dfi_init_start
- dfi_init_complete
- dfi_frequency
- dfi_dram_clk_disable
- dfi_alert_n (this signal replaces dfi_parity_error.

The DDRC does not support the following signals (defined as optional in the DFI Specification):

- dfi_data_byte_disable
- dfi_freq_ratio

9.2.2.4.8.1 Initialization

The dfi_init_start signal is used to trigger the PHY initialization by setting it to 1. It is driven by the DDRC through the APB interface with the DFIMISC.dfi_init_start read-write register field. The signal dfi_init_complete indicates that the PHY has completed its initialization. This information can be read/polled by the DDRC through the APB interface with the DFISTAT.dfi_init_complete read-only register field.

9.2.2.4.8.2 Clock Disabling

The dfi_dram_clk_disable signal depends on setting of PWRCTL.en_dfi_dram_clk_disable (see DDRC_REGS Registers). For more information about this, see “Assertion of dfi_dram_clk_disable”.

9.2.2.4.8.3 Frequency Change

The interface consists of three signals:

- dfi_init_start: Output from the controller to the PHY
- dfi_init_complete: Output from the PHY to the controller
- dfi_frequency: Output from the controller to the PHY

The procedure is as follows:

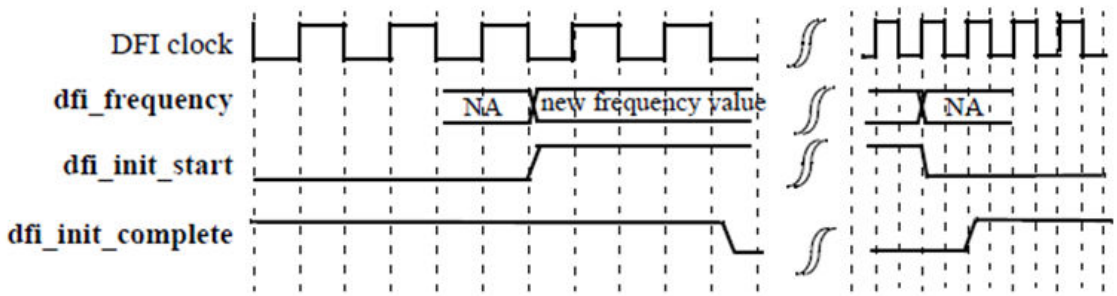


Figure 9-5. Frequency Change

dfi_frequency indicates the operating frequency of the system. This signal should change only at initialization, during a DFI frequency change operation, or other times that the system defines.This signal should be constant during normal operation.The number of supported frequencies and the mapping of signal values to clock frequencies are defined by the PHY, system, or both. For timing, the dfi_frequency signal must be set to a legal value and remain unchanged when dfi_init_start is asserted.The dfi_frequency signal can change any time when dfi_init_start is low and should be ignored at this time.

The behavior of the dfi_init_start signal is dependent on the dfi_init_complete signal.

If the PHY accepts the frequency change request, it must deassert the dfi_init_complete signal within tinit_start cycles of the dfi_init_start assertion. The MC continues to hold the dfi_init_start signal asserted until the clock frequency change has been completed. The de-assertion should be used by the PHY to reinitialize on the new clock frequency.

If the frequency change is not acknowledged (the dfi_init_complete signal remains asserted), the dfi_init_start signal must de-assert after tinit_start cycles. This requirement needs to be handled by the system as dfi_init_start is programmed through the software.

9.2.2.4.9 DFI Training Interface

The additional functions of SDRAM memories allow accurate alignment of critical timing signals. The DFI Specification version 2.1.1 accounts for these functions by providing a DFI training interface.

The DFI Specification defines the following operating modes for the training interface for write leveling, and read leveling:

- No Support
- MC Evaluation (not supported by the DDRC)

- PHY Evaluation
- PHY Independent

  These modes define whether the MC or PHY, or neither, maintains the responsibility for managing the programming of the delay lines and evaluating the response.

  The DFI training interface enables write and read leveling functions in a DFI-compliant PHY.

  In DDRC, the following modes are supported:

- “No Support and PHY Independent Mode”

While interfacing to the DDR PHY, the training is initiated by the PHY Utility Block (PUB) present in the PHY, and PHY independent training mode is used.

9.2.2.4.9.1 No Support and PHY Independent Mode

To enable the No Support and PHY independent mode, the PHY must drive the following signals to “11”:

- dfi_rdlvl_mode
- dfi_rdlvl_gate_mode
- dfi_wrlvl_mode

The above signals are inputs from the DFI-compliant PHY. For these modes, any other DFI Training signals which are outputs can be ignored while any other DFI Training signals that are inputs must be tied to all zeros.

Training is then performed automatically by the PHY, without any intervention from the DDRC. The PUB or PUBL block (part of the PHY) performs all required MRS commands to move the SDRAM to the correct state for each training stage.

If you are using a non-Synopsys PHY that supports PHY independent training mode, DDRC may have to perform these MRS commands.

However, DDRC has no knowledge on when the PHY is ready to execute each training stage, so this must be controlled by the user. For more information on mode register writes, see “Mode Register Write".

9.2.2.4.10 Low Power interface

In a DDR memory subsystem, it is advantageous to place the PHY in a low power state when the DDRC has knowledge that the memory subsystem remains idle for a period of time. Depending on the state of the system, the DDRC communicates state information to the PHY allowing the PHY to enter the appropriate power saving state. This interface consists of signals that are used to inform the PHY of a low power mode opportunity, as well as how quickly the DDRC requires the PHY to resume normal operation. This interface consists of the following signals:

- dfi_lp_req
- dfi_lp_wakeup
- dfi_lp_ack Software control is provided in DFILPCFG0/DFILPCFG1 registers (see DDRC_REGS Registers on page 464) for the following:
- This is an optional interface for a DFI-compliant PHY. The interface signals always exist in the DDRC. But the software can enable this interface in self-refresh and/or power-down and/or deep power-down mode and/or maximum power saving mode - DFILPCFG0.dfi_lp_en_pd, DFILPCFG0.dfi_lp_en_sr, DFILPCFG0.dfi_lp_en_dpd and DFILPCFG1.dfi_lp_en_mpsm respectively.
- Software control over what is driven on dfi_lp_wakeup signal (and its associated timing) in selfrefresh or power-down or deep power-down or maximum power saving mode - DFILPCFG0.dfi_lp_wakeup_pd, DFILPCFG0.dfi_lp_wakeup_sr, DFILPCFG0.dfi_lp_wakeup_dpd and DFILPCFG1.dfi_lp_wakeup_mpsm respectively.
- The timing of this DFI low power interface - DFILPCFG0.dfi_tlp_resp.
- ctl_idle output pin is a sideband signal (not specified in DFI) supported by certain Synopsys PHYs to trigger the PHY's Anti-Aging feature. This is driven the same as dfi_lp_req if enabled via software (DFIMISC.ctl_idle_en=1 and DFI Low Power Interface is also enabled via DFILPCFG0/DFILPCFG1). If configured for Shared AC, dfi_lp_req signal of both data channels must be asserted for ctl_idle to be asserted.

For more information, see "Power Saving Features".

9.2.2.4.11 PHY Master Interface

This interface consists of the following signals:

- dfi_phymstr_req
- dfi_phymstr_cs_state
- dfi_phymstr_state_sel
- dfi_phymstr_type
- dfi_phymstr_ack

For more information about the above signals, see "Signal Descriptions".

For more information about PHY Master interface, see “PHY Master Interface”.

9.2.2.5 APB Interface

The DDRC provides a dedicated APB 3.0 bus interface that is used to access the DDRC software programmable registers. The DDRC converts APB reads and writes into accesses to the internal register file. The APB data width is fixed to 32 bits for compatibility reasons with the DDR PHYs. The APB address width is 12 bits.

All APB interface signals (p*) are synchronous to pclk. pclk can be asynchronous to the core_ddrc_core_clk if the configuration parameter DDRC_P_ASYNC_EN is set. However, pclk frequency must be the same or less than the core_ddrc_core_clk frequency.

9.2.2.5.1 Register Classes

There are three classes of registers in the DDRC:

- Dynamic Registers
- Quasi Dynamic Registers
- Static Registers

Dynamic registers can be written at any time during operation.

Static registers can be written only when the controller is in reset.

Quasi dynamic registers can be written when the controller is in reset and some specific conditions outside reset. To write them, SWCTL.sw_done (see DDRC_REGS Registers on page 464) has to be set to 0 at the beginning of the programming sequence and set back to 1 at the end. After that, SWSTAT.sw_done_ack has to be polled for acknowledge.

9.2.3 Memory Map and register definition

This section includes the DDRC module memory map and detailed descriptions of all registers.

9.2.3.1 DDRC register descriptions

9.2.3.1.1 DDRC memory map

DDRC base address: 0h

Offset	Register	Width (In bits)	Access	Reset value
0h	Master Register0 (MSTR)	32	RW	0304_0001h
4h	Operating Mode Status Register (STAT)	32	RO	0000_0000h
8h	Operating Mode Status Register (MSTR1)	32	RW	0000_0000h
Ch	Operating Mode Status Register (MRCTRL3)	32	RW	0000_0003h
10h	Mode Register Read/Write Control Register 0. (MRCTRL0)	32	RW	0000_0030h
14h	Mode Register Read/Write Control Register 1 (MRCTRL1)	32	RW	0000_0000h
18h	Mode Register Read/Write Status Register (MRSTAT)	32	RO	0000_0000h
1Ch	Mode Register Read/Write Control Register 2 (MRCTRL2)	32	RW	0000_0000h
20h	Temperature Derate Enable Register (DERATEEN)	32	RW	0000_0000h
24h	Temperature Derate Interval Register (DERATEINT)	32	RW	0080_0000h
30h	Low Power Control Register (PWRCTL)	32	RW	0000_0000h
34h	Low Power Timing Register (PWRTMG)	32	RW	0040_2010h
38h	Hardware Low Power Control Register (HWLPCTL)	32	RW	0000_0003h
50h	Refresh Control Register 0 (RFSHCTL0)	32	RW	0021_0000h
54h	Refresh Control Register 1 (RFSHCTL1)	32	RW	0000_0000h
60h	Refresh Control Register 3 (RFSHCTL3)	32	RW	0000_0000h
64h	Refresh Timing Register (RFSHTMG)	32	RW	0062_008Ch
D0h	SDRAM Initialization Register 0 (INIT0)	32	RW	0002_004Eh
D4h	SDRAM Initialization Register 1 (INIT1)	32	RW	0000_0000h
D8h	SDRAM Initialization Register 2 (INIT2)	32	RW	0000_0D05h
DCh	SDRAM Initialization Register 3 (INIT3)	32	RW	0000_0510h
E0h	SDRAM Initialization Register 4 (INIT4)	32	RW	0000_0000h
E4h	SDRAM Initialization Register 5 (INIT5)	32	RW	0010_0004h
E8h	SDRAM Initialization Register 6 (INIT6)	32	RW	0000_0000h
ECh	SDRAM Initialization Register 7 (INIT7)	32	RW	0000_0000h
F0h	DIMM Control Register (DIMMCTL)	32	RW	0000_0000h
F4h	Rank Control Register (RANKCTL)	32	RW	0000_066Fh
100h	SDRAM Timing Register 0 (DRAMTMG0)	32	RW	0F10_1B0Fh
104h	SDRAM Timing Register 1 (DRAMTMG1)	32	RW	0008_0414h
108h	SDRAM Timing Register 2 (DRAMTMG2)	32	RW	0305_060Dh
10Ch	SDRAM Timing Register 3 (DRAMTMG3)	32	RW	0050_400Ch
110h	SDRAM Timing Register 4 (DRAMTMG4)	32	RW	0504_0405h
114h	SDRAM Timing Register 5 (DRAMTMG5)	32	RW	0505_0403h
118h	SDRAM Timing Register 6 (DRAMTMG6)	32	RW	0202_0005h
11Ch	SDRAM Timing Register 7 (DRAMTMG7)	32	RW	0000_0202h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
120h	SDRAM Timing Register 8 (DRAMTMG8)	32	RW	0303_4405h
124h	SDRAM Timing Register 9 (DRAMTMG9)	32	RW	0004_040Dh
128h	SDRAM Timing Register 10 (DRAMTMG10)	32	RW	001C_180Ah
12Ch	SDRAM Timing Register 11 (DRAMTMG11)	32	RW	440C_021Ch
130h	SDRAM Timing Register 12 (DRAMTMG12)	32	RW	0002_0610h
134h	SDRAM Timing Register 13 (DRAMTMG13)	32	RW	1C20_0004h
138h	SDRAM Timing Register 14 (DRAMTMG14)	32	RW	0000_00A0h
13Ch	SDRAM Timing Register 15 (DRAMTMG15)	32	RW	0000_0000h
180h	ZQ Control Register 0 (ZQCTL0)	32	RW	0200_0040h
184h	ZQ Control Register 1 (ZQCTL1)	32	RW	0200_0100h
188h	ZQ Control Register 2 (ZQCTL2)	32	RW	0000_0000h
18Ch	ZQ Status Register (ZQSTAT)	32	RO	0000_0000h
190h	DFI Timing Register 0 (DFITMG0)	32	RW	0702_0002h
194h	DFI Timing Register 1 (DFITMG1)	32	RW	0000_0404h
198h	DFI Low Power Configuration Register 0 (DFILPCFG0)	32	RW	0700_0000h
19Ch	DFI Low Power Configuration Register 1 (DFILPCFG1)	32	RW	0000_0000h
1A0h	DFI Update Register 0 (DFIUPD0)	32	RW	0040_0003h
1A4h	DFI Update Register 1 (DFIUPD1)	32	RW	0001_0001h
1A8h	DFI Update Register 2 (DFIUPD2)	32	RW	8000_0000h
1B0h	DFI Miscellaneous Control Register (DFIMISC)	32	RW	0000_0001h
1B4h	DFI Timing Register 2 (DFITMG2)	32	RW	0000_0202h
1B8h	DFI Timing Register 3 (DFITMG3)	32	RW	0000_0000h
1BCh	DFI Status Register (DFISTAT)	32	RO	0000_0000h
1C0h	DM/DBI Control Register (DBICTL)	32	RW	0000_0001h
200h	Address Map Register 0 (ADDRMAP0)	32	RW	0000_0000h
204h	Address Map Register 1 (ADDRMAP1)	32	RW	0000_0000h
208h	Address Map Register 2 (ADDRMAP2)	32	RW	0000_0000h
20Ch	Address Map Register 3 (ADDRMAP3)	32	RW	0000_0000h
210h	Address Map Register 4 (ADDRMAP4)	32	RW	0000_0000h
214h	Address Map Register 5 (ADDRMAP5)	32	RW	0000_0000h
218h	Address Map Register 6 (ADDRMAP6)	32	RW	0000_0000h
21Ch	Address Map Register 7 (ADDRMAP7)	32	RW	0000_0000h
220h	Address Map Register 8 (ADDRMAP8)	32	RW	0000_0000h
224h	Address Map Register 9 (ADDRMAP9)	32	RW	0000_0000h
228h	Address Map Register 10 (ADDRMAP10)	32	RW	0000_0000h
22Ch	Address Map Register 11 (ADDRMAP11)	32	RW	0000_0000h
240h	ODT Configuration Register (ODTCFG)	32	RW	0400_0400h
244h	ODT/Rank Map Register (ODTMAP)	32	RW	0000_2211h
250h	Scheduler Control Register (SCHED)	32	RW	0000_1005h
254h	Scheduler Control Register 1 (SCHED1)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
25Ch	High Priority Read CAM Register 1 (PERFHPR1)	32	RW	0F00_0001h
264h	Low Priority Read CAM Register 1 (PERFLPR1)	32	RW	0F00_007Fh
26Ch	Write CAM Register 1 (PERFWR1)	32	RW	0F00_007Fh
300h	Debug Register 0 (DBG0)	32	RW	0000_0000h
304h	Debug Register 1 (DBG1)	32	RW	0000_0000h
308h	CAM Debug Register (DBGCAM)	32	RO	0000_0000h
30Ch	Command Debug Register (DBGCMD)	32	RW	0000_0000h
310h	Status Debug Register (DBGSTAT)	32	RO	0000_0000h
320h	Software Register Programming Control Enable (SWCTL)	32	RW	0000_0001h
324h	Software Register Programming Control Status (SWSTAT)	32	RO	0000_0001h
36Ch	AXI Poison Configuration Register. (POISONCFG)	32	RW	0011_0011h
370h	AXI Poison Status Register (POISONSTAT)	32	RO	0000_0000h
3FCh	Port Status Register (PSTAT)	32	RO	0000_0000h
400h	Port Common Configuration Register (PCCFG)	32	RW	0000_0000h
404h	Port n Configuration Read Register (PCFGR_0)	32	RW	0000_0000h
408h	Port n Configuration Write Register (PCFGW_0)	32	RW	0000_4000h
490h	Port n Control Register (PCTRL_0)	32	RW	0000_0000h
494h	Port n Read QoS Configuration Register 0 (PCFGQOS0_0)	32	RW	0000_0000h
498h	Port n Read QoS Configuration Register 1 (PCFGQOS1_0)	32	RW	0000_0000h
49Ch	Port n Write QoS Configuration Register 0 (PCFGWQOS0_0)	32	RW	0000_0000h
4A0h	Port n Write QoS Configuration Register 1 (PCFGWQOS1_0)	32	RW	0000_0000h
2020h	[SHADOW] Temperature Derate Enable Register (DERATEEN_SHADOW)	32	RW	0000_0000h
2024h	[SHADOW] Temperature Derate Interval Register (DERATEINT_SHADOW)	32	RW	0080_0000h
2050h	[SHADOW] Refresh Control Register 0 (RFSHCTL0_SHADOW)	32	RW	0021_0000h
2064h	[SHADOW] Refresh Timing Register (RFSHTMG_SHADOW)	32	RW	0062_008Ch
20DCh	[SHADOW] SDRAM Initialization Register 3 (INIT3_SHADOW)	32	RW	0000_0510h
20E0h	[SHADOW] SDRAM Initialization Register 4 (INIT4_SHADOW)	32	RW	0000_0000h
20E8h	[SHADOW] SDRAM Initialization Register 6 (INIT6_SHADOW)	32	RW	0000_0000h
20ECh	[SHADOW] SDRAM Initialization Register 7 (INIT7_SHADOW)	32	RW	0000_0000h
2100h	[SHADOW] SDRAM Timing Register 0 (DRAMTMG0_SHADOW)	32	RW	0F10_1B0Fh
2104h	[SHADOW] SDRAM Timing Register 1 (DRAMTMG1_SHADOW)	32	RW	0008_0414h
2108h	[SHADOW] SDRAM Timing Register 2 (DRAMTMG2_SHADOW)	32	RW	0305_060Dh
210Ch	[SHADOW] SDRAM Timing Register 3 (DRAMTMG3_SHADOW)	32	RW	0050_400Ch
2110h	[SHADOW] SDRAM Timing Register 4 (DRAMTMG4_SHADOW)	32	RW	0504_0405h
2114h	[SHADOW] SDRAM Timing Register 5 (DRAMTMG5_SHADOW)	32	RW	0505_0403h
2118h	[SHADOW] SDRAM Timing Register 6 (DRAMTMG6_SHADOW)	32	RW	0202_0005h
211Ch	[SHADOW] SDRAM Timing Register 7 (DRAMTMG7_SHADOW)	32	RW	0000_0202h
2120h	[SHADOW] SDRAM Timing Register 8 (DRAMTMG8_SHADOW)	32	RW	0303_4405h

Table continues on the next page...

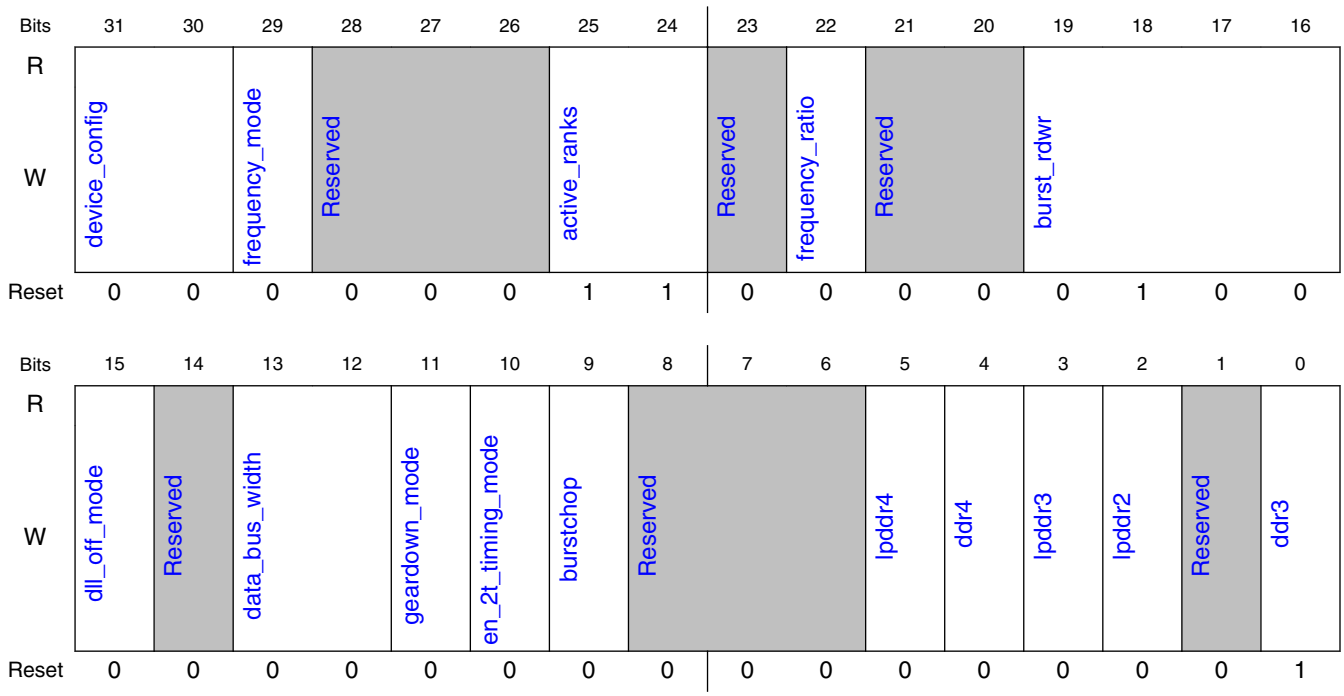
Offset	Register	Width (In bits)	Access	Reset value
2124h	[SHADOW] SDRAM Timing Register 9 (DRAMTMG9_SHADOW)	32	RW	0004_040Dh
2128h	[SHADOW] SDRAM Timing Register 10 (DRAMTMG10_SHADOW)	32	RW	001C_180Ah
212Ch	[SHADOW] SDRAM Timing Register 11 (DRAMTMG11_SHADOW)	32	RW	440C_021Ch
2130h	[SHADOW] SDRAM Timing Register 12 (DRAMTMG12_SHADOW)	32	RW	0002_0610h
2134h	[SHADOW] SDRAM Timing Register 13 (DRAMTMG13_SHADOW)	32	RW	1C20_0004h
2138h	[SHADOW] SDRAM Timing Register 14 (DRAMTMG14_SHADOW)	32	RW	0000_00A0h
213Ch	[SHADOW] SDRAM Timing Register 15 (DRAMTMG15_SHADOW)	32	RW	0000_0000h
2180h	[SHADOW] ZQ Control Register 0 (ZQCTL0_SHADOW)	32	RW	0200_0040h
2190h	[SHADOW] DFI Timing Register 0 (DFITMG0_SHADOW)	32	RW	0702_0002h
2194h	[SHADOW] DFI Timing Register 1 (DFITMG1_SHADOW)	32	RW	0000_0404h
21B4h	[SHADOW] DFI Timing Register 2 (DFITMG2_SHADOW)	32	RW	0000_0202h
21B8h	[SHADOW] DFI Timing Register 3 (DFITMG3_SHADOW)	32	RW	0000_0000h
2240h	[SHADOW] ODT Configuration Register (ODTCFG_SHADOW)	32	RW	0400_0400h

9.2.3.1.2 Master Register0 (MSTR)

9.2.3.1.2.1 Offset

Register	Offset
MSTR	0h

9.2.3.1.2.2 Diagram



9.2.3.1.2.3 Fields

Field	Function
31-30 device_config	Indicates the configuration of the device used in the system. 00b - x4 device 01b - x8 device 10b - x16 device 11b - x32 device
29 frequency_mode	Choose which registers are used. 0b - Original Registers 1b - Shadow Registers
28-26 —	Reserved
25-24 active_ranks	Only present for multi-rank configurations. Each bit represents one rank. For two-rank configurations, only bits[25:24] are present. • 1 - populated • 0 - unpopulated LSB is the lowest rank number. For 2 ranks following combinations are legal • 01 - One rank • 11 - Two ranks • Others- Reserved For 4 ranks following combinations are legal:

Table continues on the next page...

Field	Function
	• 0001 - One rank • 0011 - Two ranks • 1111 - Four ranks •
23 —	Reserved
22 frequency_ratio	Selects the Frequency Ratio 0b - 1:2 Mode 1b - 1:1 Mode
21-20 —	Reserved
19-16 burst_rdwr	SDRAM burst length used All other values are reserved. This controls the burst size used to access the SDRAM. This must match the burst length mode register setting in the SDRAM. (For BC4/8 on-the-fly mode of DDR3 and DDR4, set this field to 0x0100) Burst length of 2 is not supported with AXI ports when MEMC_BURST_LENGTH is 8. Burst length of 2 is only supported when the controller is operating in 1:1 frequency mode 0001b - Burst length of 2 (only supported for mDDR) 0010b - Burst length of 4 0100b - Burst length of 8 1000b - Burst length of 16 (only supported for mDDR, LPDDR2, and LPDDR4)
15 dll_off_mode	Set to 1 when the DDRC and DRAM has to be put in DLL-off mode for low frequency operation. Set to 0 to put DDRC and DRAM in DLL-on mode for normal frequency operation. If DDR4 CRC/parity retry is enabled (CRCPARCTL1.crc_parity_retry_enable = 1), dll_off_mode is not supported, and this bit must be set to '0'.
14 —	Reserved
13-12 data_bus_width	Selects proportion of DQ bus width that is used by the SDRAM - 00 - Full DQ bus width to SDRAM - 01 - Half DQ bus width to SDRAM - 10 - Quarter DQ bus width to SDRAM - 11 - Reserved Note that half bus width mode is only supported when the SDRAM bus width is a multiple of 16, and quarter bus width mode is only supported when the SDRAM bus width is a multiple of 32 and the configuration parameter MEMC_QBUS_SUPPORT is set. Bus width refers to DQ bus width (excluding any ECC width).
11 geardown_mode	1 indicates put the DRAM in geardown mode (2N) and 0 indicates put the DRAM in normal mode (1N). This register can be changed, only when the Controller is in self-refresh mode. This signal must be set the same value as MR3 bit A3. Note: Geardown mode is not supported if the configuration parameter MEMC_CMD_RTN2IDLE is set Note: Geardown mode is not supported if the configuration parameter DDRC_SHARED_AC is set (in Shared-AC mode) and the register value is don't care
10 en_2t_timing_mode	If 1, then DDRC uses 2T timing. Otherwise, uses 1T timing. In 2T timing, all command signals (except chip select) are held for 2 clocks on the SDRAM bus. Chip select is asserted on the second cycle of the command Note: 2T timing is not supported in LPDDR2/LPDDR3/LPDDR4 mode Note: 2T timing is not supported if the configuration parameter MEMC_CMD_RTN2IDLE is set Note: 2T timing is not supported in DDR4 geardown mode. Note: 2T timing is not supported in Shared-AC dual channel mode and the register value is don't care.
9 burstchop	When set, enable burst-chop (BC4 or 8 on-the-fly) in DDR3/DDR4. Burst Chop for Reads is exercised only in HIF configurations (DDRC_INCL_ARB not set) and if in full bus width mode (MSTR.data_bus_width = 00) and if MEMC_BURST_LENGTH=8 or 16. Burst Chop for Writes is exercised only if Partial Writes enabled (DDRC_PARTIAL_WR=1) and if CRC is disabled (CRCPARCTL1.crc_enable = 0). If DDR4 CRC/parity retry is enabled (CRCPARCTL1.crc_parity_retry_enable = 1), burst chop is not supported, and this bit must be set to '0'. BC4 (fixed) mode is not supported.
8-6	Reserved

Table continues on the next page...

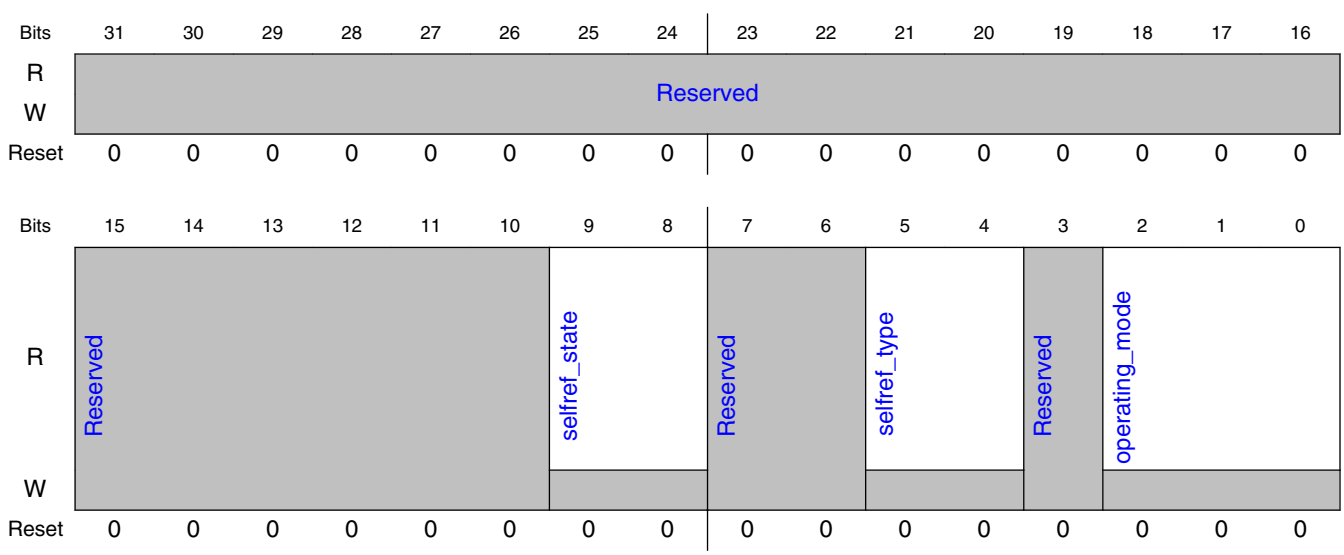
Field	Function
—	
5 lpddr4	Select LPDDR4 SDRAM - 1 - LPDDR4 SDRAM device in use. - 0 - non-LPDDR4 device in use Present only in designs configured to support LPDDR4.
4 ddr4	Select DDR4 SDRAM - 1 - DDR4 SDRAM device in use. - 0 - non-DDR4 device in use Present only in designs configured to support DDR4.
3 lpddr3	Select LPDDR3 SDRAM - 1 - LPDDR3 SDRAM device in use. - 0 - non-LPDDR3 device in use Present only in designs configured to support LPDDR3.
2 lpddr2	Select LPDDR2 SDRAM - 1 - LPDDR2 SDRAM device in use. - 0 - non-LPDDR2 device in use Present only in designs configured to support LPDDR2.
1 —	Reserved
0 ddr3	Select DDR3 SDRAM - 1 - DDR3 SDRAM device in use - 0 - non-DDR3 SDRAM device in use Only present in designs that support DDR3.

9.2.3.1.3 Operating Mode Status Register (STAT)

9.2.3.1.3.1 Offset

Register	Offset
STAT	4h

9.2.3.1.3.2 Diagram



9.2.3.1.3.3 Fields

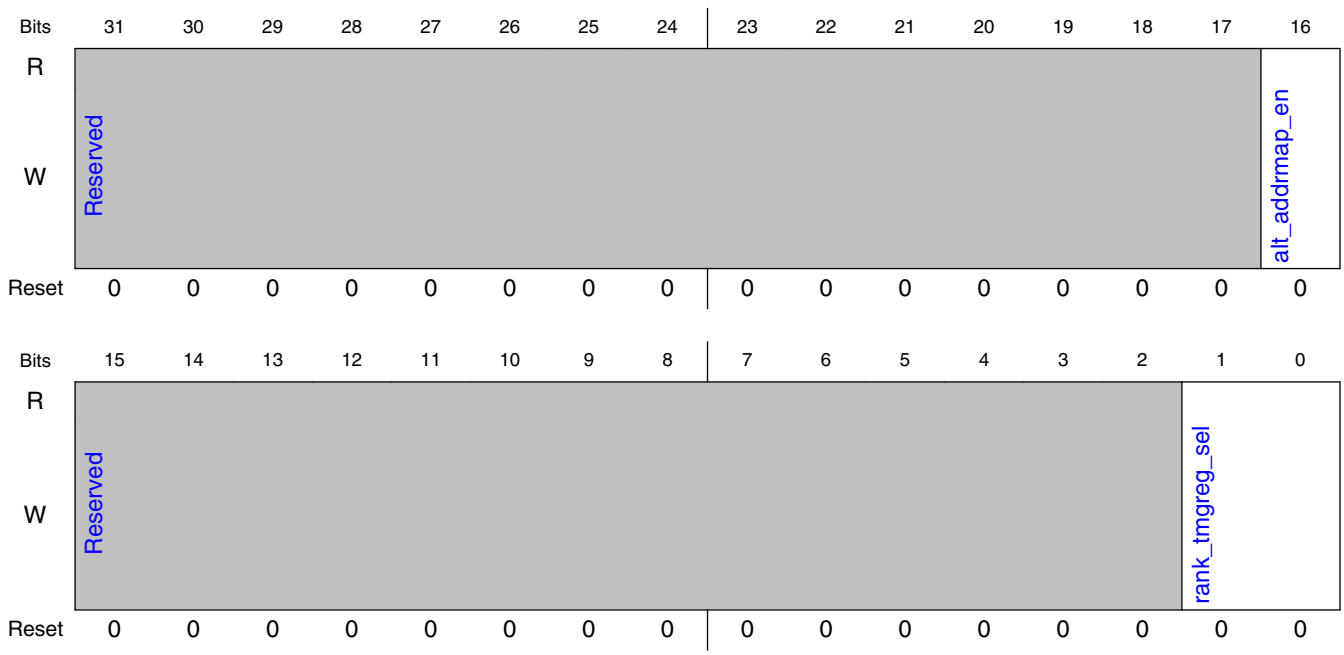
Field	Function
31-10 —	Reserved
9-8 selfref_state	Self refresh state. This indicates self refresh or self refresh power down state for LPDDR4. This register is used for frequency change and MRR/MRW access during self refresh. 00b - SDRAM is not in Self Refresh. 01b - Self refresh 1 10b - Self refresh power down 11b - Self refresh
7-6 —	Reserved
5-4 selfref_type	Flags if Self Refresh (except LPDDR4) or SR-Powerdown (LPDDR4) is entered and if it was under Automatic Self Refresh control only or not. 00b - SDRAM is not in Self Refresh (except LPDDR4) or SR-Powerdown (LPDDR4). If retry is enabled by CRCPARCTRL1.crc_parity_retry_enable, this also indicates SRE command is still in parity error window or retry is in-progress. 10b - SDRAM is in Self Refresh (except LPDDR4) or SR-Powerdown (LPDDR4), which was not caused solely under Automatic Self Refresh control. It could have been caused by Hardware Low Power Interface and/or Software (reg_ddrc_selfref_sw). If retry is enabled, this guarantees SRE command is executed correctly without parity 11b - SDRAM is in Self Refresh (except LPDDR4) or SR-Powerdown (LPDDR4), which was caused by Automatic Self Refresh only. If retry is enabled, this guarantees SRE command is executed correctly without parity error.
3 —	Reserved
2-0 operating_mode	Operating mode This is 3-bits wide in configurations with mDDR/LPDDR2/LPDDR3/LPDDR4/DDR4 support and 2-bits in all other configurations. Non-mDDR/LPDDR2/LPDDR3/LPDDR4 and non-DDR4 designs: • 00 - Init • 01 - Normal • 10 - Power-down • 11 - Self refresh mDDR/LPDDR2/LPDDR3 or DDR4 designs: • 000 - Init • 001 - Normal • 010 - Power-down • 011 - Self refresh • 1XX - Deep power-down / Maximum Power Saving Mode LPDDR4 designs: • 000 - Init • 001 - Normal • 010 - Power-down • 011 - Self refresh / Self refresh power-down

9.2.3.1.4 Operating Mode Status Register (MSTR1)

9.2.3.1.4.1 Offset

Register	Offset
MSTR1	8h

9.2.3.1.4.2 Diagram



9.2.3.1.4.3 Fields

Field	Function
31-17 —	Reserved
16 alt_addrmap_en	Enable Alternative Address Map 0b - Disable Alternative Address Map 1b - Enable Alternative Address Map
15-2 —	Reserved
1-0 rank_tmgreg_sel	rank_tmgreg_sel Indicates which register set is used for each rank. Each bit represents one rank. (LSB is the lowest rank number.)

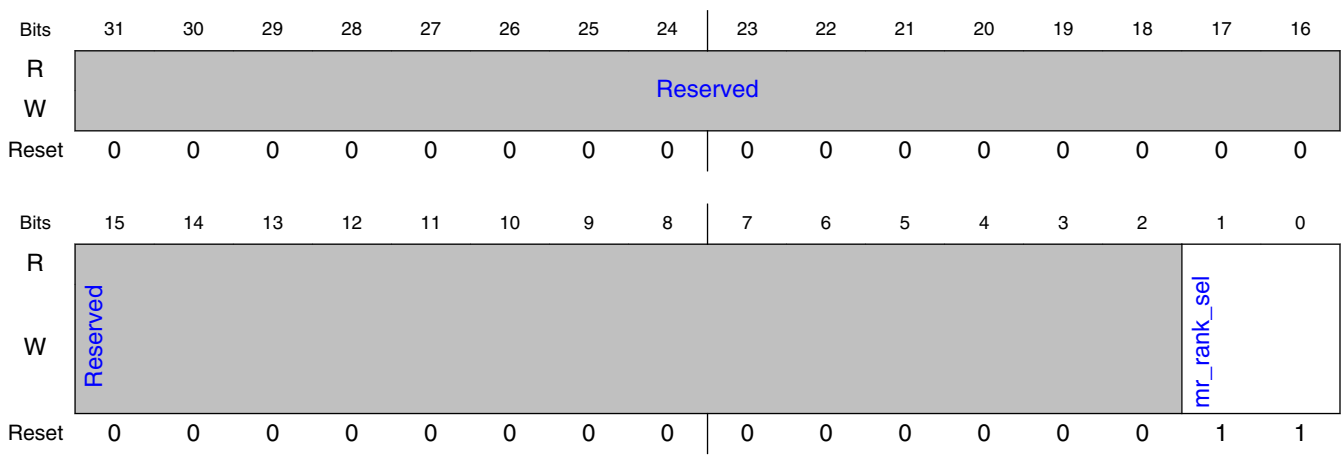
Field	Function
	NOTE: Only some of timing registers in DRAMTMGx are duplicated as MRAMTMGx for MRAM and its width is expanded for MRAM. This register can switch the duplicate registers, otherwise use DRAMTMGx. This register is only for DDR4 ST-MRAM rank, otherwise must be set to 0. This register must be set up while the Controller is in reset. 00b - USE DRAMTMGx registers for the rank 01b - USE MRAMTMGx registers for the rank

9.2.3.1.5 Operating Mode Status Register (MRCTRL3)

9.2.3.1.5.1 Offset

Register	Offset
MRCTRL3	Ch

9.2.3.1.5.2 Diagram



9.2.3.1.5.3 Fields

Field	Function
31-2 —	Reserved
1-0 mr_rank_sel	mr_rank_sel Controls which rank is accessed by MRCTRL0.mr_wr.

Field	Function
	Normally, it is desired to access all ranks, so all bits should be set to 1. However, for multi-rank UDIMMs/ RDIMMs/LRDIMMs which implement address mirroring, it may be necessary to access ranks individually. Examples (assume uMCTL2 is configured for 8 ranks): • 0x01 - select rank 0 only • 0x02 - select rank 1 only • 0x05 - select ranks 0 and 2

9.2.3.1.6 Mode Register Read/Write Control Register 0. (MRCTRL0)

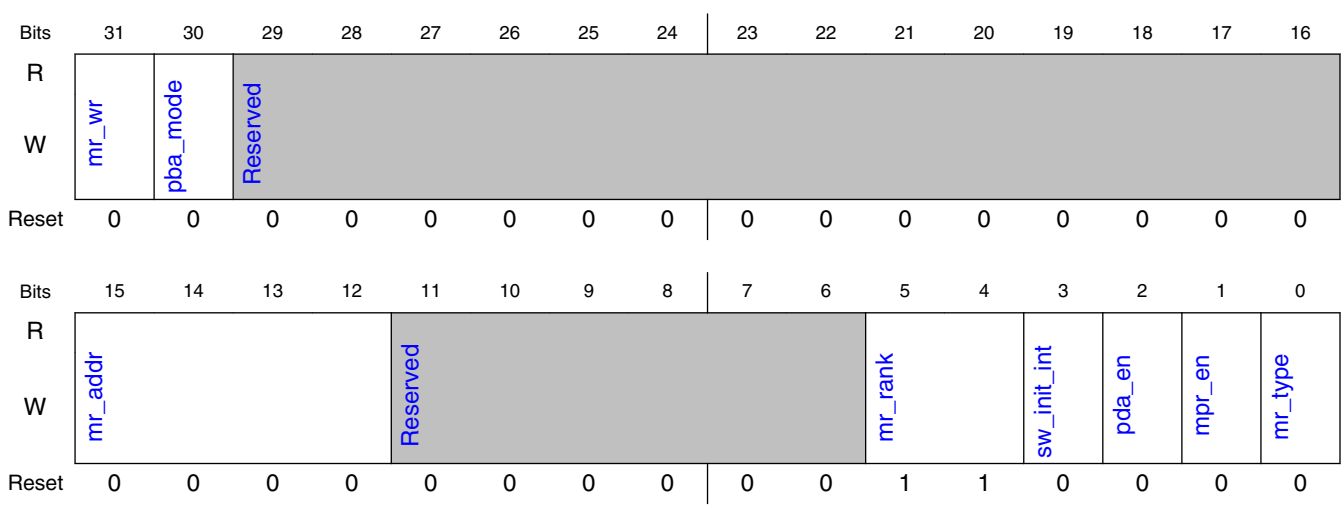
9.2.3.1.6.1 Offset

Register	Offset
MRCTRL0	10h

9.2.3.1.6.2 Function

Mode Register Read/Write Control Register 0. Note: Do not enable more than one of the following fields simultaneously: - sw_init_int - pda_en - mpr_en

9.2.3.1.6.3 Diagram



9.2.3.1.6.4 Fields

Field	Function
31 mr_wr	Setting this register bit to 1 triggers a mode register read or write operation. When the MR operation is complete, the DDRC automatically clears this bit. The other register fields of this register must be written in a separate APB transaction, before setting this mr_wr bit. It is recommended NOT to set this signal if in Init, Deep power-down or MPSM operating modes.
30 pba_mode	Indicates whether PBA access is executed. When setting this bit to 1 along with setting pda_en to 1, DDRC initiates PBA access instead of PDA access. - 0 - Per DRAM Addressability mode - 1 - Per Buffer Addressability mode The completion of PBA access is confirmed by MRSTAT.pda_done in the same way as PDA.
29-16 —	Reserved
15-12 mr_addr	Address of the mode register that is to be written to. Don't Care for LPDDR2/LPDDR3/LPDDR4 (see MRCTRL1.mr_data for mode register addressing in LPDDR2/LPDDR3/LPDDR4) This signal is also used for writing to control words of the register chip on RDIMMs/LRDIMMs. In that case, it corresponds to the bank address bits sent to the RDIMM/LRDIMM In case of DDR4, the bit[3:2] corresponds to the bank group bits. Therefore, the bit[3] as well as the bit[2:0] must be set to an appropriate value which is considered both the Address Mirroring of UDIMMs/RDIMMs/LRDIMMs and the Output Inversion of RDIMMs/LRDIMMs. 0000b - MR0 0001b - MR1 0010b - MR2 0011b - MR3 0100b - MR4 0101b - MR5 0110b - MR6 0111b - MR7
11-6 —	Reserved
5-4 mr_rank	Controls which rank is accessed by MRCTRL0.mr_wr. Normally, it is desired to access all ranks, so all bits should be set to 1. However, for multi-rank UDIMMs/RDIMMs/LRDIMMs which implement address mirroring, it may be necessary to access ranks individually. Examples (assume DDRC is configured for 4 ranks): 0x1 - select rank 0 only 0x2 - select rank 1 only 0x5 - select ranks 0 and 2 0xA - select ranks 1 and 3 0xF - select ranks 0, 1, 2 and 3
3 sw_init_int	Indicates whether Software intervention is allowed via MRCTRL0/MRCTRL1 before automatic SDRAM initialization routine or not. For DDR4, this bit can be used to initialize the DDR4 RCD (MR7) before automatic SDRAM initialization. For LPDDR4, this bit can be used to program additional mode registers before automatic SDRAM initialization if necessary. In LPDDR4 independent channel mode, note that this must be programmed to both channels beforehand. Note that this must be cleared to 0 after completing Software operation. Otherwise, SDRAM initialization routine will not re-start. 0b - Software intervention is not allowed 1b - Software intervention is allowed
2 pda_en	Indicates whether the mode register operation is MRS in PDA mode or not. Note that when pba_mode=1, PBA access is initiated instead of PDA access. 0b - MRS 1b - MRS in Per DRAM Addressability
1 mpr_en	Indicates whether the mode register operation is MRS or WR/RD for MPR (only supported for DDR4). 0b - MRS 1b - WR/RD for MPR
0 mr_type	Indicates whether the mode register operation is read or write. Only used for LPDDR2/LPDDR3/LPDDR4/DDR4.

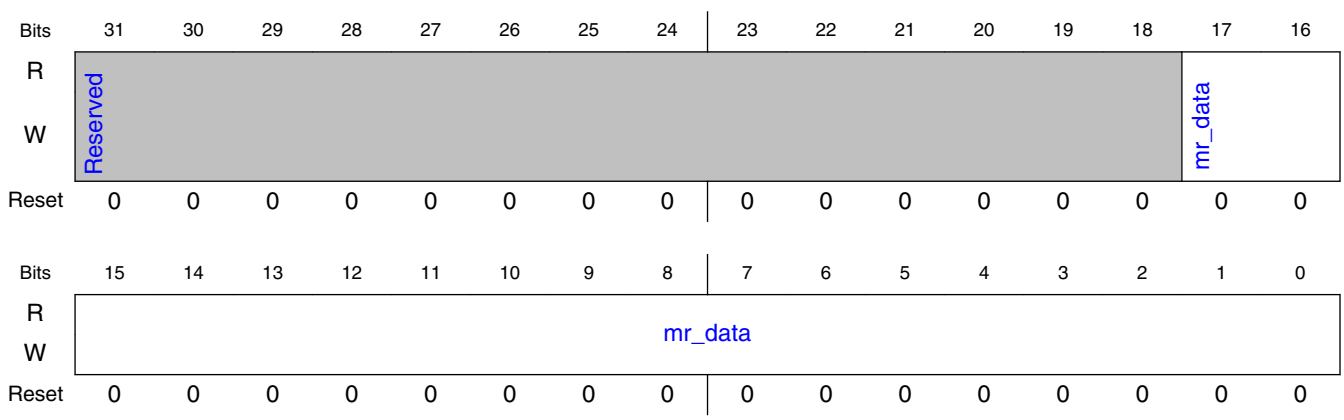
Field	Function
	0b - Write 1b - Read

9.2.3.1.7 Mode Register Read/Write Control Register 1 (MRCTRL1)

9.2.3.1.7.1 Offset

Register	Offset
MRCTRL1	14h

9.2.3.1.7.2 Diagram



9.2.3.1.7.3 Fields

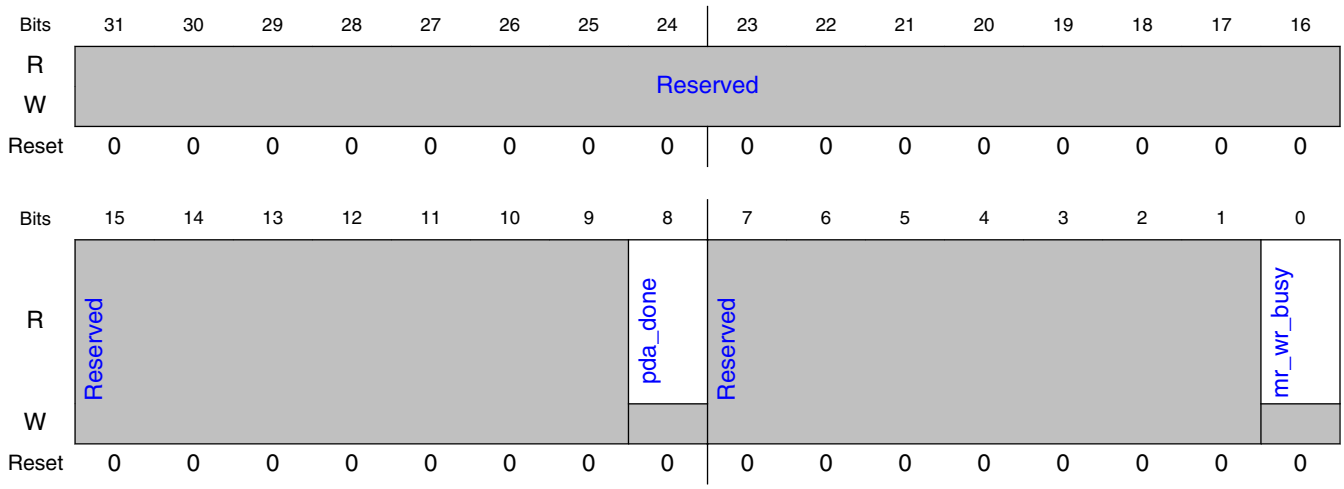
Field	Function
31-18 —	Reserved
17-0 mr_data	Mode register write data for all non-LPDDR2/non-LPDDR3/non-LPDDR4 modes. For LPDDR2/LPDDR3/LPDDR4, MRCTRL1[15:0] are interpreted as [15:8] MR Address [7:0] MR data for writes, don't care for reads. This is 18-bits wide in configurations with DDR4 support and 16-bits in all other configurations.

9.2.3.1.8 Mode Register Read/Write Status Register (MRSTAT)

9.2.3.1.8.1 Offset

Register	Offset
MRSTAT	18h

9.2.3.1.8.2 Diagram



9.2.3.1.8.3 Fields

Field	Function
31-9 —	Reserved
8 pda_done	The SoC core may initiate a MR write operation in PDA/PBA mode only if this signal is low. This signal goes high when three consecutive MRS commands related to the PDA/PBA mode are issued to the SDRAM. This signal goes low when MRCTRL0.pda_en becomes 0. Therefore, it is recommended to write MRCTRL0.pda_en to 0 after this signal goes high in order to prepare to perform PDA operation next time 0b - Indicates that mode register write operation related to PDA/PBA is in progress or has not started yet. 1b - Indicates that mode register write operation related to PDA/PBA has competed.
7-1 —	Reserved
0 mr_wr_busy	The SoC core may initiate a MR write operation only if this signal is low. This signal goes high in the clock after the DDRC accepts the MRW/MRR request. It goes low when the MRW/MRR command is issued to the SDRAM. It is recommended not to perform MRW/MRR commands when 'MRSTAT.mr_wr_busy' is high. 0b - Indicates that the SoC core can initiate a mode register write operation 1b - Indicates that mode register write operation is in progress

9.2.3.1.9 Mode Register Read/Write Control Register 2 (MRCTRL2)

9.2.3.1.9.1 Offset

Register	Offset
MRCTRL2	1Ch

9.2.3.1.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	mr_device_sel															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	mr_device_sel															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.9.3 Fields

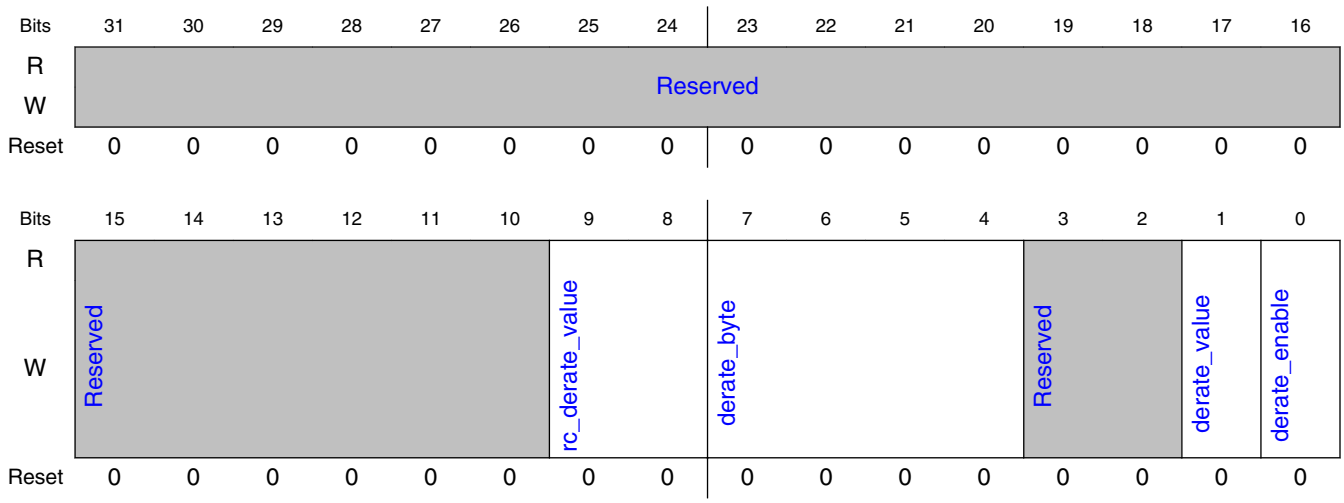
Field	Function
31-0 mr_device_sel	Indicates the device(s) to be selected during the MRS that happens in PDA mode. Each bit is associated with one device. For example, bit[0] corresponds to Device 0, bit[1] to Device 1 etc. A '1' should be programmed to indicate that the MRS command should be applied to that device. A '0' indicates that the MRS commands should be skipped for that device.

9.2.3.1.10 Temperature Derate Enable Register (DERATEEN)

9.2.3.1.10.1 Offset

Register	Offset
DERATEEN	20h

9.2.3.1.10.2 Diagram



9.2.3.1.10.3 Fields

Field	Function
31-10 —	Reserved
9-8 rc_derate_value	Derate value of tRC for LPDDR4. Present only in designs configured to support LPDDR4. The required number of cycles for derating can be determined by dividing 3.75ns by the core_ddrc_core_clk period, and rounding up the next integer. 00b - Derating uses +1 01b - Derating uses +2 10b - Derating uses +3 11b - Derating uses +4
7-4 derate_byte	Derate byte Present only in designs configured to support LPDDR2/LPDDR3/LPDDR4 Indicates which byte of the MRR data is used for derating. The maximum valid value depends on MEMC_DRAM_TOTAL_DATA_WIDTH.
3-2 —	Reserved
1 derate_value	Derate value. Present only in designs configured to support LPDDR2/LPDDR3/LPDDR4 Set to 0 for all LPDDR2 speed grades as derating value of +1.875 ns is less than a core_ddrc_core_clk period. For LPDDR3/4, if the period of core_ddrc_core_clk is less than 1.875ns, this register field should be set to 1; otherwise it should be set to 0. 0b - Derating uses +1 1b - Derating uses +2
0 derate_enable	Enables derating. Present only in designs configured to support LPDDR2/LPDDR3/LPDDR4. This field must be set to '0' for non-LPDDR2/LPDDR3/LPDDR4 mode. 0b - Timing parameter derating is disabled 1b - Timing parameter derating is enabled using MR4 read value.

9.2.3.1.11 Temperature Derate Interval Register (DERATEINT)

9.2.3.1.11.1 Offset

Register	Offset
DERATEINT	24h

9.2.3.1.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	mr4_read_interval															
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	mr4_read_interval															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.11.3 Fields

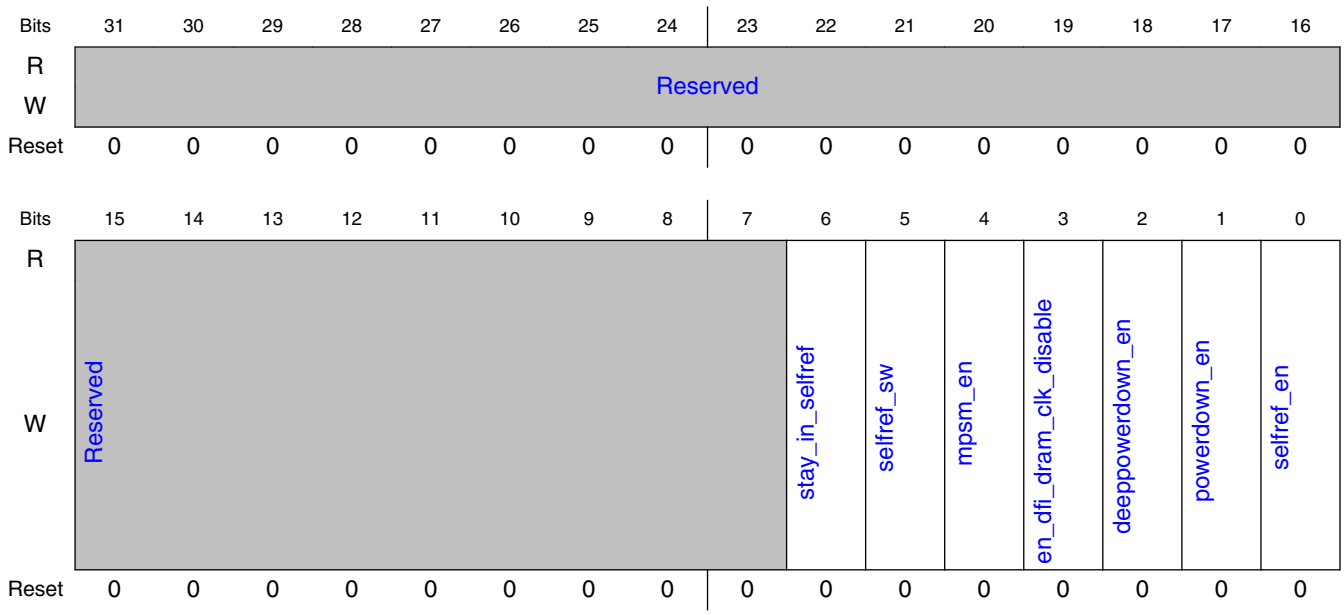
Field	Function
31-0 mr4_read_interval	Interval between two MR4 reads, used to derate the timing parameters. Present only in designs configured to support LPDDR2/LPDDR3/LPDDR4. This register must not be set to zero. Unit: DFI clock cycle.

9.2.3.1.12 Low Power Control Register (PWRCTL)

9.2.3.1.12.1 Offset

Register	Offset
PWRCTL	30h

9.2.3.1.12.2 Diagram



9.2.3.1.12.3 Fields

Field	Function
31-7 —	Reserved
6 stay_in_selfref	Self refresh state is an intermediate state to enter to Self refresh power down state or exit Self refresh power down state for LPDDR4. This register controls transition from the Self refresh state. - 1 - Prohibit transition from Self refresh state - 0 - Allow transition from Self refresh state 0b - 1b -
5 selfref_sw	A value of 1 to this register causes system to move to Self Refresh state immediately, as long as it is not in INIT or DPD/MPSM operating_mode. This is referred to as Software Entry/Exit to Self Refresh. 0b - Software Exit from Self Refresh 1b - Software Entry to Self Refresh
4 mpsm_en	When this is 1, the DDRC puts the SDRAM into maximum power saving mode when the transaction store is empty. This register must be reset to '0' to bring DDRC out of maximum power saving mode. Present only in designs configured to support DDR4. For non-DDR4, this register should not be set to 1. Note that MPSM is not supported when using a DDR PHY, if the PHY parameter DDRC_AC_CS_USE is disabled, as the MPSM exit sequence requires the chip-select signal to toggle. FOR PERFORMANCE ONLY.
3 en_dfi_dram_clk_disable	Enable the assertion of dfi_dram_clk_disable whenever a clock is not required by the SDRAM. If set to 0, dfi_dram_clk_disable is never asserted. Assertion of dfi_dram_clk_disable is as follows: In DDR2/DDR3, can only be asserted in Self Refresh. In DDR4, can be asserted in following: in Self Refresh in Maximum Power Saving Mode In mDDR/LPDDR2/LPDDR3, can be asserted in following: in Self Refresh in Power Down in Deep Power Down during Normal operation (Clock Stop) In LPDDR4, can be asserted in following: in Self Refresh Power Down in Power Down during Normal operation (Clock Stop)
2 deeppowerdown_en	When this is 1, DDRC puts the SDRAM into deep power-down mode when the transaction store is empty. This register must be reset to '0' to bring DDRC out of deep power-down mode. Controller performs automatic SDRAM initialization on deep power-down exit. Present only in designs configured to support

Table continues on the next page...

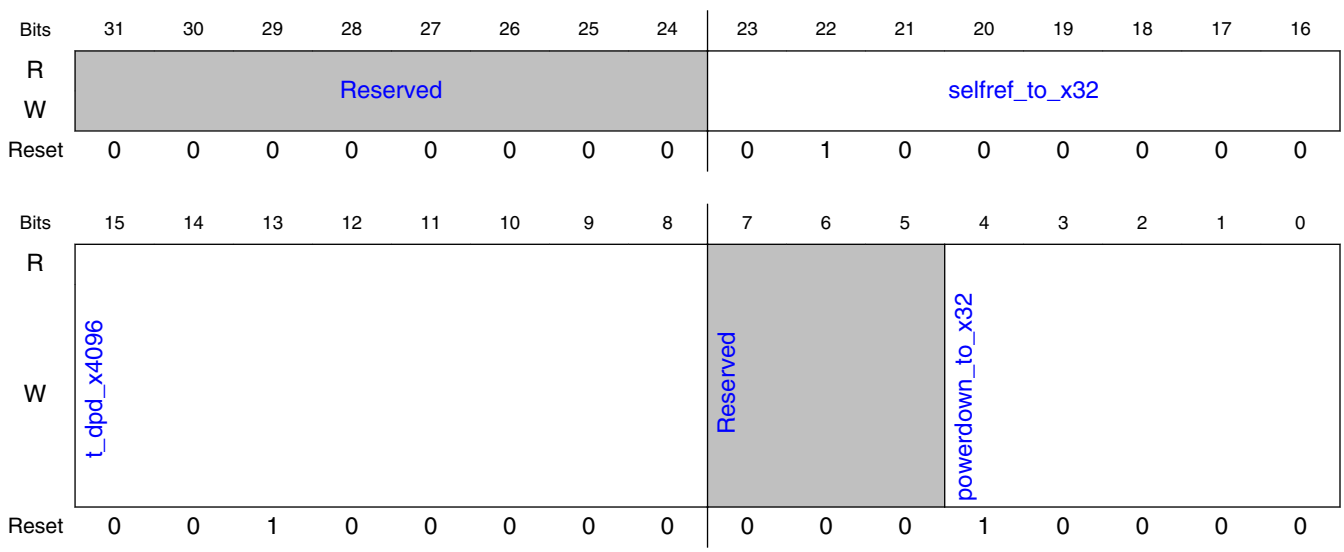
Field	Function
	mDDR or LPDDR2 or LPDDR3. For non-mDDR/non-LPDDR2/non-LPDDR3, this register should not be set to 1. FOR PERFORMANCE ONLY.
1 powerdown_en	If true then the DDRC goes into power-down after a programmable number of cycles "maximum idle clocks before power down" (PWRTMG.powerdown_to_x32). This register bit may be re-programmed during the course of normal operation.
0 selfref_en	If true then the DDRC puts the SDRAM into Self Refresh after a programmable number of cycles "maximum idle clocks before Self Refresh (PWRTMG.selfref_to_x32)". This register bit may be re-programmed during the course of normal operation.

9.2.3.1.13 Low Power Timing Register (PWRTMG)

9.2.3.1.13.1 Offset

Register	Offset
PWRTMG	34h

9.2.3.1.13.2 Diagram



9.2.3.1.13.3 Fields

Field	Function
31-24 —	Reserved

Table continues on the next page...

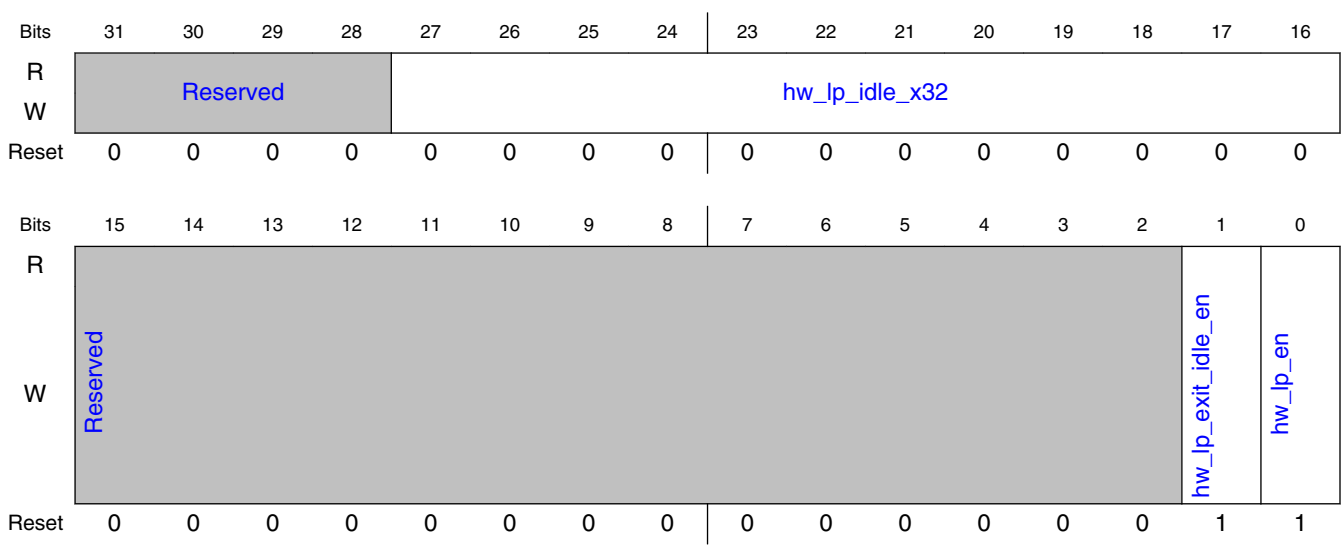
Field	Function
23-16 selfref_to_x32	After this many clocks of the DDRC command channel being idle the DDRC automatically puts the SDRAM into Self Refresh. The DDRC command channel is considered idle when there are no HIF commands outstanding. This must be enabled in the PWRCTL.selfref_en. Unit: Multiples of 32 DFI clocks. FOR PERFORMANCE ONLY.
15-8 t_dpd_x4096	Minimum deep power-down time. For mDDR, value from the JEDEC specification is 0 as mDDR exits from deep power-down mode immediately after PWRCTL.deeppowerdown_en is de-asserted. For LPDDR2/LPDDR3, value from the JEDEC specification is 500us. Unit: Multiples of 4096 DFI clocks. Present only in designs configured to support mDDR, LPDDR2 or LPDDR3. FOR PERFORMANCE ONLY.
7-5 —	Reserved
4-0 powerdown_to_x32	After this many clocks of the DDRC command channel being idle the DDRC automatically puts the SDRAM into power-down. The DDRC command channel is considered idle when there are no HIF commands outstanding. This must be enabled in the PWRCTL.powerdown_en. Unit: Multiples of 32 DFI clocks FOR PERFORMANCE ONLY.

9.2.3.1.14 Hardware Low Power Control Register (HWLPCTL)

9.2.3.1.14.1 Offset

Register	Offset
HWLPCTL	38h

9.2.3.1.14.2 Diagram



9.2.3.1.14.3 Fields

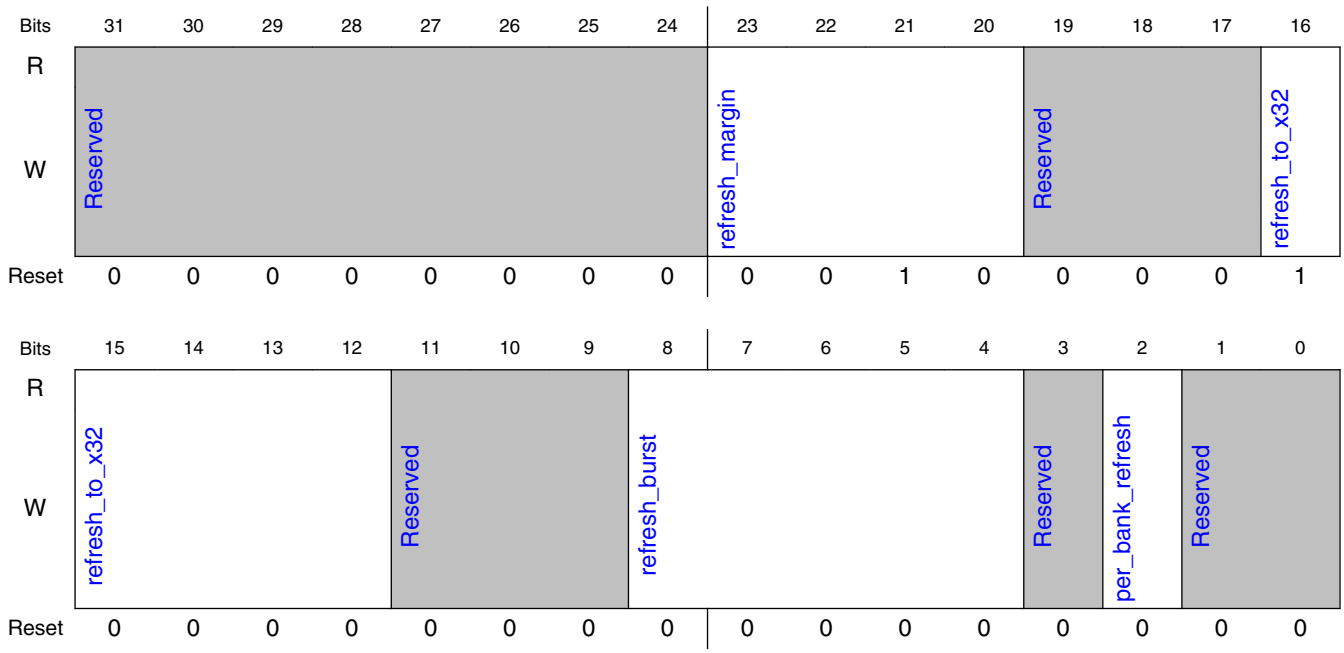
Field	Function
31-28 —	Reserved
27-16 hw_lp_idle_x32	Hardware idle period. The cactive_ddrc output is driven low if the DDRC command channel is idle for hw_lp_idle * 32 cycles if not in INIT or DPD/MPSM operating_mode. The DDRC command channel is considered idle when there are no HIF commands outstanding. The hardware idle function is disabled when hw_lp_idle_x32=0. Unit: Multiples of 32 DFI clocks. FOR PERFORMANCE ONLY.
15-2 —	Reserved
1 hw_lp_exit_idle_en	When this bit is programmed to 1 the cactive_in_ddrc pin of the DDRC can be used to exit from the automatic clock stop, automatic power down or automatic self-refresh modes. Note, it will not cause exit of Self-Refresh that was caused by Hardware Low Power Interface and/or Software (PWRCTL.selfref_sw).
0 hw_lp_en	Enable for Hardware Low Power Interface.

9.2.3.1.15 Refresh Control Register 0 (RFSHCTL0)

9.2.3.1.15.1 Offset

Register	Offset
RFSHCTL0	50h

9.2.3.1.15.2 Diagram



9.2.3.1.15.3 Fields

Field	Function
31-24 —	Reserved
23-20 refresh_margin	Threshold value in number of DFI clock cycles before the critical refresh or page timer expires. A critical refresh is to be issued before this threshold is reached. It is recommended that this not be changed from the default value, currently shown as 0x2. It must always be less than internally used t_rfc_nom_x32. Note that, in LPDDR2/LPDDR3/LPDDR4, internally used t_rfc_nom_x32 may be equal to RFSHTMG.t_rfc_nom_x32>>2 if derating is enabled (DERATEEN.derate_enable=1). Otherwise, internally used t_rfc_nom_x32 will be equal to RFSHTMG.t_rfc_nom_x32. Unit: Multiples of 32 DFI clocks.
19-17 —	Reserved
16-12 refresh_to_x32	If the refresh timer (tRFCnom, also known as tREFI) has expired at least once, but it has not expired (RFSHCTL0.refresh_burst+1) times yet, then a speculative refresh may be performed. A speculative refresh is a refresh performed at a time when refresh would be useful, but before it is absolutely required. When the SDRAM bus is idle for a period of time determined by this RFSHCTL0.refresh_to_x32 and the refresh timer has expired at least once since the last refresh, then a speculative refresh is performed. Speculative refreshes continues successively until there are no refreshes pending or until new reads or writes are issued to the DDRC. FOR PERFORMANCE ONLY. Unit: Multiples of 32 DFI clocks.
11-9 —	Reserved
8-4 refresh_burst	The programmed value + 1 is the number of refresh timeouts that is allowed to accumulate before traffic is blocked and the refreshes are forced to execute. Closing pages to perform a refresh is a one-time penalty that must be paid for each group of refreshes. Therefore, performing refreshes in a burst reduces

Table continues on the next page...

Field	Function
	the per-refresh penalty of these page closings. Higher numbers for RFSHCTL.refresh_burst slightly increases utilization; lower numbers decreases the worst-case latency associated with refreshes. - 0 - single refresh - 1 - burst-of-2 refresh - 7 - burst-of-8 refresh For information on burst refresh feature refer to section 3.9 of DDR2 JEDEC specification - JESD79-2F.pdf. For DDR2/3, the refresh is always per-rank and not per-bank. The rank refresh can be accumulated over 8*tREFI cycles using the burst refresh feature. In DDR4 mode, according to Fine Granularity feature, 8 refreshes can be postponed in 1X mode, 16 refreshes in 2X mode and 32 refreshes in 4X mode. If using PHY-initiated updates, care must be taken in the setting of RFSHCTL0.refresh_burst, to ensure that tRFCmax is not violated due to a PHY-initiated update occurring shortly before a refresh burst was due. In this situation, the refresh burst will be delayed until the PHY-initiated update is complete.
3 —	Reserved
2 per_bank_refresh	Per bank refresh allows traffic to flow to other banks. Per bank refresh is not supported by all LPDDR2 devices but should be supported by all LPDDR3/LPDDR4 devices. Present only in designs configured to support LPDDR2/LPDDR3/LPDDR4 0b - All bank refresh 1b - Per bank refresh
1-0 —	Reserved

9.2.3.1.16 Refresh Control Register 1 (RFSHCTL1)

9.2.3.1.16.1 Offset

Register	Offset
RFSHCTL1	54h

9.2.3.1.16.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				refresh_timer1_start_value_x32											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				refresh_timer0_start_value_x32											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.16.3 Fields

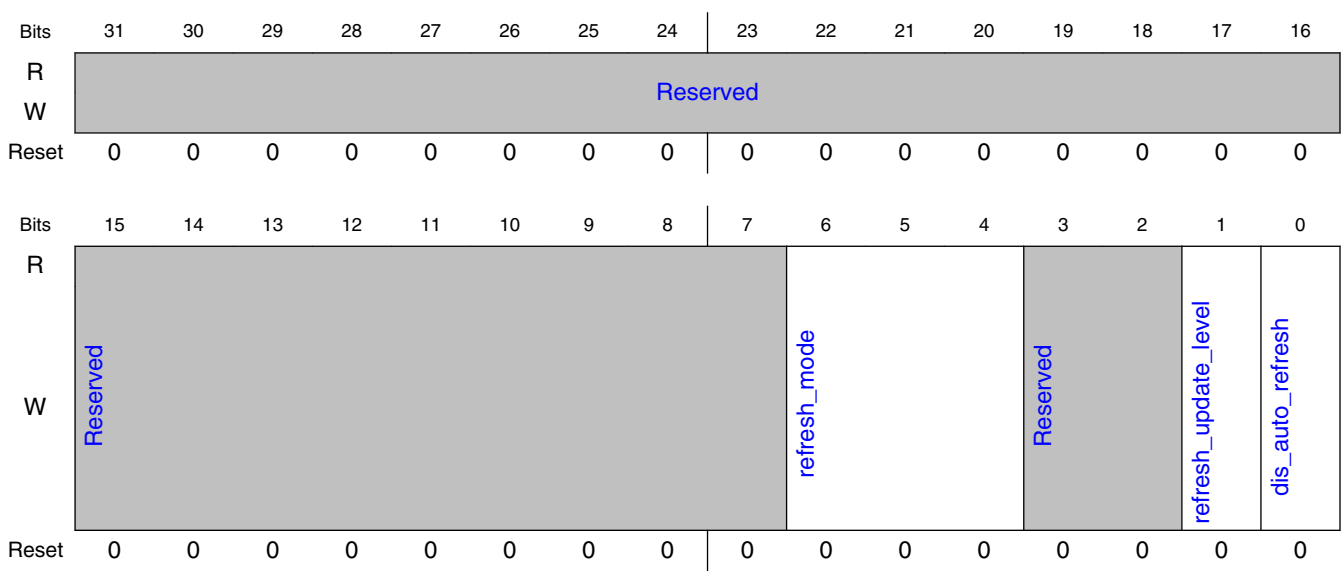
Field	Function
31-28 —	Reserved
27-16 refresh_timer1_start_value_x32	Refresh timer start for rank 1 (only present in multi-rank configurations). This is useful in staggering the refreshes to multiple ranks to help traffic to proceed. This is explained in Refresh Controls section of architecture chapter. Unit: Multiples of 32 DFI clock cycles. FOR PERFORMANCE ONLY.
15-12 —	Reserved
11-0 refresh_timer0_start_value_x32	Refresh timer start for rank 0 (only present in multi-rank configurations). This is useful in staggering the refreshes to multiple ranks to help traffic to proceed. This is explained in Refresh Controls section of architecture chapter. Unit: Multiples of 32 DFI clock cycles. FOR PERFORMANCE ONLY.

9.2.3.1.17 Refresh Control Register 3 (RFSHCTL3)

9.2.3.1.17.1 Offset

Register	Offset
RFSHCTL3	60h

9.2.3.1.17.2 Diagram



9.2.3.1.17.3 Fields

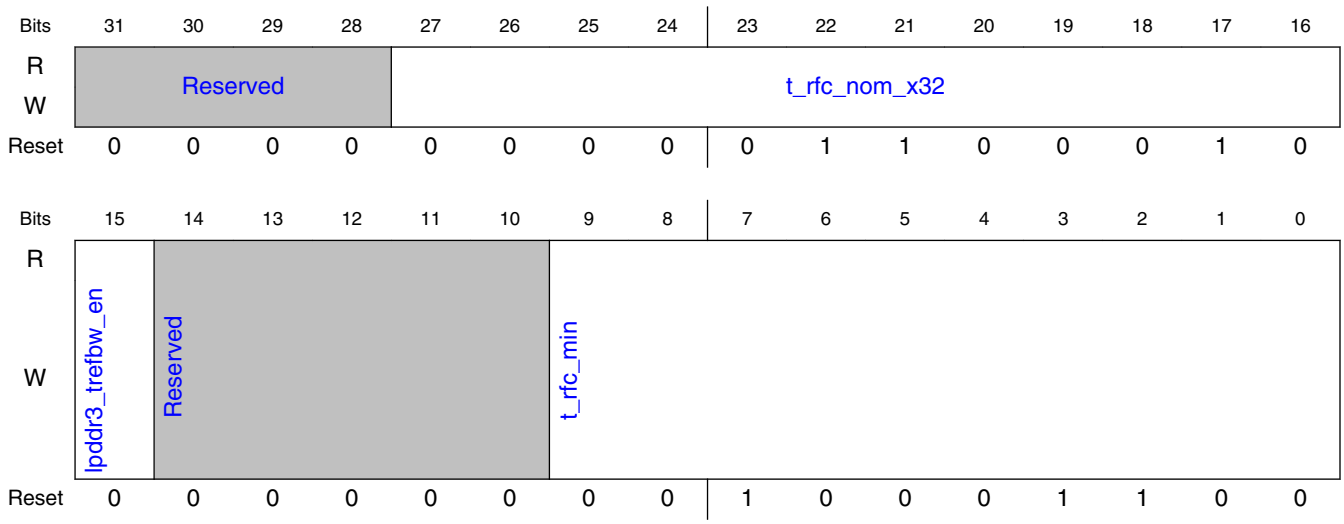
Field	Function
31-7 —	Reserved
6-4 refresh_mode	Fine Granularity Refresh Mode - 000 - Fixed 1x (Normal mode) - 001 - Fixed 2x - 010 - Fixed 4x - 101 - Enable on the fly 2x (not supported) - 110 - Enable on the fly 4x (not supported) - Everything else - reserved Note: Only Fixed 1x mode is supported if RFSHCTL3.dis_auto_refresh = 1. Note: The on-the-fly modes are not supported in this version of the DDRC. Note: This must be set up while the Controller is in reset or while the Controller is in self-refresh mode. Changing this during normal operation is not allowed. Making this a dynamic register will be supported in future version of the DDRC. Note: This register field has effect only if a DDR4 SDRAM device is in use (MSTR.ddr4 = 1).
3-2 —	Reserved
1 refresh_update_level	Toggle this signal (either from 0 to 1 or from 1 to 0) to indicate that the refresh register(s) have been updated. refresh_update_level must not be toggled when the DDRC is in reset (core_ddrc_rstn = 0). The refresh register(s) are automatically updated when exiting reset.
0 dis_auto_refresh	When '1', disable auto-refresh generated by the DDRC. When auto-refresh is disabled, the SoC core must generate refreshes using the registers reg_ddrc_rank0_refresh, reg_ddrc_rank1_refresh, reg_ddrc_rank2_refresh and reg_ddrc_rank3_refresh. When dis_auto_refresh transitions from 0 to 1, any pending refreshes are immediately scheduled by the DDRC. If DDR4 CRC/parity retry is enabled (CRCPARCTL1.crc_parity_retry_enable = 1), disable auto-refresh is not supported, and this bit must be set to '0'. (DDR4 only) If FGR mode is enabled (RFSHCTL3.refresh_mode > 0), disable auto-refresh is not supported, and this bit must be set to '0'. This register field is changeable on the fly.

9.2.3.1.18 Refresh Timing Register (RFSHTMG)

9.2.3.1.18.1 Offset

Register	Offset
RFSHTMG	64h

9.2.3.1.18.2 Diagram



9.2.3.1.18.3 Fields

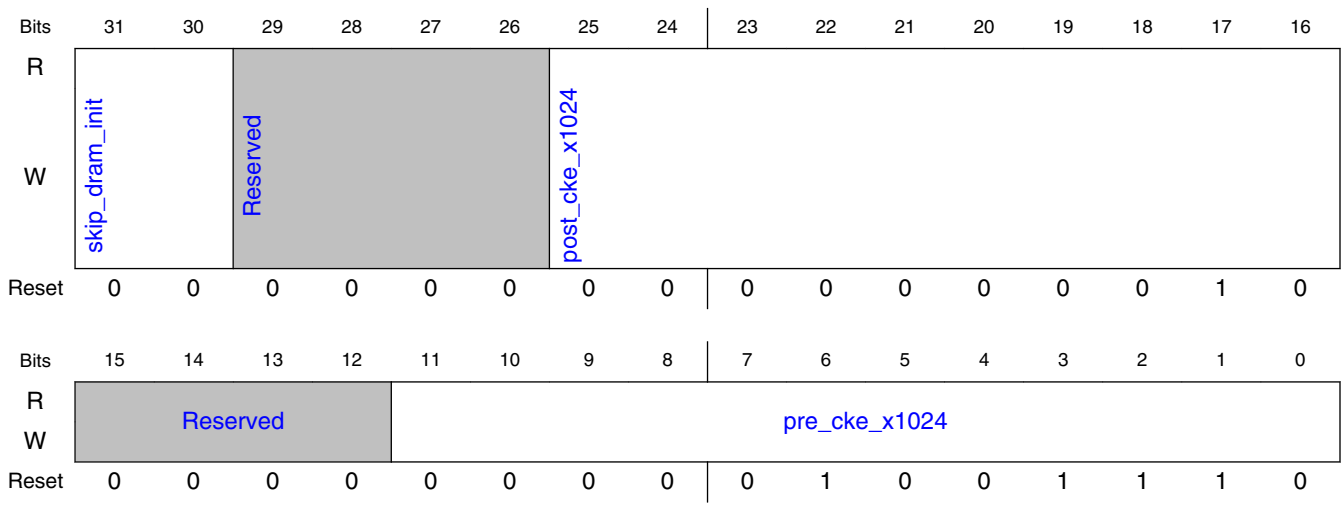
Field	Function
31-28 —	Reserved
27-16 t_rfc_nom_x32	tREFI: Average time interval between refreshes per rank (Specification: 7.8us for DDR2, DDR3 and DDR4. See JEDEC specification for mDDR, LPDDR2, LPDDR3 and LPDDR4). For LPDDR2/LPDDR3/LPDDR4: - if using all-bank refreshes (RFSHCTL0.per_bank_refresh = 0), this register should be set to tREFIab - if using per-bank refreshes (RFSHCTL0.per_bank_refresh = 1), this register should be set to tREFIpb When the controller is operating in 1:2 frequency ratio mode, program this to (tREFI/2), no rounding up. In DDR4 mode, tREFI value is different depending on the refresh mode. The user should program the appropriate value from the spec based on the value programmed in the refresh mode register. Note that RFSHTMG.t_rfc_nom_x32 * 32 must be greater than RFSHTMG.t_rfc_min, and RFSHTMG.t_rfc_nom_x32 must be greater than 0x1. - Non-DDR4 or DDR4 Fixed 1x mode: RFSHTMG.t_rfc_nom_x32 must be less than or equal to 0xFFE. - DDR4 Fixed 2x mode: RFSHTMG.t_rfc_nom_x32 must be less than or equal to 0x7FF. - DDR4 Fixed 4x mode: RFSHTMG.t_rfc_nom_x32 must be less than or equal to 0x3FF. Unit: Multiples of 32 clocks.
15 lpddr3_trefbw_en	Used only when LPDDR3 memory type is connected. Should only be changed when DDRC is in reset. Specifies whether to use the tREFBW parameter (required by some LPDDR3 devices which comply with earlier versions of the LPDDR3 JEDEC specification) or not: - 0 - tREFBW parameter not used - 1 - tREFBW parameter used
14-10 —	Reserved
9-0 t_rfc_min	tRFC (min): Minimum time from refresh to refresh or activate. When the controller is operating in 1:1 mode, t_rfc_min should be set to RoundUp(tRFCmin/tCK). When the controller is operating in 1:2 mode, t_rfc_min should be set to RoundUp(RoundUp(tRFCmin/tCK)/2). In LPDDR2/LPDDR3/LPDDR4 mode: - if using all-bank refreshes, the tRFCmin value in the above equations is equal to tRFCab - if using per-bank refreshes, the tRFCmin value in the above equations is equal to tRFCpb In DDR4 mode, the tRFCmin value in the above equations is different depending on the refresh mode (fixed 1X,2X,4X) and the device density. The user should program the appropriate value from the spec based on the 'refresh_mode' and the device density that is used. Unit: Clocks.

9.2.3.1.19 SDRAM Initialization Register 0 (INIT0)

9.2.3.1.19.1 Offset

Register	Offset
INIT0	D0h

9.2.3.1.19.2 Diagram



9.2.3.1.19.3 Fields

Field	Function
31-30 skip_dram_init	If lower bit is enabled the SDRAM initialization routine is skipped. The upper bit decides what state the controller starts up in when reset is removed - 00 - SDRAM Intialization routine is run after power-up - 01 - SDRAM Initialization routine is skipped after power-up. Controller starts up in Normal Mode - 11 - SDRAM Initialization routine is skipped after power-up. Controller starts up in Self-refresh Mode - 10 - SDRAM Initialization routine is run after power-up. 00b - SDRAM Initialization routine is run after power-up 01b - SDRAM Initialization routine is skipped after power-up 10b - SDRAM Initialization routine is run after power-up 11b - SDRAM Initialization routine is skipped after power-up
29-26 —	Reserved
25-16 post_cke_x1024	Cycles to wait after driving CKE high to start the SDRAM initialization sequence. Unit: 1024 DFI clock cycles. DDR2 typically requires a 400 ns delay, requiring this value to be programmed to 2 at all clock speeds. LPDDR2/LPDDR3 typically requires this to be programmed for a delay of 200 us. LPDDR4 typically requires this to be programmed for a delay of 2 us. When the controller is operating in 1:2 frequency ratio mode, program this to JEDEC spec value divided by 2, and round it up to the next integer value.

Table continues on the next page...

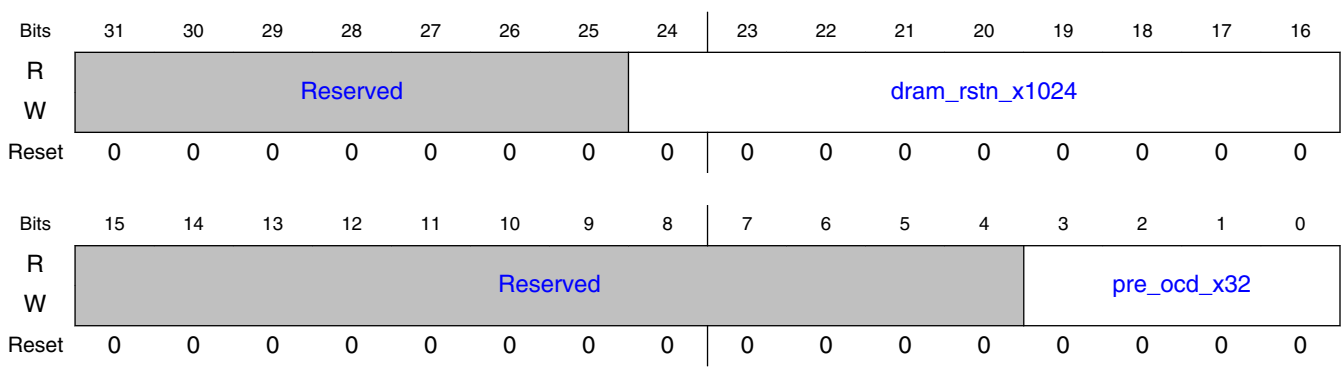
Field	Function
15-12 —	Reserved
11-0 pre_cke_x1024	Cycles to wait after reset before driving CKE high to start the SDRAM initialization sequence. Unit: 1024 DFI clock cycles. DDR2 specifications typically require this to be programmed for a delay of >= 200 us. LPDDR2/LPDDR3: tINIT1 of 100 ns (min) LPDDR4: tINIT3 of 2 ms (min) When the controller is operating in 1:2 frequency ratio mode, program this to JEDEC spec value divided by 2, and round it up to the next integer value. For DDR3/DDR4 RDIMMs, this should include the time needed to satisfy tSTAB

9.2.3.1.20 SDRAM Initialization Register 1 (INIT1)

9.2.3.1.20.1 Offset

Register	Offset
INIT1	D4h

9.2.3.1.20.2 Diagram



9.2.3.1.20.3 Fields

Field	Function
31-25 —	Reserved
24-16 dram_rstn_x1024	Number of cycles to assert SDRAM reset signal during init sequence. This is only present for designs supporting DDR3, DDR4 or LPDDR4 devices. For use with a DDR PHY, this should be set to a minimum of 1. When the controller is operating in 1:2 frequency ratio mode, program this to JEDEC spec value divided by 2, and round it up to the next integer value. Unit: 1024 DFI clock cycles.
15-4 —	Reserved

Table continues on the next page...

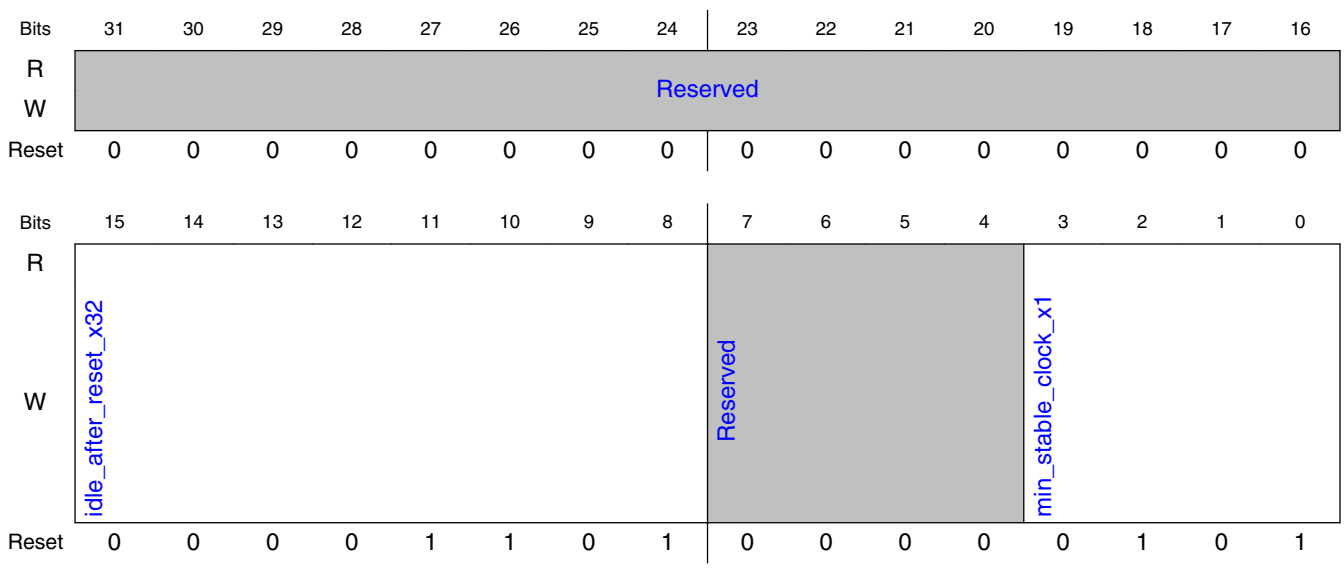
Field	Function
3-0 pre_ocd_x32	Wait period before driving the OCD complete command to SDRAM. Unit: Counts of a global timer that pulses every 32 DFI clock cycles. There is no known specific requirement for this; it may be set to zero.

9.2.3.1.21 SDRAM Initialization Register 2 (INIT2)

9.2.3.1.21.1 Offset

Register	Offset
INIT2	D8h

9.2.3.1.21.2 Diagram



9.2.3.1.21.3 Fields

Field	Function
31-16 —	Reserved
15-8 idle_after_reset_x32	Idle time after the reset command, tINIT4. Present only in designs configured to support LPDDR2. When the controller is operating in 1:2 frequency ratio mode, program this to JEDEC spec value divided by 2, and round it up to the next integer value. Unit: 32 DFI clock cycles.
7-4 —	Reserved

Table continues on the next page...

Field	Function
3-0 min_stable_clock_x1	Time to wait after the first CKE high, tINIT2. Present only in designs configured to support LPDDR2/LPDDR3. LPDDR2/LPDDR3 typically requires 5 x tCK delay. When the controller is operating in 1:2 frequency ratio mode, program this to JEDEC spec value divided by 2, and round it up to the next integer value. Unit: DFI clock cycles.

9.2.3.1.22 SDRAM Initialization Register 3 (INIT3)

9.2.3.1.22.1 Offset

Register	Offset
INIT3	DCh

9.2.3.1.22.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	mr															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	emr															
Reset	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0

9.2.3.1.22.3 Fields

Field	Function
31-16 mr	DDR2: Value to write to MR register. Bit 8 is for DLL and the setting here is ignored. The DDRC sets this bit appropriately. DDR3/DDR4: Value loaded into MR0 register. mDDR: Value to write to MR register. LPDDR2/LPDDR3/LPDDR4 - Value to write to MR1 register
15-0 emr	DDR2: Value to write to EMR register. Bits 9:7 are for OCD and the setting in this register is ignored. The DDRC sets those bits appropriately. DDR3/DDR4: Value to write to MR1 register Set bit 7 to 0. If PHY-evaluation mode training is enabled, this bit is set appropriately by the DDRC during write leveling. mDDR: Value to write to EMR register. LPDDR2/LPDDR3/LPDDR4 - Value to write to MR2 register

9.2.3.1.23 SDRAM Initialization Register 4 (INIT4)

9.2.3.1.23.1 Offset

Register	Offset
INIT4	E0h

9.2.3.1.23.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	emr2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	emr3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.23.3 Fields

Field	Function
31-16 emr2	DDR2: Value to write to EMR2 register. DDR3/DDR4: Value to write to MR2 register LPDDR2/LPDDR3/ LPDDR4: Value to write to MR3 register mDDR: Unused
15-0 emr3	DDR2: Value to write to EMR3 register. DDR3/DDR4: Value to write to MR3 register mDDR/LPDDR2/ LPDDR3: Unused LPDDR4: Value to write to MR13 register

9.2.3.1.24 SDRAM Initialization Register 5 (INIT5)

9.2.3.1.24.1 Offset

Register	Offset
INIT5	E4h

9.2.3.1.24.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								dev_zqinit_x32							
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						max_auto_init_x1024									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

9.2.3.1.24.3 Fields

Field	Function
31-24 —	Reserved
23-16 dev_zqinit_x32	ZQ initial calibration, tZQINIT. Present only in designs configured to support DDR3 or DDR4 or LPDDR2/LPDDR3. DDR3 typically requires 512 SDRAM clock cycles. DDR4 requires 1024 SDRAM clock cycles. LPDDR2/LPDDR3 requires 1 us. When the controller is operating in 1:2 frequency ratio mode, program this to JEDEC spec value divided by 2, and round it up to the next integer value. Unit: 32 DFI clock cycles.
15-10 —	Reserved
9-0 max_auto_init_x1024	Maximum duration of the auto initialization, tINIT5. Present only in designs configured to support LPDDR2/LPDDR3. LPDDR2/LPDDR3 typically requires 10 us. Unit: 1024 DFI clock cycles.

9.2.3.1.25 SDRAM Initialization Register 6 (INIT6)

9.2.3.1.25.1 Offset

Register	Offset
INIT6	E8h

9.2.3.1.25.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	mr4															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	mr5															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.25.3 Fields

Field	Function
31-16 mr4	DDR4- Value to be loaded into SDRAM MR4 registers. Used in DDR4 designs only.
15-0 mr5	DDR4- Value to be loaded into SDRAM MR5 registers. Used in DDR4 designs only.

9.2.3.1.26 SDRAM Initialization Register 7 (INIT7)

9.2.3.1.26.1 Offset

Register	Offset
INIT7	ECh

9.2.3.1.26.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	mr6															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.26.3 Fields

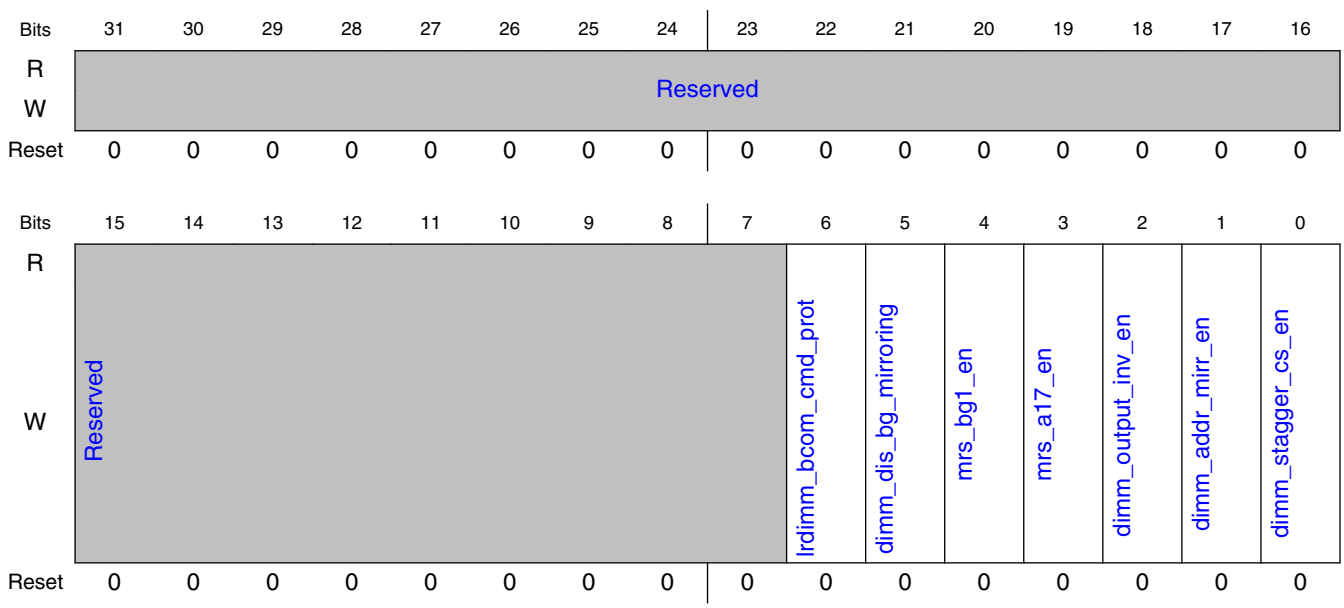
Field	Function
31-16 mr6	DDR4- Value to be loaded into SDRAM MR6 registers. Used in DDR4 designs only.
15-0 —	Reserved

9.2.3.1.27 DIMM Control Register (DIMMCTL)

9.2.3.1.27.1 Offset

Register	Offset
DIMMCTL	F0h

9.2.3.1.27.2 Diagram



9.2.3.1.27.3 Fields

Field	Function
31-7	Reserved

Table continues on the next page...

Field	Function
—	
6 lrdimm_bcom_cmd_prot	Protects the timing restrictions (tBCW/tMRC) between consecutive BCOM commands defined in the Data Buffer specification. When using DDR4 LRDIMM, this bit must be set to 1. Otherwise, this bit must be set to 0.
5 dimm_dis_bg_mirroring	Disabling Address Mirroring for BG bits. When this is set to 1, BG0 and BG1 are NOT swapped even if Address Mirroring is enabled. This will be required for DDR4 DIMMs with x16 devices. 0b - BG0 and BG1 are swapped if address mirroring is enabled. 1b - BG0 and BG1 are NOT swapped.
4 mrs_bg1_en	Enable for BG1 bit of MRS command. BG1 bit of the mode register address is specified as RFU (Reserved for Future Use) and must be programmed to 0 during MRS. In case where DRAMs which do not have BG1 are attached and both the CA parity and the Output Inversion are enabled, this must be set to 0, so that the calculation of CA parity will not include BG1 bit. Note: This has no effect on the address of any other memory accesses, or of software-driven mode register accesses. If address mirroring is enabled, this is applied to BG1 of even ranks and BG0 of odd ranks. 0b - Disabled 1b - Enabled
3 mrs_a17_en	Enable for A17 bit of MRS command. A17 bit of the mode register address is specified as RFU (Reserved for Future Use) and must be programmed to 0 during MRS. In case where DRAMs which do not have A17 are attached and the Output Inversion are enabled, this must be set to 0, so that the calculation of CA parity will not include A17 bit. Note: This has no effect on the address of any other memory accesses, or of software-driven mode register accesses. 0b - Disabled 1b - Enabled
2 dimm_output_inv_en	Output Inversion Enable (for DDR4 RDIMM/LRDIMM implementations only). DDR4 RDIMM/LRDIMM implements the Output Inversion feature by default, which means that the following address, bank address and bank group bits of B-side DRAMs are inverted: A3-A9, A11, A13, A17, BA0-BA1, BG0-BG1. Setting this bit ensures that, for mode register accesses generated by the DDRC during the automatic initialization routine and enabling of a particular DDR4 feature, separate A-side and B-side mode register accesses are generated. For B-side mode register accesses, these bits are inverted within the DDRC to compensate for this RDIMM/LRDIMM inversion. It is recommended to set this bit always, if using DDR4 RDIMMs/LRDIMMs. Note: This has no effect on the address of any other memory accesses, or of software-driven mode register accesses. 0b - Do not implement output inversion for B-side DRAMs. 1b - Implement output inversion for B-side DRAMs.
1 dimm_addr_mirr_en	Address Mirroring Enable (for multi-rank UDIMM implementations and multi-rank DDR4 RDIMM/LRDIMM implementations). Some UDIMMs and DDR4 RDIMMs/LRDIMMs implement address mirroring for odd ranks, which means that the following address, bank address and bank group bits are swapped: (A3, A4), (A5, A6), (A7, A8), (BA0, BA1) and also (A11, A13), (BG0, BG1) for the DDR4. Setting this bit ensures that, for mode register accesses during the automatic initialization routine, these bits are swapped within the DDRC to compensate for this UDIMM/RDIMM/LRDIMM swapping. In addition to the automatic initialization routine, in case of DDR4 UDIMM/RDIMM/LRDIMM, they are swapped during the automatic MRS access to enable/disable of a particular DDR4 feature. Note: This has no effect on the address of any other memory accesses, or of software-driven mode register accesses. This is not supported for mDDR, LPDDR2, LPDDR3 or LPDDR4 SDRAMs. Note: In case of x16 DDR4 DIMMs, BG1 output of MRS for the odd ranks is same as BG0 because BG1 is invalid, hence dimm_dis_bg_mirroring register must be set to 1. 0b - Do not implement address mirroring 1b - For odd ranks, implement address mirroring for MRS commands to during initialization and for any automatic DDR4 MRS commands (to be used if UDIMM/RDIMM/LRDIMM implements address mirroring)
0 dimm_stagger_cs_en	Staggering enable for multi-rank accesses (for multi-rank UDIMM, RDIMM and LRDIMM implementations only). This is not supported for mDDR, LPDDR2, LPDDR3 or LPDDR4 SDRAMs. Even if this bit is set it does not take care of software driven MR commands (via MRCTRL0/MRCTRL1), where software is responsible to send them to separate ranks as appropriate.

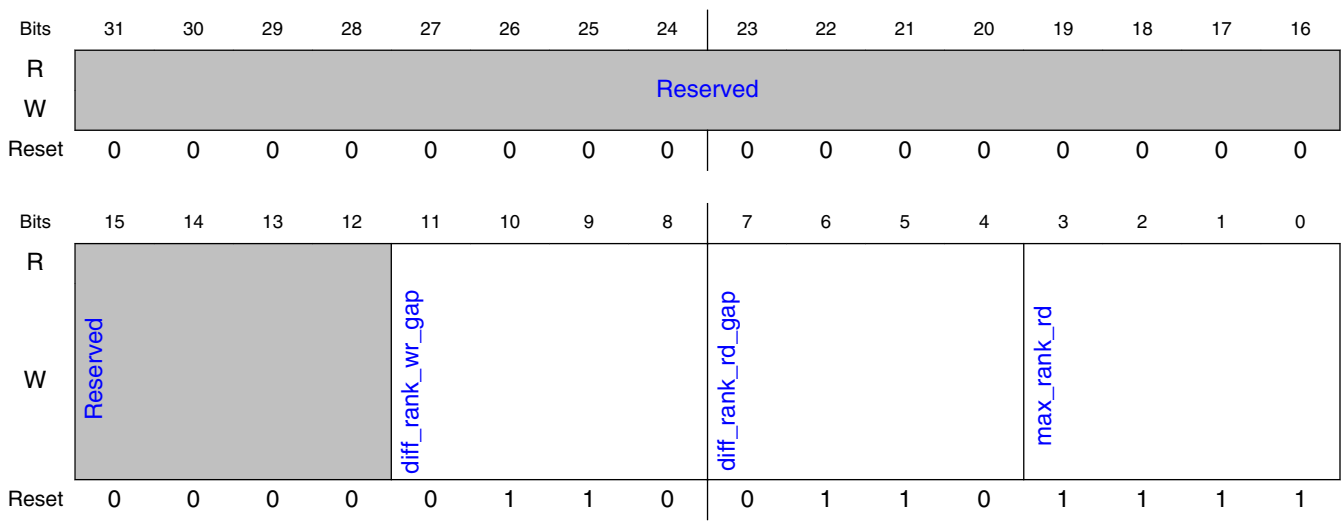
Field	Function
	0b - Do not stagger accesses 1b - (non-DDR4) Send all commands to even and odd ranks separately 1b - (DDR4) Send MRS commands to each ranks separately

9.2.3.1.28 Rank Control Register (RANKCTL)

9.2.3.1.28.1 Offset

Register	Offset
RANKCTL	F4h

9.2.3.1.28.2 Diagram



9.2.3.1.28.3 Fields

Field	Function
31-12 —	Reserved
11-8 diff_rank_wr_gap	Only present for multi-rank configurations. Indicates the number of clocks of gap in data responses when performing consecutive writes to different ranks. This is used to switch the delays in the PHY to match the rank requirements. This value should consider both PHY requirement and ODT requirement. - PHY requirement: tphy_wrcsgap + 1 (see PHY databook for value of tphy_wrcsgap) If CRC feature is enabled, should be increased by 1. If write preamble is set to 2tCK(DDR4/LPDDR4 only), should be increased by 1. If write postamble is set to 1.5tCK(LPDDR4 only), should be increased by 1. - ODT requirement: The value programmed in this register takes care of the ODT switch off timing requirement when switching

Table continues on the next page...

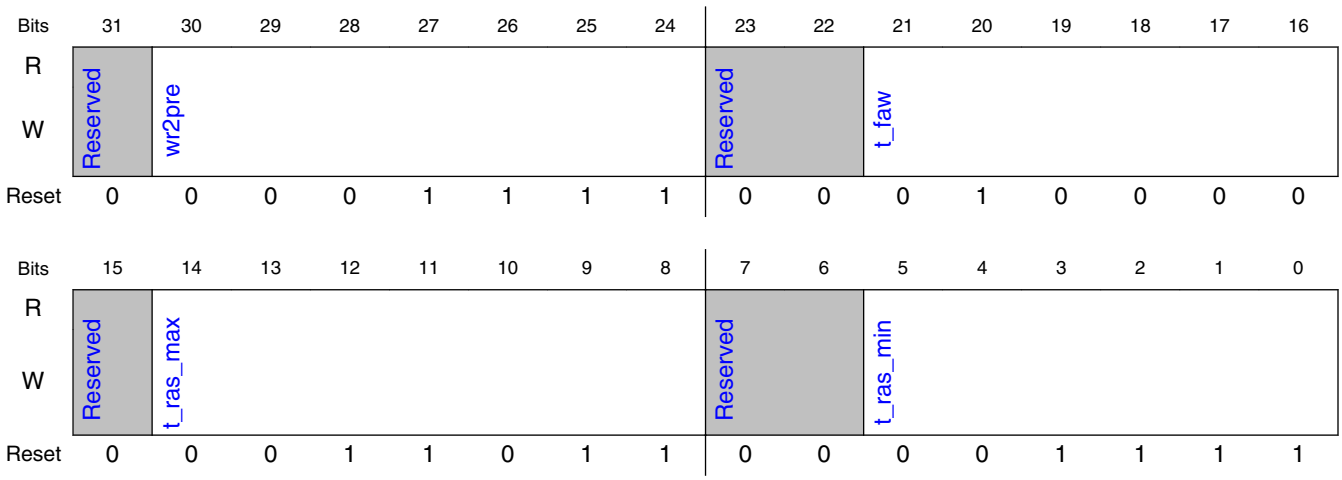
Field	Function
	ranks during writes. For LPDDR4, the requirement is ODTLoff - ODTLon - BL/2 + 1 When the controller is operating in 1:1 mode, program this to the larger of PHY requirement or ODT requirement. When the controller is operating in 1:2 mode, program this to the larger value divided by two and round it up to the next integer. Note that, if using DDR4-LRDIMM, refer to TWRWR timing requirements in JEDEC DDR4 Data Buffer (DDR4DB01) Specification.
7-4 diff_rank_rd_gap	Only present for multi-rank configurations. Indicates the number of clocks of gap in data responses when performing consecutive reads to different ranks. This is used to switch the delays in the PHY to match the rank requirements. This value should consider both PHY requirement and ODT requirement. - PHY requirement: tphy_rdcsgap + 1 (see PHY databook for value of tphy_rdcsgap) If read preamble is set to 2tCK(DDR4/LPDDR4 only), should be increased by 1. If read postamble is set to 1.5tCK(LPDDR4 only), should be increased by 1. - ODT requirement: The value programmed in this register takes care of the ODT switch off timing requirement when switching ranks during reads. When the controller is operating in 1:1 mode, program this to the larger of PHY requirement or ODT requirement. When the controller is operating in 1:2 mode, program this to the larger value divided by two and round it up to the next integer. Note that, if using DDR4-LRDIMM, refer to TRDRD timing requirements in JEDEC DDR4 Data Buffer (DDR4DB01) Specification.
3-0 max_rank_rd	Only present for multi-rank configurations. Background: Reads to the same rank can be performed back-to-back. Reads to different ranks require additional gap dictated by the register RANKCTL.diff_rank_rd_gap. This is to avoid possible data bus contention as well as to give PHY enough time to switch the delay when changing ranks. The DDRC arbitrates for bus access on a cycle-by-cycle basis; therefore after a read is scheduled, there are few clock cycles (determined by the value on RANKCTL.diff_rank_rd_gap register) in which only reads from the same rank are eligible to be scheduled. This prevents reads from other ranks from having fair access to the data bus. This parameter represents the maximum number of reads that can be scheduled consecutively to the same rank. After this number is reached, a delay equal to RANKCTL.diff_rank_rd_gap is inserted by the scheduler to allow all ranks a fair opportunity to be scheduled. Higher numbers increase bandwidth utilization, lower numbers increase fairness. This feature can be DISABLED by setting this register to 0. When set to 0, the Controller will stay on the same rank as long as commands are available for it. Minimum programmable value is 0 (feature disabled) and maximum programmable value is 0xF. FOR PERFORMANCE ONLY.

9.2.3.1.29 SDRAM Timing Register 0 (DRAMTMG0)

9.2.3.1.29.1 Offset

Register	Offset
DRAMTMG0	100h

9.2.3.1.29.2 Diagram



9.2.3.1.29.3 Fields

Field	Function
31 —	Reserved
30-24 wr2pre	Minimum time between write and precharge to same bank. Unit: Clocks Specifications: WL + BL/2 + tWR = approximately 8 cycles + 15 ns = 14 clocks @400MHz and less for lower frequencies where: - WL = write latency - BL = burst length. This must match the value programmed in the BL bit of the mode register to the SDRAM. BST (burst terminate) is not supported at present. - tWR = Write recovery time. This comes directly from the SDRAM specification. Add one extra cycle for LPDDR2/LPDDR3/LPDDR4 for this parameter. When the controller is operating in 1:2 frequency ratio mode, 1T mode, divide the above value by 2. No rounding up. When the controller is operating in 1:2 frequency ratio mode, 2T mode or LPDDR4 mode, divide the above value by 2 and round it up to the next integer value. Note that, depending on the PHY, if using LRDIMM, it may be necessary to adjust the value of this parameter to compensate for the extra cycle of latency through the LRDIMM.
23-22 —	Reserved
21-16 t_faw	tFAW Valid only when 8 or more banks(or banks x bank groups) are present. In 8-bank design, at most 4 banks must be activated in a rolling window of tFAW cycles. When the controller is operating in 1:2 frequency ratio mode, program this to (tFAW/2) and round up to next integer value. In a 4-bank design, set this register to 0x1 independent of the 1:1/1:2 frequency mode. Unit: Clocks
15 —	Reserved
14-8 t_ras_max	tRAS(max): Maximum time between activate and precharge to same bank. This is the maximum time that a page can be kept open Minimum value of this register is 1. Zero is invalid. When the controller is operating in 1:2 frequency ratio mode, program this to (tRAS(max)-1)/2. No rounding up. Unit: Multiples of 1024 clocks.
7-6 —	Reserved
5-0 t_ras_min	tRAS(min): Minimum time between activate and precharge to the same bank. When the controller is operating in 1:2 frequency mode, 1T mode, program this to tRAS(min)/2. No rounding up. When the

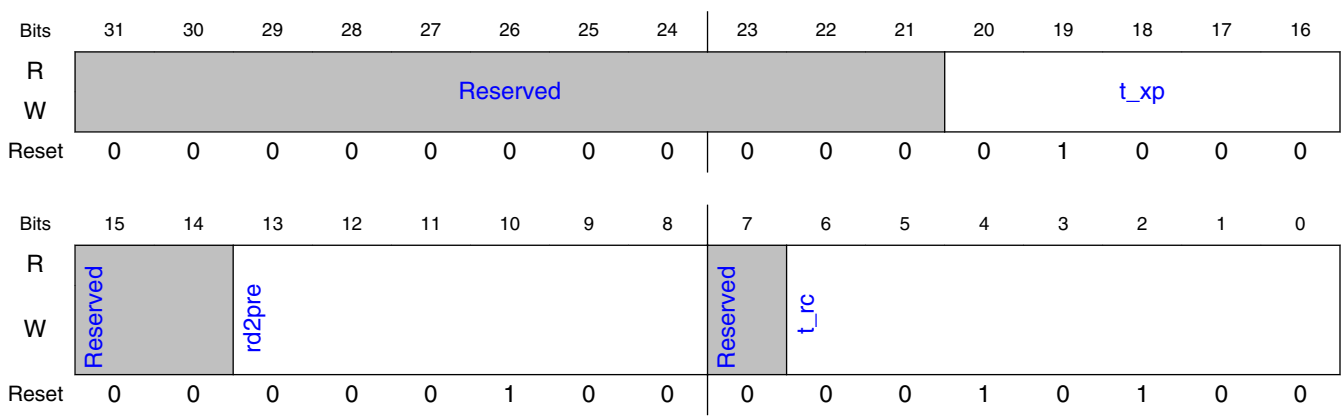
Field	Function
	controller is operating in 1:2 frequency ratio mode, 2T mode or LPDDR4 mode, program this to (tRAS(min)/2) and round it up to the next integer value. Unit: Clocks

9.2.3.1.30 SDRAM Timing Register 1 (DRAMTMG1)

9.2.3.1.30.1 Offset

Register	Offset
DRAMTMG1	104h

9.2.3.1.30.2 Diagram



9.2.3.1.30.3 Fields

Field	Function
31-21 —	Reserved
20-16 t_xp	tXP: Minimum time after power-down exit to any operation. For DDR3, this should be programmed to tXPDLL if slow powerdown exit is selected in MR0[12]. If C/A parity for DDR4 is used, set to (tXP+PL) instead. When the controller is operating in 1:2 frequency ratio mode, program this to (tXP/2) and round it up to the next integer value. Units: Clocks
15-14 —	Reserved
13-8 rd2pre	tRTP: Minimum time from read to precharge of same bank. - DDR2: tAL + BL/2 + max(tRTP, 2) - 2 - DDR3: tAL + max (tRTP, 4) - DDR4: Max of following two equations: tAL + max (tRTP, 4) or, RL + BL/2 - tRP (*). - mDDR: BL/2 - LPDDR2: Depends on if it's LPDDR2-S2 or LPDDR2-S4: LPDDR2-S2: BL/2 + tRTP - 1. LPDDR2-S4: BL/2 + max(tRTP,2) - 2. - LPDDR3: BL/2 + max(tRTP,4) - 4 - LPDDR4: BL/2 + max(tRTP,8) - 8 (*) When both DDR4 SDRAM and ST-MRAM are used simultaneously, use SDRAM's

Table continues on the next page...

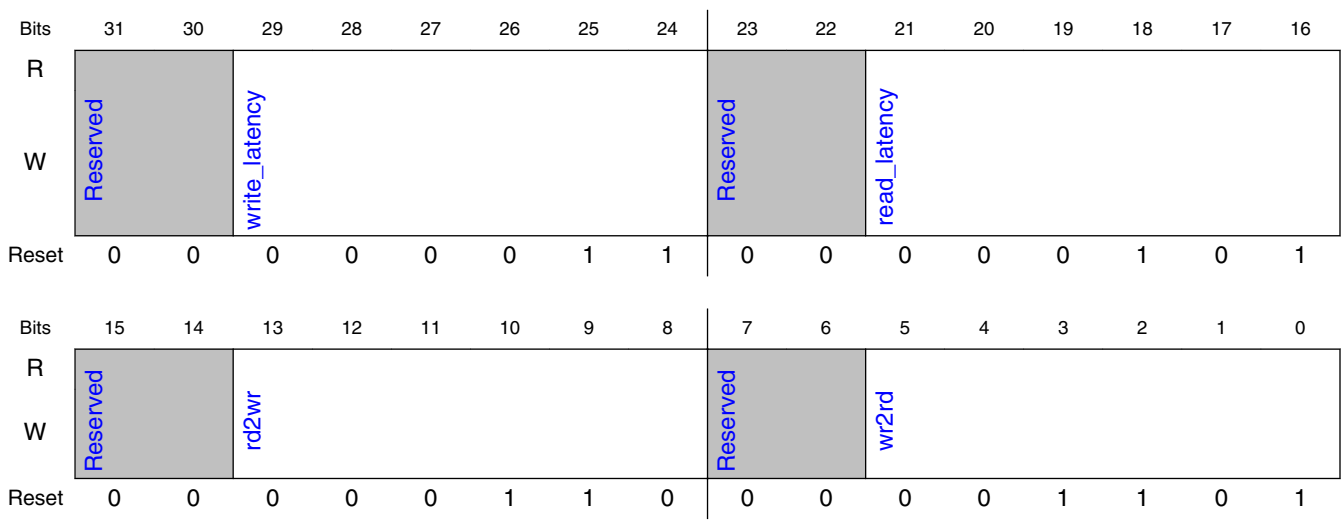
Field	Function
	tRP value for calculation. When the controller is operating in 1:2 mode, 1T mode, divide the above value by 2. No rounding up. When the controller is operating in 1:2 mode, 2T mode or LPDDR4 mode, divide the above value by 2 and round it up to the next integer value. Unit: Clocks.
7 —	Reserved
6-0 t_rc	tRC: Minimum time between activates to same bank. When the controller is operating in 1:2 frequency ratio mode, program this to (tRC/2) and round up to next integer value. Unit: Clocks.

9.2.3.1.31 SDRAM Timing Register 2 (DRAMTMG2)

9.2.3.1.31.1 Offset

Register	Offset
DRAMTMG2	108h

9.2.3.1.31.2 Diagram



9.2.3.1.31.3 Fields

Field	Function
31-30 —	Reserved

Table continues on the next page...

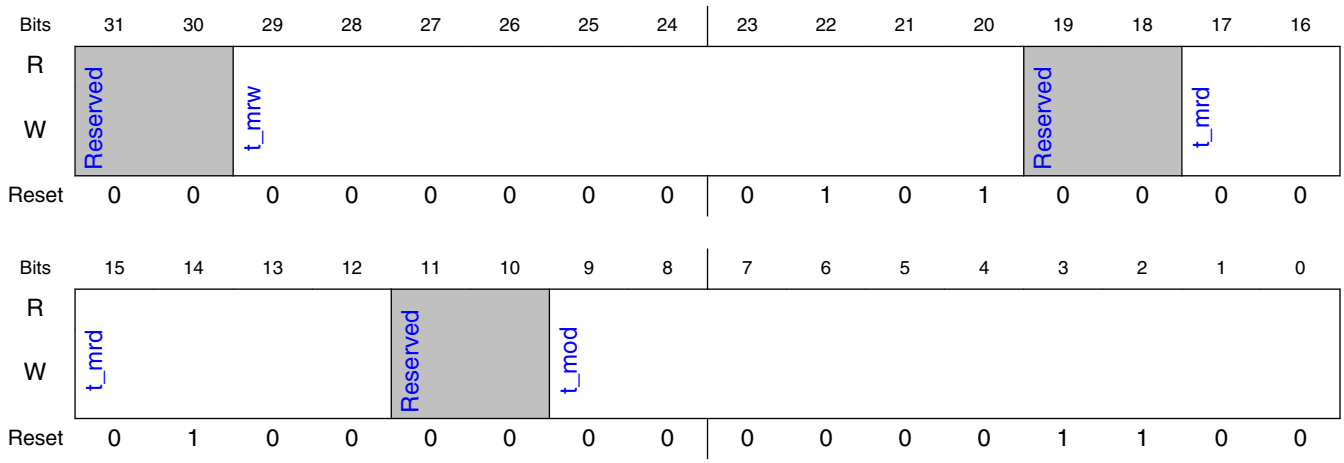
Field	Function
29-24 write_latency	Set to WL Time from write command to write data on SDRAM interface. This must be set to WL. For mDDR, it should normally be set to 1. Note that, depending on the PHY, if using RDIMM/LRDIMM, it may be necessary to adjust the value of WL to compensate for the extra cycle of latency through the RDIMM/LRDIMM. When the controller is operating in 1:2 frequency ratio mode, divide the value calculated using the above equation by 2, and round it up to next integer. This register field is not required for DDR2 and DDR3 (except if MEMC_TRAINING is set), as the DFI read and write latencies defined in DFITMG0 and DFITMG1 are sufficient for those protocols Unit: clocks
23-22 —	Reserved
21-16 read_latency	Set to RL Time from read command to read data on SDRAM interface. This must be set to RL. Note that, depending on the PHY, if using RDIMM/LRDIMM, it may be necessary to adjust the value of RL to compensate for the extra cycle of latency through the RDIMM/LRDIMM. When the controller is operating in 1:2 frequency ratio mode, divide the value calculated using the above equation by 2, and round it up to next integer. This register field is not required for DDR2 and DDR3 (except if MEMC_TRAINING is set), as the DFI read and write latencies defined in DFITMG0 and DFITMG1 are sufficient for those protocols Unit: clocks
15-14 —	Reserved
13-8 rd2wr	DDR2/3/mDDR: RL + BL/2 + 2 - WL DDR4: RL + BL/2 + 1 + WR_PREAMBLE - WL LPDDR2/LPDDR3: RL + BL/2 + RU(tDQSCKmax/tCK) + 1 - WL LPDDR4(DQ ODT is Disabled): RL + BL/2 + RU(tDQSCKmax/tCK) + WR_PREAMBLE + RD_POSTAMBLE - WL LPDDR4(DQ ODT is Enabled) : RL + BL/2 + RU(tDQSCKmax/tCK) + RD_POSTAMBLE - ODTLon - RU(tODTon(min)/tCK) Minimum time from read command to write command. Include time for bus turnaround and all per-bank, per-rank, and global constraints. Please see the relevant PHY databook for details of what should be included here. Unit: Clocks. Where: - WL = write latency - BL = burst length. This must match the value programmed in the BL bit of the mode register to the SDRAM - RL = read latency = CAS latency - WR_PREAMBLE = write preamble. This is unique to DDR4 and LPDDR4. - RD_POSTAMBLE = read postamble. This is unique to LPDDR4. For LPDDR2/LPDDR3/LPDDR4, if derating is enabled (DERATEEN.derate_enable=1), derated tDQSCKmax should be used. When the controller is operating in 1:2 frequency ratio mode, divide the value calculated using the above equation by 2, and round it up to next integer. Note that, depending on the PHY, if using LRDIMM, it may be necessary to adjust the value of this parameter to compensate for the extra cycle of latency through the LRDIMM.
7-6 —	Reserved
5-0 wr2rd	DDR4: CWL + PL + BL/2 + tWTR_L Others: CWL + BL/2 + tWTR In DDR4, minimum time from write command to read command for same bank group. In others, minimum time from write command to read command. Includes time for bus turnaround, recovery times, and all per-bank, per-rank, and global constraints. Unit: Clocks. Where: - CWL = CAS write latency - PL = Parity latency - BL = burst length. This must match the value programmed in the BL bit of the mode register to the SDRAM - tWTR_L = internal write to read command delay for same bank group. This comes directly from the SDRAM specification. - tWTR = internal write to read command delay. This comes directly from the SDRAM specification. Add one extra cycle for LPDDR2/LPDDR3/LPDDR4 operation. When the controller is operating in 1:2 mode, divide the value calculated using the above equation by 2, and round it up to next integer.

9.2.3.1.32 SDRAM Timing Register 3 (DRAMTMG3)

9.2.3.1.32.1 Offset

Register	Offset
DRAMTMG3	10Ch

9.2.3.1.32.2 Diagram



9.2.3.1.32.3 Fields

Field	Function
31-30 —	Reserved
29-20 t_mrw	Time to wait after a mode register write or read (MRW or MRR). Present only in designs configured to support LPDDR2, LPDDR3 or LPDDR4. LPDDR2 typically requires value of 5. LPDDR3 typically requires value of 10. LPDDR4: Set this to the larger of tMRW and tMRWCKEL. For LPDDR2, this register is used for the time from a MRW/MRR to all other commands. When the controller is operating in 1:2 frequency ratio mode, program this to the above values divided by 2 and round it up to the next integer value. For LDPDR3, this register is used for the time from a MRW/MRR to a MRW/MRR.
19-18 —	Reserved
17-12 t_mrd	tMRD: Cycles to wait after a mode register write or read. Depending on the connected SDRAM, tMRD represents: DDR2/mDDR: Time from MRS to any command DDR3/4: Time from MRS to MRS command LPDDR2: not used LPDDR3/4: Time from MRS to non-MRS command. When the controller is operating in 1:2 frequency ratio mode, program this to (tMRD/2) and round it up to the next integer value. If C/A parity for DDR4 is used, set to tMRD_PAR(tMOD+PL) instead.
11-10 —	Reserved
9-0 t_mod	tMOD: Parameter used only in DDR3 and DDR4. Cycles between load mode command and following non-load mode command. If C/A parity for DDR4 is used, set to tMOD_PAR(tMOD+PL) instead. Set to tMOD if controller is operating in 1:1 frequency ratio mode, or tMOD/2 (rounded up to next integer) if controller is operating in 1:2 frequency ratio mode. Note that if using RDIMM/LRDIMM, depending on the

Field	Function
	PHY, it may be necessary to adjust the value of this parameter to compensate for the extra cycle of latency applied to mode register writes by the RDIMM/LRDIMM chip. Also note that if using LRDIMM, the minimum value of this register is tMRD_L2 if controller is operating in 1:1 frequency ratio mode, or tMRD_L2/2 (rounded up to next integer) if controller is operating in 1:2 frequency ratio mode.

9.2.3.1.33 SDRAM Timing Register 4 (DRAMTMG4)

9.2.3.1.33.1 Offset

Register	Offset
DRAMTMG4	110h

9.2.3.1.33.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			t_rcd					Reserved				t_ccd			
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				t_rrd				Reserved			t_rp				
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1

9.2.3.1.33.3 Fields

Field	Function
31-29 —	Reserved
28-24 t_rcd	tRCD - tAL: Minimum time from activate to read or write command to same bank. When the controller is operating in 1:2 frequency ratio mode, program this to ((tRCD - tAL)/2) and round it up to the next integer value. Minimum value allowed for this register is 1, which implies minimum (tRCD - tAL) value to be 2 when the controller is operating in 1:2 frequency ratio mode. Unit: Clocks.
23-20 —	Reserved
19-16 t_ccd	DDR4: tCCD_L: This is the minimum time between two reads or two writes for same bank group. Others: tCCD: This is the minimum time between two reads or two writes. When the controller is operating in 1:2 frequency ratio mode, program this to (tCCD_L/2 or tCCD/2) and round it up to the next integer value. Unit: clocks.

Table continues on the next page...

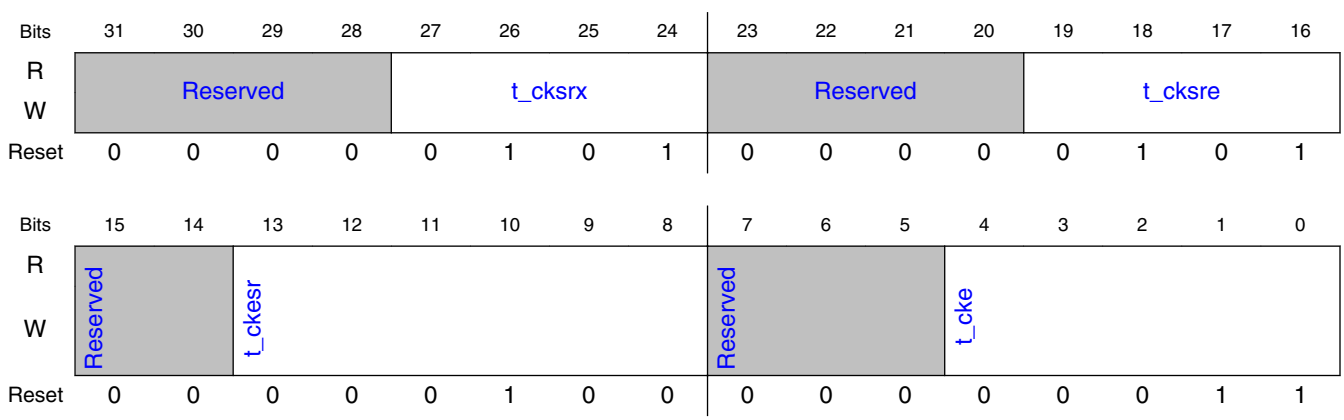
Field	Function
15-12 —	Reserved
11-8 t_rrd	DDR4: tRRD_L: Minimum time between activates from bank "a" to bank "b" for same bank group. Others: tRRD: Minimum time between activates from bank "a" to bank "b" When the controller is operating in 1:2 frequency ratio mode, program this to (tRRD_L/2 or tRRD/2) and round it up to the next integer value. Unit: Clocks.
7-5 —	Reserved
4-0 t_rp	tRP: Minimum time from precharge to activate of same bank. When the controller is operating in 1:1 frequency ratio mode, t_rp should be set to RoundUp(tRP/tCK). When the controller is operating in 1:2 frequency ratio mode, t_rp should be set to RoundDown(RoundUp(tRP/tCK)/2) + 1. When the controller is operating in 1:2 frequency ratio mode in LPDDR4, t_rp should be set to RoundUp(RoundUp(tRP/tCK)/2). Unit: Clocks.

9.2.3.1.34 SDRAM Timing Register 5 (DRAMTMG5)

9.2.3.1.34.1 Offset

Register	Offset
DRAMTMG5	114h

9.2.3.1.34.2 Diagram



9.2.3.1.34.3 Fields

Field	Function
31-28	Reserved

Table continues on the next page...

Field	Function
—	
27-24 t_cksrx	This is the time before Self Refresh Exit that CK is maintained as a valid clock before issuing SRX. Specifies the clock stable time before SRX. Recommended settings: - mDDR: 1 - LPDDR2: 2 - LPDDR3: 2 - LPDDR4: tCKCKEH - DDR2: 1 - DDR3: tCKSRX - DDR4: tCKSRX When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer.
23-20 —	Reserved
19-16 t_cksre	This is the time after Self Refresh Down Entry that CK is maintained as a valid clock. Specifies the clock disable delay after SRE. Recommended settings: - mDDR: 0 - LPDDR2: 2 - LPDDR3: 2 - LPDDR4: tCKCKEL - DDR2: 1 - DDR3: max (10 ns, 5 tCK) - DDR4: max (10 ns, 5 tCK) (+ PL(parity latency)(*)) (*)Only if CRCPARCTL1.caparity_disable_before_sr=0, this register should be increased by PL. When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer.
15-14 —	Reserved
13-8 t_ckesr	Minimum CKE low width for Self refresh or Self refresh power down entry to exit timing in memory clock cycles. Recommended settings: - mDDR: tRFC - LPDDR2: tCKESR - LPDDR3: tCKESR - LPDDR4: max(tCKELPD, tSR) - DDR2: tCKE - DDR3: tCKE + 1 - DDR4: tCKE + 1 (+ PL(parity latency)(*)) (*)Only if CRCPARCTL1.caparity_disable_before_sr=0, this register should be increased by PL. When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer.
7-5 —	Reserved
4-0 t_cke	Minimum number of cycles of CKE HIGH/LOW during power-down and self refresh. - LPDDR2/LPDDR3 mode: Set this to the larger of tCKE or tCKESR - LPDDR4 mode: Set this to the larger of tCKE, tCKELPD or tSR. - Non-LPDDR2/non-LPDDR3/non-LPDDR4 designs: Set this to tCKE value. When the controller is operating in 1:2 frequency ratio mode, program this to (value described above)/2 and round it up to the next integer value. Unit: Clocks.

9.2.3.1.35 SDRAM Timing Register 6 (DRAMTMG6)

9.2.3.1.35.1 Offset

Register	Offset
DRAMTMG6	118h

9.2.3.1.35.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				t_ckdpde				Reserved				t_ckdpdx			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												t_ckcsx			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

9.2.3.1.35.3 Fields

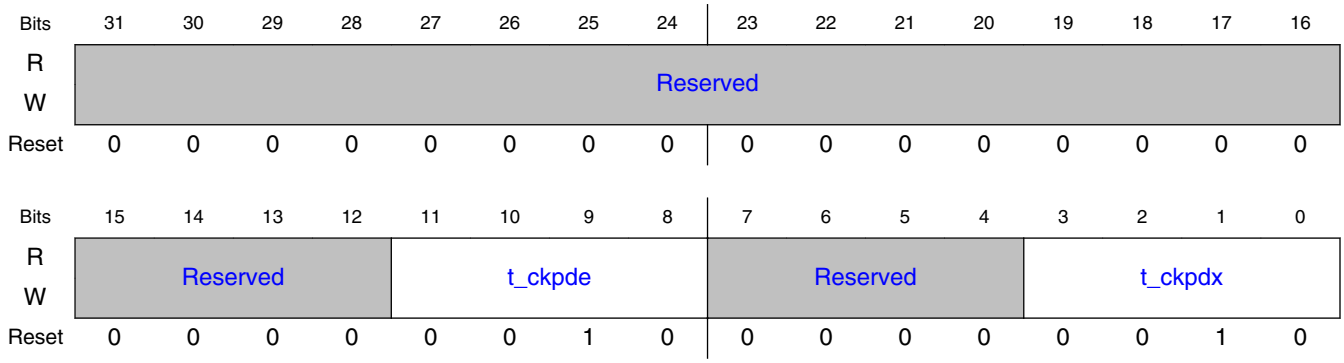
Field	Function
31-28 —	Reserved
27-24 t_ckdpde	This is the time after Deep Power Down Entry that CK is maintained as a valid clock. Specifies the clock disable delay after DPDE. Recommended settings: - mDDR: 0 - LPDDR2: 2 - LPDDR3: 2 When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. This is only present for designs supporting mDDR or LPDDR2/LPDDR3 devices.
23-20 —	Reserved
19-16 t_ckdpdx	This is the time before Deep Power Down Exit that CK is maintained as a valid clock before issuing DPDX. Specifies the clock stable time before DPDX. Recommended settings: - mDDR: 1 - LPDDR2: 2 - LPDDR3: 2 When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. This is only present for designs supporting mDDR or LPDDR2 devices.
15-4 —	Reserved
3-0 t_ckcsx	This is the time before Clock Stop Exit that CK is maintained as a valid clock before issuing Clock Stop Exit. Specifies the clock stable time before next command after Clock Stop Exit. Recommended settings: - mDDR: 1 - LPDDR2: tXP + 2 - LPDDR3: tXP + 2 - LPDDR4: tXP + 2 When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. This is only present for designs supporting mDDR or LPDDR2/LPDDR3/LPDDR4 devices.

9.2.3.1.36 SDRAM Timing Register 7 (DRAMTMG7)

9.2.3.1.36.1 Offset

Register	Offset
DRAMTMG7	11Ch

9.2.3.1.36.2 Diagram



9.2.3.1.36.3 Fields

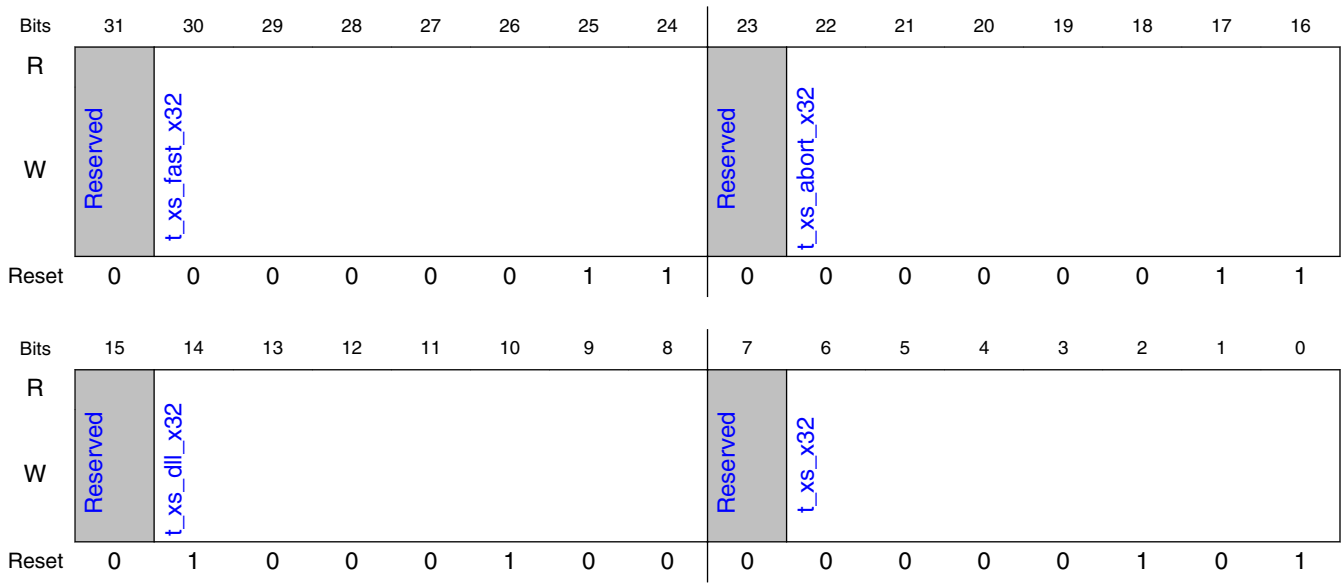
Field	Function
31-12 —	Reserved
11-8 t_ckpde	This is the time after Power Down Entry that CK is maintained as a valid clock. Specifies the clock disable delay after PDE. Recommended settings: - mDDR: 0 - LPDDR2: 2 - LPDDR3: 2 - LPDDR4: tCKCKEL When using DDR2/3/4 SDRAM, this register should be set to the same value as DRAMTMG5.t_cksre. When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. This is only present for designs supporting mDDR or LPDDR2/ LPDDR3/LPDDR4 devices.
7-4 —	Reserved
3-0 t_ckpdx	This is the time before Power Down Exit that CK is maintained as a valid clock before issuing PDX. Specifies the clock stable time before PDX. Recommended settings: - mDDR: 0 - LPDDR2: 2 - LPDDR3: 2 - LPDDR4: 2 When using DDR2/3/4 SDRAM, this register should be set to the same value as DRAMTMG5.t_cksrx. When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. This is only present for designs supporting mDDR or LPDDR2/LPDDR3/LPDDR4 devices.

9.2.3.1.37 SDRAM Timing Register 8 (DRAMTMG8)

9.2.3.1.37.1 Offset

Register	Offset
DRAMTMG8	120h

9.2.3.1.37.2 Diagram



9.2.3.1.37.3 Fields

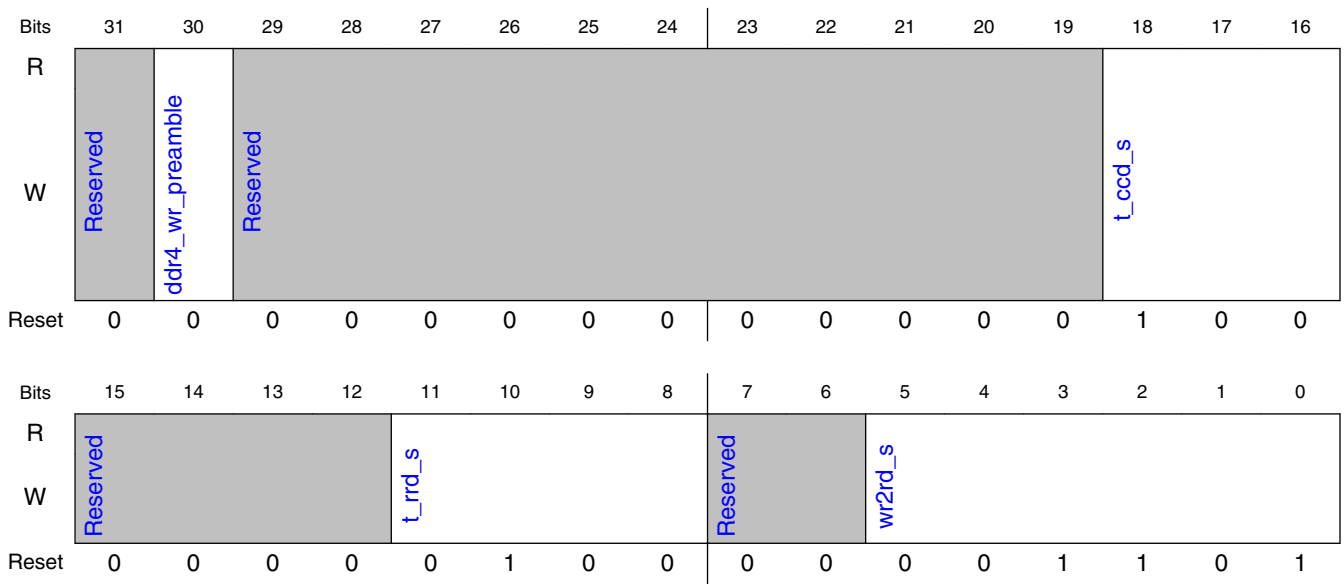
Field	Function
31 —	Reserved
30-24 t_xs_fast_x32	tXS_FAST: Exit Self Refresh to ZQCL, ZQCS and MRS (only CL, WR, RTP and Geardown mode). When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Unit: Multiples of 32 clocks. Note: This is applicable to only ZQCL/ZQCS commands. Note: Ensure this is less than or equal to t_xs_x32.
23 —	Reserved
22-16 t_xs_abort_x32	tXS_ABORT: Exit Self Refresh to commands not requiring a locked DLL in Self Refresh Abort. When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Unit: Multiples of 32 clocks. Note: Ensure this is less than or equal to t_xs_x32.
15 —	Reserved
14-8 t_xs_dll_x32	tXSDLL: Exit Self Refresh to commands requiring a locked DLL. When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Unit: Multiples of 32 clocks. Note: Used only for DDR2, DDR3 and DDR4 SDRAMs.
7 —	Reserved
6-0 t_xs_x32	tXS: Exit Self Refresh to commands not requiring a locked DLL. When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Unit: Multiples of 32 clocks. Note: Used only for DDR2, DDR3 and DDR4 SDRAMs.

9.2.3.1.38 SDRAM Timing Register 9 (DRAMTMG9)

9.2.3.1.38.1 Offset

Register	Offset
DRAMTMG9	124h

9.2.3.1.38.2 Diagram



9.2.3.1.38.3 Fields

Field	Function
31 —	Reserved
30 ddr4_wr_preamble	DDR4 Write preamble mode - 0: 1tCK preamble - 1: 2tCK preamble Present only with MEMC_FREQ_RATIO=2
29-19 —	Reserved
18-16 t_ccd_s	tCCD_S: This is the minimum time between two reads or two writes for different bank group. For bank switching (from bank "a" to bank "b"), the minimum time is this value + 1. When the controller is operating in 1:2 frequency ratio mode, program this to (tCCD_S/2) and round it up to the next integer value. Present only in designs configured to support DDR4. Unit: clocks.
15-12 —	Reserved

Table continues on the next page...

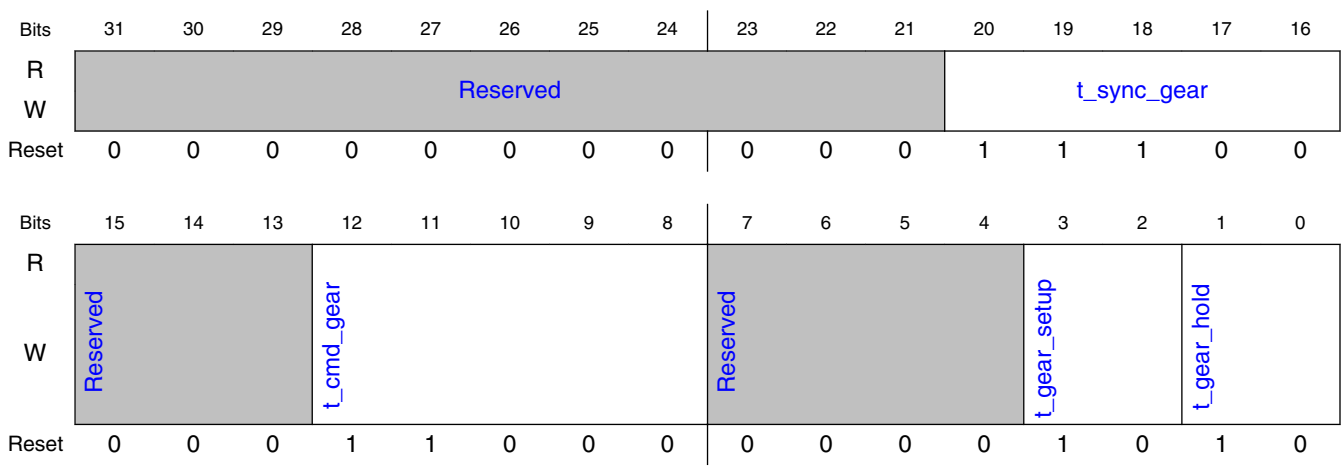
Field	Function
11-8 t_rrd_s	tRRD_S: Minimum time between activates from bank "a" to bank "b" for different bank group. When the controller is operating in 1:2 frequency ratio mode, program this to (tRRD_S/2) and round it up to the next integer value. Present only in designs configured to support DDR4. Unit: Clocks.
7-6 —	Reserved
5-0 wr2rd_s	CWL + PL + BL/2 + tWTR_S Minimum time from write command to read command for different bank group. Includes time for bus turnaround, recovery times, and all per-bank, per-rank, and global constraints. Present only in designs configured to support DDR4. Unit: Clocks. Where: - CWL = CAS write latency - PL = Parity latency - BL = burst length. This must match the value programmed in the BL bit of the mode register to the SDRAM - tWTR_S = internal write to read command delay for different bank group. This comes directly from the SDRAM specification. When the controller is operating in 1:2 mode, divide the value calculated using the above equation by 2, and round it up to next integer.

9.2.3.1.39 SDRAM Timing Register 10 (DRAMTMG10)

9.2.3.1.39.1 Offset

Register	Offset
DRAMTMG10	128h

9.2.3.1.39.2 Diagram



9.2.3.1.39.3 Fields

Field	Function
31-21	Reserved

Table continues on the next page...

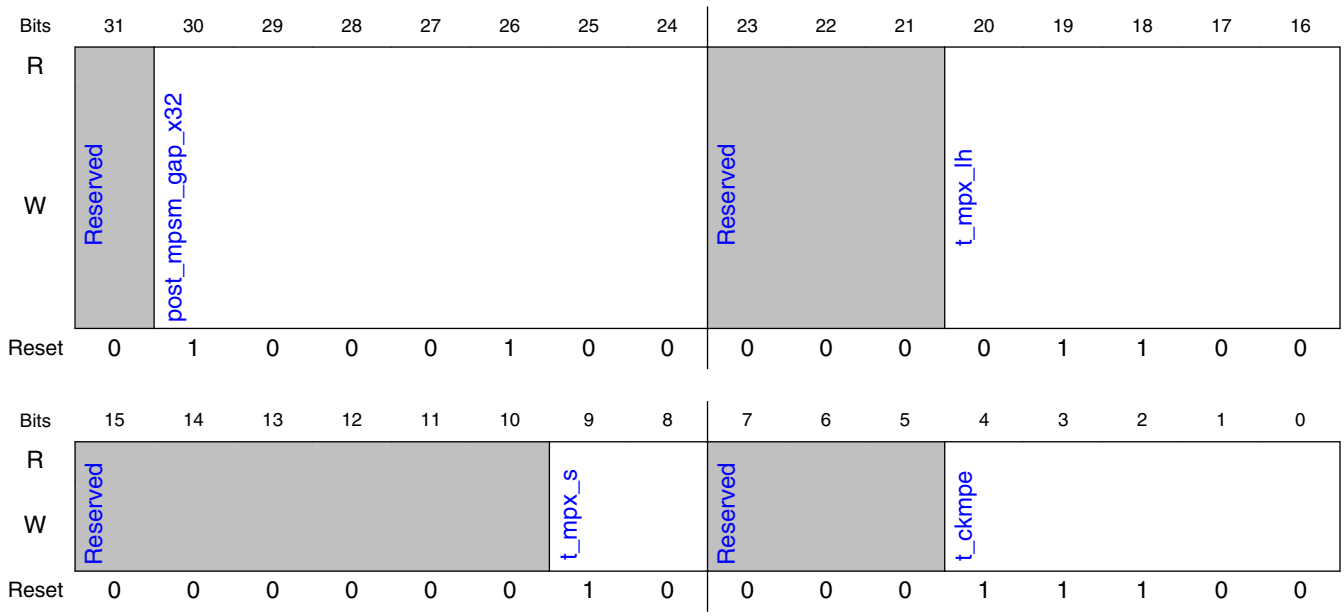
Field	Function
—	
20-16 t_sync_gear	Indicates the time between MRS command and the sync pulse time. This must be even number of clocks. For DDR4-2666 and DDR4-3200, this parameter is defined as tMOD(min)+4nCK tMOD(min) is greater of 24nCK or 15ns 15ns / .625ns = 24 Max value for this register is 24+4 = 28 When the controller is operating in 1:2 mode, program this to (tSYNC_GEAR/2) and round it up to the next integer value. Unit: Clocks
15-13 —	Reserved
12-8 t_cmd_gear	Sync pulse to first valid command. For DDR4-2666 and DDR4-3200, this parameter is defined as tMOD(min) tMOD(min) is greater of 24nCK or 15ns 15ns / .625ns = 24 Max value for this register is 24 When the controller is operating in 1:2 mode, program this to (tCMD_GEAR/2) and round it up to the next integer value. Unit: Clocks
7-4 —	Reserved
3-2 t_gear_setup	Geardown setup time. Minimum value of this register is 1. Zero is invalid. For DDR4-2666 and DDR4-3200, this parameter is defined as 2 clks When the controller is operating in 1:2 frequency ratio mode, program this to (tGEAR_setup/2) and round it up to the next integer value. Unit: Clocks
1-0 t_gear_hold	Geardown hold time. Minimum value of this register is 1. Zero is invalid. For DDR4-2666 and DDR4-3200, this parameter is defined as 2 clks When the controller is operating in 1:2 frequency ratio mode, program this to (tGEAR_hold/2) and round it up to the next integer value. Unit: Clocks

9.2.3.1.40 SDRAM Timing Register 11 (DRAMTMG11)

9.2.3.1.40.1 Offset

Register	Offset
DRAMTMG11	12Ch

9.2.3.1.40.2 Diagram



9.2.3.1.40.3 Fields

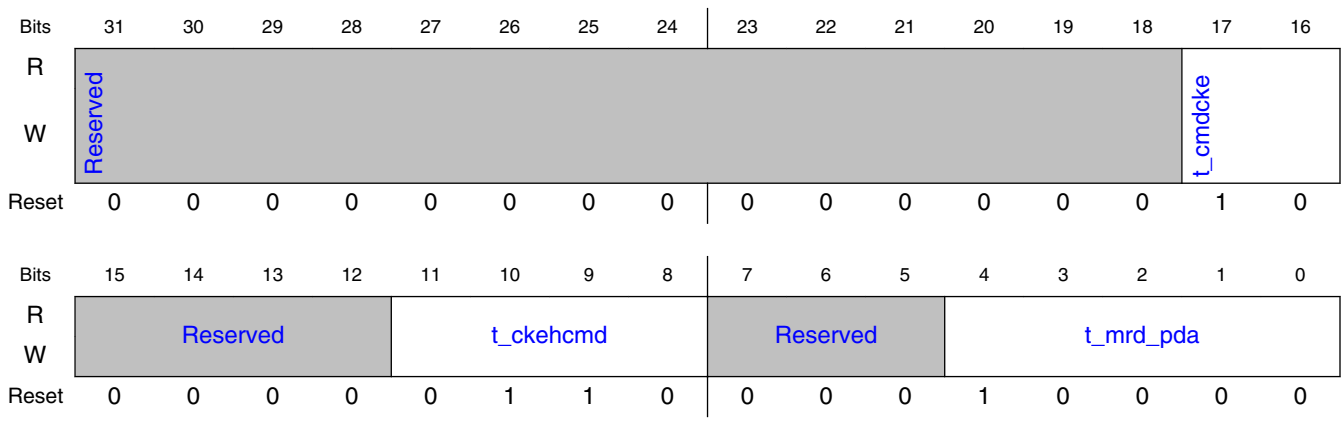
Field	Function
31 —	Reserved
30-24 post_mpsm_gap_x32	tXMPDLL: This is the minimum Exit MPSM to commands requiring a locked DLL. When the controller is operating in 1:2 frequency ratio mode, program this to (tXMPDLL/2) and round it up to the next integer value. Present only in designs configured to support DDR4. Unit: Multiples of 32 clocks.
23-21 —	Reserved
20-16 t_mpx_lh	tMPX_LH: This is the minimum CS_n Low hold time to CKE rising edge. When the controller is operating in 1:2 frequency ratio mode, program this to RoundUp(tMPX_LH/2)+1. Present only in designs configured to support DDR4. Unit: clocks.
15-10 —	Reserved
9-8 t_mpx_s	tMPX_S: Minimum time CS setup time to CKE. When the controller is operating in 1:2 frequency ratio mode, program this to (tMPX_S/2) and round it up to the next integer value. Present only in designs configured to support DDR4. Unit: Clocks.
7-5 —	Reserved
4-0 t_ckmpe	tCKMPE: Minimum valid clock requirement after MPSM entry. Present only in designs configured to support DDR4. Unit: Clocks. When the controller is operating in 1:2 frequency ratio mode, divide the value calculated using the above equation by 2, and round it up to next integer.

9.2.3.1.41 SDRAM Timing Register 12 (DRAMTMG12)

9.2.3.1.41.1 Offset

Register	Offset
DRAMTMG12	130h

9.2.3.1.41.2 Diagram



9.2.3.1.41.3 Fields

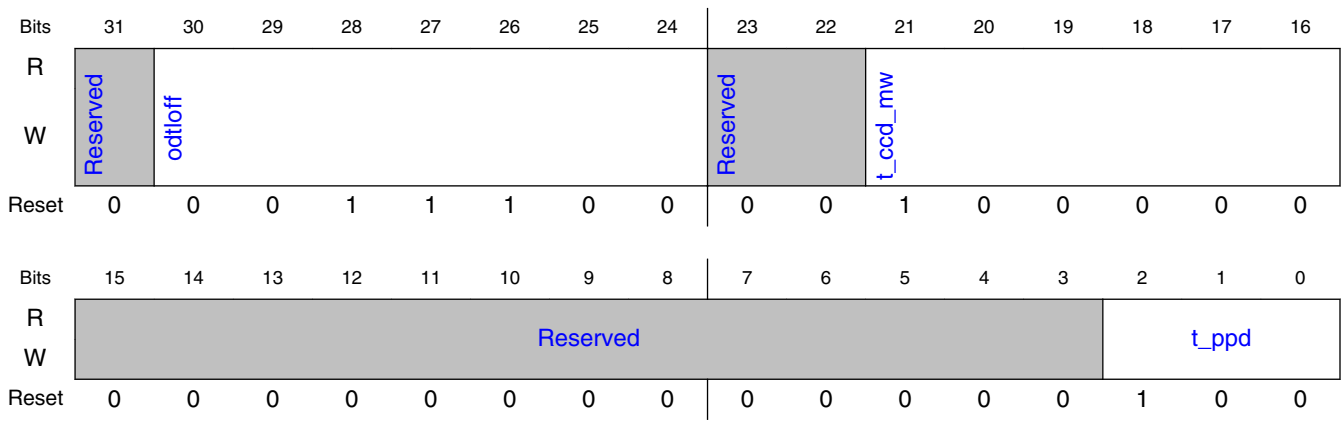
Field	Function
31-18 —	Reserved
17-16 t_cmdcke	tCMDCKE: Delay from valid command to CKE input LOW. Set this to the larger of tESCKE or tCMDCKE When the controller is operating in 1:2 frequency ratio mode, program this to (max(tESCKE, tCMDCKE)/2) and round it up to the next integer value.
15-12 —	Reserved
11-8 t_ckehcmd	tCKEHCMD: Valid command requirement after CKE input HIGH. When the controller is operating in 1:2 frequency ratio mode, program this to (tCKEHCMD/2) and round it up to the next integer value.
7-5 —	Reserved
4-0 t_mrd_pda	tMRD_PDA: This is the Mode Register Set command cycle time in PDA mode. When the controller is operating in 1:2 frequency ratio mode, program this to (tMRD_PDA/2) and round it up to the next integer value.

9.2.3.1.42 SDRAM Timing Register 13 (DRAMTMG13)

9.2.3.1.42.1 Offset

Register	Offset
DRAMTMG13	134h

9.2.3.1.42.2 Diagram



9.2.3.1.42.3 Fields

Field	Function
31 —	Reserved
30-24 odtloff	LPDDR4: tODTLoff: This is the latency from CAS-2 command to tODToff reference. When the controller is operating in 1:2 frequency ratio mode, program this to (tODTLoff/2) and round it up to the next integer value. Unit: Clocks.
23-22 —	Reserved
21-16 t_ccd_mw	LPDDR4: tCCDMW: This is the minimum time from write or masked write to masked write command for same bank. When the controller is operating in 1:2 frequency ratio mode, program this to (tCCDMW/2) and round it up to the next integer value. Unit: Clocks.
15-3 —	Reserved
2-0 t_ppd	LPDDR4: tPPD: This is the minimum time from precharge to precharge command. When the controller is operating in 1:2 frequency ratio mode, program this to (tPPD/2) and round it up to the next integer value. Unit: Clocks.

9.2.3.1.43 SDRAM Timing Register 14 (DRAMTMG14)

9.2.3.1.43.1 Offset

Register	Offset
DRAMTMG14	138h

9.2.3.1.43.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				t_xsr											
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0

9.2.3.1.43.3 Fields

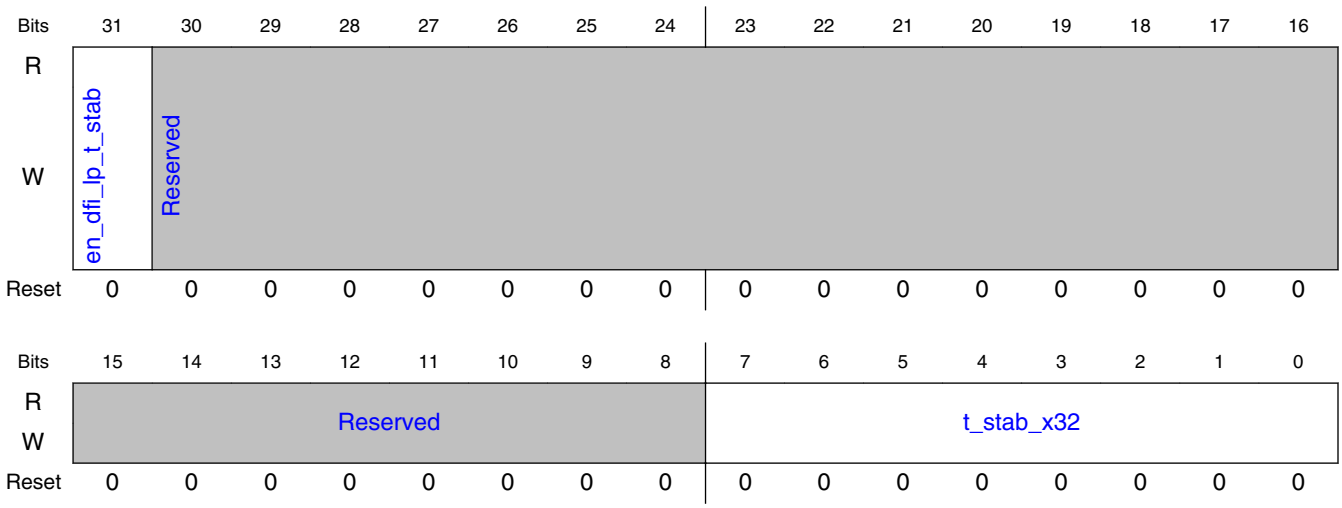
Field	Function
31-12 —	Reserved
11-0 t_xsr	tXSR: Exit Self Refresh to any command. When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Note: Used only for mDDR/LPDDR2/LPDDR3/LPDDR4 mode.

9.2.3.1.44 SDRAM Timing Register 15 (DRAMTMG15)

9.2.3.1.44.1 Offset

Register	Offset
DRAMTMG15	13Ch

9.2.3.1.44.2 Diagram



9.2.3.1.44.3 Fields

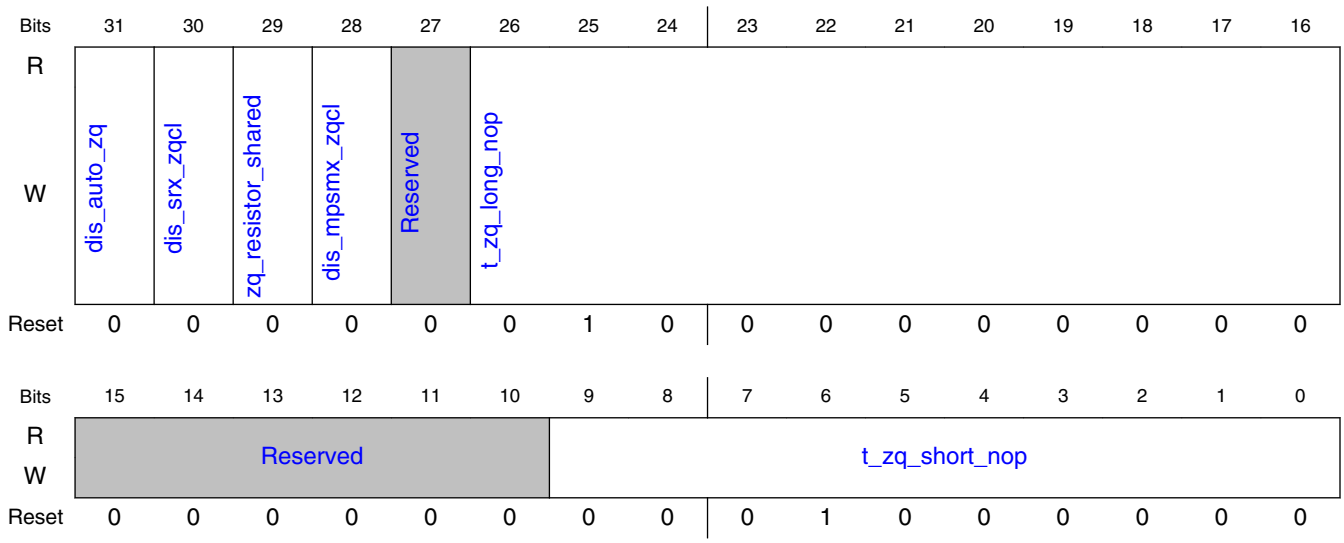
Field	Function
31 en_dfi_lp_t_stab	Enable DFI tSTAB 0b - Disable using tSTAB when exiting DFI LP 1b - Enable using tSTAB when exiting DFI LP. Needs to be set when the PHY is stopping the clock during DFI LP to save maximum power.
30-8 —	Reserved
7-0 t_stab_x32	tSTAB: Stabilization time. It is required in the following two cases for DDR3/DDR4 RDIMM : - when exiting power saving mode, if the clock was stopped, after re-enabling it the clock must be stable for a time specified by tSTAB - in the case of input clock frequency change (DDR4) - after issuing control words that refers to clock timing (Specification: 6us for DDR3, 5us for DDR4) When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. Unit: Multiples of 32 clock cycles.

9.2.3.1.45 ZQ Control Register 0 (ZQCTL0)

9.2.3.1.45.1 Offset

Register	Offset
ZQCTL0	180h

9.2.3.1.45.2 Diagram



9.2.3.1.45.3 Fields

Field	Function
31 dis_auto_zq	Disable Auto ZQCS/MPC 0b - Internally generate ZQCS/MPC(ZQ calibration) commands based on ZQCTL1.t_zq_short_interval_x1024. 1b - Disable DDRC generation of ZQCS/MPC(ZQ calibration) command. Register DBGCMD.zq_calib_short can be used instead to issue ZQ calibration request from APB module.
30 dis_srx_zqcl	Disable ZQCL/MPC 0b - Enable issuing of ZQCL/MPC(ZQ calibration) command at Self-Refresh/SR-Powerdown exit. Only applicable when run in DDR3 or DDR4 or LPDDR2 or LPDDR3 or LPDDR4 mode. This is only present for designs supporting DDR3/DDR4 or LPDDR2/LPDDR3/LPDDR4 devices. 1b - Disable issuing of ZQCL/MPC(ZQ calibration) command at Self-Refresh/SR-Powerdown exit. Only applicable when run in DDR3 or DDR4 or LPDDR2 or LPDDR3 or LPDDR4 mode.
29 zq_resistor_shared	ZQ resistor sharing 0b - ZQ resistor is not shared. This is only present for designs supporting DDR3/DDR4 or LPDDR2/LPDDR3/LPDDR4 devices. 1b - Denotes that ZQ resistor is shared between ranks. Means ZQinit/ZQCL/ZQCS/MPC(ZQ calibration) commands are sent to one rank at a time with tZQinit/tZQCL/tZQCS/tZQCAL/tZQLAT timing met between commands so that commands to different ranks do not overlap.
28 dis_mpsmx_zqcl	Do not issue ZQCL command at Maximum Power Save Mode exit if the DDRC_SHARED_AC configuration parameter is set. Program it to 1'b1. The software can send ZQCS after exiting MPSM mode. 0b - Enable issuing of ZQCL command at Maximum Power Saving Mode exit. Only applicable when run in DDR4 mode. This is only present for designs supporting DDR4 devices. 1b - Disable issuing of ZQCL command at Maximum Power Saving Mode exit. Only applicable when run in DDR4 mode.
27 —	Reserved
26-16 t_zq_long_nop	tZQoper for DDR3/DDR4, tZQCL for LPDDR2/LPDDR3, tZQCAL for LPDDR4: Number of DFI clock cycles of NOP required after a ZQCL (ZQ calibration long)/MPC(ZQ Start) command is issued to

Table continues on the next page...

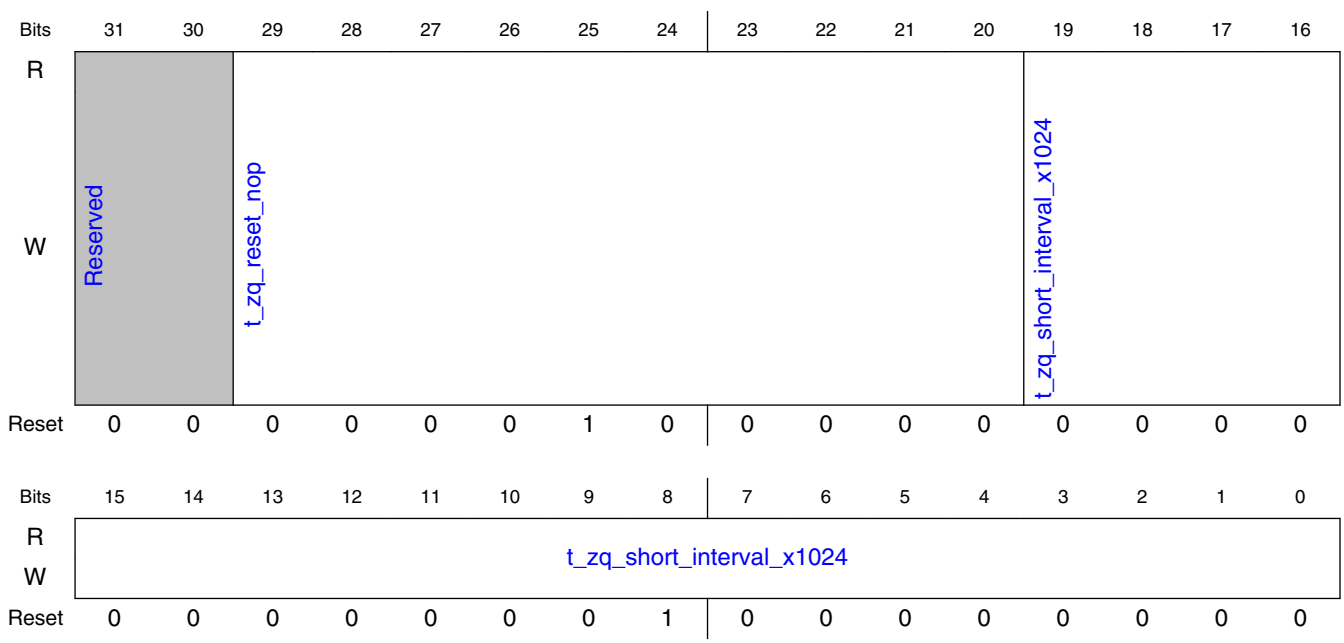
Field	Function
	SDRAM. When the controller is operating in 1:2 frequency ratio mode: DDR3/DDR4: program this to tZQoper/2 and round it up to the next integer value. LPDDR2/LPDDR3: program this to tZQCL/2 and round it up to the next integer value. LPDDR4: program this to tZQCAL/2 and round it up to the next integer value. This is only present for designs supporting DDR3/DDR4 or LPDDR2/LPDDR3/LPDDR4 devices.
15-10 —	Reserved
9-0 t_zq_short_nop	tZQCS for DDR3/DD4/LPDDR2/LPDDR3, tZQLAT for LPDDR4: Number of DFI clock cycles of NOP required after a ZQCS (ZQ calibration short)/MPC(ZQ Latch) command is issued to SDRAM. When the controller is operating in 1:2 frequency ratio mode, program this to tZQCS/2 and round it up to the next integer value. This is only present for designs supporting DDR3/DDR4 or LPDDR2/LPDDR3/LPDDR4 devices.

9.2.3.1.46 ZQ Control Register 1 (ZQCTL1)

9.2.3.1.46.1 Offset

Register	Offset
ZQCTL1	184h

9.2.3.1.46.2 Diagram



9.2.3.1.46.3 Fields

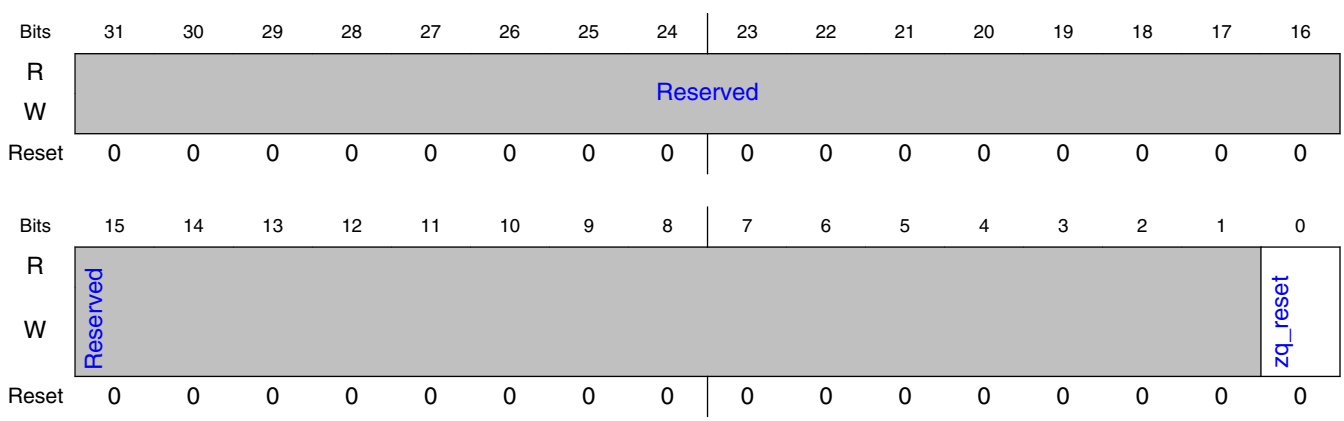
Field	Function
31-30 —	Reserved
29-20 t_zq_reset_nop	tZQReset: Number of DFI clock cycles of NOP required after a ZQReset (ZQ calibration Reset) command is issued to SDRAM. When the controller is operating in 1:2 frequency ratio mode, program this to tZQReset/2 and round it up to the next integer value. This is only present for designs supporting LPDDR2/LPDDR3/LPDDR4 devices.
19-0 t_zq_short_interval_x1024	Average interval to wait between automatically issuing ZQCS (ZQ calibration short)/MPC(ZQ calibration) commands to DDR3/DDR4/LPDDR2/LPDDR3/LPDDR4 devices. Meaningless, if ZQCTL0.dis_auto_zq=1. Unit: 1024 DFI clock cycles. This is only present for designs supporting DDR3/DDR4 or LPDDR2/LPDDR3/LPDDR4 devices.

9.2.3.1.47 ZQ Control Register 2 (ZQCTL2)

9.2.3.1.47.1 Offset

Register	Offset
ZQCTL2	188h

9.2.3.1.47.2 Diagram



9.2.3.1.47.3 Fields

Field	Function
31-1 —	Reserved

Table continues on the next page...

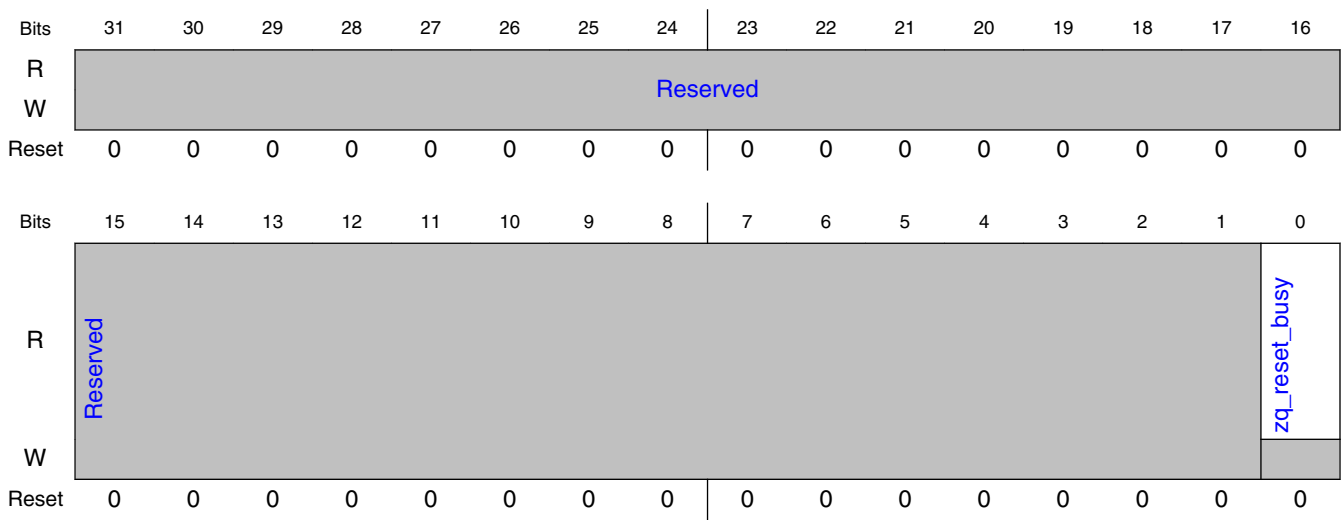
Field	Function
0 zq_reset	Setting this register bit to 1 triggers a ZQ Reset operation. When the ZQ Reset operation is complete, the DDRC automatically clears this bit. It is recommended NOT to set this signal if in Init, Self-Refresh(except LPDDR4) or SR-Powerdown(LPDDR4) or Deep power-down operating modes. This is only present for designs supporting LPDDR2/LPDDR3/LPDDR4 devices.

9.2.3.1.48 ZQ Status Register (ZQSTAT)

9.2.3.1.48.1 Offset

Register	Offset
ZQSTAT	18Ch

9.2.3.1.48.2 Diagram



9.2.3.1.48.3 Fields

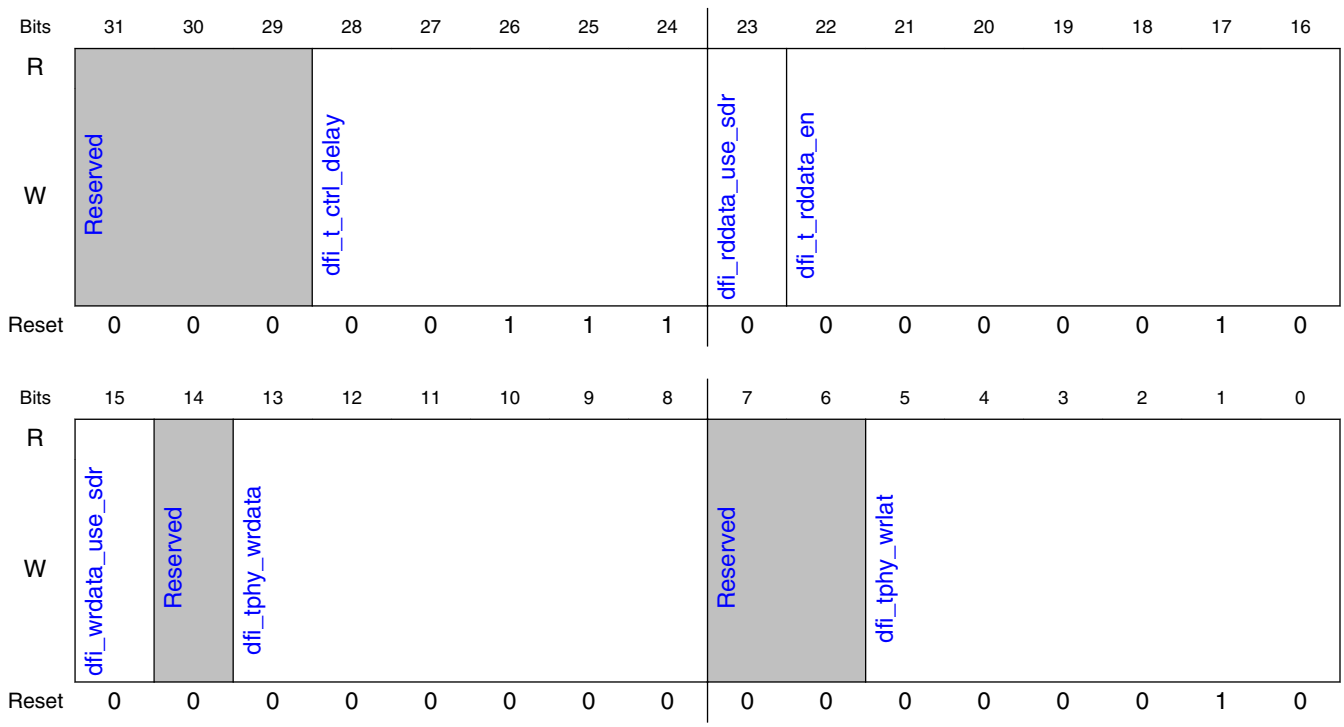
Field	Function
31-1 —	Reserved
0 zq_reset_busy	SoC core may initiate a ZQ Reset operation only if this signal is low. This signal goes high in the clock after the DDRC accepts the ZQ Reset request. It goes low when the ZQ Reset command is issued to the SDRAM and the associated NOP period is over. It is recommended not to perform ZQ Reset commands when this signal is high. 0b - Indicates that the SoC core can initiate a ZQ Reset operation 1b - Indicates that ZQ Reset operation is in progress

9.2.3.1.49 DFI Timing Register 0 (DFITMG0)

9.2.3.1.49.1 Offset

Register	Offset
DFITMG0	190h

9.2.3.1.49.2 Diagram



9.2.3.1.49.3 Fields

Field	Function
31-29 —	Reserved
28-24 dfi_t_ctrl_delay	Specifies the number of DFI clock cycles after an assertion or de-assertion of the DFI control signals that the control signals at the PHY-DRAM interface reflect the assertion or de-assertion. If the DFI clock and the memory clock are not phase-aligned, this timing parameter should be rounded up to the next integer value. Note that if using RDIMM/LRDIMM, it is necessary to increment this parameter by RDIMM's/ LRDIMM's extra cycle of latency in terms of DFI clock.
23	Defines whether dfi_rddata_en/dfi_rddata/dfi_rddata_valid is generated using HDR (DFI clock) or SDR (DFI PHY clock) values. Selects whether value in DFITMG0.dfi_t_rddata_en is in terms of HDR (DFI

Table continues on the next page...

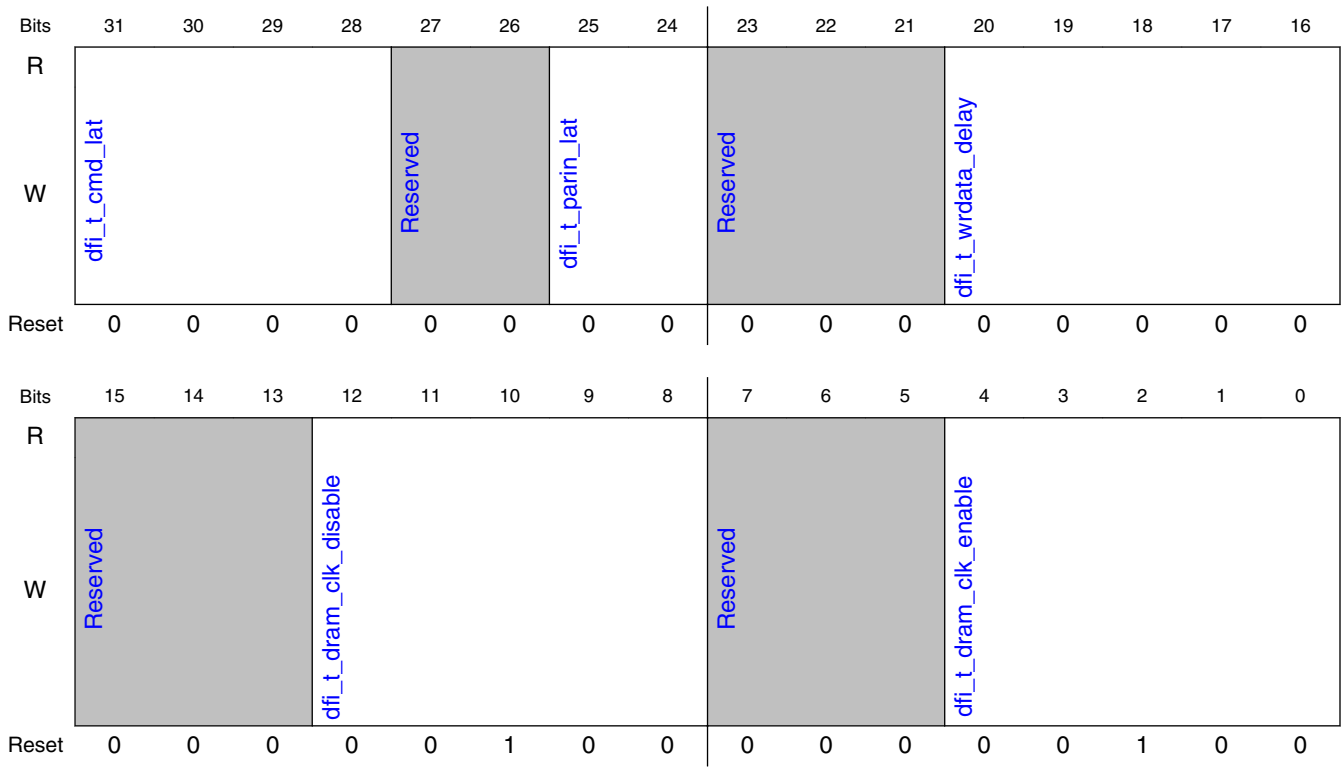
Field	Function
dfi_rddata_use_sdr	clock) or SDR (DFI PHY clock) cycles: - 0 in terms of HDR (DFI clock) cycles - 1 in terms of SDR (DFI PHY clock) cycles Refer to PHY specification for correct value.
22-16 dfi_t_rddata_en	Time from the assertion of a read command on the DFI interface to the assertion of the dfi_rddata_en signal. Refer to PHY specification for correct value. This corresponds to the DFI parameter trddata_en. Note that, depending on the PHY, if using RDIMM/LRDIMM, it may be necessary to use the adjusted value of CL in the calculation of trddata_en. This is to compensate for the extra cycle(s) of latency through the RDIMM/LRDIMM. Unit: DFI clock cycles or DFI PHY clock cycles, depending on DFITMG0.dfi_rddata_use_sdr.
15 dfi_wrdata_use_sdr	Defines whether dfi_wrdata_en/dfi_wrdata/dfi_wrdata_mask is generated using HDR (DFI clock) or SDR (DFI PHY clock) values Selects whether value in DFITMG0.dfi_tphy_wrlat is in terms of HDR (DFI clock) or SDR (DFI PHY clock) cycles Selects whether value in DFITMG0.dfi_tphy_wrdata is in terms of HDR (DFI clock) or SDR (DFI PHY clock) cycles - 0 in terms of HDR (DFI clock) cycles - 1 in terms of SDR (DFI PHY clock) cycles Refer to PHY specification for correct value.
14 —	Reserved
13-8 dfi_tphy_wrdata	Specifies the number of clock cycles between when dfi_wrdata_en is asserted to when the associated write data is driven on the dfi_wrdata signal. This corresponds to the DFI timing parameter tphy_wrdata. Refer to PHY specification for correct value. Note, max supported value is 8. Unit: DFI clock cycles or DFI PHY clock cycles, depending on DFITMG0.dfi_wrdata_use_sdr.
7-6 —	Reserved
5-0 dfi_tphy_wrlat	Write latency Number of clocks from the write command to write data enable (dfi_wrdata_en). This corresponds to the DFI timing parameter tphy_wrlat. Refer to PHY specification for correct value.Note that, depending on the PHY, if using RDIMM/LRDIMM, it may be necessary to use the adjusted value of CL in the calculation of tphy_wrlat. This is to compensate for the extra cycle(s) of latency through the RDIMM/LRDIMM. Unit: DFI clock cycles or DFI PHY clock cycles, depending on DFITMG0.dfi_wrdata_use_sdr.

9.2.3.1.50 DFI Timing Register 1 (DFITMG1)

9.2.3.1.50.1 Offset

Register	Offset
DFITMG1	194h

9.2.3.1.50.2 Diagram



9.2.3.1.50.3 Fields

Field	Function
31-28 dfi_t_cmd_lat	Specifies the number of DFI PHY clock cycles between when the dfi_cs signal is asserted and when the associated command is driven. This field is used for CAL mode, should be set to '0' or the value which matches the CAL mode register setting in the DRAM. If the PHY can add the latency for CAL mode, this should be set to '0'. Valid Range: 0, 3, 4, 5, 6, and 8
27-26 —	Reserved
25-24 dfi_t_parin_lat	Specifies the number of DFI PHY clock cycles between when the dfi_cs signal is asserted and when the associated dfi_parity_in signal is driven.
23-21 —	Reserved
20-16 dfi_t_wrdata_delay	Specifies the number of DFI clock cycles between when the dfi_wrdata_en signal is asserted and when the corresponding write data transfer is completed on the DRAM bus. This corresponds to the DFI timing parameter twrdata_delay. Refer to PHY specification for correct value. For DFI 3.0 PHY, set to twrdata_delay, a new timing parameter introduced in DFI 3.0. For DFI 2.1 PHY, set to tphy_wrdata + (delay of DFI write data to the DRAM). Value to be programmed is in terms of DFI clocks, not PHY clocks. In FREQ_RATIO=2, divide PHY's value by 2 and round up to next integer. If using DFITMG0.dfi_wrdata_use_sdr=1, add 1 to the value. Unit: Clocks
15-13 —	Reserved

Table continues on the next page...

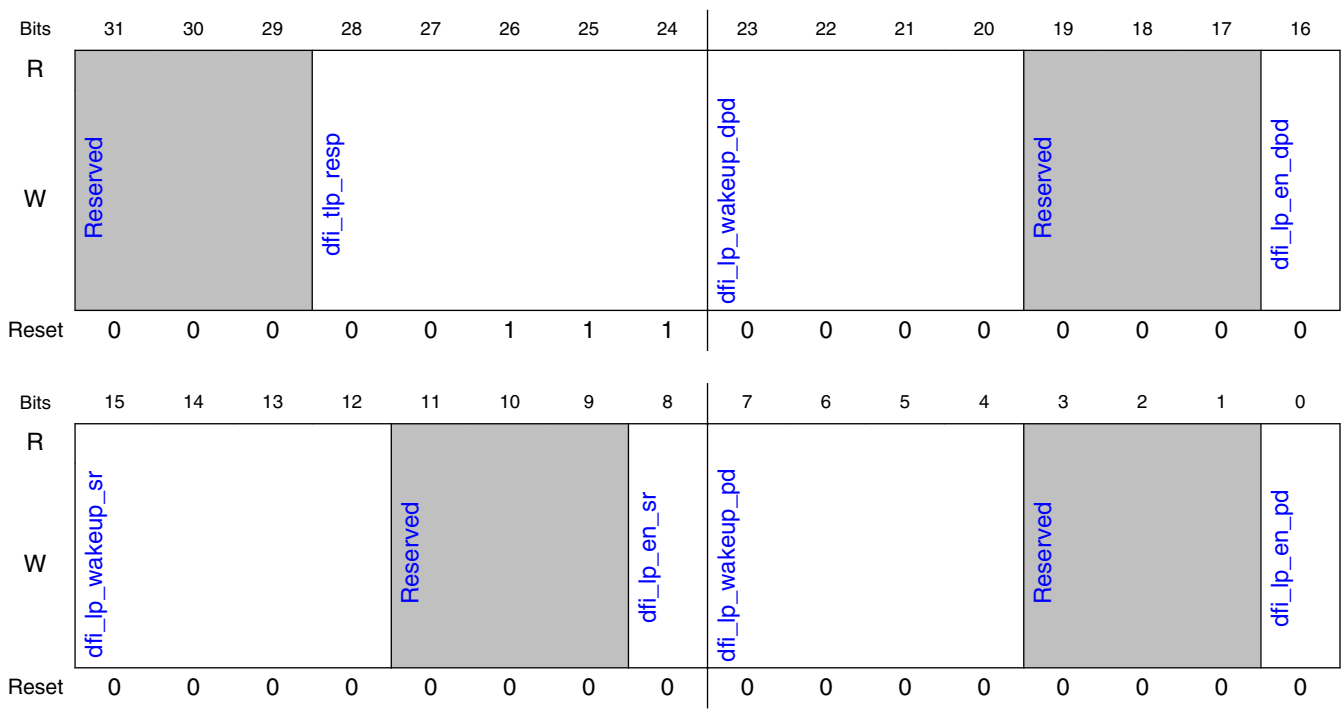
Field	Function
12-8 dfi_t_dram_clk_disable	Specifies the number of DFI clock cycles from the assertion of the dfi_dram_clk_disable signal on the DFI until the clock to the DRAM memory devices, at the PHY-DRAM boundary, maintains a low value. If the DFI clock and the memory clock are not phase aligned, this timing parameter should be rounded up to the next integer value.
7-5 —	Reserved
4-0 dfi_t_dram_clk_enable	Specifies the number of DFI clock cycles from the de-assertion of the dfi_dram_clk_disable signal on the DFI until the first valid rising edge of the clock to the DRAM memory devices, at the PHY-DRAM boundary. If the DFI clock and the memory clock are not phase aligned, this timing parameter should be rounded up to the next integer value.

9.2.3.1.51 DFI Low Power Configuration Register 0 (DFILPCFG0)

9.2.3.1.51.1 Offset

Register	Offset
DFILPCFG0	198h

9.2.3.1.51.2 Diagram



9.2.3.1.51.3 Fields

Field	Function
31-29 —	Reserved
28-24 dfi_tlp_resp	Setting in DFI clock cycles for DFI's tlp_resp time. Same value is used for both Power Down, Self Refresh, Deep Power Down and Maximum Power Saving modes. DFI 2.1 specification onwards, recommends using a fixed value of 7 always.
23-20 dfi_lp_wakeup_dpd	Value in DFI clock cycles to drive on dfi_lp_wakeup signal when Deep Power Down mode is entered. Determines the DFI's tlp_wakeup time: This is only present for designs supporting mDDR or LPDDR2/ LPDDR3 devices. 0000b - 16 cycles 0001b - 32 cycles 0010b - 64 cycles 0011b - 128 cycles 0100b - 256 cycles 0101b - 512 cycles 0110b - 1024 cycles 0111b - 2048 cycles 1000b - 4096 cycles 1001b - 8192 cycles 1010b - 16384 cycles 1011b - 32768 cycles 1100b - 65536 cycles 1101b - 131072 cycles 1110b - 262144 cycles 1111b - Unlimited cycles
19-17 —	Reserved
16 dfi_lp_en_dpd	Enables DFI Low Power interface handshaking during Deep Power Down Entry/Exit. - 0 - Disabled - 1 - Enabled This is only present for designs supporting mDDR or LPDDR2/LPDDR3 devices.
15-12 dfi_lp_wakeup_sr	Value in DFI clpck cycles to drive on dfi_lp_wakeup signal when Self Refresh mode is entered. Determines the DFI's tlp_wakeup time: 0000b - 16 cycles 0001b - 32 cycles 0010b - 64 cycles 0011b - 128 cycles 0100b - 256 cycles 0101b - 512 cycles 0110b - 1024 cycles 0111b - 2048 cycles 1000b - 4096 cycles 1001b - 8192 cycles 1010b - 16384 cycles 1011b - 32768 cycles 1100b - 65536 cycles 1101b - 131072 cycles 1110b - 262144 cycles 1111b - Unlimited cycles
11-9 —	Reserved
8	Enables DFI Low Power interface handshaking during Self Refresh Entry/Exit. - 0 - Disabled - 1 - Enabled 0b - Disabled

Table continues on the next page...

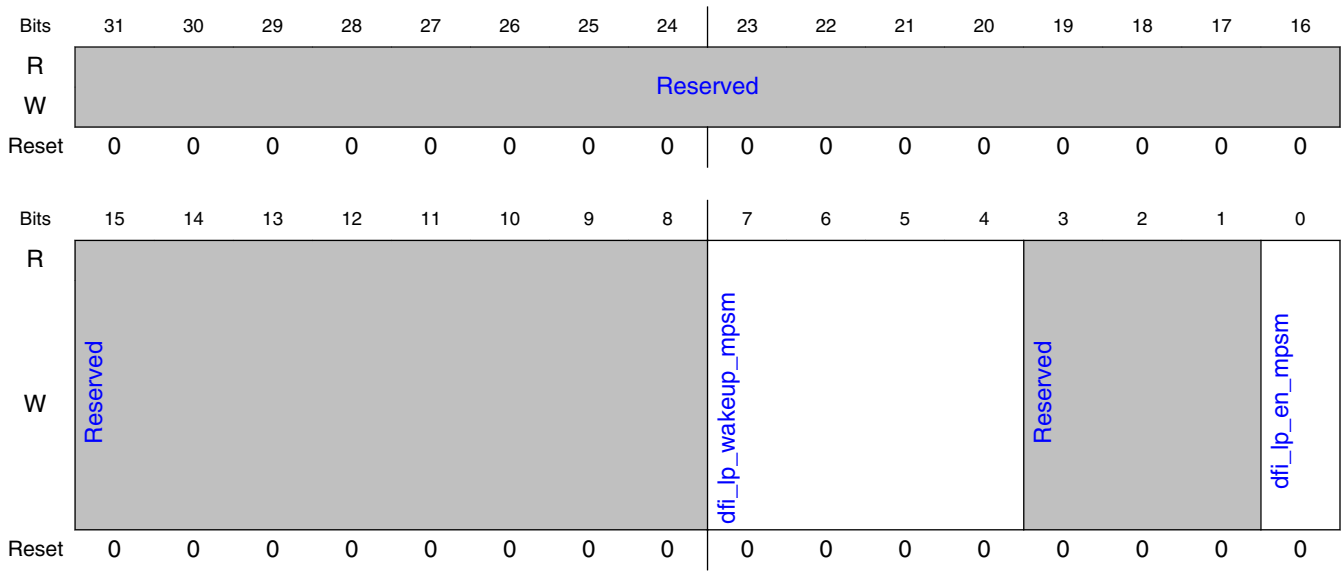
Field	Function
dfi_lp_en_sr	1b - Enabled
7-4 dfi_lp_wakeup_pd	Value in DFI clock cycles to drive on dfi_lp_wakeup signal when Power Down mode is entered. Determines the DFI's tlp_wakeup time: 0000b - 16 cycles 0001b - 32 cycles 0010b - 64 cycles 0011b - 128 cycles 0100b - 256 cycles 0101b - 512 cycles 0110b - 1024 cycles 0111b - 2048 cycles 1000b - 4096 cycles 1001b - 8192 cycles 1010b - 16384 cycles 1011b - 32768 cycles 1100b - 65536 cycles 1101b - 131072 cycles 1110b - 262144 cycles 1111b - Unlimited cycles
3-1 —	Reserved
0 dfi_lp_en_pd	Enables DFI Low Power interface handshaking during Power Down Entry/Exit. - 0 - Disabled - 1 - Enabled

9.2.3.1.52 DFI Low Power Configuration Register 1 (DFILPCFG1)

9.2.3.1.52.1 Offset

Register	Offset
DFILPCFG1	19Ch

9.2.3.1.52.2 Diagram



9.2.3.1.52.3 Fields

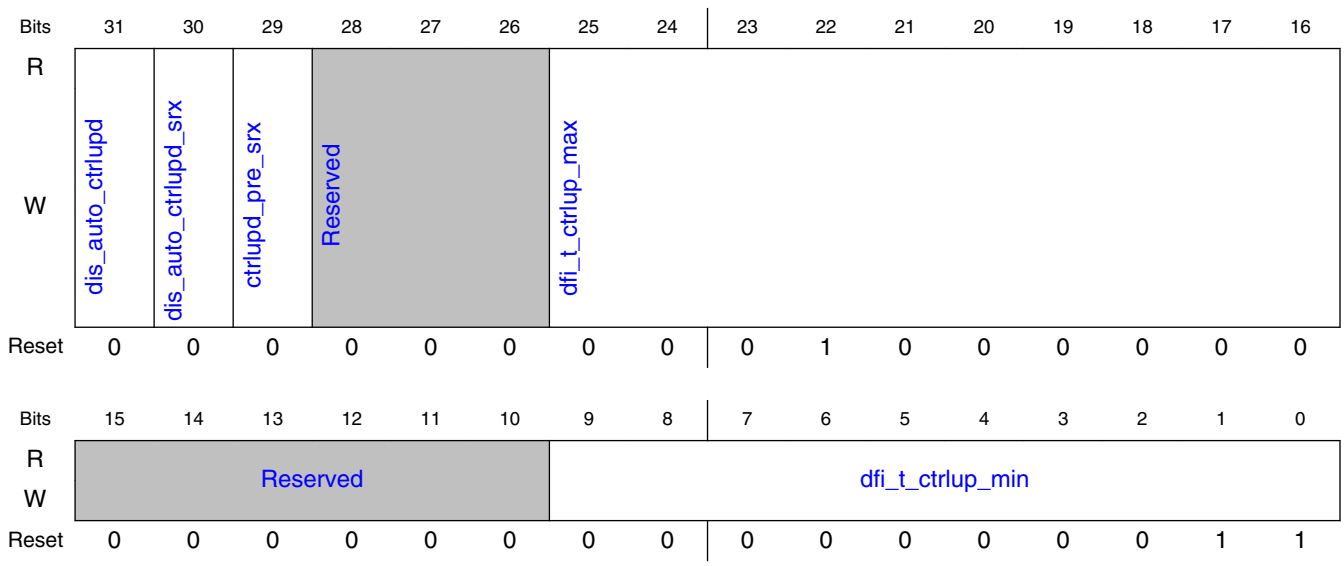
Field	Function
31-8 —	Reserved
7-4 dfi_lp_wakeup_mpsm	Value in DFI clock cycles to drive on dfi_lp_wakeup signal when Maximum Power Saving Mode is entered. Determines the DFI's tlp_wakeup time: 0000b - 16 cycles 0001b - 32 cycles 0010b - 64 cycles 0011b - 128 cycles 0100b - 256 cycles 0101b - 512 cycles 0110b - 1024 cycles 0111b - 2048 cycles 1000b - 4096 cycles 1001b - 8192 cycles 1010b - 16384 cycles 1011b - 32768 cycles 1100b - 65536 cycles 1101b - 131072 cycles 1110b - 262144 cycles 1111b - Unlimited cycles
3-1 —	Reserved
0 dfi_lp_en_mpsm	Enables DFI Low Power interface handshaking during Maximum Power Saving Mode Entry/Exit. - 0 - Disabled - 1 - Enabled This is only present for designs supporting DDR4 devices.

9.2.3.1.53 DFI Update Register 0 (DFIUPD0)

9.2.3.1.53.1 Offset

Register	Offset
DFIUPD0	1A0h

9.2.3.1.53.2 Diagram



9.2.3.1.53.3 Fields

Field	Function
31 dis_auto_ctrlupd	automatic dfi_ctrlupd_req generation by the DDRC 0b - DDRC issues dfi_ctrlupd_req periodically. 1b - disable the automatic dfi_ctrlupd_req generation by the DDRC. The core must issue the dfi_ctrlupd_req signal using register reg_ddrc_ctrlupd.
30 dis_auto_ctrlupd_srx	Auto ctrlupd request generation 0b - DDRC issues a dfi_ctrlupd_req before or after exiting self-refresh, depending on DFIUPD0.ctrlupd_pre_srx. 1b - disable the automatic dfi_ctrlupd_req generation by the DDRC at self-refresh exit.
29 ctrlupd_pre_srx	Selects dfi_ctrlupd_req requirements at SRX: - 0 : send ctrlupd after SRX - 1 : send ctrlupd before SRX If DFIUPD0.dis_auto_ctrlupd_srx=1, this register has no impact, because no dfi_ctrlupd_req will be issued when SRX. 0b - send ctrlupd after SRX 1b - send ctrlupd before SRX
28-26 —	Reserved

Table continues on the next page...

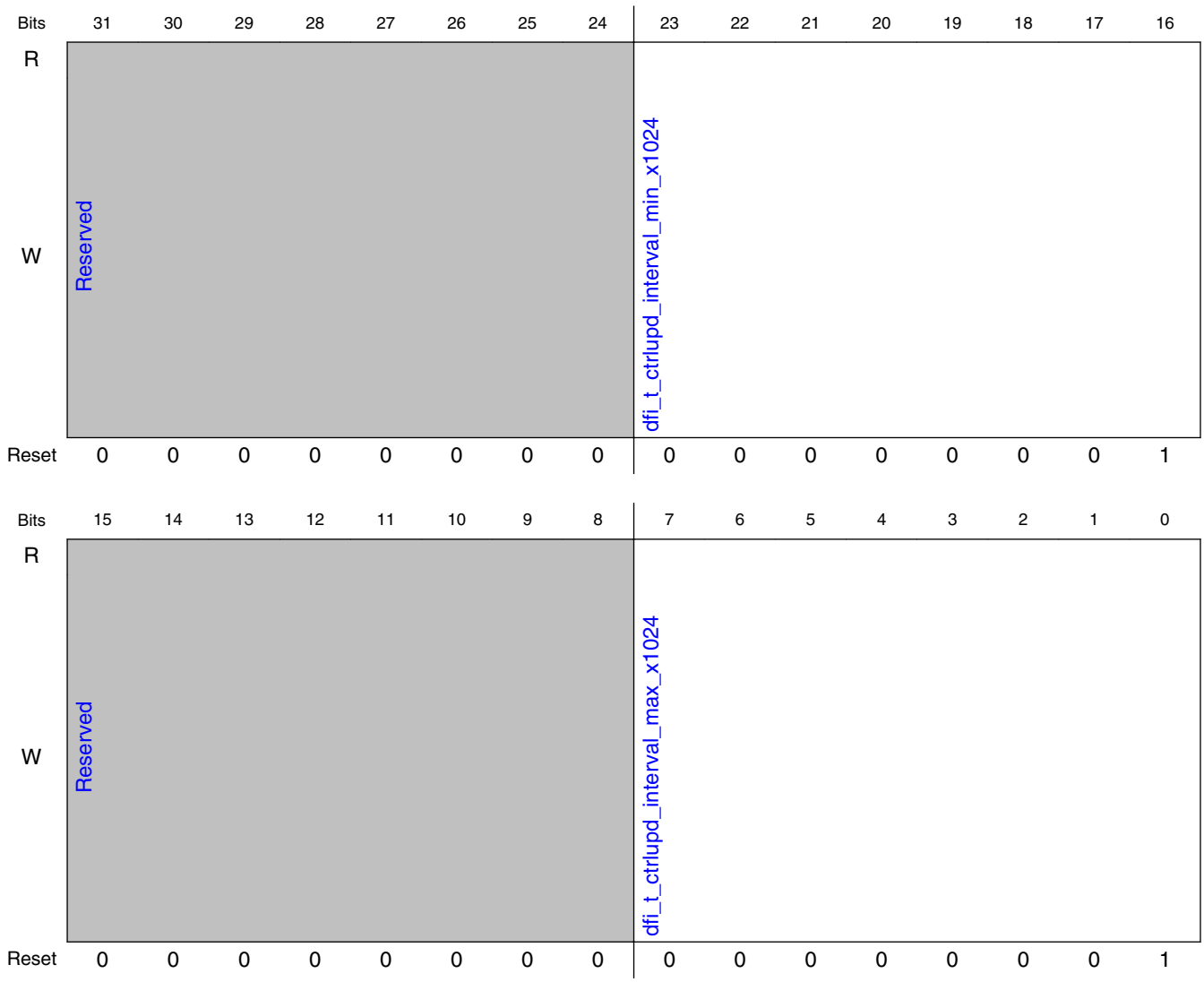
Field	Function
25-16 dfi_t_ctrlup_max	Specifies the maximum number of DFI clock cycles that the dfi_ctrlupd_req signal can assert. Lowest value to assign to this variable is 0x40.
15-10 —	Reserved
9-0 dfi_t_ctrlup_min	Specifies the minimum number of DFI clock cycles that the dfi_ctrlupd_req signal must be asserted. The DDRC expects the PHY to respond within this time. If the PHY does not respond, the DDRC will de-assert dfi_ctrlupd_req after dfi_t_ctrlup_min + 2 cycles. Lowest value to assign to this variable is 0x3.

9.2.3.1.54 DFI Update Register 1 (DFIUPD1)

9.2.3.1.54.1 Offset

Register	Offset
DFIUPD1	1A4h

9.2.3.1.54.2 Diagram



9.2.3.1.54.3 Fields

Field	Function
31-24 —	Reserved
23-16 dfi_t_ctrlupd_interval_min_x1024	This is the minimum amount of time between DDRC initiated DFI update requests (which is executed whenever the DDRC is idle). Set this number higher to reduce the frequency of update requests, which can have a small impact on the latency of the first read request when the DDRC is idle. Minimum allowed value for this field is 1. Unit: 1024 DFI clock cycles
15-8 —	Reserved

Table continues on the next page...

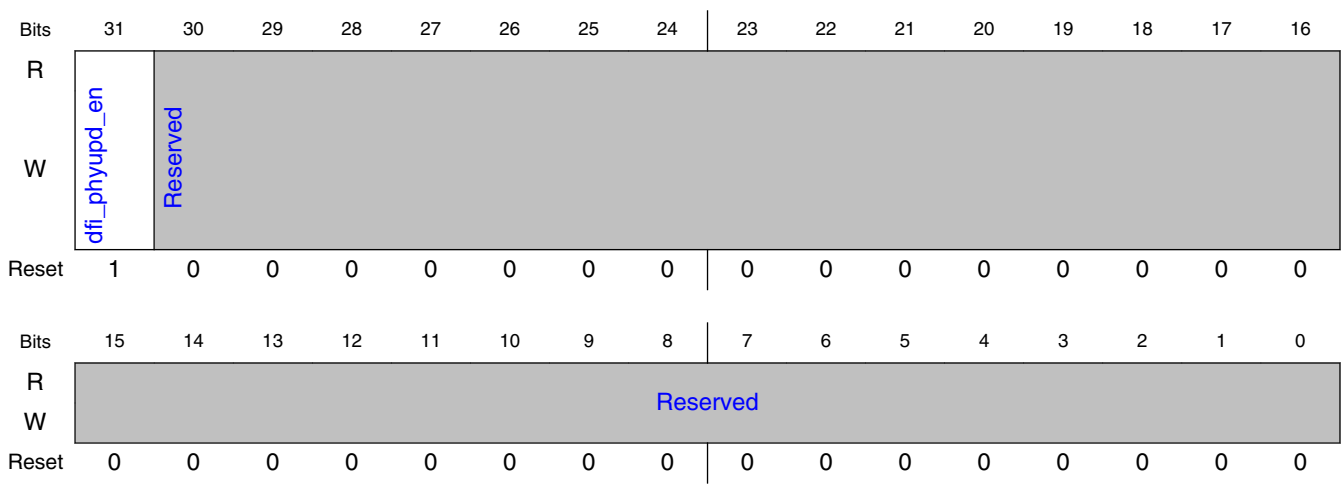
Field	Function
7-0 dfi_t_ctrlupd_interval_max_x1024	This is the maximum amount of time between DDRC initiated DFI update requests. This timer resets with each update request; when the timer expires dfi_ctrlupd_req is sent and traffic is blocked until the dfi_ctrlupd_ackx is received. PHY can use this idle time to recalibrate the delay lines to the DLLs. The DFI controller update is also used to reset PHY FIFO pointers in case of data capture errors. Updates are required to maintain calibration over PVT, but frequent updates may impact performance. Minimum allowed value for this field is 1. Note: Value programmed for DFIUPD1.dfi_t_ctrlupd_interval_max_x1024 must be greater than DFIUPD1.dfi_t_ctrlupd_interval_min_x1024. Unit: 1024 DFI clock cycles

9.2.3.1.55 DFI Update Register 2 (DFIUPD2)

9.2.3.1.55.1 Offset

Register	Offset
DFIUPD2	1A8h

9.2.3.1.55.2 Diagram



9.2.3.1.55.3 Fields

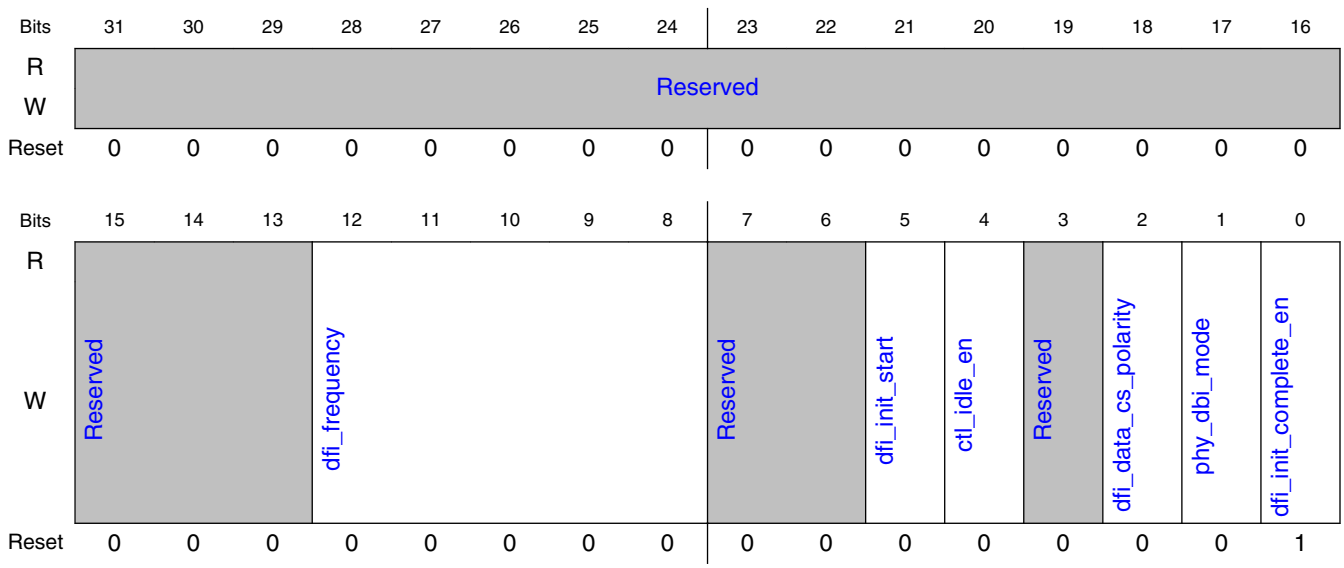
Field	Function
31 dfi_phyupd_en	Enables the support for acknowledging PHY-initiated updates: 0b - Disabled 1b - Enabled
30-0 —	Reserved

9.2.3.1.56 DFI Miscellaneous Control Register (DFIMISC)

9.2.3.1.56.1 Offset

Register	Offset
DFIMISC	1B0h

9.2.3.1.56.2 Diagram



9.2.3.1.56.3 Fields

Field	Function
31-13 —	Reserved
12-8 dfi_frequency	Indicates the operating frequency of the system. The number of supported frequencies and the mapping of signal values to clock frequencies are defined by the PHY.
7-6 —	Reserved
5 dfi_init_start	PHY init start request signal.When asserted it triggers the PHY init start request
4 ctl_idle_en	Enables support of ctl_idle signal, which is non-DFI related pin specific to certain PHYs. See signal description of ctl_idle signal for further details of ctl_idle functionality.
3	Reserved

Table continues on the next page...

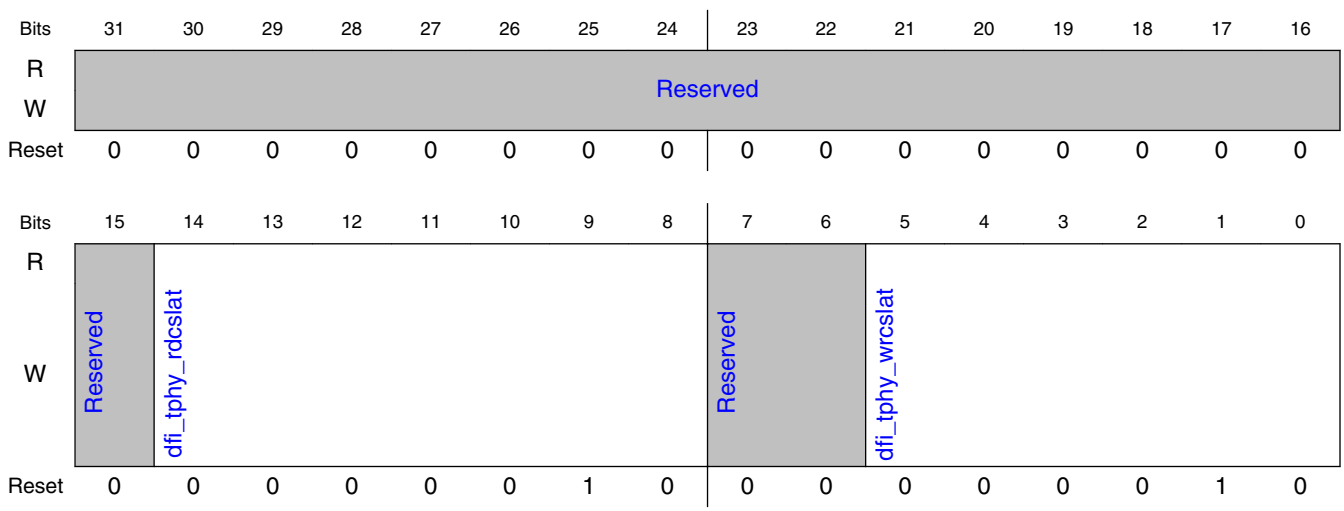
Field	Function
—	
2 dfi_data_cs_polarity	Defines polarity of dfi_wrdata_cs and dfi_rddata_cs signals. 0b - Signals are active low 1b - Signals are active high
1 phy_dbi_mode	DBI implemented in DDRC or PHY. Present only in designs configured to support DDR4 and LPDDR4. 0b - DDRC implements DBI functionality. 1b - PHY implements DBI functionality.
0 dfi_init_complete_en	PHY initialization complete enable signal. When asserted the dfi_init_complete signal can be used to trigger SDRAM initialisation

9.2.3.1.57 DFI Timing Register 2 (DFITMG2)

9.2.3.1.57.1 Offset

Register	Offset
DFITMG2	1B4h

9.2.3.1.57.2 Diagram



9.2.3.1.57.3 Fields

Field	Function
31-15	Reserved

Table continues on the next page...

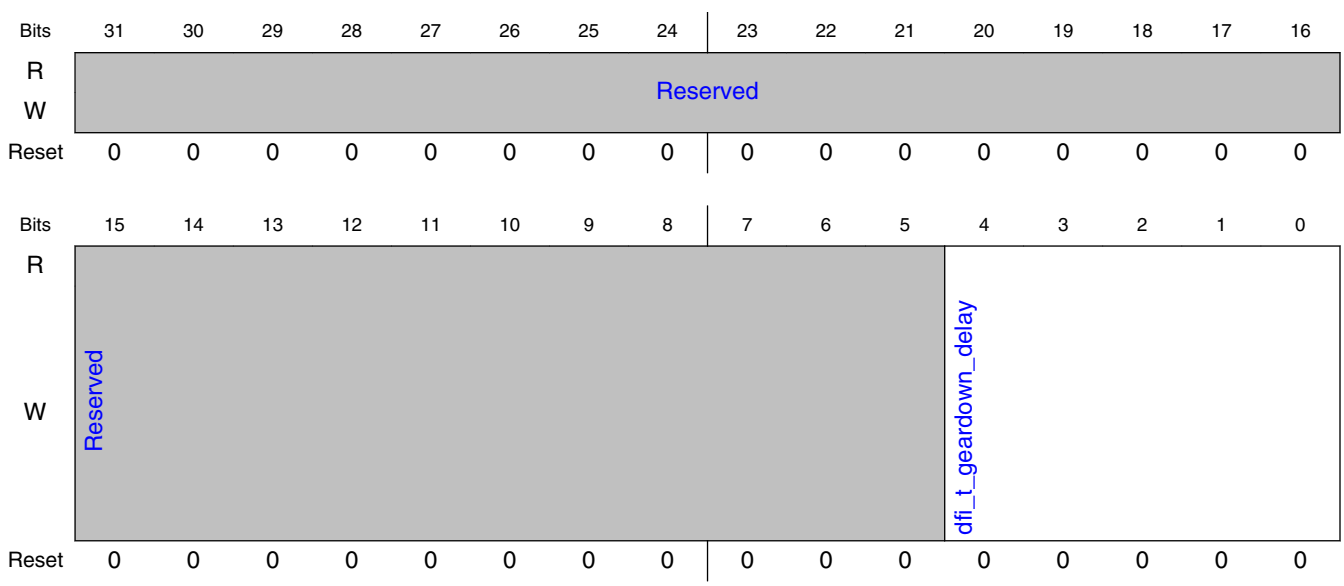
Field	Function
—	
14-8 dfi_tphy_rdcslat	Number of DFI PHY clock cycles between when a read command is sent on the DFI control interface and when the associated dfi_rddata_cs signal is asserted. This corresponds to the DFI timing parameter tphy_rdcslat. Refer to PHY specification for correct value.
7-6 —	Reserved
5-0 dfi_tphy_wrcslat	Number of DFI PHY clock cycles between when a write command is sent on the DFI control interface and when the associated dfi_wrdata_cs signal is asserted. This corresponds to the DFI timing parameter tphy_wrcslat. Refer to PHY specification for correct value.

9.2.3.1.58 DFI Timing Register 3 (DFITMG3)

9.2.3.1.58.1 Offset

Register	Offset
DFITMG3	1B8h

9.2.3.1.58.2 Diagram



9.2.3.1.58.3 Fields

Field	Function
31-5	Reserved

Table continues on the next page...

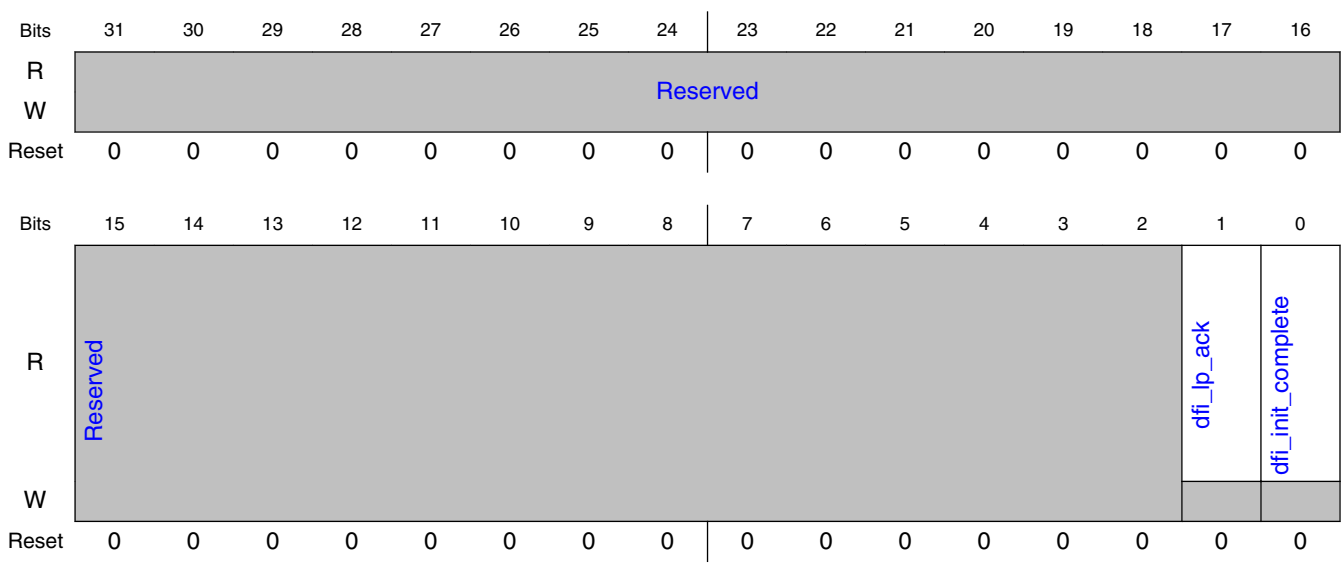
Field	Function
—	
4-0 dfi_t_geardown_delay	The delay from dfi_geardown_en assertion to the time of the PHY being ready to receive commands. Refer to PHY specification for correct value. When the controller is operating in 1:2 frequency ratio mode, program this to (tgeardown_delay/2) and round it up to the next integer value. Unit: Clocks

9.2.3.1.59 DFI Status Register (DFISTAT)

9.2.3.1.59.1 Offset

Register	Offset
DFISTAT	1BCh

9.2.3.1.59.2 Diagram



9.2.3.1.59.3 Fields

Field	Function
31-2 —	Reserved
1 dfi_lp_ack	Stores the value of the dfi_lp_ack input to the controller.

Table continues on the next page...

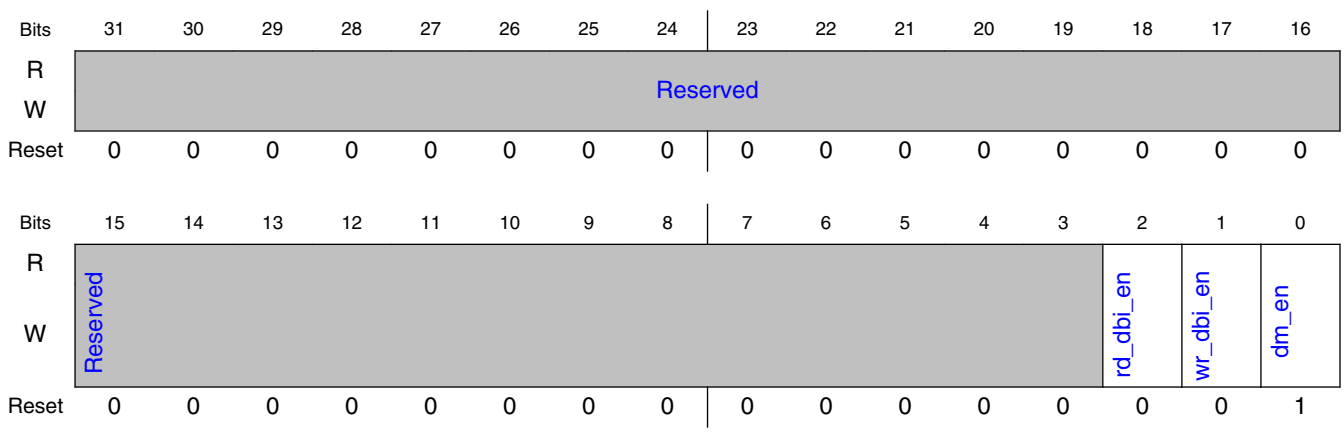
Field	Function
0 dfi_init_complete	The status flag register which announces when the DFI initialization has been completed. The DFI INIT triggered by dfi_init_start signal and then the dfi_init_complete flag is polled to know when the initialization is done.

9.2.3.1.60 DM/DBI Control Register (DBICTL)

9.2.3.1.60.1 Offset

Register	Offset
DBICTL	1C0h

9.2.3.1.60.2 Diagram



9.2.3.1.60.3 Fields

Field	Function
31-3 —	Reserved
2 rd_dbi_en	Read DBI enable signal in DDRC. - 0 - Read DBI is disabled. - 1 - Read DBI is enabled. This signal must be set the same value as DRAM's mode register. - DDR4: MR5 bit A12. When x4 devices are used, this signal must be set to 0. - LPDDR4: MR3[6]
1 wr_dbi_en	This signal must be set the same value as DRAM's mode register. - DDR4: MR5 bit A11. When x4 devices are used, this signal must be set to 0. - LPDDR4: MR3[7] 0b - Write DBI is disabled 1b - Write DBI is enabled.

Table continues on the next page...

Field	Function
0 dm_en	DM enable signal in DDRC. This signal must be set the same logical value as DRAM's mode register. - DDR4: Set this to same value as MR5 bit A10. When x4 devices are used, this signal must be set to 0. - LPDDR4: Set this to inverted value of MR13[5] which is opposite polarity from this signal 0b - DM is disabled 1b - DM is enabled

9.2.3.1.61 Address Map Register 0 (ADDRMAP0)

9.2.3.1.61.1 Offset

Register	Offset
ADDRMAP0	200h

9.2.3.1.61.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											addrmap_cs_bit0				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.61.3 Fields

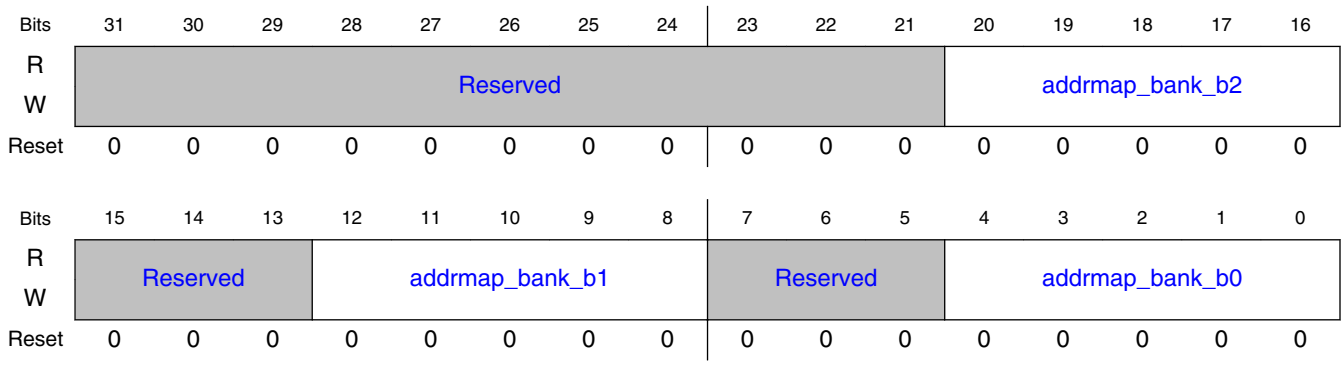
Field	Function
31-5 —	Reserved
4-0 addrmap_cs_bit0	Selects the HIF address bit used as rank address bit 0. Valid Range: 0 to 28, and 31 Internal Base: 6 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 31, rank address bit 0 is set to 0.

9.2.3.1.62 Address Map Register 1 (ADDRMAP1)

9.2.3.1.62.1 Offset

Register	Offset
ADDRMAP1	204h

9.2.3.1.62.2 Diagram



9.2.3.1.62.3 Fields

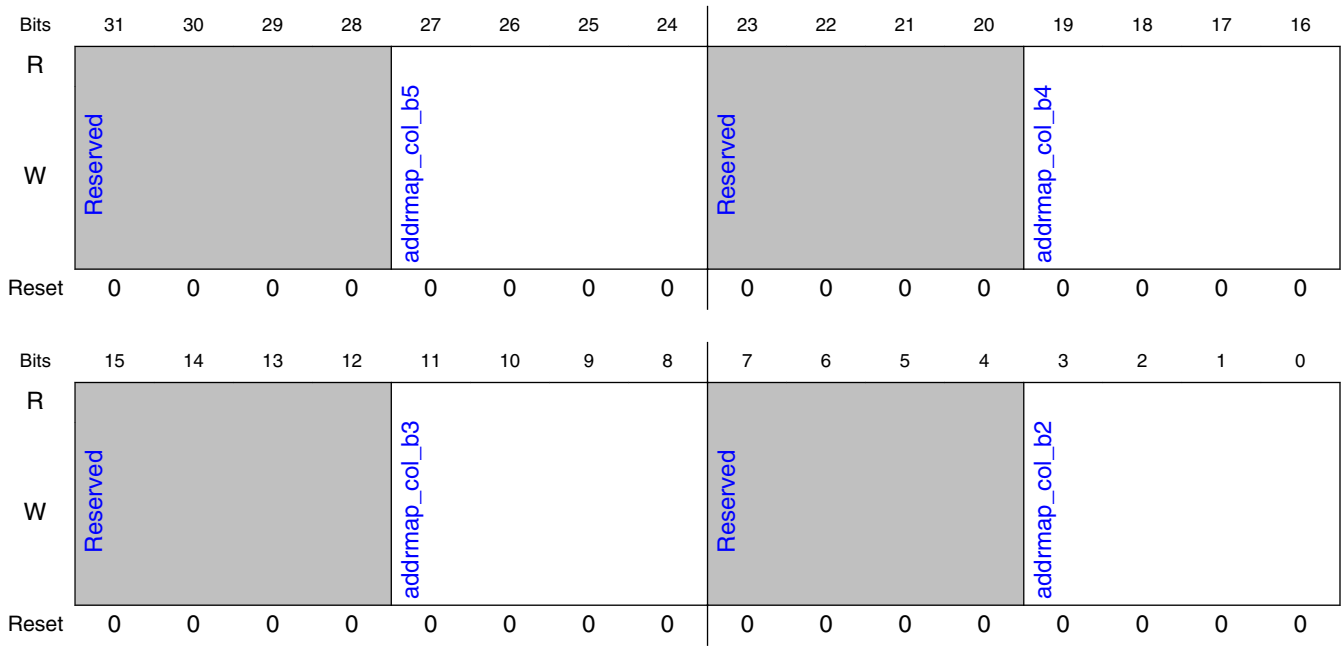
Field	Function
31-21 —	Reserved
20-16 addrmap_bank_b2	Selects the HIF address bit used as bank address bit 2. Valid Range: 0 to 30 and 31 Internal Base: 4 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 31, bank address bit 2 is set to 0.
15-13 —	Reserved
12-8 addrmap_bank_b1	Selects the HIF address bits used as bank address bit 1. Valid Range: 0 to 31 Internal Base: 3 The selected HIF address bit for each of the bank address bits is determined by adding the internal base to the value of this field.
7-5 —	Reserved
4-0 addrmap_bank_b0	Selects the HIF address bits used as bank address bit 0. Valid Range: 0 to 31 Internal Base: 2 The selected HIF address bit for each of the bank address bits is determined by adding the internal base to the value of this field.

9.2.3.1.63 Address Map Register 2 (ADDRMAP2)

9.2.3.1.63.1 Offset

Register	Offset
ADDRMAP2	208h

9.2.3.1.63.2 Diagram



9.2.3.1.63.3 Fields

Field	Function
31-28 —	Reserved
27-24 addrmap_col_b5	- Full bus width mode: Selects the HIF address bit used as column address bit 5. - Half bus width mode: Selects the HIF address bit used as column address bit 6. - Quarter bus width mode: Selects the HIF address bit used as column address bit 7 . Valid Range: 0 to 7, and 15 Internal Base: 5 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, this column address bit is set to 0.
23-20 —	Reserved
19-16 addrmap_col_b4	- Full bus width mode: Selects the HIF address bit used as column address bit 4. - Half bus width mode: Selects the HIF address bit used as column address bit 5. - Quarter bus width mode: Selects the HIF address bit used as column address bit 6. Valid Range: 0 to 7, and 15 Internal Base: 4 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, this column address bit is set to 0.
15-12	Reserved

Table continues on the next page...

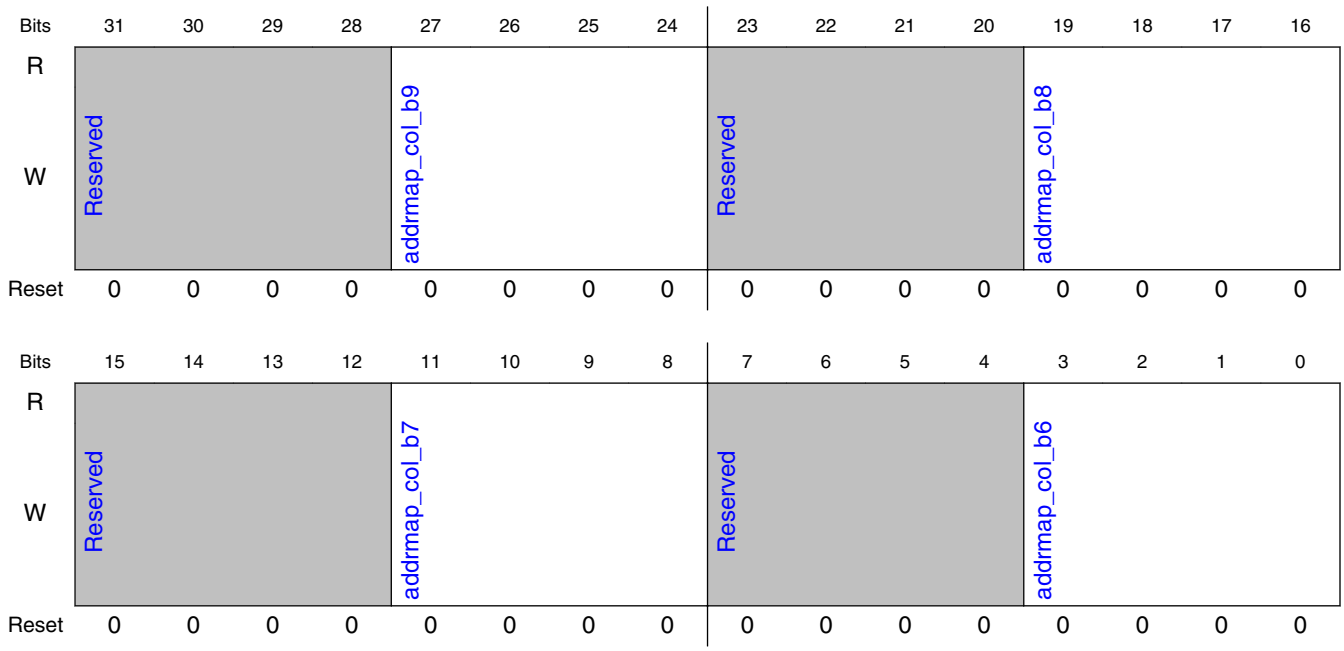
Field	Function
—	
11-8 addrmap_col_b3	- Full bus width mode: Selects the HIF address bit used as column address bit 3. - Half bus width mode: Selects the HIF address bit used as column address bit 4. - Quarter bus width mode: Selects the HIF address bit used as column address bit 5. Valid Range: 0 to 7 Internal Base: 3 The selected HIF address bit is determined by adding the internal base to the value of this field. Note, if DDRC_INCL_ARB=1, MEMC_BURST_LENGTH=16, Full bus width (MSTR.data_bus_width=00) and BL16 (MSTR.burst_rdwr=1000), it is recommended to program this to 0.
7-4 —	Reserved
3-0 addrmap_col_b2	- Full bus width mode: Selects the HIF address bit used as column address bit 2. - Half bus width mode: Selects the HIF address bit used as column address bit 3. - Quarter bus width mode: Selects the HIF address bit used as column address bit 4. Valid Range: 0 to 7 Internal Base: 2 The selected HIF address bit is determined by adding the internal base to the value of this field. Note, if DDRC_INCL_ARB=1 and MEMC_BURST_LENGTH=8, it is required to program this to 0 unless: - in Half or Quarter bus width (MSTR.data_bus_width!=00) and - PCCFG.bl_exp_mode==1 and either - In DDR4 and ADDRMAP8.addrmap_bg_b0==0 or - In LPDDR4 and ADDRMAP1.addrmap_bank_b0==0 If DDRC_INCL_ARB=1 and MEMC_BURST_LENGTH=16, it is required to program this to 0 unless: - in Half or Quarter bus width (MSTR.data_bus_width!=00) and - PCCFG.bl_exp_mode==1 and - In DDR4 and ADDRMAP8.addrmap_bg_b0==0 Otherwise, if MEMC_BURST_LENGTH=8 and Full Bus Width (MSTR.data_bus_width==00), it is recommended to program this to 0 so that HIF[2] maps to column address bit 2. If MEMC_BURST_LENGTH=16 and Full Bus Width (MSTR.data_bus_width==00), it is recommended to program this to 0 so that HIF[2] maps to column address bit 2. If MEMC_BURST_LENGTH=16 and Half Bus Width (MSTR.data_bus_width==01), it is recommended to program this to 0 so that HIF[2] maps to column address bit 3.

9.2.3.1.64 Address Map Register 3 (ADDRMAP3)

9.2.3.1.64.1 Offset

Register	Offset
ADDRMAP3	20Ch

9.2.3.1.64.2 Diagram



9.2.3.1.64.3 Fields

Field	Function
31-28 —	Reserved
27-24 addrmap_col_b9	- Full bus width mode: Selects the HIF address bit used as column address bit 9. - Half bus width mode: Selects the HIF address bit used as column address bit 11 (10 in LPDDR2/LPDDR3 mode). - Quarter bus width mode: Selects the HIF address bit used as column address bit 13 (11 in LPDDR2/LPDDR3 mode). Valid Range: 0 to 7, and 15 Internal Base: 9 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, this column address bit is set to 0. Note: Per JEDEC DDR2/3/mDDR specification, column address bit 10 is reserved for indicating auto-precharge, and hence no source address bit can be mapped to column address bit 10. In LPDDR2/LPDDR3, there is a dedicated bit for auto-precharge in the CA bus and hence column bit 10 is used.
23-20 —	Reserved
19-16 addrmap_col_b8	- Full bus width mode: Selects the HIF address bit used as column address bit 8. - Half bus width mode: Selects the HIF address bit used as column address bit 9. - Quarter bus width mode: Selects the HIF address bit used as column address bit 11 (10 in LPDDR2/LPDDR3 mode). Valid Range: 0 to 7, and 15 Internal Base: 8 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, this column address bit is set to 0. Note: Per JEDEC DDR2/3/mDDR specification, column address bit 10 is reserved for indicating auto-precharge, and hence no source address bit can be mapped to column address bit 10. In LPDDR2/LPDDR3, there is a dedicated bit for auto-precharge in the CA bus and hence column bit 10 is used.
15-12 —	Reserved

Table continues on the next page...

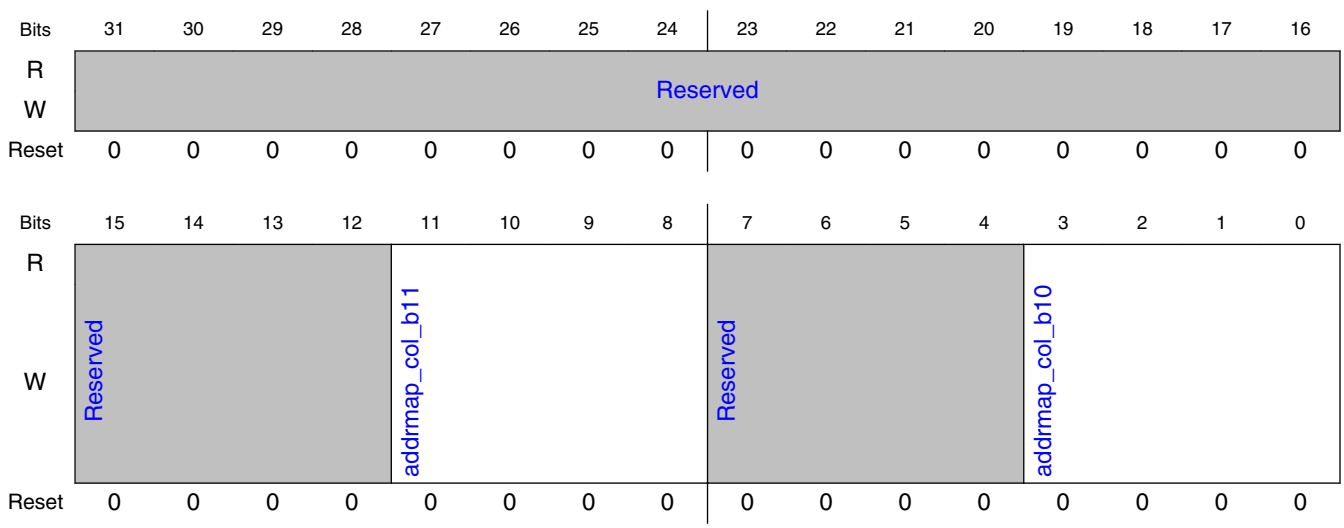
Field	Function
11-8 addrmap_col_b7	- Full bus width mode: Selects the HIF address bit used as column address bit 7. - Half bus width mode: Selects the HIF address bit used as column address bit 8. - Quarter bus width mode: Selects the HIF address bit used as column address bit 9. Valid Range: 0 to 7, and 15 Internal Base: 7 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, this column address bit is set to 0.
7-4 —	Reserved
3-0 addrmap_col_b6	- Full bus width mode: Selects the HIF address bit used as column address bit 6. - Half bus width mode: Selects the HIF address bit used as column address bit 7. - Quarter bus width mode: Selects the HIF address bit used as column address bit 8. Valid Range: 0 to 7, and 15 Internal Base: 6 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, this column address bit is set to 0.

9.2.3.1.65 Address Map Register 4 (ADDRMAP4)

9.2.3.1.65.1 Offset

Register	Offset
ADDRMAP4	210h

9.2.3.1.65.2 Diagram



9.2.3.1.65.3 Fields

Field	Function
31-12	Reserved

Table continues on the next page...

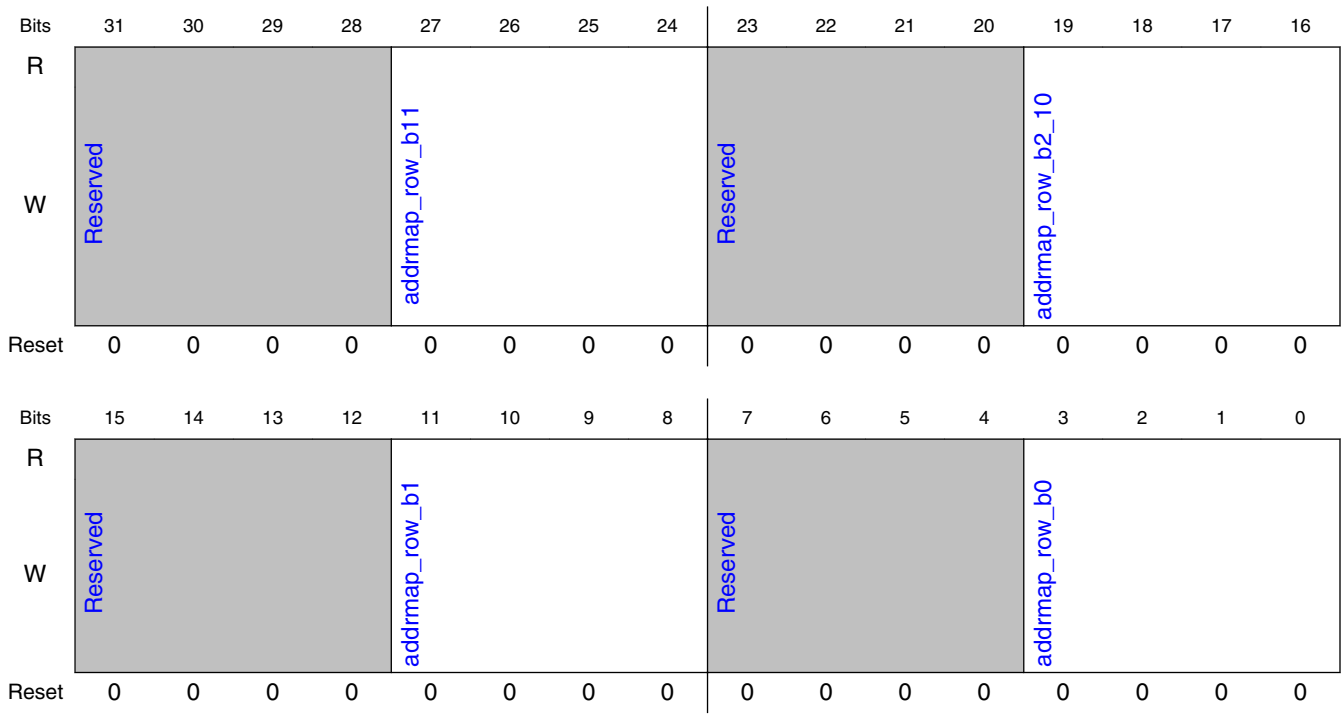
Field	Function
—	
11-8 addrmap_col_b11	- Full bus width mode: Selects the HIF address bit used as column address bit 13 (11 in LPDDR2/LPDDR3 mode). - Half bus width mode: Unused. To make it unused, this should be tied to 4'hF. - Quarter bus width mode: Unused. To make it unused, this must be tied to 4'hF. Valid Range: 0 to 7, and 15 Internal Base: 11 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, this column address bit is set to 0. Note: Per JEDEC DDR2/3/mDDR specification, column address bit 10 is reserved for indicating auto-precharge, and hence no source address bit can be mapped to column address bit 10. In LPDDR2/LPDDR3, there is a dedicated bit for auto-precharge in the CA bus and hence column bit 10 is used.
7-4 —	Reserved
3-0 addrmap_col_b10	- Full bus width mode: Selects the HIF address bit used as column address bit 11 (10 in LPDDR2/LPDDR3 mode). - Half bus width mode: Selects the HIF address bit used as column address bit 13 (11 in LPDDR2/LPDDR3 mode). - Quarter bus width mode: UNUSED. To make it unused, this must be tied to 4'hF. Valid Range: 0 to 7, and 15 Internal Base: 10 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, this column address bit is set to 0. Note: Per JEDEC DDR2/3/mDDR specification, column address bit 10 is reserved for indicating auto-precharge, and hence no source address bit can be mapped to column address bit 10. In LPDDR2/LPDDR3, there is a dedicated bit for auto-precharge in the CA bus and hence column bit 10 is used.

9.2.3.1.66 Address Map Register 5 (ADDRMAP5)

9.2.3.1.66.1 Offset

Register	Offset
ADDRMAP5	214h

9.2.3.1.66.2 Diagram



9.2.3.1.66.3 Fields

Field	Function
31-28 —	Reserved
27-24 addrmap_row_b11	Selects the HIF address bit used as row address bit 11. Valid Range: 0 to 11, and 15 Internal Base: 17 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, row address bit 11 is set to 0.
23-20 —	Reserved
19-16 addrmap_row_b2_10	Selects the HIF address bits used as row address bits 2 to 10. Valid Range: 0 to 11, and 15 Internal Base: 8 (for row address bit 2), 9 (for row address bit 3), 10 (for row address bit 4) etc increasing to 16 (for row address bit 10) The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. When value 15 is used the values of row address bits 2 to 10 are defined by registers ADDRMAP9, ADDRMAP10, ADDRMAP11.
15-12 —	Reserved
11-8 addrmap_row_b1	Selects the HIF address bits used as row address bit 1. Valid Range: 0 to 11 Internal Base: 7 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field.
7-4 —	Reserved

Table continues on the next page...

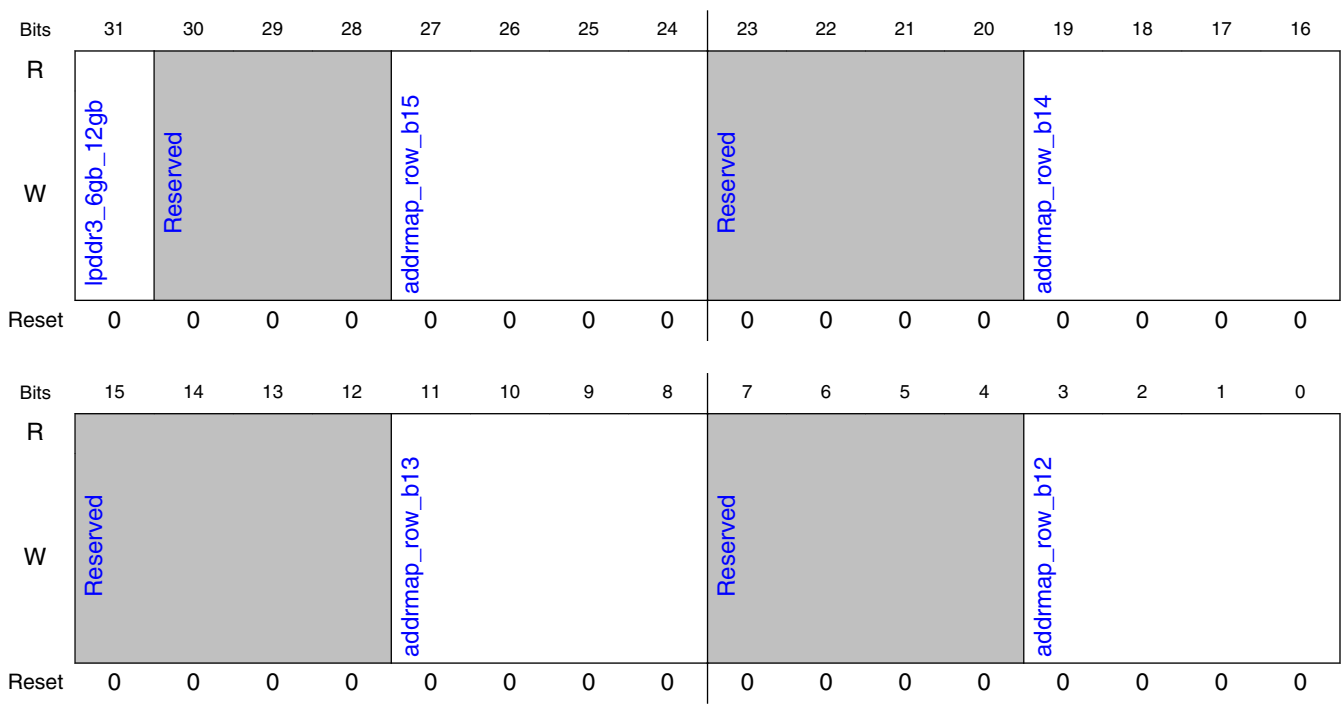
Field	Function
3-0 addrmap_row_b0	Selects the HIF address bits used as row address bit 0. Valid Range: 0 to 11 Internal Base: 6 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field.

9.2.3.1.67 Address Map Register 6 (ADDRMAP6)

9.2.3.1.67.1 Offset

Register	Offset
ADDRMAP6	218h

9.2.3.1.67.2 Diagram



9.2.3.1.67.3 Fields

Field	Function
31 lpddr3_6gb_12gb	Set this to 1 if there is an LPDDR3 SDRAM 6Gb or 12Gb device in use. - 1 - LPDDR3 SDRAM 6Gb/12Gb device in use. Every address having row[14:13]==2'b11 is considered as invalid - 0 - non-LPDDR3 6Gb/12Gb device in use. All addresses are valid Present only in designs configured to support LPDDR3.

Table continues on the next page...

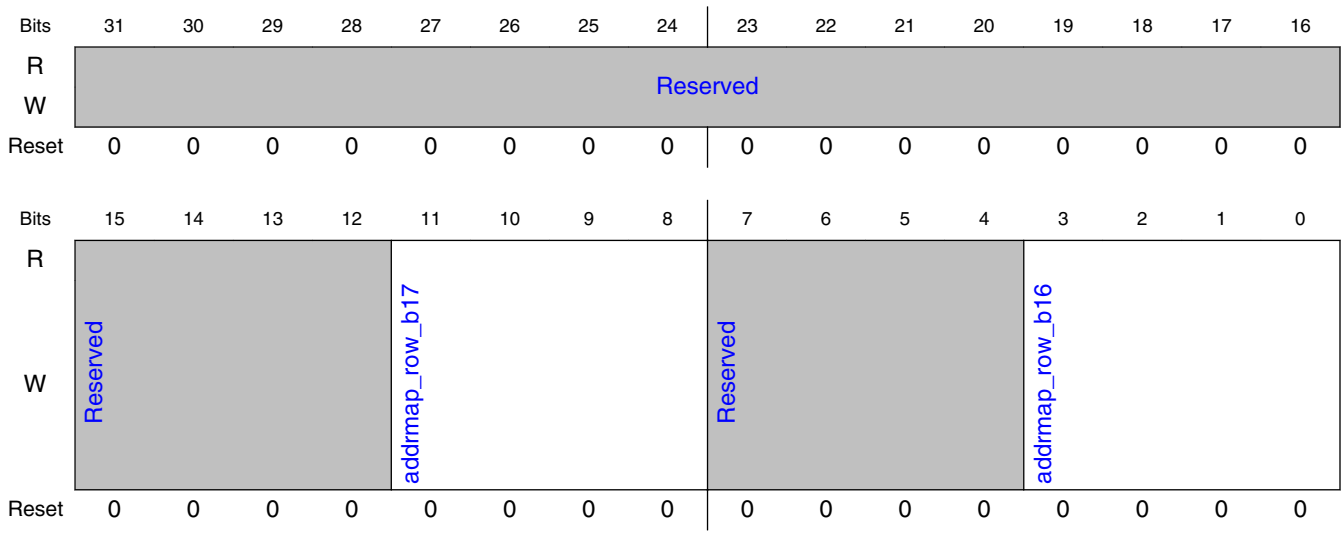
Field	Function
30-28 —	Reserved
27-24 addrmap_row_b15	Selects the HIF address bit used as row address bit 15. Valid Range: 0 to 11, and 15 Internal Base: 21 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, row address bit 15 is set to 0.
23-20 —	Reserved
19-16 addrmap_row_b14	Selects the HIF address bit used as row address bit 14. Valid Range: 0 to 11, and 15 Internal Base: 20 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, row address bit 14 is set to 0.
15-12 —	Reserved
11-8 addrmap_row_b13	Selects the HIF address bit used as row address bit 13. Valid Range: 0 to 11, and 15 Internal Base: 19 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, row address bit 13 is set to 0.
7-4 —	Reserved
3-0 addrmap_row_b12	Selects the HIF address bit used as row address bit 12. Valid Range: 0 to 11, and 15 Internal Base: 18 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, row address bit 12 is set to 0.

9.2.3.1.68 Address Map Register 7 (ADDRMAP7)

9.2.3.1.68.1 Offset

Register	Offset
ADDRMAP7	21Ch

9.2.3.1.68.2 Diagram



9.2.3.1.68.3 Fields

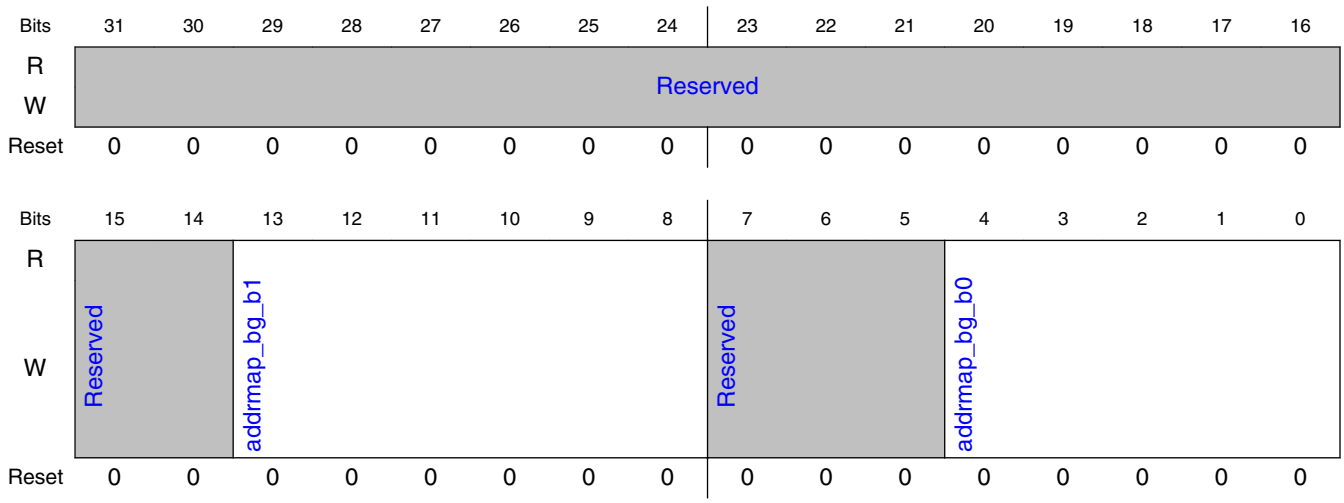
Field	Function
31-12 —	Reserved
11-8 addrmap_row_b17	Selects the HIF address bit used as row address bit 17. Valid Range: 0 to 11, and 15 Internal Base: 23 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, row address bit 17 is set to 0.
7-4 —	Reserved
3-0 addrmap_row_b16	Selects the HIF address bit used as row address bit 16. Valid Range: 0 to 11, and 15 Internal Base: 22 The selected HIF address bit is determined by adding the internal base to the value of this field. If set to 15, row address bit 16 is set to 0.

9.2.3.1.69 Address Map Register 8 (ADDRMAP8)

9.2.3.1.69.1 Offset

Register	Offset
ADDRMAP8	220h

9.2.3.1.69.2 Diagram



9.2.3.1.69.3 Fields

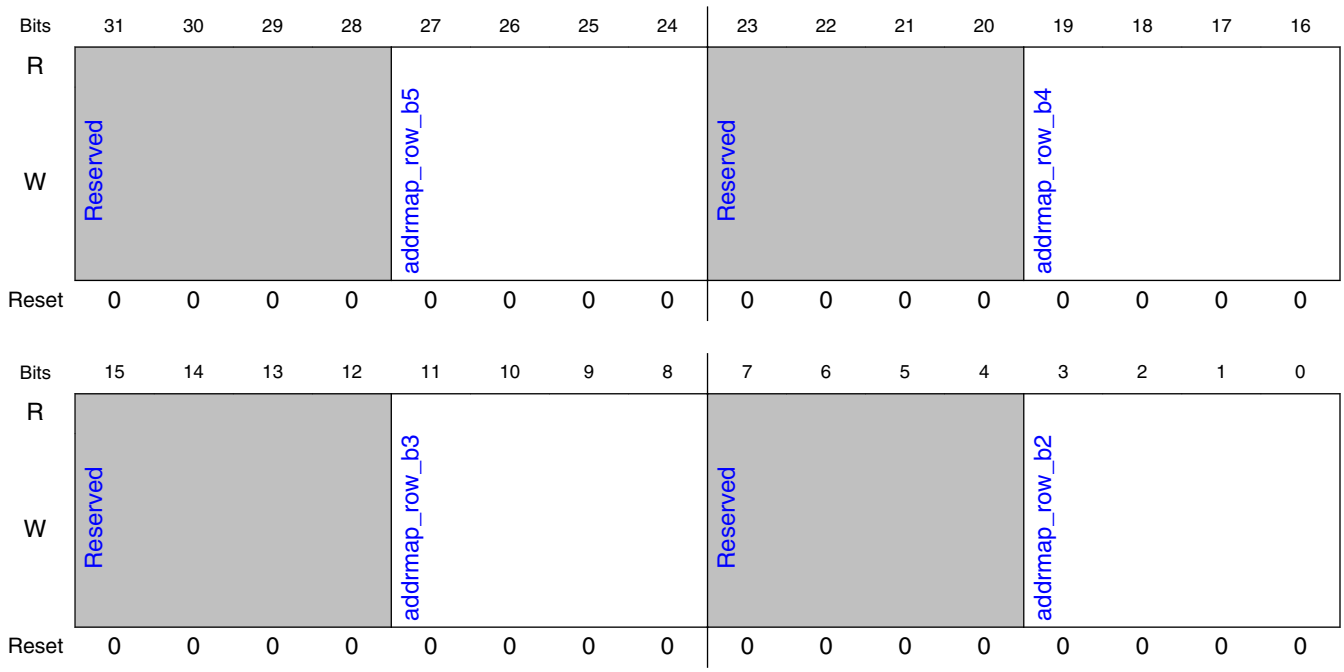
Field	Function
31-14 —	Reserved
13-8 addrmap_bg_b1	Selects the HIF address bits used as bank group address bit 1. Valid Range: 0 to 31, and 63 Internal Base: 3 The selected HIF address bit for each of the bank group address bits is determined by adding the internal base to the value of this field. If set to 63, bank group address bit 1 is set to 0.
7-5 —	Reserved
4-0 addrmap_bg_b0	Selects the HIF address bits used as bank group address bit 0. Valid Range: 0 to 31 Internal Base: 2 The selected HIF address bit for each of the bank group address bits is determined by adding the internal base to the value of this field.

9.2.3.1.70 Address Map Register 9 (ADDRMAP9)

9.2.3.1.70.1 Offset

Register	Offset
ADDRMAP9	224h

9.2.3.1.70.2 Diagram



9.2.3.1.70.3 Fields

Field	Function
31-28 —	Reserved
27-24 addrmap_row_b5	Selects the HIF address bits used as row address bit 5. Valid Range: 0 to 11 Internal Base: 11 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15.
23-20 —	Reserved
19-16 addrmap_row_b4	Selects the HIF address bits used as row address bit 4. Valid Range: 0 to 11 Internal Base: 10 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15.
15-12 —	Reserved
11-8 addrmap_row_b3	Selects the HIF address bits used as row address bit 3. Valid Range: 0 to 11 Internal Base: 9 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15.
7-4 —	Reserved

Table continues on the next page...

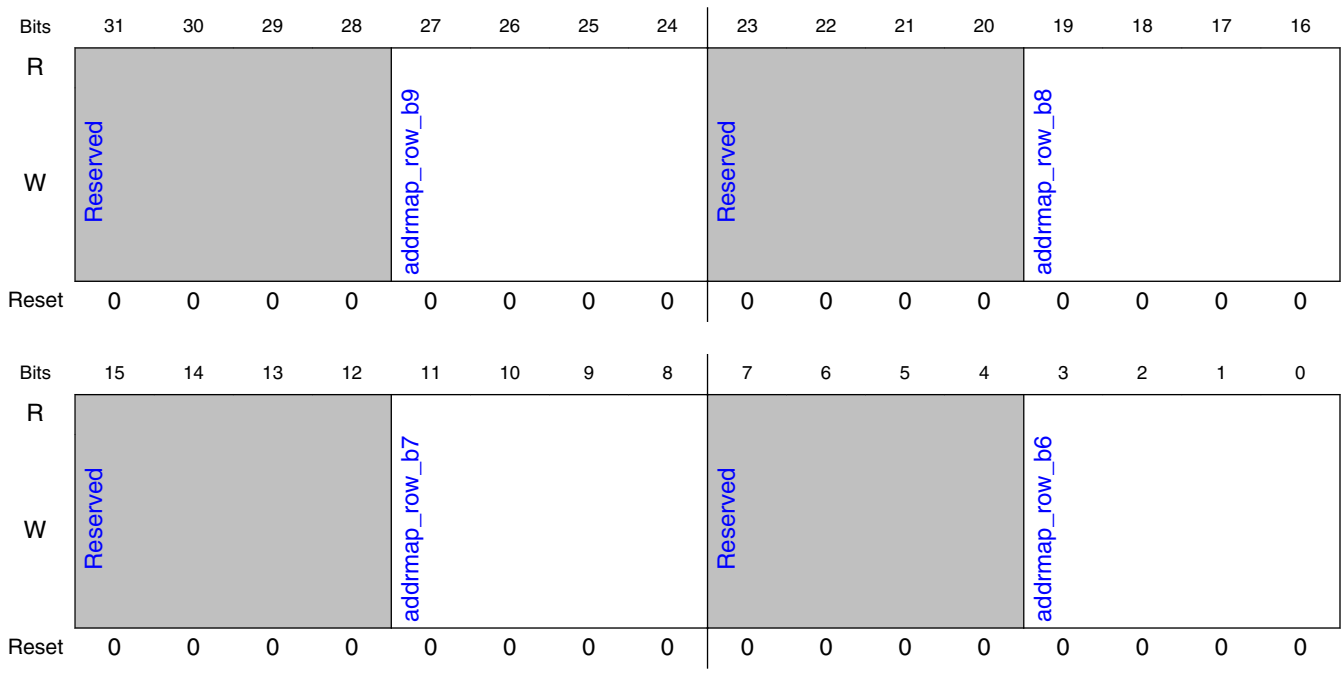
Field	Function
3-0 addrmap_row_b2	Selects the HIF address bits used as row address bit 2. Valid Range: 0 to 11 Internal Base: 8 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15.

9.2.3.1.71 Address Map Register 10 (ADDRMAP10)

9.2.3.1.71.1 Offset

Register	Offset
ADDRMAP10	228h

9.2.3.1.71.2 Diagram



9.2.3.1.71.3 Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

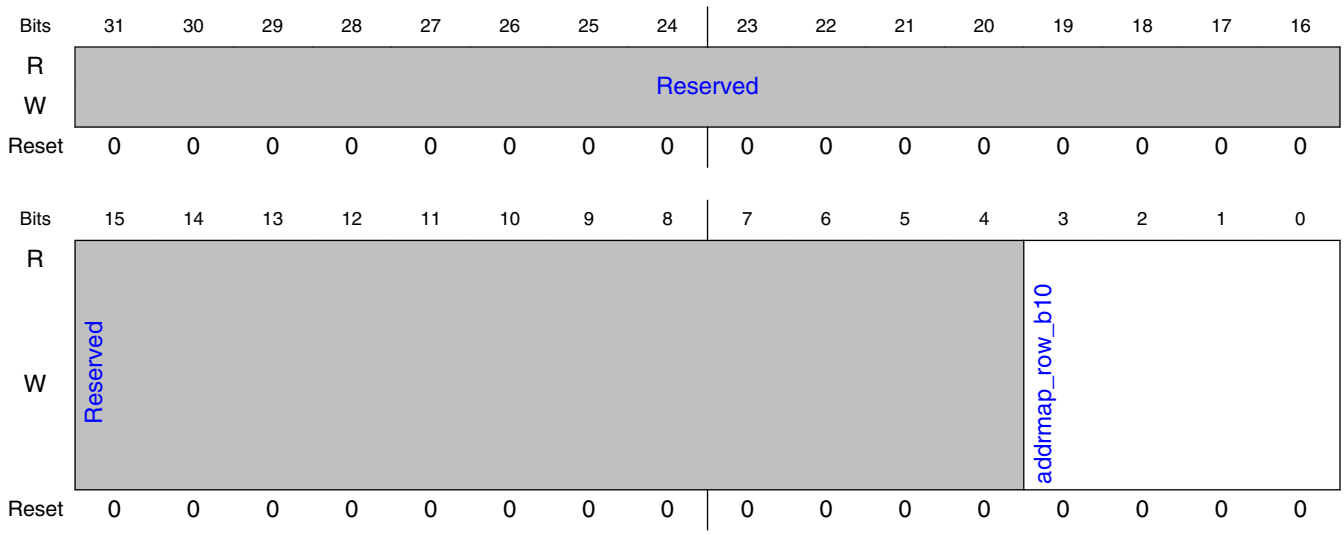
Field	Function
27-24 addrmap_row_b9	Selects the HIF address bits used as row address bit 9. Valid Range: 0 to 11 Internal Base: 15 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15.
23-20 —	Reserved
19-16 addrmap_row_b8	Selects the HIF address bits used as row address bit 8. Valid Range: 0 to 11 Internal Base: 14 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15.
15-12 —	Reserved
11-8 addrmap_row_b7	Selects the HIF address bits used as row address bit 7. Valid Range: 0 to 11 Internal Base: 13 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15.
7-4 —	Reserved
3-0 addrmap_row_b6	Selects the HIF address bits used as row address bit 6. Valid Range: 0 to 11 Internal Base: 12 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15.

9.2.3.1.72 Address Map Register 11 (ADDRMAP11)

9.2.3.1.72.1 Offset

Register	Offset
ADDRMAP11	22Ch

9.2.3.1.72.2 Diagram



9.2.3.1.72.3 Fields

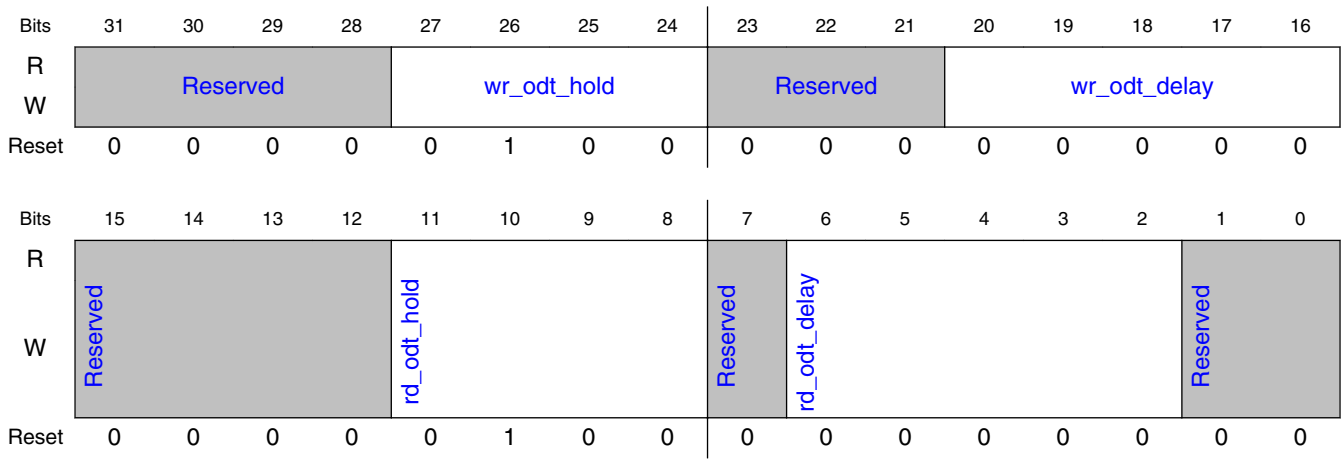
Field	Function
31-4 —	Reserved
3-0 addrmap_row_b10	Selects the HIF address bits used as row address bit 10. Valid Range: 0 to 11 Internal Base: 16 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15.

9.2.3.1.73 ODT Configuration Register (ODTCFG)

9.2.3.1.73.1 Offset

Register	Offset
ODTCFG	240h

9.2.3.1.73.2 Diagram



9.2.3.1.73.3 Fields

Field	Function
31-28 —	Reserved
27-24 wr_odt_hold	DFI PHY clock cycles to hold ODT for a write command. The minimum supported value is 2. Recommended values: DDR2: - BL8: 0x5 (DDR2-400/533/667), 0x6 (DDR2-800), 0x7 (DDR2-1066) - BL4: 0x3 (DDR2-400/533/667), 0x4 (DDR2-800), 0x5 (DDR2-1066) DDR3: - BL8: 0x6 DDR4: - BL8: 5 + WR_PREAMBLE + CRC_MODE WR_PREAMBLE = 1 (1tCK write preamble), 2 (2tCK write preamble) CRC_MODE = 0 (not CRC mode), 1 (CRC mode) LPDDR3: - BL8: 7 + RU(tODTon(max)/tCK)
23-21 —	Reserved
20-16 wr_odt_delay	The delay, in DFI PHY clock cycles, from issuing a write command to setting ODT values associated with that command. ODT setting must remain constant for the entire time that DQS is driven by the DDRC. Recommended values: DDR2: - CWL + AL - 3 (DDR2-400/533/667), CWL + AL - 4 (DDR2-800), CWL + AL - 5 (DDR2-1066) If (CWL + AL - 3 < 0), DDRC does not support ODT for write operation. DDR3: - 0x0 DDR4: - DFITMG1.dfi_t_cmd_lat (to adjust for CAL mode) LPDDR3: - WL - 1 - RU(tODTon(max)/tCK))
15-12 —	Reserved
11-8 rd_odt_hold	DFI PHY clock cycles to hold ODT for a read command. The minimum supported value is 2. Recommended values: DDR2: - BL8: 0x6 (not DDR2-1066), 0x7 (DDR2-1066) - BL4: 0x4 (not DDR2-1066), 0x5 (DDR2-1066) DDR3: - BL8 - 0x6 DDR4: - BL8: 5 + RD_PREAMBLE RD_PREAMBLE = 1 (1tCK write preamble), 2 (2tCK write preamble) LPDDR3: - BL8: 5 + RU(tDQSCK(max)/tCK) - RD(tDQSCK(min)/tCK) + RU(tODTon(max)/tCK)
7 —	Reserved
6-2 rd_odt_delay	The delay, in DFI PHY clock cycles, from issuing a read command to setting ODT values associated with that command. ODT setting must remain constant for the entire time that DQS is driven by the DDRC. Recommended values: DDR2: - CL + AL - 4 (not DDR2-1066), CL + AL - 5 (DDR2-1066) If (CL + AL - 4 < 0), DDRC does not support ODT for read operation. DDR3: - CL - CWL DDR4: - CL - CWL - RD_PREAMBLE + WR_PREAMBLE + DFITMG1.dfi_t_cmd_lat (to adjust for CAL mode) WR_PREAMBLE = 1 (1tCK write preamble), 2 (2tCK write preamble) RD_PREAMBLE = 1 (1tCK write

Table continues on the next page...

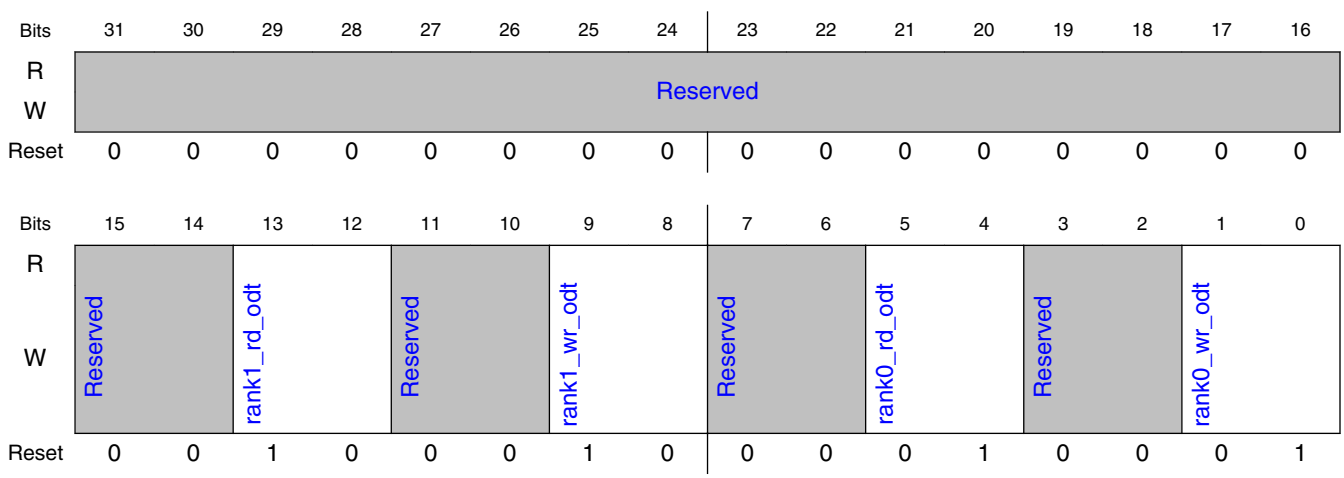
Field	Function
	preamble), 2 (2tCK write preamble) If (CL - CWL - RD_PREAMBLE + WR_PREAMBLE) < 0, DDRC does not support ODT for read operation. LPDDR3: - RL + RD(tDQSCK(min)/tCK) - 1 - RU(tODTon(max)/tCK)
1-0 —	Reserved

9.2.3.1.74 ODT/Rank Map Register (ODTMAP)

9.2.3.1.74.1 Offset

Register	Offset
ODTMAP	244h

9.2.3.1.74.2 Diagram



9.2.3.1.74.3 Fields

Field	Function
31-14 —	Reserved
13-12 rank1_rd_odt	Indicates which remote ODTs must be turned on during a read from rank 1. Each rank has a remote ODT (in the SDRAM) which can be turned on by setting the appropriate bit here. Rank 0 is controlled by the LSB; rank 1 is controlled by bit next to the LSB, etc. For each rank, set its bit to 1 to enable its ODT. Present only in configurations that have 2 or more ranks
11-10 —	Reserved

Table continues on the next page...

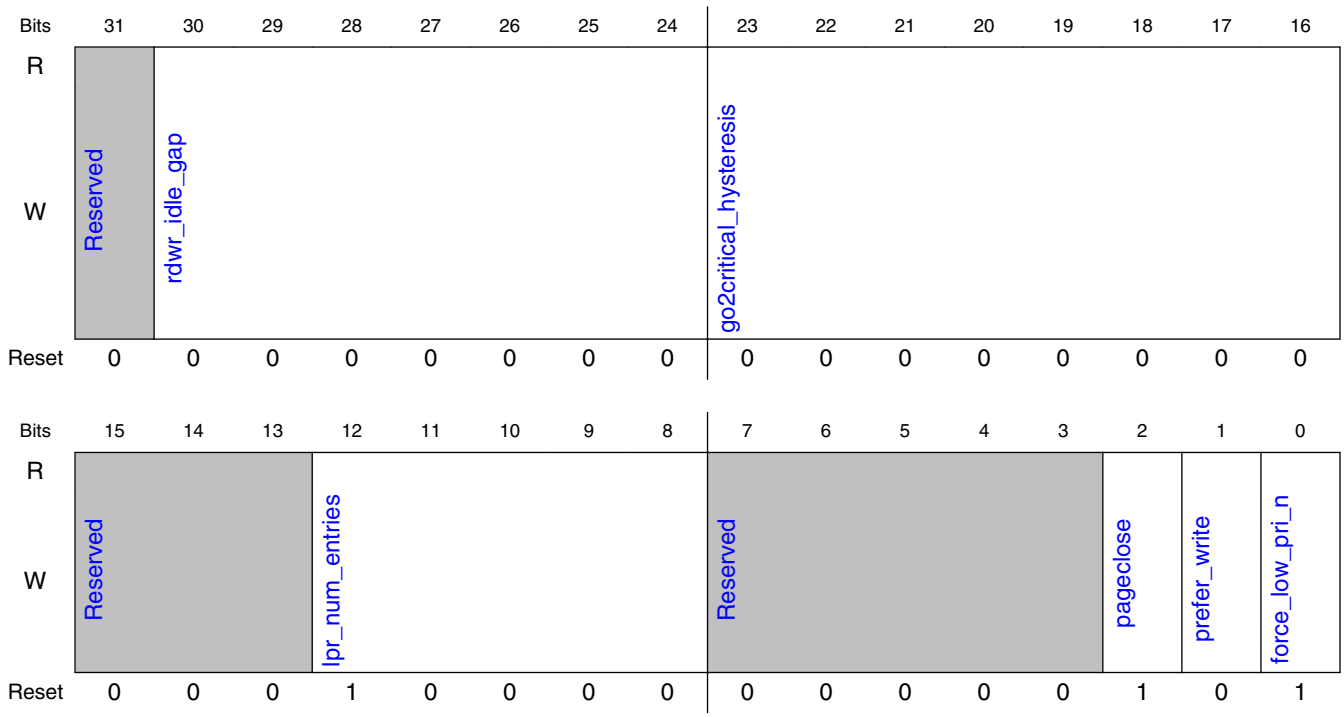
Field	Function
9-8 rank1_wr_odt	Indicates which remote ODTs must be turned on during a write to rank 1. Each rank has a remote ODT (in the SDRAM) which can be turned on by setting the appropriate bit here. Rank 0 is controlled by the LSB; rank 1 is controlled by bit next to the LSB, etc. For each rank, set its bit to 1 to enable its ODT. Present only in configurations that have 2 or more ranks
7-6 —	Reserved
5-4 rank0_rd_odt	Indicates which remote ODTs must be turned on during a read from rank 0. Each rank has a remote ODT (in the SDRAM) which can be turned on by setting the appropriate bit here. Rank 0 is controlled by the LSB; rank 1 is controlled by bit next to the LSB, etc. For each rank, set its bit to 1 to enable its ODT.
3-2 —	Reserved
1-0 rank0_wr_odt	Indicates which remote ODTs must be turned on during a write to rank 0. Each rank has a remote ODT (in the SDRAM) which can be turned on by setting the appropriate bit here. Rank 0 is controlled by the LSB; rank 1 is controlled by bit next to the LSB, etc. For each rank, set its bit to 1 to enable its ODT.

9.2.3.1.75 Scheduler Control Register (SCHED)

9.2.3.1.75.1 Offset

Register	Offset
SCHED	250h

9.2.3.1.75.2 Diagram



9.2.3.1.75.3 Fields

Field	Function
31 —	Reserved
30-24 rdwr_idle_gap	When the preferred transaction store is empty for these many clock cycles, switch to the alternate transaction store if it is non-empty. The read transaction store (both high and low priority) is the default preferred transaction store and the write transaction store is the alternative store. When prefer write over read is set this is reversed. 0x0 is a legal value for this register. When set to 0x0, the transaction store switching will happen immediately when the switching conditions become true. FOR PERFORMANCE ONLY
23-16 go2critical_hysteresis	UNUSED
15-13 —	Reserved
12-8 lpr_num_entries	Number of entries in the low priority transaction store is this value + 1. (MEMC_NO_OF_ENTRY - (SCHED.lpr_num_entries + 1)) is the number of entries available for the high priority transaction store. Setting this to maximum value allocates all entries to low priority transaction store. Setting this to 0 allocates 1 entry to low priority transaction store and the rest to high priority transaction store. Note: In ECC configurations, the numbers of write and low priority read credits issued is one less than in the non-ECC case. One entry each is reserved in the write and low-priority read CAMs for storing the RMW requests arising out of single bit error correction RMW operation.
7-3	Reserved

Table continues on the next page...

Field	Function
—	
2 pageclose	If true, bank is kept open only while there are page hit transactions available in the CAM to that bank. The last read or write command in the CAM with a bank and page hit will be executed with auto-precharge if SCHED1.pageclose_timer=0. Even if this register set to 1 and SCHED1.pageclose_timer is set to 0, explicit precharge (and not auto-precharge) may be issued in some cases where there is a mode switch between Write and Read or between LPR and HPR. The Read and Write commands that are executed as part of the ECC scrub requests are also executed without auto-precharge. If false, the bank remains open until there is a need to close it (to open a different page, or for page timeout or refresh timeout) - also known as open page policy. The open page policy can be overridden by setting the per-command-autopre bit on the HIF interface (hif_cmd_autopre). The pageclose feature provids a midway between Open and Close page policies. FOR PERFORMANCE ONLY.
1 prefer_write	If set then the bank selector prefers writes over reads. FOR DEBUG ONLY.
0 force_low_pri_n	Active low signal. When asserted ('0'), all incoming transactions are forced to low priority. This implies that all High Priority Read (HPR) and Variable Priority Read commands (VPR) will be treated as Low Priority Read (LPR) commands. On the write side, all Variable Priority Write (VPW) commands will be treated as Normal Priority Write (NPW) commands. Forcing the incoming transactions to low priority implicitly turns off Bypass path for read commands. FOR PERFORMANCE ONLY.

9.2.3.1.76 Scheduler Control Register 1 (SCHED1)

9.2.3.1.76.1 Offset

Register	Offset
SCHED1	254h

9.2.3.1.76.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								pageclose_timer							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.76.3 Fields

Field	Function
31-8 —	Reserved
7-0 pageclose_timer	This field works in conjunction with SCHED.pageclose. It only has meaning if SCHED.pageclose==1. If SCHED.pageclose==1 and pageclose_timer==0, then an auto-precharge may be scheduled for last read or write command in the CAM with a bank and page hit. Note, sometimes an explicit precharge is scheduled instead of the auto-precharge. See SCHED.pageclose for details of when this may happen. If SCHED.pageclose==1 and pageclose_timer>0, then an auto-precharge is not scheduled for last read or write command in the CAM with a bank and page hit. Instead, a timer is started, with pageclose_timer as the initial value. There is a timer on a per bank basis. The timer decrements unless the next read or write in the CAM to a bank is a page hit. It gets reset to pageclose_timer value if the next read or write in the CAM to a bank is a page hit. Once the timer has reached zero, an explcit precharge will be attempted to be scheduled.

9.2.3.1.77 High Priority Read CAM Register 1 (PERFHPR1)

9.2.3.1.77.1 Offset

Register	Offset
PERFHPR1	25Ch

9.2.3.1.77.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	hpr_xact_run_length								Reserved							
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	hpr_max_starve															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

9.2.3.1.77.3 Fields

Field	Function
31-24	Number of transactions that are serviced once the HPR queue goes critical is the smaller of: - (a) This number - (b) Number of transactions available. Unit: Transaction. FOR PERFORMANCE ONLY.

Table continues on the next page...

Field	Function
hpr_xact_run_length	
23-16 —	Reserved
15-0 hpr_max_starve	Number of DFI clocks that the HPR queue can be starved before it goes critical. The minimum valid functional value for this register is 0x1. Programming it to 0x0 will disable the starvation functionality; during normal operation, this function should not be disabled as it will cause excessive latencies. FOR PERFORMANCE ONLY.

9.2.3.1.78 Low Priority Read CAM Register 1 (PERFLPR1)

9.2.3.1.78.1 Offset

Register	Offset
PERFLPR1	264h

9.2.3.1.78.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	lpr_xact_run_length								Reserved							
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	lpr_max_starve															
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1

9.2.3.1.78.3 Fields

Field	Function
31-24 lpr_xact_run_length	Number of transactions that are serviced once the LPR queue goes critical is the smaller of: - (a) This number - (b) Number of transactions available. Unit: Transaction. FOR PERFORMANCE ONLY.
23-16 —	Reserved
15-0 lpr_max_starve	Number of DFI clocks that the LPR queue can be starved before it goes critical. The minimum valid functional value for this register is 0x1. Programming it to 0x0 will disable the starvation functionality;

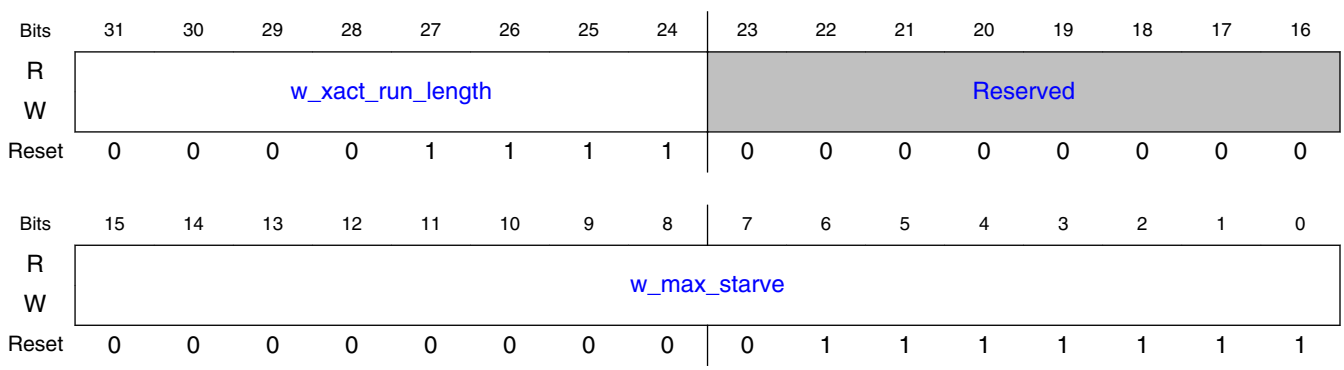
Field	Function
	during normal operation, this function should not be disabled as it will cause excessive latencies. FOR PERFORMANCE ONLY.

9.2.3.1.79 Write CAM Register 1 (PERFWR1)

9.2.3.1.79.1 Offset

Register	Offset
PERFWR1	26Ch

9.2.3.1.79.2 Diagram



9.2.3.1.79.3 Fields

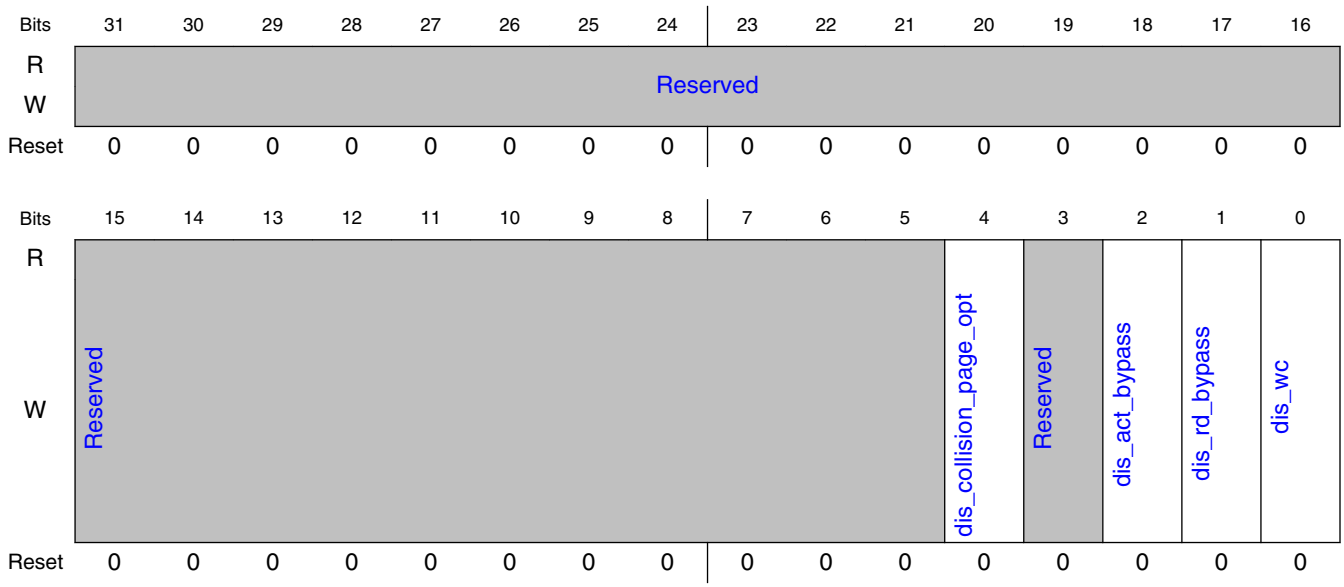
Field	Function
31-24 w_xact_run_length	Number of transactions that are serviced once the WR queue goes critical is the smaller of: - (a) This number - (b) Number of transactions available. Unit: Transaction. FOR PERFORMANCE ONLY.
23-16 —	Reserved
15-0 w_max_starve	Number of DFI clocks that the WR queue can be starved before it goes critical. The minimum valid functional value for this register is 0x1. Programming it to 0x0 will disable the starvation functionality; during normal operation, this function should not be disabled as it will cause excessive latencies. FOR PERFORMANCE ONLY.

9.2.3.1.80 Debug Register 0 (DBG0)

9.2.3.1.80.1 Offset

Register	Offset
DBG0	300h

9.2.3.1.80.2 Diagram



9.2.3.1.80.3 Fields

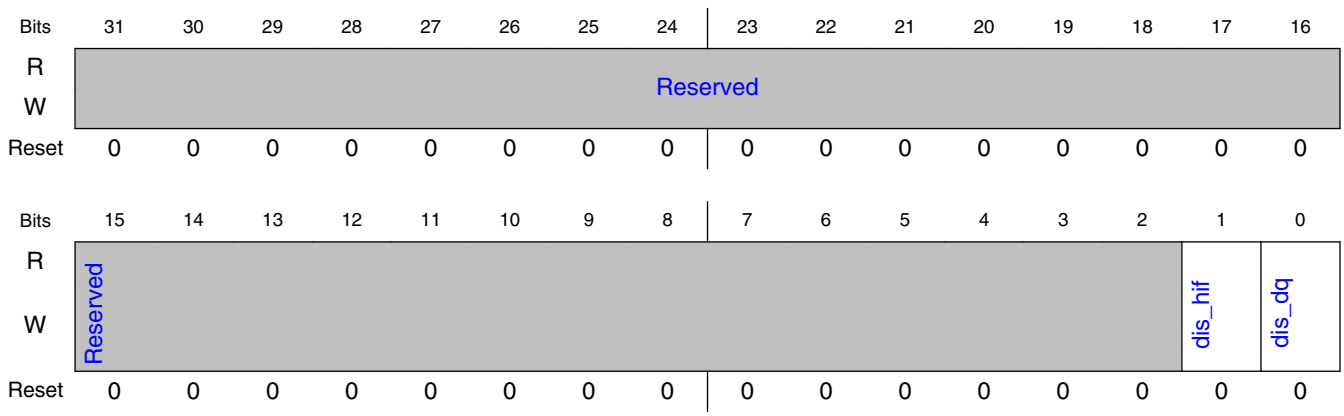
Field	Function
31-5 —	Reserved
4 dis_collision_page_opt	When this is set to '0', auto-precharge is disabled for the flushed command in a collision case. Collision cases are write followed by read to same address, read followed by write to same address, or write followed by write to same address with DBG0.dis_wc bit = 1 (where same address comparisons exclude the two address bits representing critical word). FOR DEBUG ONLY.
3 —	Reserved
2 dis_act_bypass	Only present in designs supporting activate bypass. When 1, disable bypass path for high priority read activates FOR DEBUG ONLY.
1 dis_rd_bypass	Only present in designs supporting read bypass. When 1, disable bypass path for high priority read page hits FOR DEBUG ONLY.
0 dis_wc	When 1, disable write combine. FOR DEBUG ONLY

9.2.3.1.81 Debug Register 1 (DBG1)

9.2.3.1.81.1 Offset

Register	Offset
DBG1	304h

9.2.3.1.81.2 Diagram



9.2.3.1.81.3 Fields

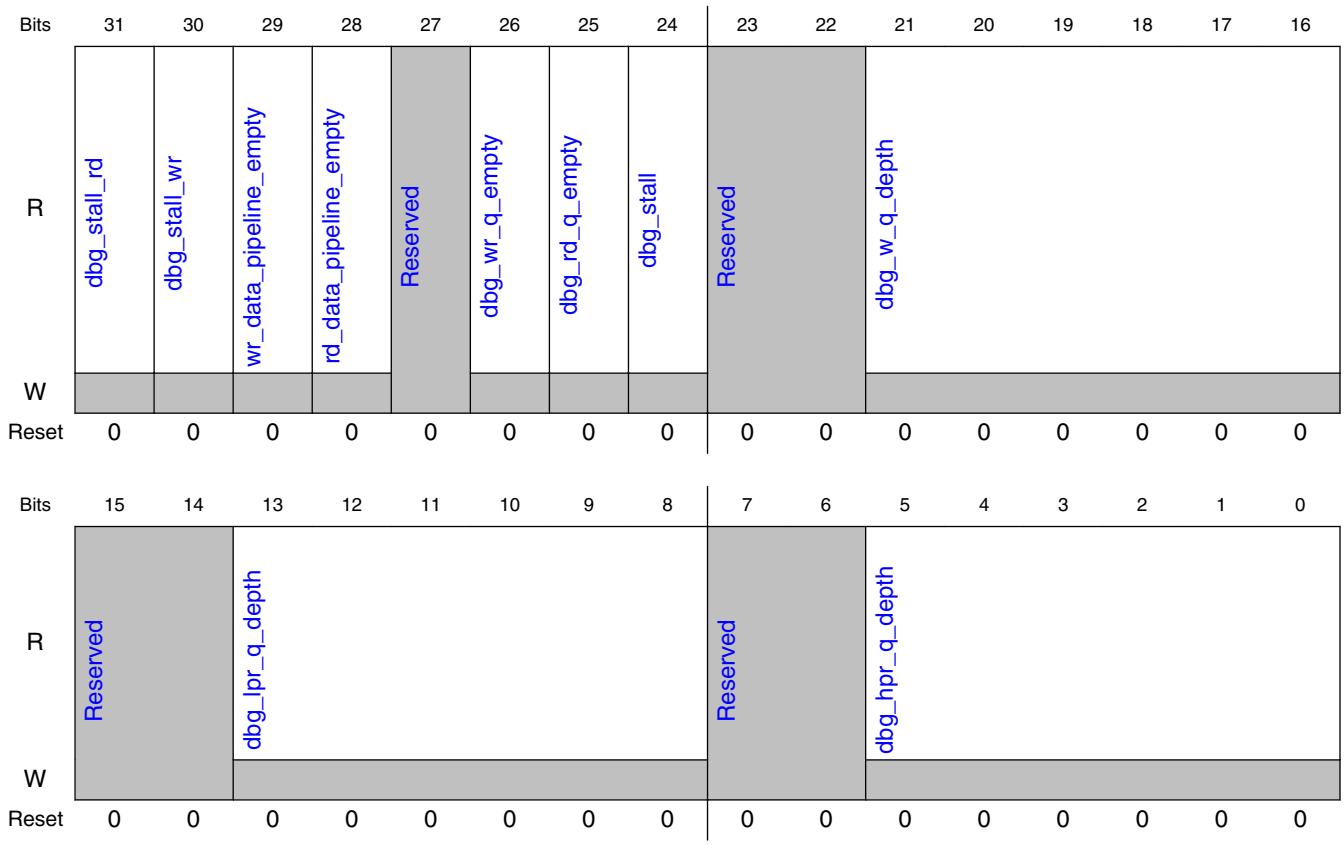
Field	Function
31-2 —	Reserved
1 dis_hif	When 1, DDRC asserts the HIF command signal hif_cmd_stall. DDRC will ignore the hif_cmd_valid and all other associated request signals. This bit is intended to be switched on-the-fly.
0 dis_dq	When 1, DDRC will not de-queue any transactions from the CAM. Bypass is also disabled. All transactions are queued in the CAM. No reads or writes are issued to SDRAM as long as this is asserted. This bit may be used to prevent reads or writes being issued by the DDRC, which makes it safe to modify certain register fields associated with reads and writes (see User Guide for details). After setting this bit, it is strongly recommended to poll DBGCAM.wr_data_pipeline_empty and DBGCAM.rd_data_pipeline_empty, before making changes to any registers which affect reads and writes. This will ensure that the relevant logic in the DDRC is idle. This bit is intended to be switched on-the-fly.

9.2.3.1.82 CAM Debug Register (DBGCAM)

9.2.3.1.82.1 Offset

Register	Offset
DBGCAM	308h

9.2.3.1.82.2 Diagram



9.2.3.1.82.3 Fields

Field	Function
31 dbg_stall_rd	Stall for Read channel FOR DEBUG ONLY
30 dbg_stall_wr	Stall for Write channel FOR DEBUG ONLY
29 wr_data_pipeline_empty	This bit indicates that the write data pipeline on the DFI interface is empty. This register is intended to be polled at least twice after setting DBG1.dis_dq, to ensure that all remaining commands/data have completed.

Table continues on the next page...

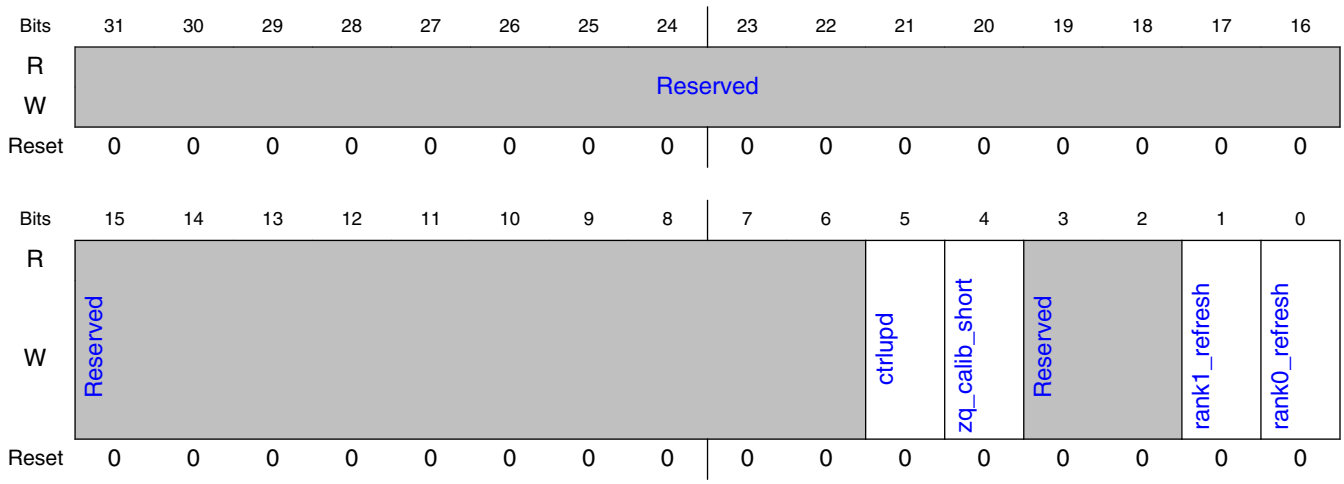
Field	Function
28 rd_data_pipeline_empty	This bit indicates that the read data pipeline on the DFI interface is empty. This register is intended to be polled at least twice after setting DBG1.dis_dq, to ensure that all remaining commands/data have completed.
27 —	Reserved
26 dbg_wr_q_empty	When 1, all the Write command queues and Write data buffers inside DDRC are empty. This register is to be used for debug purpose. An example use-case scenario: When Controller enters Self-Refresh using the Low-Power entry sequence, Controller is expected to have executed all the commands in its queues and the write and read data drained. Hence this register should be 1 at that time. FOR DEBUG ONLY
25 dbg_rd_q_empty	When 1, all the Read command queues and Read data buffers inside DDRC are empty. This register is to be used for debug purpose. An example use-case scenario: When Controller enters Self-Refresh using the Low-Power entry sequence, Controller is expected to have executed all the commands in its queues and the write and read data drained. Hence this register should be 1 at that time. FOR DEBUG ONLY
24 dbg_stall	Stall FOR DEBUG ONLY
23-22 —	Reserved
21-16 dbg_w_q_depth	Write queue depth The last entry of WR queue is reserved for ECC SCRUB operation. This entry is not included in the calculation of the queue depth. FOR DEBUG ONLY
15-14 —	Reserved
13-8 dbg_lpr_q_depth	Low priority read queue depth The last entry of Lpr queue is reserved for ECC SCRUB operation. This entry is not included in the calculation of the queue depth. FOR DEBUG ONLY
7-6 —	Reserved
5-0 dbg_hpr_q_depth	High priority read queue depth FOR DEBUG ONLY

9.2.3.1.83 Command Debug Register (DBGCMD)

9.2.3.1.83.1 Offset

Register	Offset
DBGCMD	30Ch

9.2.3.1.83.2 Diagram



9.2.3.1.83.3 Fields

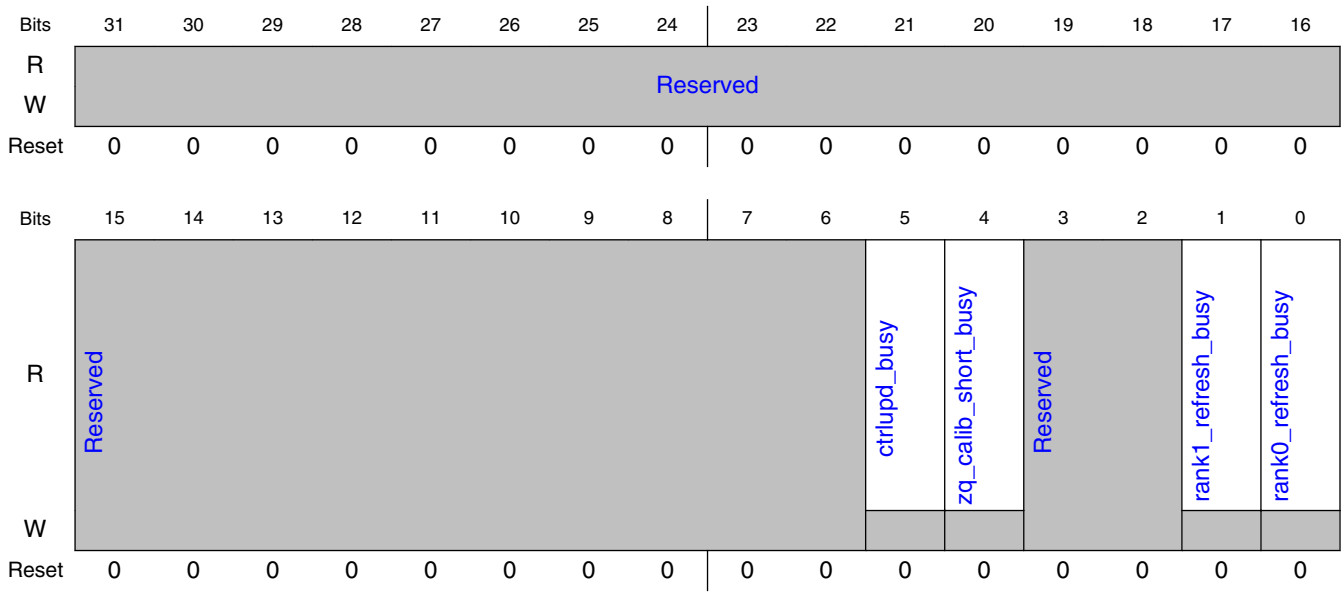
Field	Function
31-6 —	Reserved
5 ctrlupd	Setting this register bit to 1 indicates to the DDRC to issue a dfi_ctrlupd_req to the PHY. When this request is stored in the DDRC, the bit is automatically cleared. This operation must only be performed when DFIUPD0.dis_auto_ctrlupd=1.
4 zq_calib_short	Setting this register bit to 1 indicates to the DDRC to issue a ZQCS (ZQ calibration short)/MPC(ZQ calibration) command to the SDRAM. When this request is stored in the DDRC, the bit is automatically cleared. This operation can be performed only when ZQCTL0.dis_auto_zq=1. It is recommended NOT to set this register bit if in Init operating mode. This register bit is ignored when in Self-Refresh(except LPDDR4) and SR-Powerdown(LPDDR4) and Deep power-down operating modes and Maximum Power Saving Mode.
3-2 —	Reserved
1 rank1_refresh	Setting this register bit to 1 indicates to the DDRC to issue a refresh to rank 1. Writing to this bit causes DBGSTAT.rank1_refresh_busy to be set. When DBGSTAT.rank1_refresh_busy is cleared, the command has been stored in DDRC. For 3DS configuration, refresh is sent to rank index 1. This operation can be performed only when RFSHCTL3.dis_auto_refresh=1. It is recommended NOT to set this register bit if in Init or Deep power-down operating modes or Maximum Power Saving Mode.
0 rank0_refresh	Setting this register bit to 1 indicates to the DDRC to issue a refresh to rank 0. Writing to this bit causes DBGSTAT.rank0_refresh_busy to be set. When DBGSTAT.rank0_refresh_busy is cleared, the command has been stored in DDRC. For 3DS configuration, refresh is sent to rank index 0. This operation can be performed only when RFSHCTL3.dis_auto_refresh=1. It is recommended NOT to set this register bit if in Init or Deep power-down operating modes or Maximum Power Saving Mode.

9.2.3.1.84 Status Debug Register (DBGSTAT)

9.2.3.1.84.1 Offset

Register	Offset
DBGSTAT	310h

9.2.3.1.84.2 Diagram



9.2.3.1.84.3 Fields

Field	Function
31-6 —	Reserved
5 ctrlupd_busy	SoC core may initiate a ctrlupd operation only if this signal is low. This signal goes high in the clock after the DDRC accepts the ctrlupd request. It goes low when the ctrlupd operation is initiated in the DDRC. It is recommended not to perform ctrlupd operations when this signal is high. - 0 - Indicates that the SoC core can initiate a ctrlupd operation - 1 - Indicates that ctrlupd operation has not been initiated yet in the DDRC
4 zq_calib_short_busy	SoC core may initiate a ZQCS (ZQ calibration short) operation only if this signal is low. This signal goes high in the clock after the DDRC accepts the ZQCS request. It goes low when the ZQCS operation is initiated in the DDRC. It is recommended not to perform ZQCS operations when this signal is high. - 0 - Indicates that the SoC core can initiate a ZQCS operation - 1 - Indicates that ZQCS operation has not been initiated yet in the DDRC
3-2 —	Reserved
1 rank1_refresh_busy	SoC core may initiate a rank1_refresh operation (refresh operation to rank 1) only if this signal is low. This signal goes high in the clock after DBGCMD.rank1_refresh is set to one. It goes low when the rank1_refresh operation is stored in the DDRC. It is recommended not to perform rank1_refresh

Table continues on the next page...

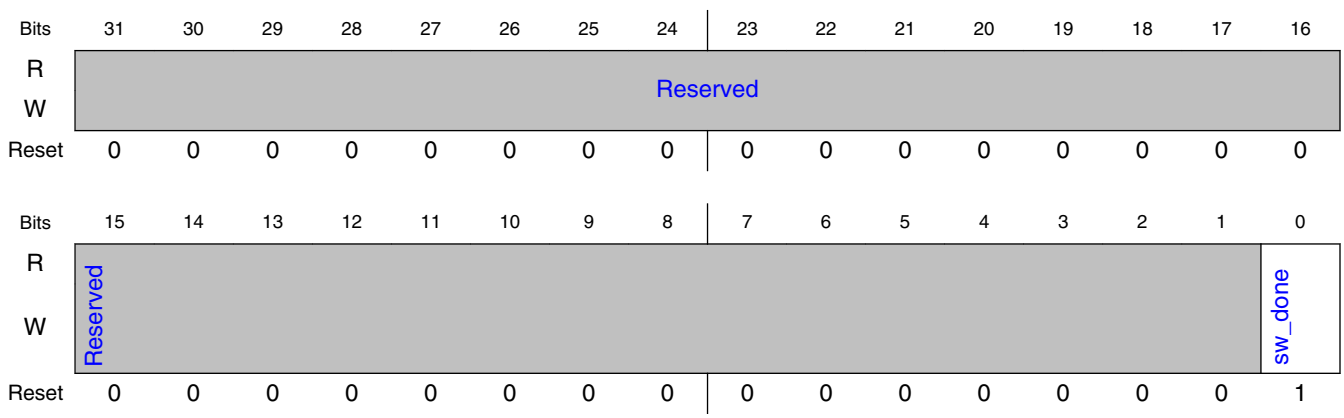
Field	Function
	operations when this signal is high. - 0 - Indicates that the SoC core can initiate a rank1_refresh operation - 1 - Indicates that rank1_refresh operation has not been stored yet in the DDRC
0 rank0_refresh_busy	SoC core may initiate a rank0_refresh operation (refresh operation to rank 0) only if this signal is low. This signal goes high in the clock after DBGCMD.rank0_refresh is set to one. It goes low when the rank0_refresh operation is stored in the DDRC. It is recommended not to perform rank0_refresh operations when this signal is high. - 0 - Indicates that the SoC core can initiate a rank0_refresh operation - 1 - Indicates that rank0_refresh operation has not been stored yet in the DDRC

9.2.3.1.85 Software Register Programming Control Enable (SWCTL)

9.2.3.1.85.1 Offset

Register	Offset
SWCTL	320h

9.2.3.1.85.2 Diagram



9.2.3.1.85.3 Fields

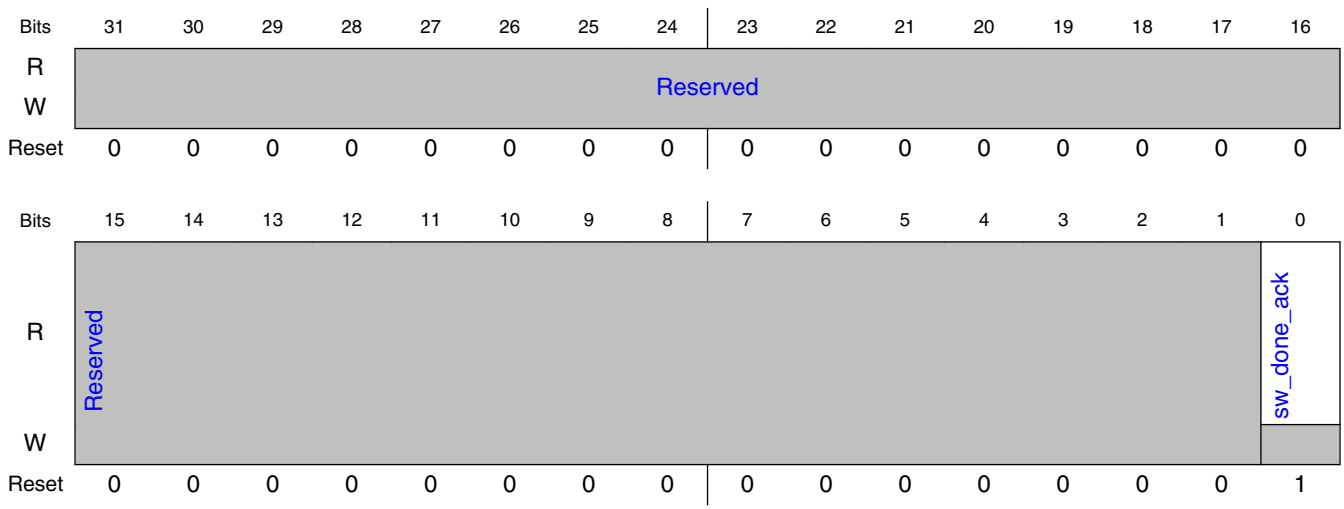
Field	Function
31-1 —	Reserved
0 sw_done	Enable quasi-dynamic register programming outside reset. Program register to 0 to enable quasi-dynamic programming. Set back register to 1 once programming is done.

9.2.3.1.86 Software Register Programming Control Status (SWSTAT)

9.2.3.1.86.1 Offset

Register	Offset
SWSTAT	324h

9.2.3.1.86.2 Diagram



9.2.3.1.86.3 Fields

Field	Function
31-1 —	Reserved
0 sw_done_ack	Register programming done. This register is the echo of SWCTL.sw_done. Wait for sw_done value 1 to propagate to sw_done_ack at the end of the programming sequence to ensure that the correct registers values are propagated to the destination clock domains.

9.2.3.1.87 AXI Poison Configuration Register. (POISONCFG)

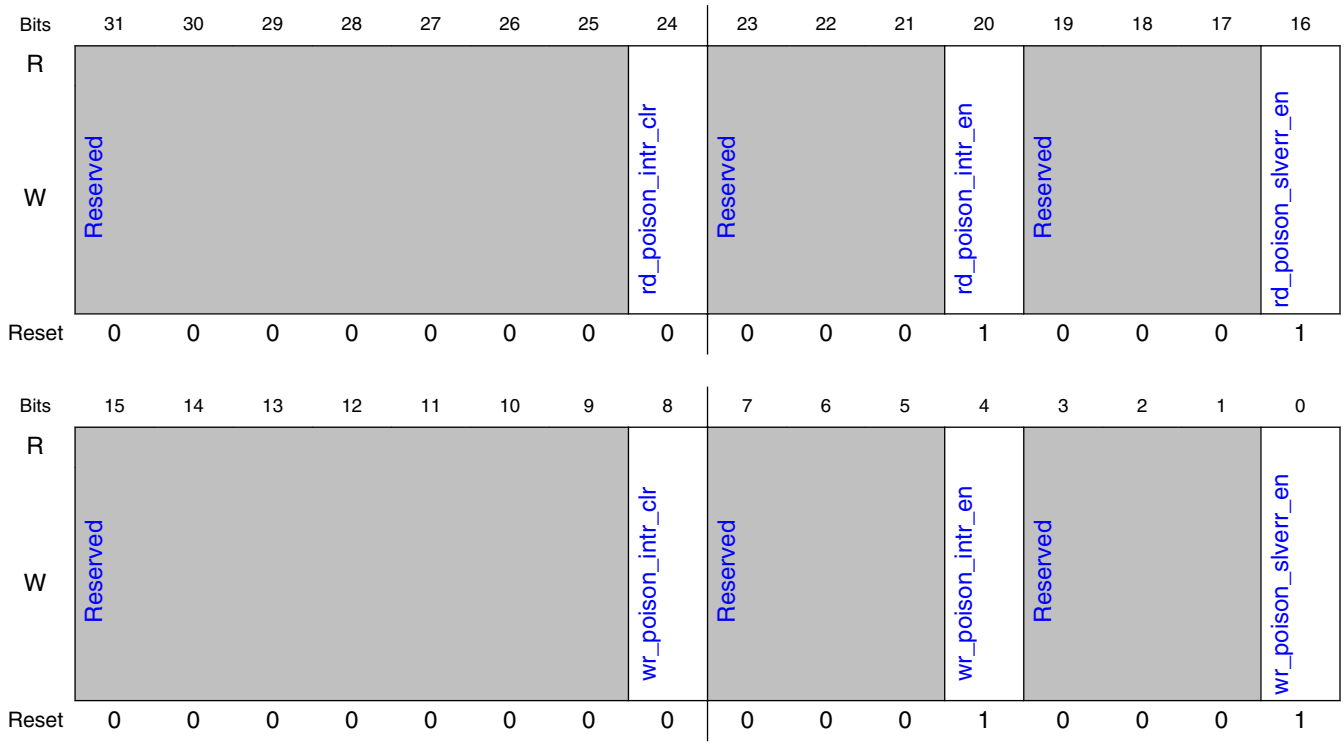
9.2.3.1.87.1 Offset

Register	Offset
POISONCFG	36Ch

9.2.3.1.87.2 Function

AXI Poison Configuration Register. Common for all AXI ports

9.2.3.1.87.3 Diagram



9.2.3.1.87.4 Fields

Field	Function
31-25 —	Reserved
24 rd_poison_intr_clr	Interrupt clear for read transaction poisoning. Allow 2/3 clock cycles for correct value to propagate to core logic and clear the interrupts.
23-21 —	Reserved
20 rd_poison_intr_en	If set to 1, enables interrupts for read transaction poisoning
19-17 —	Reserved
16	If set to 1, enables SLVERR response for read transaction poisoning

Table continues on the next page...

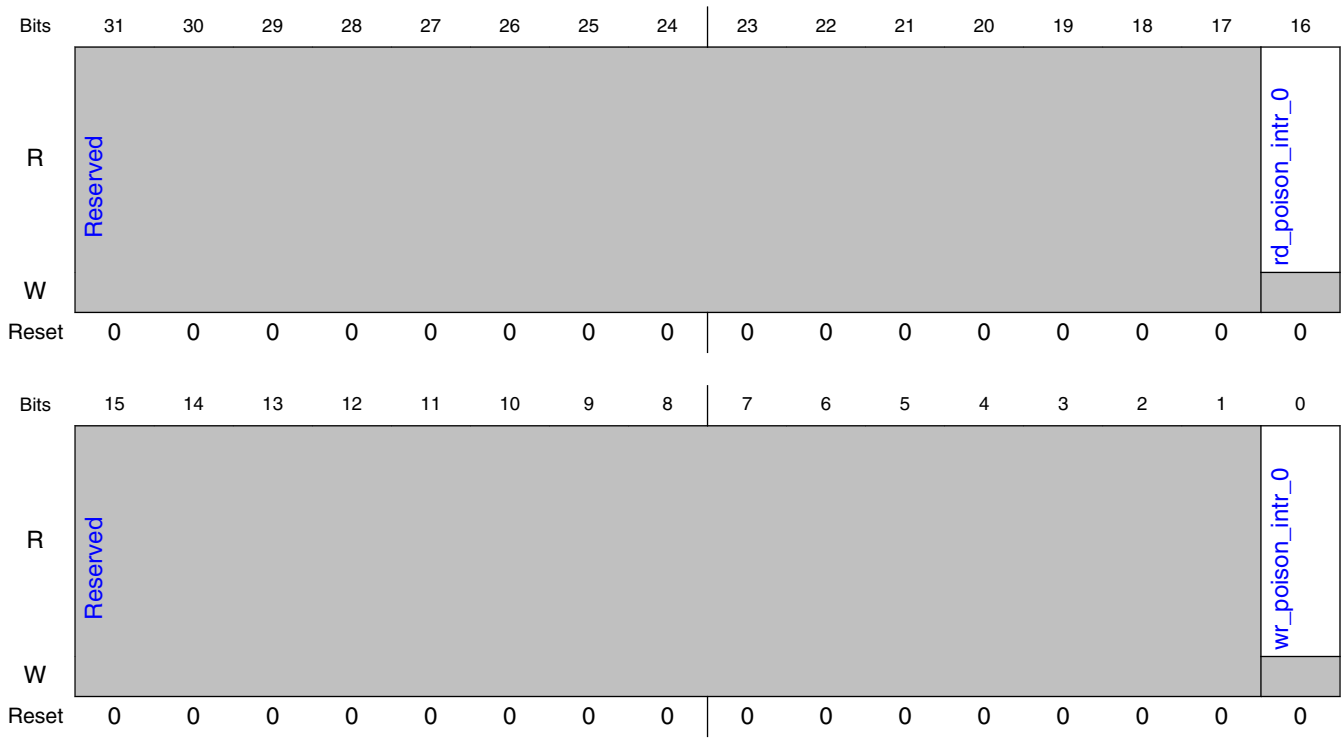
Field	Function
rd_poison_slverr_en	
15-9 —	Reserved
8 wr_poison_intr_clr	Interrupt clear for write transaction poisoning. Allow 2/3 clock cycles for correct value to propagate to core logic and clear the interrupts.
7-5 —	Reserved
4 wr_poison_intr_en	If set to 1, enables interrupts for write transaction poisoning
3-1 —	Reserved
0 wr_poison_slverr_en	If set to 1, enables SLVERR response for write transaction poisoning

9.2.3.1.88 AXI Poison Status Register (POISONSTAT)

9.2.3.1.88.1 Offset

Register	Offset
POISONSTAT	370h

9.2.3.1.88.2 Diagram



9.2.3.1.88.3 Fields

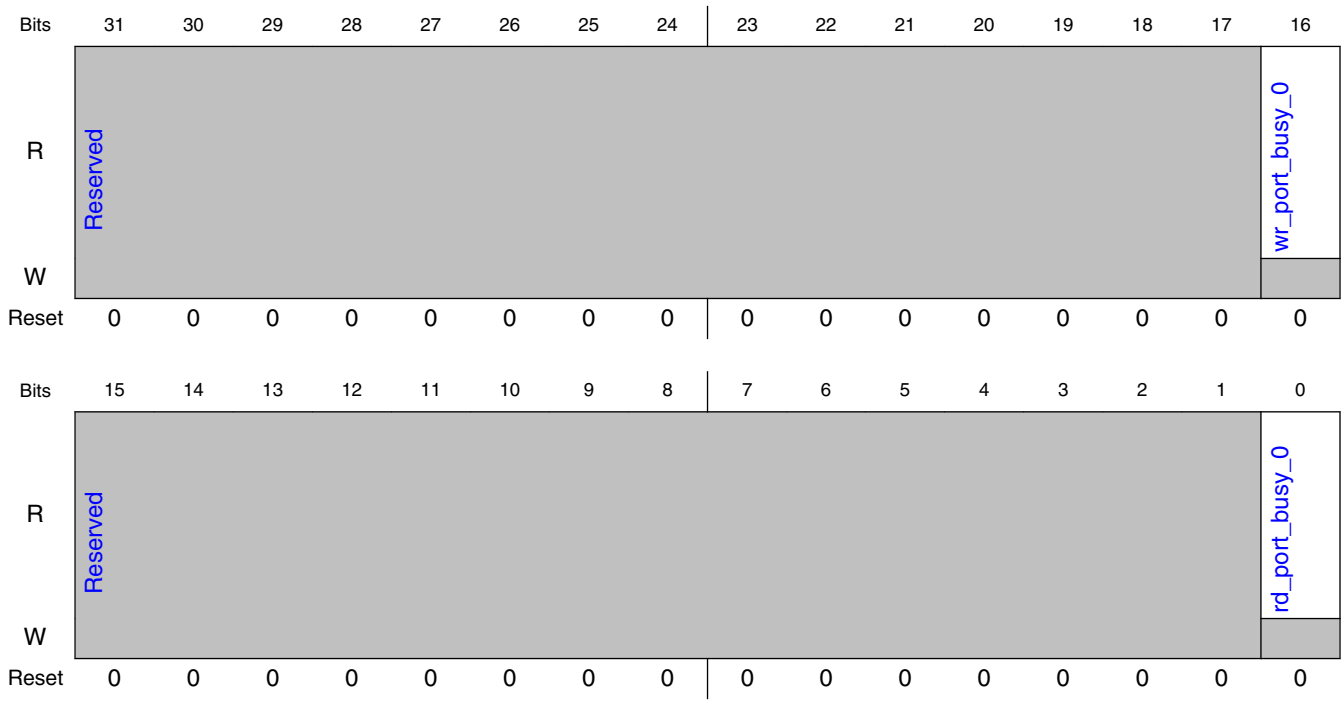
Field	Function
31-17 —	Reserved
16 rd_poison_intr_0	Read transaction poisoning error interrupt for port 0. This register is a APB clock copy (double register synchronizer) of the interrupt asserted when a transaction is poisoned on the corresponding AXI port's read address channel. Bit 0 corresponds to Port 0, and so on. Interrupt is cleared by register rd_poison_intr_clr, then value propagated to APB clock.
15-1 —	Reserved
0 wr_poison_intr_0	Write transaction poisoning error interrupt for port 0. This register is a APB clock copy (double register synchronizer) of the interrupt asserted when a transaction is poisoned on the corresponding AXI port's write address channel. Bit 0 corresponds to Port 0, and so on. Interrupt is cleared by register wr_poison_intr_clr, then value propagated to APB clock.

9.2.3.1.89 Port Status Register (PSTAT)

9.2.3.1.89.1 Offset

Register	Offset
PSTAT	3FCh

9.2.3.1.89.2 Diagram



9.2.3.1.89.3 Fields

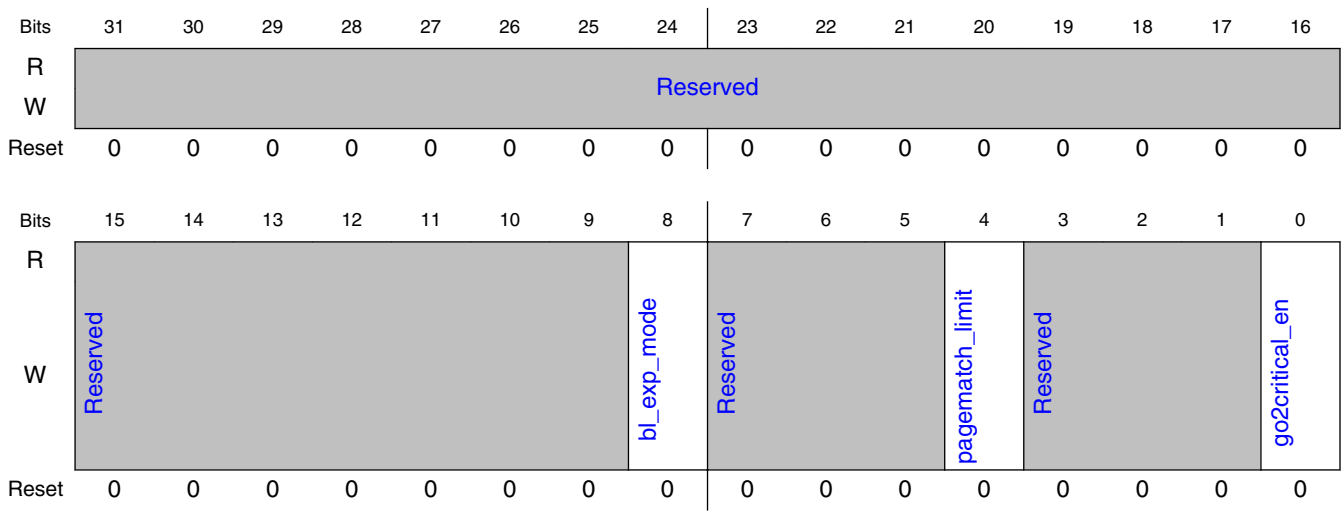
Field	Function
31-17 —	Reserved
16 wr_port_busy_0	Indicates if there are outstanding writes for AXI port 0.
15-1 —	Reserved
0 rd_port_busy_0	Indicates if there are outstanding reads for AXI port 0.

9.2.3.1.90 Port Common Configuration Register (PCCFG)

9.2.3.1.90.1 Offset

Register	Offset
PCCFG	400h

9.2.3.1.90.2 Diagram



9.2.3.1.90.3 Fields

Field	Function
31-9 —	Reserved
8 bl_exp_mode	Burst length expansion mode. By default (i.e. bl_exp_mode==0) XPI expands every AXI burst into multiple HIF commands, using the memory burst length as a unit. If set to 1, then XPI will use half of the memory burst length as a unit. This applies to both reads and writes. When MSTR.data_bus_width==00, setting bl_exp_mode to 1 has no effect. This can be used in cases where Partial Writes is enabled (DDRC_PARTIAL_WR=1), in order to avoid or minimize t_ccd_l penalty in DDR4 and t_ccd_mw penalty in LPDDR4. Hence, bl_exp_mode=1 is only recommended if DDR4 or LPDDR4. Note that if DBICTL.reg_ddrc_dm_en=0, functionality is not supported in the following cases: - DDRC_PARTIAL_WR=0 - DDRC_PARTIAL_WR=1, MSTR.data_bus_width=01, MEMC_BURST_LENGTH=8 and MSTR.burst_rdwr=1000 (LPDDR4 only) - DDRC_PARTIAL_WR=1, MSTR.data_bus_width=01, MEMC_BURST_LENGTH=4 and MSTR.burst_rdwr=0100 (DDR4 only), with either MSTR.reg_ddrc_burstchop=0 or CRCPARCTL1.reg_ddrc_crc_enable=1 Functionality is also not supported if Data Channel Interleave is enabled
7-5 —	Reserved

Table continues on the next page...

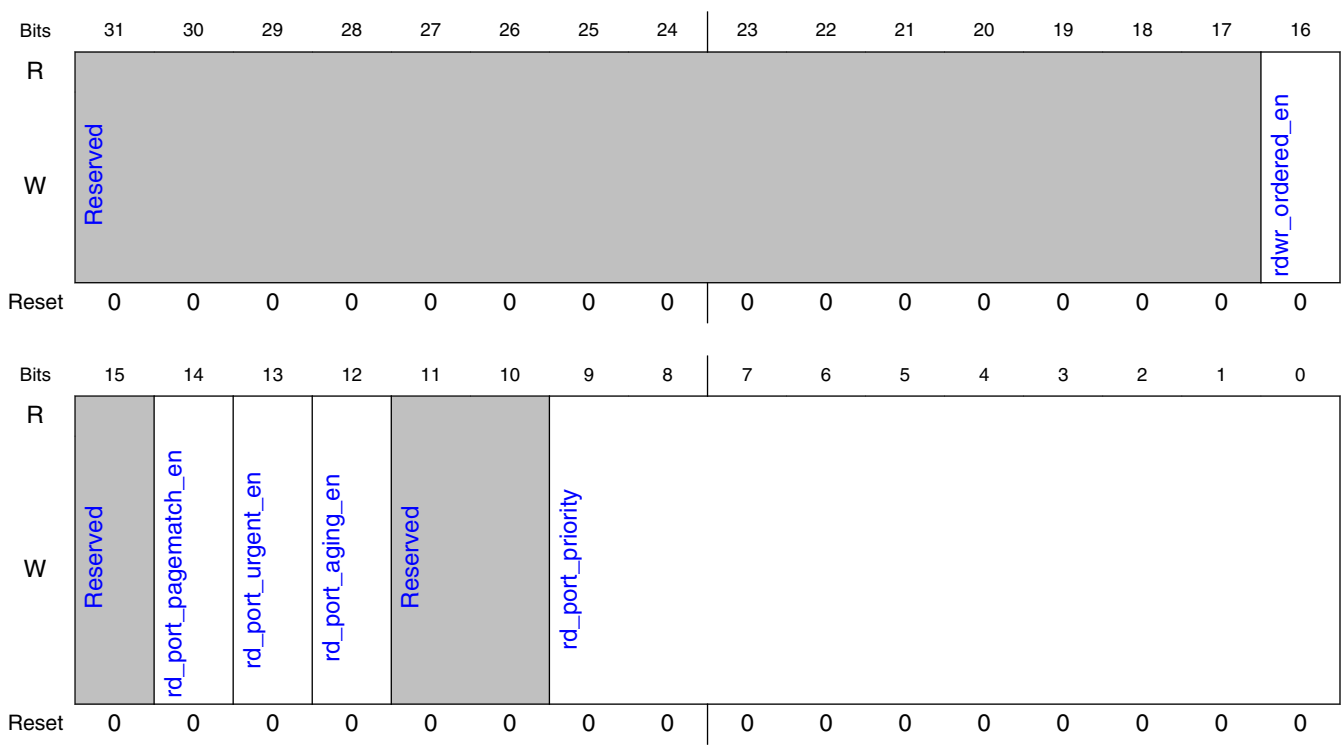
Field	Function
4 pagematch_limit	Page match four limit. If set to 1, limits the number of consecutive same page DDRC transactions that can be granted by the Port Arbiter to four when Page Match feature is enabled. If set to 0, there is no limit imposed on number of consecutive same page DDRC transactions.
3-1 —	Reserved
0 go2critical_en	If set to 1 (enabled), sets co_gs_go2critical_wr and co_gs_go2critical_lpr/co_gs_go2critical_hpr signals going to DDRC based on urgent input (awurgent, arurgent) coming from AXI master. If set to 0 (disabled), co_gs_go2critical_wr and co_gs_go2critical_lpr/co_gs_go2critical_hpr signals at DDRC are driven to 1b'0.

9.2.3.1.91 Port n Configuration Read Register (PCFGR_0)

9.2.3.1.91.1 Offset

Register	Offset
PCFGR_0	404h

9.2.3.1.91.2 Diagram



9.2.3.1.91.3 Fields

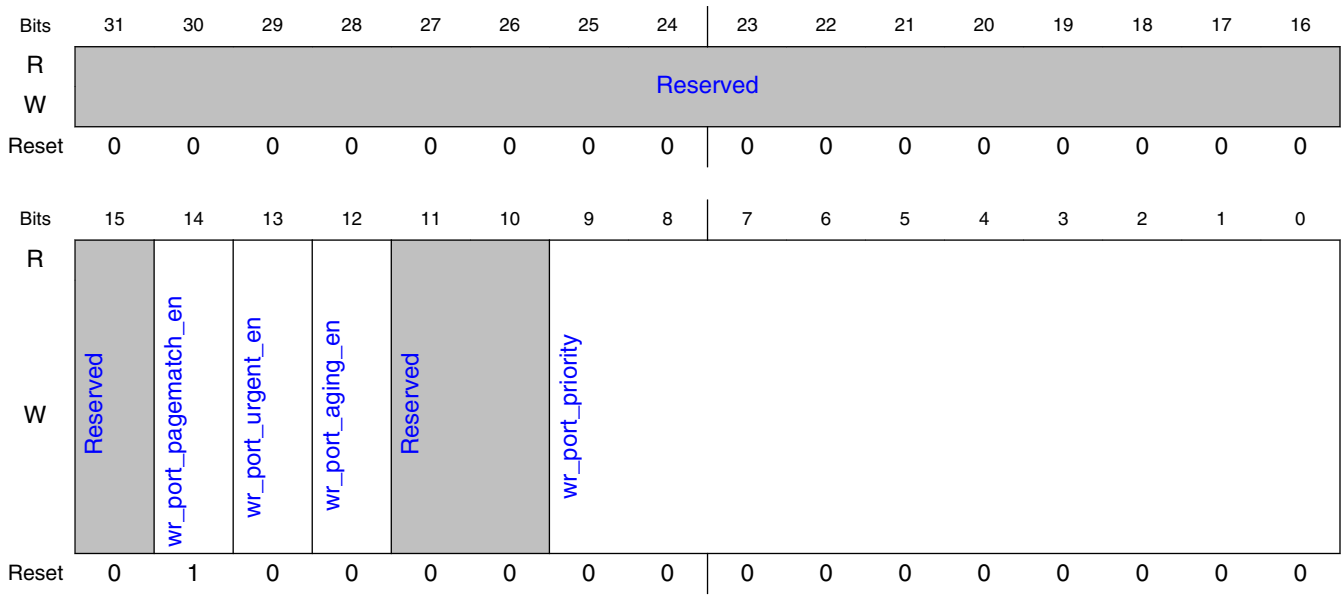
Field	Function
31-17 —	Reserved
16 rdwr_ordered_en	Enable ordered read/writes. If set to 1, preserves the ordering between read transaction and write transaction issued to the same address, on a given port. In other words, the controller ensures that all same address read and write commands from the application port interface are transported to the DFI interface in the order of acceptance. This feature is useful in cases where software coherency is desired for masters issuing back-to-back read/write transactions without waiting for write/read responses. Note that this register has an effect only if necessary logic is instantiated via the DDRC_RDWR_ORDERED_n parameter.
15 —	Reserved
14 rd_port_pagematch_en	If set to 1, enables the Page Match feature. If enabled, once a requesting port is granted, the port is continued to be granted if the following immediate commands are to the same memory page (same bank and same row). See also related PCCFG.pagematch_limit register.
13 rd_port_urgent_en	If set to 1, enables the AXI urgent sideband signal (arurgent). When enabled and arurgent is asserted by the master, that port becomes the highest priority and co_gs_go2critical_lpr/co_gs_go2critical_hpr signal to DDRC is asserted if enabled in PCCFG.go2critical_en register. Note that arurgent signal can be asserted anytime and as long as required which is independent of address handshaking (it is not associated with any particular command).
12 rd_port_aging_en	If set to 1, enables aging function for the read channel of the port.
11-10 —	Reserved
9-0 rd_port_priority	Determines the initial load value of read aging counters. These counters will be parallel loaded after reset, or after each grant to the corresponding port. The aging counters down-count every clock cycle where the port is requesting but not granted. The higher significant 5-bits of the read aging counter sets the priority of the read channel of a given port. Port's priority will increase as the higher significant 5-bits of the counter starts to decrease. When the aging counter becomes 0, the corresponding port channel will have the highest priority level (timeout condition - Priority0). For multi-port configurations, the aging counters cannot be used to set port priorities when external dynamic priority inputs (arqos) are enabled (timeout is still applicable). For single port configurations, the aging counters are only used when they timeout (become 0) to force read-write direction switching. In this case, external dynamic priority input, arqos (for reads only) can still be used to set the DDRC read priority (2 priority levels: low priority read - LPR, high priority read - HPR) on a command by command basis. Note: The two LSBs of this register field are tied internally to 2'b00.

9.2.3.1.92 Port n Configuration Write Register (PCFGW_0)

9.2.3.1.92.1 Offset

Register	Offset
PCFGW_0	408h

9.2.3.1.92.2 Diagram



9.2.3.1.92.3 Fields

Field	Function
31-15 —	Reserved
14 wr_port_pagematch_en	If set to 1, enables the Page Match feature. If enabled, once a requesting port is granted, the port is continued to be granted if the following immediate commands are to the same memory page (same bank and same row). See also related PCCFG.pagematch_limit register.
13 wr_port_urgent_en	If set to 1, enables the AXI urgent sideband signal (awurgent). When enabled and awurgent is asserted by the master, that port becomes the highest priority and co_gs_go2critical_wr signal to DDRC is asserted if enabled in PCCFG.go2critical_en register. Note that awurgent signal can be asserted anytime and as long as required which is independent of address handshaking (it is not associated with any particular command).
12 wr_port_aging_en	If set to 1, enables aging function for the write channel of the port.
11-10 —	Reserved
9-0 wr_port_priority	Determines the initial load value of write aging counters. These counters will be parallel loaded after reset, or after each grant to the corresponding port. The aging counters down-count every clock cycle where the port is requesting but not granted. The higher significant 5-bits of the write aging counter sets the initial priority of the write channel of a given port. Port's priority will increase as the higher significant 5-bits of the counter starts to decrease. When the aging counter becomes 0, the corresponding port channel will have the highest priority level. For multi-port configurations, the aging counters cannot be used to set port priorities when external dynamic priority inputs (awqos) are enabled (timeout is still applicable). For single port configurations, the aging counters are only used when they timeout (become

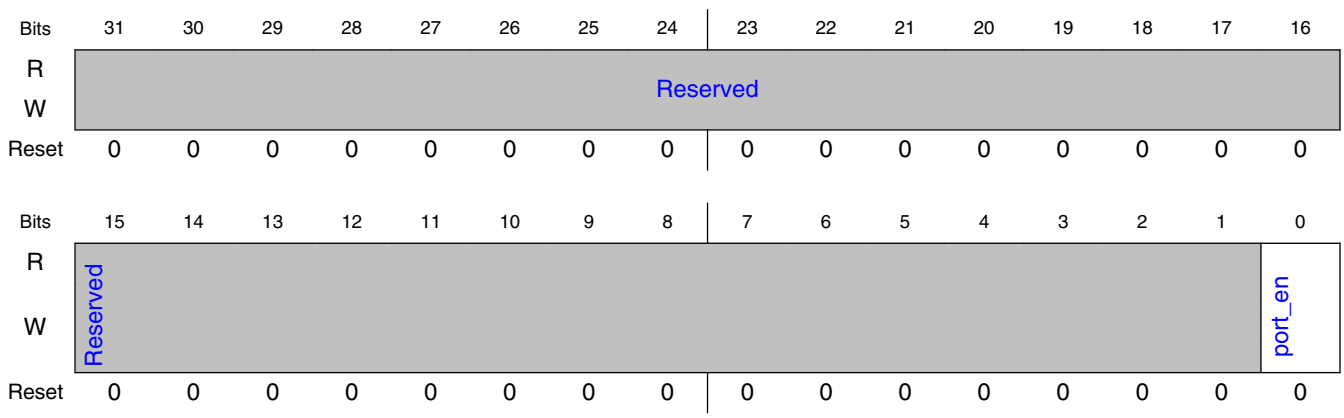
Field	Function
	0) to force read-write direction switching. Note: The two LSBs of this register field are tied internally to 2'b00.

9.2.3.1.93 Port n Control Register (PCTRL_0)

9.2.3.1.93.1 Offset

Register	Offset
PCTRL_0	490h

9.2.3.1.93.2 Diagram



9.2.3.1.93.3 Fields

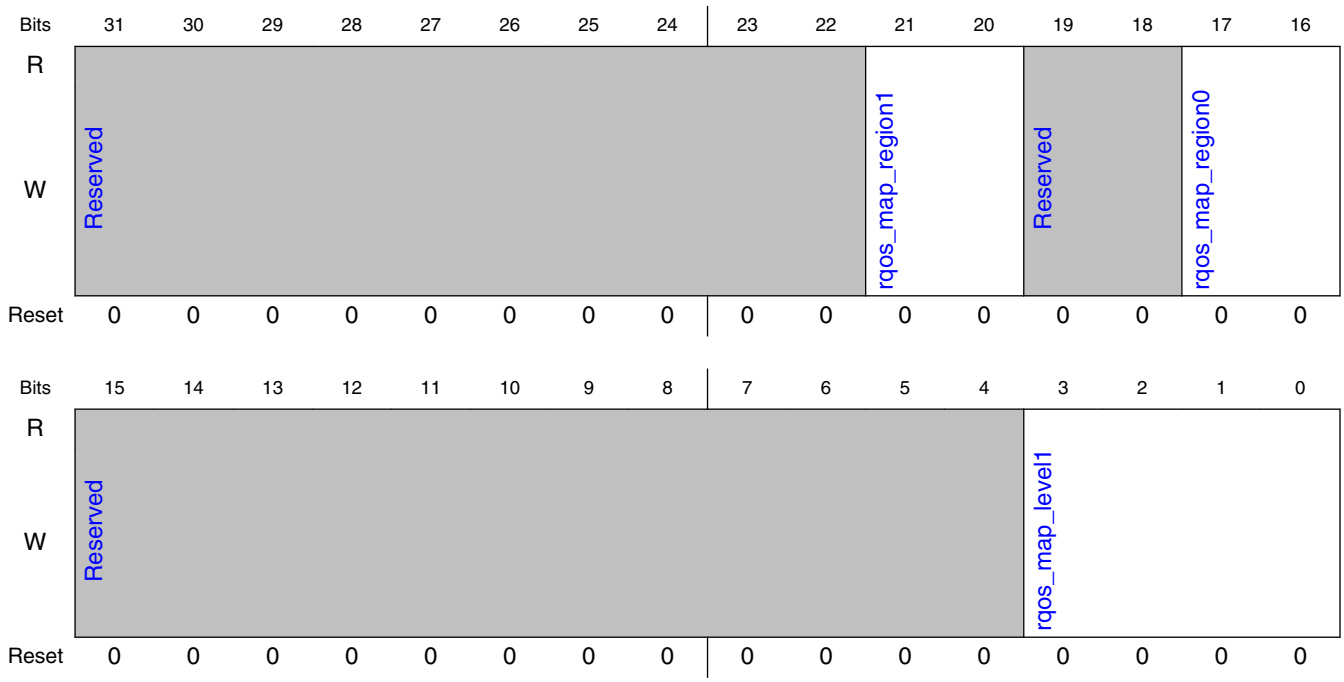
Field	Function
31-1 —	Reserved
0 port_en	Enables AXI port n.

9.2.3.1.94 Port n Read QoS Configuration Register 0 (PCFGQOS0_0)

9.2.3.1.94.1 Offset

Register	Offset
PCFGQOS0_0	494h

9.2.3.1.94.2 Diagram



9.2.3.1.94.3 Fields

Field	Function
31-22 —	Reserved
21-20 rqos_map_region1	This bitfield indicates the traffic class of region 1. Valid values are: 0 : LPR, 1: VPR, 2: HPR. For dual address queue configurations, region1 maps to the blue address queue. In this case, valid values are 0: LPR and 1: VPR only. When VPR support is disabled (DDRC_VPR_EN = 0) and traffic class of region 1 is set to 1 (VPR), VPR traffic is aliased to LPR traffic.
19-18 —	Reserved
17-16 rqos_map_region0	This bitfield indicates the traffic class of region 0. Valid values are: 0: LPR, 1: VPR, 2: HPR. For dual address queue configurations, region 0 maps to the blue address queue. In this case, valid values are: 0: LPR and 1: VPR only. When VPR support is disabled (DDRC_VPR_EN = 0) and traffic class of region0 is set to 1 (VPR), VPR traffic is aliased to LPR traffic.
15-4 —	Reserved

Table continues on the next page...

Field	Function
3-0 rqos_map_level1	Separation level1 indicating the end of region0 mapping; start of region0 is 0. Possible values for level1 are 0 to 13 (for dual RAQ) or 0 to 14 (for single RAQ) which corresponds to arqos. Note that for PA, arqos values are used directly as port priorities, where the higher the value corresponds to higher port priority. All of the map_level* registers must be set to distinct values.

9.2.3.1.95 Port n Read QoS Configuration Register 1 (PCFGQOS1_0)

9.2.3.1.95.1 Offset

Register	Offset
PCFGQOS1_0	498h

9.2.3.1.95.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved					rqos_map_timeoutr										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved					rqos_map_timeoutb										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.95.3 Fields

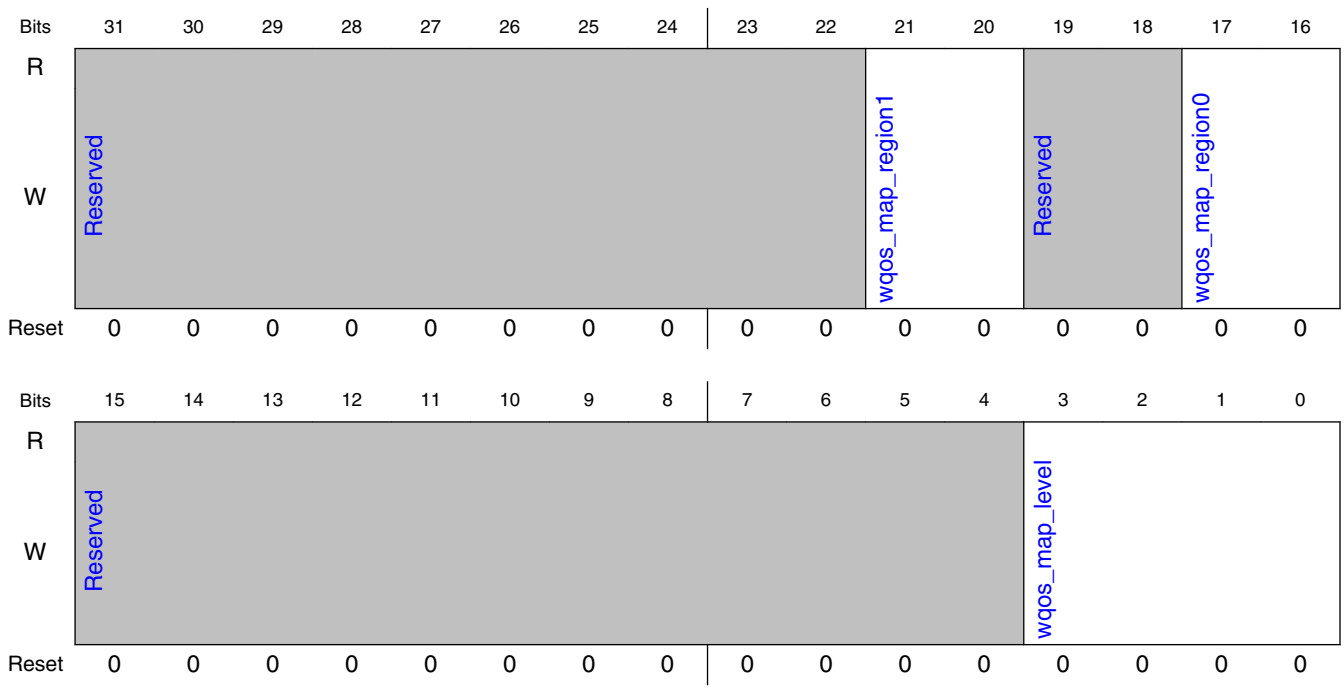
Field	Function
31-27 —	Reserved
26-16 rqos_map_timeoutr	Specifies the timeout value for transactions mapped to the red address queue.
15-11 —	Reserved
10-0 rqos_map_timeoutb	Specifies the timeout value for transactions mapped to the blue address queue.

9.2.3.1.96 Port n Write QoS Configuration Register 0 (PCFGWQOS0_0)

9.2.3.1.96.1 Offset

Register	Offset
PCFGWQOS0_0	49Ch

9.2.3.1.96.2 Diagram



9.2.3.1.96.3 Fields

Field	Function
31-22 —	Reserved
21-20 wqos_map_region1	This bitfield indicates the traffic class of region 1. Valid values are: 0: NPW, 1: VPW. When VPW support is disabled (DDRC_VPW_EN = 0) and traffic class of region 1 is set to 1 (VPW), VPW traffic is aliased to LPW traffic.
19-18 —	Reserved

Table continues on the next page...

Field	Function
17-16 wqos_map_region0	This bitfield indicates the traffic class of region 0. Valid values are: 0: NPW, 1: VPW. When VPW support is disabled (DDRC_VPW_EN = 0) and traffic class of region0 is set to 1 (VPW), VPW traffic is aliased to NPW traffic.
15-4 —	Reserved
3-0 wqos_map_level	Separation level indicating the end of region0 mapping; start of region0 is 0. Possible values for level1 are 0 to 14 which corresponds to awqos. Note that for PA, awqos values are used directly as port priorities, where the higher the value corresponds to higher port priority.

9.2.3.1.97 Port n Write QoS Configuration Register 1 (PCFGWQOS1_0)

9.2.3.1.97.1 Offset

Register	Offset
PCFGWQOS1_0	4A0h

9.2.3.1.97.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved					wqos_map_timeout										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.97.3 Fields

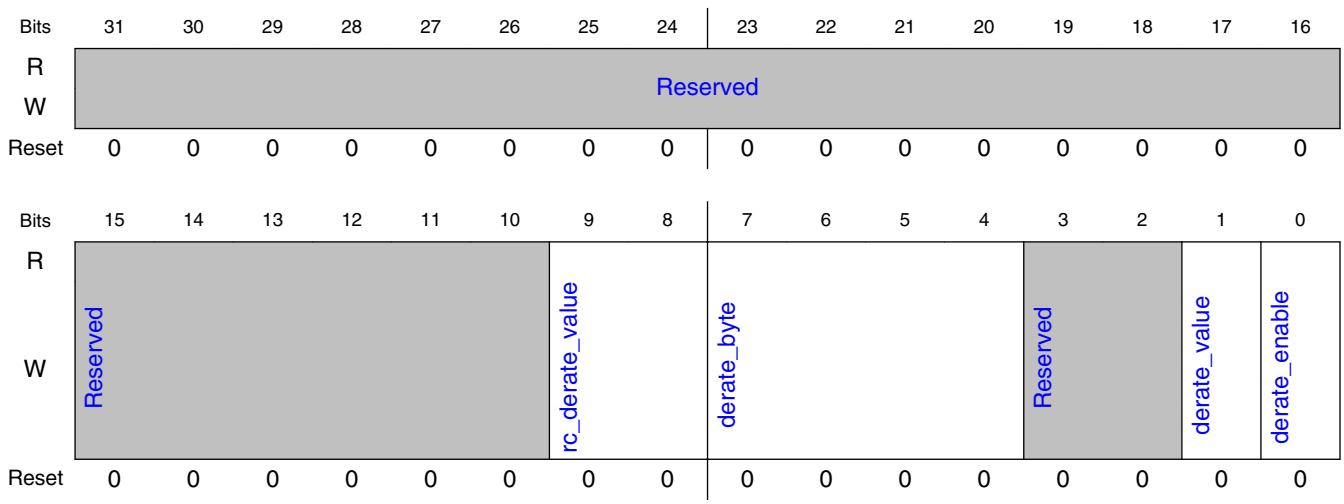
Field	Function
31-11 —	Reserved
10-0 wqos_map_timeout	Specifies the timeout value for write transactions.

9.2.3.1.98 [SHADOW] Temperature Derate Enable Register (DERATEEN_SHADOW)

9.2.3.1.98.1 Offset

Register	Offset
DERATEEN_SHADOW	2020h

9.2.3.1.98.2 Diagram



9.2.3.1.98.3 Fields

Field	Function
31-10 —	Reserved
9-8 rc_derate_value	Derate value of tRC for LPDDR4 - 0 - Derating uses +1. - 1 - Derating uses +2. - 2 - Derating uses +3. - 3 - Derating uses +4. Present only in designs configured to support LPDDR4. The required number of cycles for derating can be determined by dividing 3.75ns by the core_ddrc_core_clk period, and rounding up the next integer.
7-4 derate_byte	Derate byte Present only in designs configured to support LPDDR2/LPDDR3/LPDDR4 Indicates which byte of the MRR data is used for derating. The maximum valid value depends on MEMC_DRAM_TOTAL_DATA_WIDTH.
3-2 —	Reserved
1 derate_value	Derate value - 0 - Derating uses +1. - 1 - Derating uses +2. Present only in designs configured to support LPDDR2/LPDDR3/LPDDR4 Set to 0 for all LPDDR2 speed grades as derating value of +1.875 ns is less than a core_ddrc_core_clk period. For LPDDR3/4, if the period of core_ddrc_core_clk is less than 1.875ns, this register field should be set to 1; otherwise it should be set to 0.

Table continues on the next page...

Field	Function
0 derate_enable	Enables derating - 0 - Timing parameter derating is disabled - 1 - Timing parameter derating is enabled using MR4 read value. Present only in designs configured to support LPDDR2/LPDDR3/LPDDR4 This field must be set to '0' for non-LPDDR2/LPDDR3/LPDDR4 mode.

9.2.3.1.99 [SHADOW] Temperature Derate Interval Register (DERATEINT_SHADOW)

9.2.3.1.99.1 Offset

Register	Offset
DERATEINT_SHADOW	2024h

9.2.3.1.99.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	mr4_read_interval															
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	mr4_read_interval															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.99.3 Fields

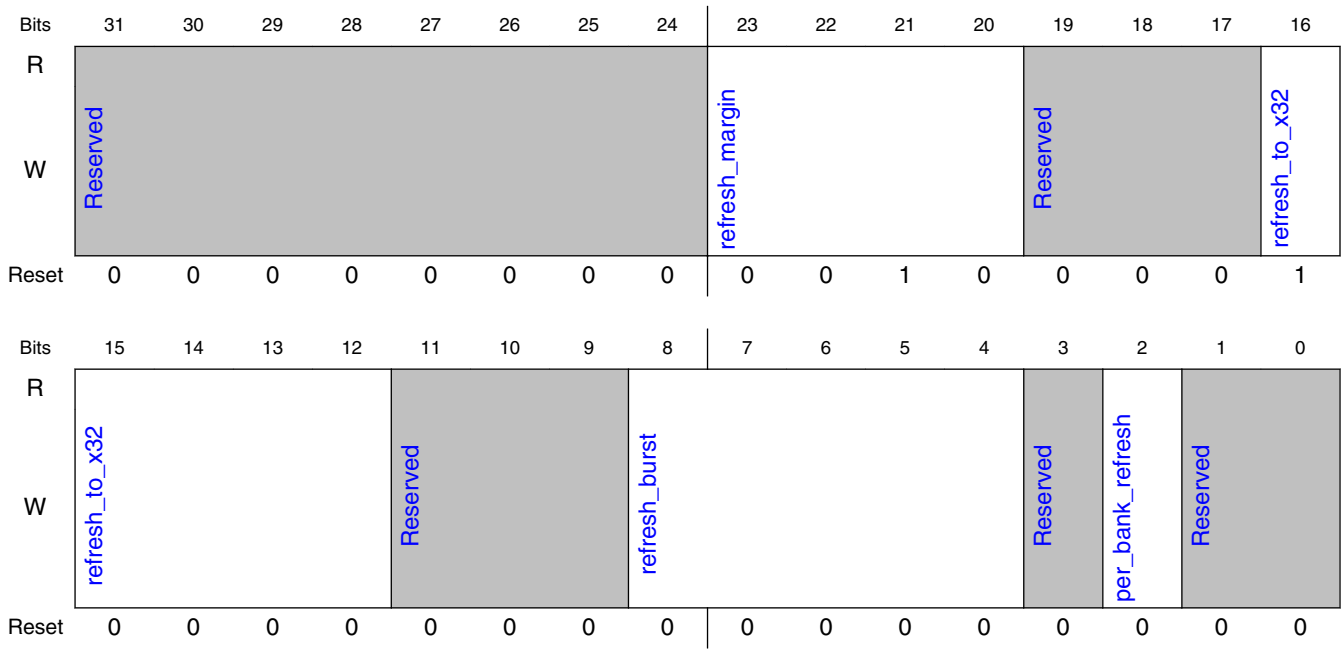
Field	Function
31-0 mr4_read_interval	Interval between two MR4 reads, used to derate the timing parameters. Present only in designs configured to support LPDDR2/LPDDR3/LPDDR4. This register must not be set to zero. Unit: DFI clock cycle.

9.2.3.1.100 [SHADOW] Refresh Control Register 0 (RFSHCTL0_SHADOW)

9.2.3.1.100.1 Offset

Register	Offset
RFSHCTL0_SHADOW	2050h

9.2.3.1.100.2 Diagram



9.2.3.1.100.3 Fields

Field	Function
31-24 —	Reserved
23-20 refresh_margin	Threshold value in number of DFI clock cycles before the critical refresh or page timer expires. A critical refresh is to be issued before this threshold is reached. It is recommended that this not be changed from the default value, currently shown as 0x2. It must always be less than internally used t_rfc_nom_x32. Note that, in LPDDR2/LPDDR3/LPDDR4, internally used t_rfc_nom_x32 may be equal to RFSHTMG.t_rfc_nom_x32>>2 if derating is enabled (DERATEEN.derate_enable=1). Otherwise, internally used t_rfc_nom_x32 will be equal to RFSHTMG.t_rfc_nom_x32. Unit: Multiples of 32 DFI clocks.
19-17 —	Reserved
16-12 refresh_to_x32	If the refresh timer (tRFCnom, also known as tREFI) has expired at least once, but it has not expired (RFSHCTL0.refresh_burst+1) times yet, then a speculative refresh may be performed. A speculative refresh is a refresh performed at a time when refresh would be useful, but before it is absolutely required. When the SDRAM bus is idle for a period of time determined by this RFSHCTL0.refresh_to_x32 and the refresh timer has expired at least once since the last refresh, then a speculative refresh is performed.

Table continues on the next page...

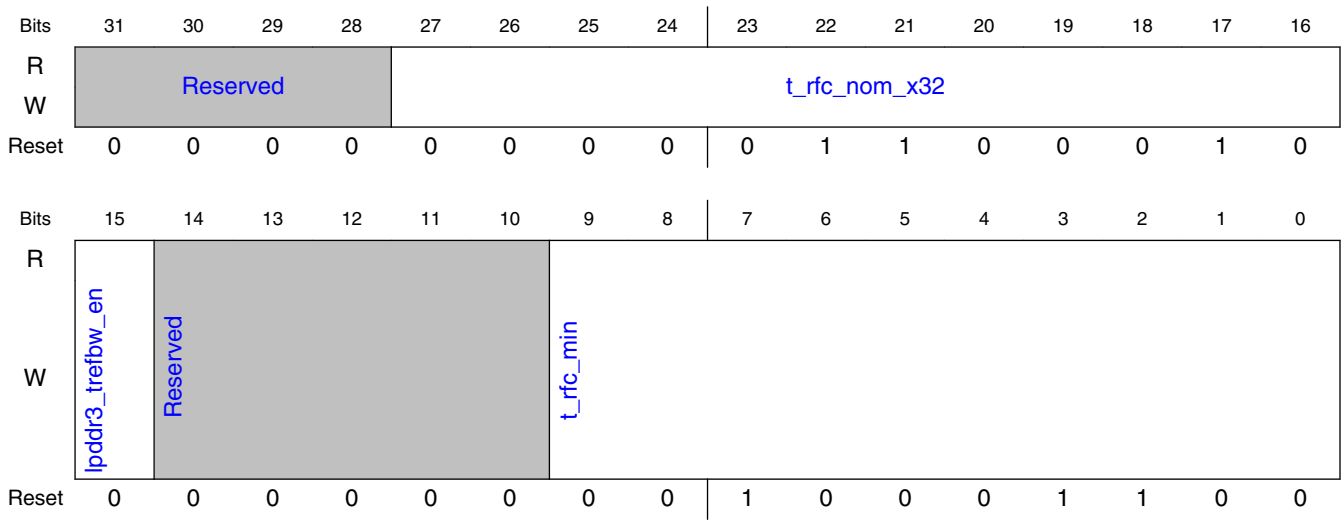
Field	Function
	Speculative refreshes continues successively until there are no refreshes pending or until new reads or writes are issued to the DDRC. FOR PERFORMANCE ONLY. Unit: Multiples of 32 DFI clocks.
11-9 —	Reserved
8-4 refresh_burst	The programmed value + 1 is the number of refresh timeouts that is allowed to accumulate before traffic is blocked and the refreshes are forced to execute. Closing pages to perform a refresh is a one-time penalty that must be paid for each group of refreshes. Therefore, performing refreshes in a burst reduces the per-refresh penalty of these page closings. Higher numbers for RFSHCTL.refresh_burst slightly increases utilization; lower numbers decreases the worst-case latency associated with refreshes. - 0 - single refresh - 1 - burst-of-2 refresh - 7 - burst-of-8 refresh For information on burst refresh feature refer to section 3.9 of DDR2 JEDEC specification - JESD79-2F.pdf. For DDR2/3, the refresh is always per-rank and not per-bank. The rank refresh can be accumulated over 8*tREFI cycles using the burst refresh feature. In DDR4 mode, according to Fine Granularity feature, 8 refreshes can be postponed in 1X mode, 16 refreshes in 2X mode and 32 refreshes in 4X mode. If using PHY-initiated updates, care must be taken in the setting of RFSHCTL0.refresh_burst, to ensure that tRFCmax is not violated due to a PHY-initiated update occurring shortly before a refresh burst was due. In this situation, the refresh burst will be delayed until the PHY-initiated update is complete.
3 —	Reserved
2 per_bank_refresh	- 1 - Per bank refresh; - 0 - All bank refresh. Per bank refresh allows traffic to flow to other banks. Per bank refresh is not supported by all LPDDR2 devices but should be supported by all LPDDR3/LPDDR4 devices. Present only in designs configured to support LPDDR2/LPDDR3/LPDDR4
1-0 —	Reserved

9.2.3.1.101 [SHADOW] Refresh Timing Register (RFSHTMG_SHADOW)

9.2.3.1.101.1 Offset

Register	Offset
RFSHTMG_SHADOW	2064h

9.2.3.1.101.2 Diagram



9.2.3.1.101.3 Fields

Field	Function
31-28 —	Reserved
27-16 t_rfc_nom_x32	tREFI: Average time interval between refreshes per rank (Specification: 7.8us for DDR2, DDR3 and DDR4. See JEDEC specification for mDDR, LPDDR2, LPDDR3 and LPDDR4). For LPDDR2/LPDDR3/LPDDR4: - if using all-bank refreshes (RFSHCTL0.per_bank_refresh = 0), this register should be set to tREFIab - if using per-bank refreshes (RFSHCTL0.per_bank_refresh = 1), this register should be set to tREFIpb When the controller is operating in 1:2 frequency ratio mode, program this to (tREFI/2), no rounding up. In DDR4 mode, tREFI value is different depending on the refresh mode. The user should program the appropriate value from the spec based on the value programmed in the refresh mode register. Note that RFSHTMG.t_rfc_nom_x32 * 32 must be greater than RFSHTMG.t_rfc_min, and RFSHTMG.t_rfc_nom_x32 must be greater than 0x1. - Non-DDR4 or DDR4 Fixed 1x mode: RFSHTMG.t_rfc_nom_x32 must be less than or equal to 0xFFE. - DDR4 Fixed 2x mode: RFSHTMG.t_rfc_nom_x32 must be less than or equal to 0x7FF. - DDR4 Fixed 4x mode: RFSHTMG.t_rfc_nom_x32 must be less than or equal to 0x3FF. Unit: Multiples of 32 clocks.
15 lpddr3_trefbw_en	Used only when LPDDR3 memory type is connected. Should only be changed when DDRC is in reset. Specifies whether to use the tREFBW parameter (required by some LPDDR3 devices which comply with earlier versions of the LPDDR3 JEDEC specification) or not: - 0 - tREFBW parameter not used - 1 - tREFBW parameter used
14-10 —	Reserved
9-0 t_rfc_min	tRFC (min): Minimum time from refresh to refresh or activate. When the controller is operating in 1:1 mode, t_rfc_min should be set to RoundUp(tRFCmin/tCK). When the controller is operating in 1:2 mode, t_rfc_min should be set to RoundUp(RoundUp(tRFCmin/tCK)/2). In LPDDR2/LPDDR3/LPDDR4 mode: - if using all-bank refreshes, the tRFCmin value in the above equations is equal to tRFCab - if using per-bank refreshes, the tRFCmin value in the above equations is equal to tRFCpb In DDR4 mode, the tRFCmin value in the above equations is different depending on the refresh mode (fixed 1X,2X,4X) and the device density. The user should program the appropriate value from the spec based on the 'refresh_mode' and the device density that is used. Unit: Clocks.

9.2.3.1.102 [SHADOW] SDRAM Initialization Register 3 (INIT3_SHADOW)

9.2.3.1.102.1 Offset

Register	Offset
INIT3_SHADOW	20DCh

9.2.3.1.102.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	mr															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	emr															
Reset	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0

9.2.3.1.102.3 Fields

Field	Function
31-16 mr	DDR2: Value to write to MR register. Bit 8 is for DLL and the setting here is ignored. The DDRC sets this bit appropriately. DDR3/DDR4: Value loaded into MR0 register. mDDR: Value to write to MR register. LPDDR2/LPDDR3/LPDDR4 - Value to write to MR1 register
15-0 emr	DDR2: Value to write to EMR register. Bits 9:7 are for OCD and the setting in this register is ignored. The DDRC sets those bits appropriately. DDR3/DDR4: Value to write to MR1 register Set bit 7 to 0. If PHY-evaluation mode training is enabled, this bit is set appropriately by the DDRC during write leveling. mDDR: Value to write to EMR register. LPDDR2/LPDDR3/LPDDR4 - Value to write to MR2 register

9.2.3.1.103 [SHADOW] SDRAM Initialization Register 4 (INIT4_SHADOW)

9.2.3.1.103.1 Offset

Register	Offset
INIT4_SHADOW	20E0h

9.2.3.1.103.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	emr2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	emr3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.103.3 Fields

Field	Function
31-16 emr2	DDR2: Value to write to EMR2 register. DDR3/DDR4: Value to write to MR2 register LPDDR2/LPDDR3/ LPDDR4: Value to write to MR3 register mDDR: Unused
15-0 emr3	DDR2: Value to write to EMR3 register. DDR3/DDR4: Value to write to MR3 register mDDR/LPDDR2/ LPDDR3: Unused LPDDR4: Value to write to MR13 register

9.2.3.1.104 [SHADOW] SDRAM Initialization Register 6 (INIT6_SHADOW)

9.2.3.1.104.1 Offset

Register	Offset
INIT6_SHADOW	20E8h

9.2.3.1.104.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	mr4															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	mr5															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.2.3.1.104.3 Fields

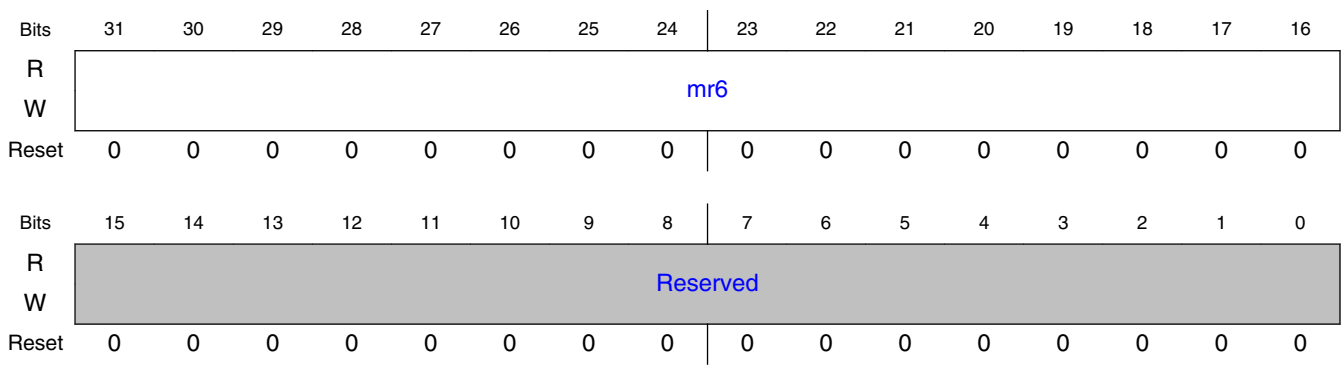
Field	Function
31-16 mr4	DDR4- Value to be loaded into SDRAM MR4 registers. Used in DDR4 designs only.
15-0 mr5	DDR4- Value to be loaded into SDRAM MR5 registers. Used in DDR4 designs only.

9.2.3.1.105 [SHADOW] SDRAM Initialization Register 7 (INIT7_SHADOW)

9.2.3.1.105.1 Offset

Register	Offset
INIT7_SHADOW	20ECh

9.2.3.1.105.2 Diagram



9.2.3.1.105.3 Fields

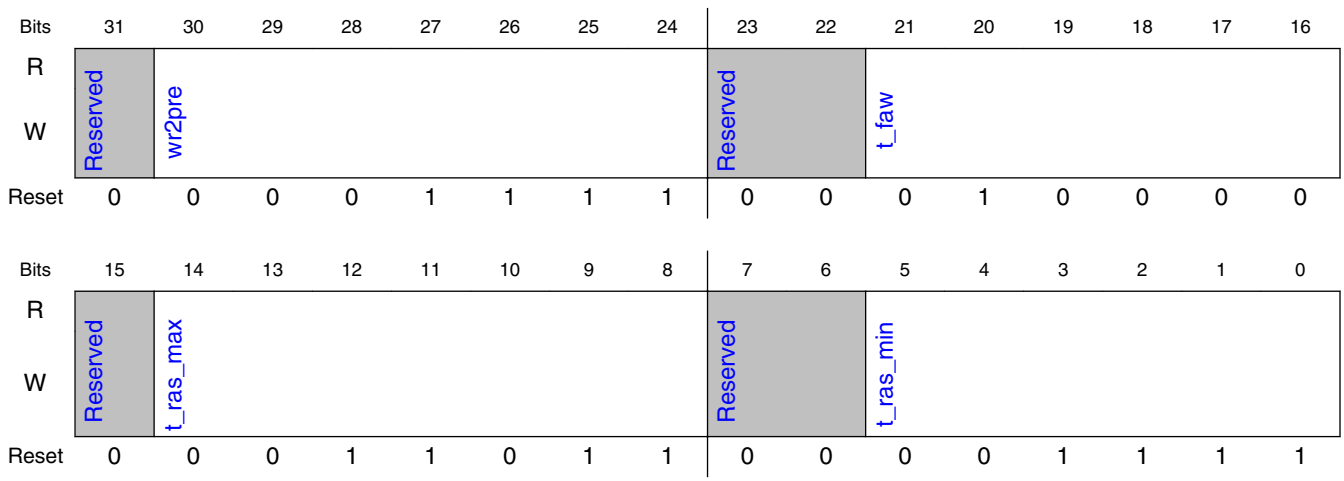
Field	Function
31-16 mr6	DDR4- Value to be loaded into SDRAM MR6 registers. Used in DDR4 designs only.
15-0 —	Reserved

9.2.3.1.106 [SHADOW] SDRAM Timing Register 0 (DRAMTMG0_SHADOW)

9.2.3.1.106.1 Offset

Register	Offset
DRAMTMG0_SHADOW	2100h

9.2.3.1.106.2 Diagram



9.2.3.1.106.3 Fields

Field	Function
31 —	Reserved
30-24 wr2pre	Minimum time between write and precharge to same bank. Unit: Clocks Specifications: WL + BL/2 + tWR = approximately 8 cycles + 15 ns = 14 clocks @400MHz and less for lower frequencies where: - WL = write latency - BL = burst length. This must match the value programmed in the BL bit of the mode register to the SDRAM. BST (burst terminate) is not supported at present. - tWR = Write recovery time. This comes directly from the SDRAM specification. Add one extra cycle for LPDDR2/LPDDR3/LPDDR4 for this parameter. When the controller is operating in 1:2 frequency ratio mode, 1T mode, divide the above value by 2. No rounding up. When the controller is operating in 1:2 frequency ratio mode, 2T mode or LPDDR4 mode, divide the above value by 2 and round it up to the next integer value. Note that, depending on the PHY, if using LRDIMM, it may be necessary to adjust the value of this parameter to compensate for the extra cycle of latency through the LRDIMM.
23-22 —	Reserved
21-16 t_faw	tFAW Valid only when 8 or more banks(or banks x bank groups) are present. In 8-bank design, at most 4 banks must be activated in a rolling window of tFAW cycles. When the controller is operating in 1:2 frequency ratio mode, program this to (tFAW/2) and round up to next integer value. In a 4-bank design, set this register to 0x1 independent of the 1:1/1:2 frequency mode. Unit: Clocks

Table continues on the next page...

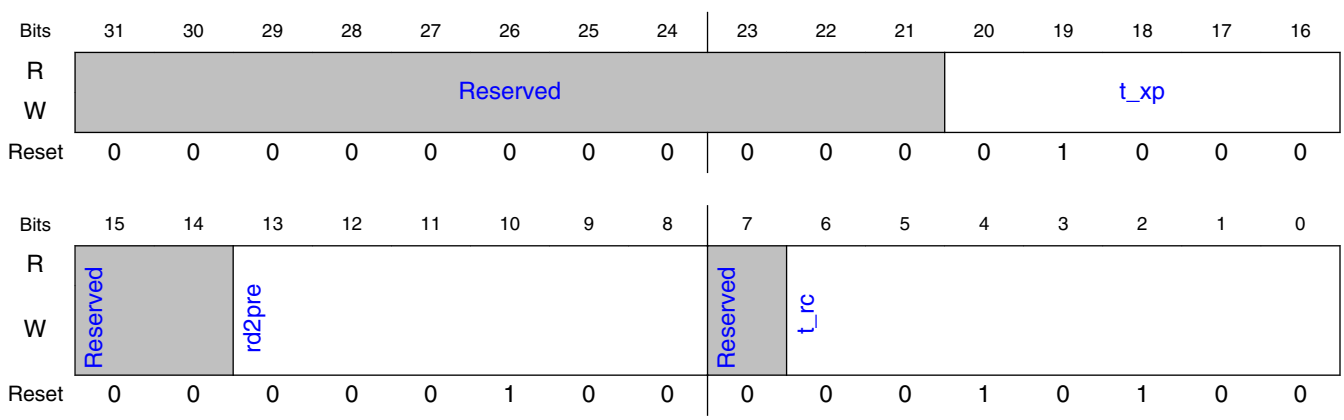
Field	Function
15 —	Reserved
14-8 t_ras_max	tRAS(max): Maximum time between activate and precharge to same bank. This is the maximum time that a page can be kept open Minimum value of this register is 1. Zero is invalid. When the controller is operating in 1:2 frequency ratio mode, program this to (tRAS(max)-1)/2. No rounding up. Unit: Multiples of 1024 clocks.
7-6 —	Reserved
5-0 t_ras_min	tRAS(min): Minimum time between activate and precharge to the same bank. When the controller is operating in 1:2 frequency mode, 1T mode, program this to tRAS(min)/2. No rounding up. When the controller is operating in 1:2 frequency ratio mode, 2T mode or LPDDR4 mode, program this to (tRAS(min)/2) and round it up to the next integer value. Unit: Clocks

9.2.3.1.107 [SHADOW] SDRAM Timing Register 1 (DRAMTMG1_SHADOW)

9.2.3.1.107.1 Offset

Register	Offset
DRAMTMG1_SHADOW	2104h

9.2.3.1.107.2 Diagram



9.2.3.1.107.3 Fields

Field	Function
31-21 —	Reserved

Table continues on the next page...

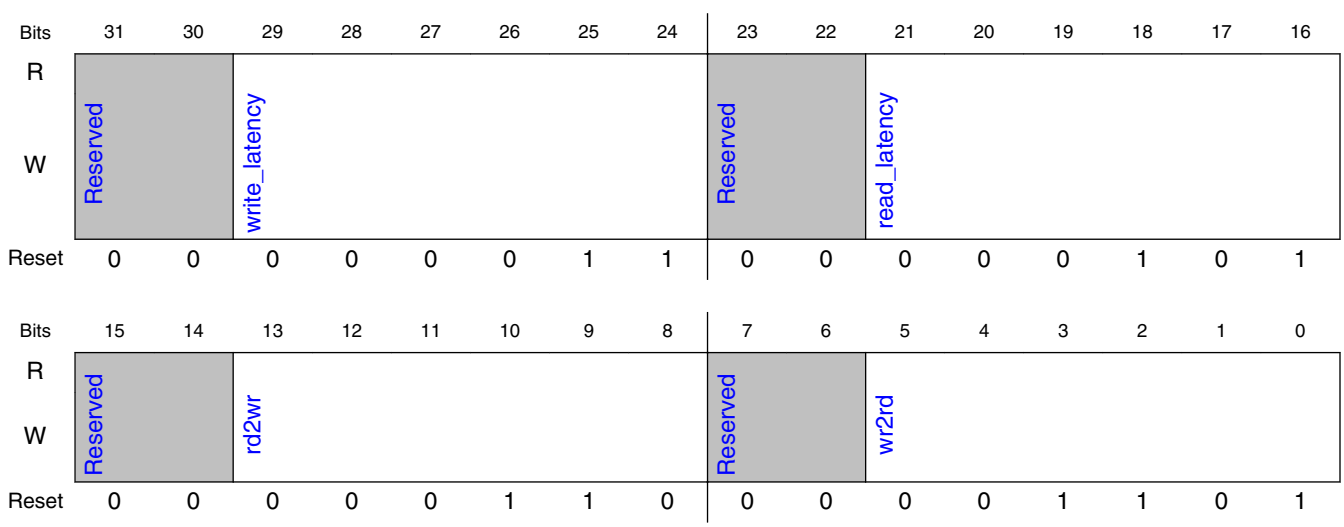
Field	Function
20-16 t_xp	tXP: Minimum time after power-down exit to any operation. For DDR3, this should be programmed to tXPDLL if slow powerdown exit is selected in MR0[12]. If C/A parity for DDR4 is used, set to (tXP+PL) instead. When the controller is operating in 1:2 frequency ratio mode, program this to (tXP/2) and round it up to the next integer value. Units: Clocks
15-14 —	Reserved
13-8 rd2pre	tRTP: Minimum time from read to precharge of same bank. - DDR2: tAL + BL/2 + max(tRTP, 2) - 2 - DDR3: tAL + max (tRTP, 4) - DDR4: Max of following two equations: tAL + max (tRTP, 4) or, RL + BL/2 - tRP (*). - mDDR: BL/2 - LPDDR2: Depends on if it's LPDDR2-S2 or LPDDR2-S4: LPDDR2-S2: BL/2 + tRTP - 1. LPDDR2-S4: BL/2 + max(tRTP,2) - 2. - LPDDR3: BL/2 + max(tRTP,4) - 4 - LPDDR4: BL/2 + max(tRTP,8) - 8 (*) When both DDR4 SDRAM and ST-MRAM are used simultaneously, use SDRAM's tRP value for calculation. When the controller is operating in 1:2 mode, 1T mode, divide the above value by 2. No rounding up. When the controller is operating in 1:2 mode, 2T mode or LPDDR4 mode, divide the above value by 2 and round it up to the next integer value. Unit: Clocks.
7 —	Reserved
6-0 t_rc	tRC: Minimum time between activates to same bank. When the controller is operating in 1:2 frequency ratio mode, program this to (tRC/2) and round up to next integer value. Unit: Clocks.

9.2.3.1.108 [SHADOW] SDRAM Timing Register 2 (DRAMTMG2_SHADOW)

9.2.3.1.108.1 Offset

Register	Offset
DRAMTMG2_SHADOW	2108h

9.2.3.1.108.2 Diagram



9.2.3.1.108.3 Fields

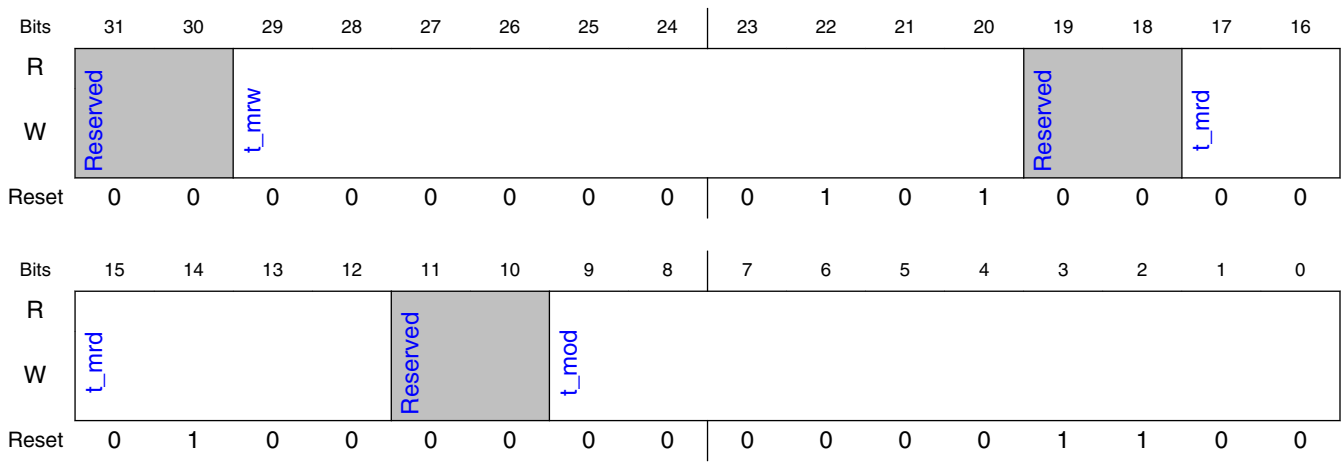
Field	Function
31-30 —	Reserved
29-24 write_latency	Set to WL Time from write command to write data on SDRAM interface. This must be set to WL. For mDDR, it should normally be set to 1. Note that, depending on the PHY, if using RDIMM/LRDIMM, it may be necessary to adjust the value of WL to compensate for the extra cycle of latency through the RDIMM/LRDIMM. When the controller is operating in 1:2 frequency ratio mode, divide the value calculated using the above equation by 2, and round it up to next integer. This register field is not required for DDR2 and DDR3 (except if MEMC_TRAINING is set), as the DFI read and write latencies defined in DFITMG0 and DFITMG1 are sufficient for those protocols Unit: clocks
23-22 —	Reserved
21-16 read_latency	Set to RL Time from read command to read data on SDRAM interface. This must be set to RL. Note that, depending on the PHY, if using RDIMM/LRDIMM, it may be necessary to adjust the value of RL to compensate for the extra cycle of latency through the RDIMM/LRDIMM. When the controller is operating in 1:2 frequency ratio mode, divide the value calculated using the above equation by 2, and round it up to next integer. This register field is not required for DDR2 and DDR3 (except if MEMC_TRAINING is set), as the DFI read and write latencies defined in DFITMG0 and DFITMG1 are sufficient for those protocols Unit: clocks
15-14 —	Reserved
13-8 rd2wr	DDR2/3/mDDR: RL + BL/2 + 2 - WL DDR4: RL + BL/2 + 1 + WR_PREAMBLE - WL LPDDR2/LPDDR3: RL + BL/2 + RU(tDQSCKmax/tCK) + 1 - WL LPDDR4(DQ ODT is Disabled): RL + BL/2 + RU(tDQSCKmax/tCK) + WR_PREAMBLE + RD_POSTAMBLE - WL LPDDR4(DQ ODT is Enabled) : RL + BL/2 + RU(tDQSCKmax/tCK) + RD_POSTAMBLE - ODTLon - RU(tODTon(min)/tCK) Minimum time from read command to write command. Include time for bus turnaround and all per-bank, per-rank, and global constraints. Please see the relevant PHY databook for details of what should be included here. Unit: Clocks. Where: - WL = write latency - BL = burst length. This must match the value programmed in the BL bit of the mode register to the SDRAM - RL = read latency = CAS latency - WR_PREAMBLE = write preamble. This is unique to DDR4 and LPDDR4. - RD_POSTAMBLE = read postamble. This is unique to LPDDR4. For LPDDR2/LPDDR3/LPDDR4, if derating is enabled (DERATEEN.derate_enable=1), derated tDQSCKmax should be used. When the controller is operating in 1:2 frequency ratio mode, divide the value calculated using the above equation by 2, and round it up to next integer. Note that, depending on the PHY, if using LRDIMM, it may be necessary to adjust the value of this parameter to compensate for the extra cycle of latency through the LRDIMM.
7-6 —	Reserved
5-0 wr2rd	DDR4: CWL + PL + BL/2 + tWTR_L Others: CWL + BL/2 + tWTR In DDR4, minimum time from write command to read command for same bank group. In others, minimum time from write command to read command. Includes time for bus turnaround, recovery times, and all per-bank, per-rank, and global constraints. Unit: Clocks. Where: - CWL = CAS write latency - PL = Parity latency - BL = burst length. This must match the value programmed in the BL bit of the mode register to the SDRAM - tWTR_L = internal write to read command delay for same bank group. This comes directly from the SDRAM specification. - tWTR = internal write to read command delay. This comes directly from the SDRAM specification. Add one extra cycle for LPDDR2/LPDDR3/LPDDR4 operation. When the controller is operating in 1:2 mode, divide the value calculated using the above equation by 2, and round it up to next integer.

9.2.3.1.109 [SHADOW] SDRAM Timing Register 3 (DRAMTMG3_SHADOW)

9.2.3.1.109.1 Offset

Register	Offset
DRAMTMG3_SHADOW	210Ch

9.2.3.1.109.2 Diagram



9.2.3.1.109.3 Fields

Field	Function
31-30 —	Reserved
29-20 t_mrw	Time to wait after a mode register write or read (MRW or MRR). Present only in designs configured to support LPDDR2, LPDDR3 or LPDDR4. LPDDR2 typically requires value of 5. LPDDR3 typically requires value of 10. LPDDR4: Set this to the larger of tMRW and tMRWCKEL. For LPDDR2, this register is used for the time from a MRW/MRR to all other commands. When the controller is operating in 1:2 frequency ratio mode, program this to the above values divided by 2 and round it up to the next integer value. For LDPDR3, this register is used for the time from a MRW/MRR to a MRW/MRR.
19-18 —	Reserved
17-12 t_mrd	tMRD: Cycles to wait after a mode register write or read. Depending on the connected SDRAM, tMRD represents: DDR2/mDDR: Time from MRS to any command DDR3/4: Time from MRS to MRS command LPDDR2: not used LPDDR3/4: Time from MRS to non-MRS command. When the controller is operating in 1:2 frequency ratio mode, program this to (tMRD/2) and round it up to the next integer value. If C/A parity for DDR4 is used, set to tMRD_PAR(tMOD+PL) instead.
11-10 —	Reserved

Table continues on the next page...

Field	Function
9-0 t_mod	tMOD: Parameter used only in DDR3 and DDR4. Cycles between load mode command and following non-load mode command. If C/A parity for DDR4 is used, set to tMOD_PAR(tMOD+PL) instead. Set to tMOD if controller is operating in 1:1 frequency ratio mode, or tMOD/2 (rounded up to next integer) if controller is operating in 1:2 frequency ratio mode. Note that if using RDIMM/LRDIMM, depending on the PHY, it may be necessary to adjust the value of this parameter to compensate for the extra cycle of latency applied to mode register writes by the RDIMM/LRDIMM chip. Also note that if using LRDIMM, the minimum value of this register is tMRD_L2 if controller is operating in 1:1 frequency ratio mode, or tMRD_L2/2 (rounded up to next integer) if controller is operating in 1:2 frequency ratio mode.

9.2.3.1.110 [SHADOW] SDRAM Timing Register 4 (DRAMTMG4_SHADOW)

9.2.3.1.110.1 Offset

Register	Offset
DRAMTMG4_SHADOW	2110h

9.2.3.1.110.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			t_rcd					Reserved				t_ccd			
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				t_rrd				Reserved				t_rp			
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1

9.2.3.1.110.3 Fields

Field	Function
31-29 —	Reserved
28-24 t_rcd	tRCD - tAL: Minimum time from activate to read or write command to same bank. When the controller is operating in 1:2 frequency ratio mode, program this to ((tRCD - tAL)/2) and round it up to the next integer value. Minimum value allowed for this register is 1, which implies minimum (tRCD - tAL) value to be 2 when the controller is operating in 1:2 frequency ratio mode. Unit: Clocks.
23-20 —	Reserved

Table continues on the next page...

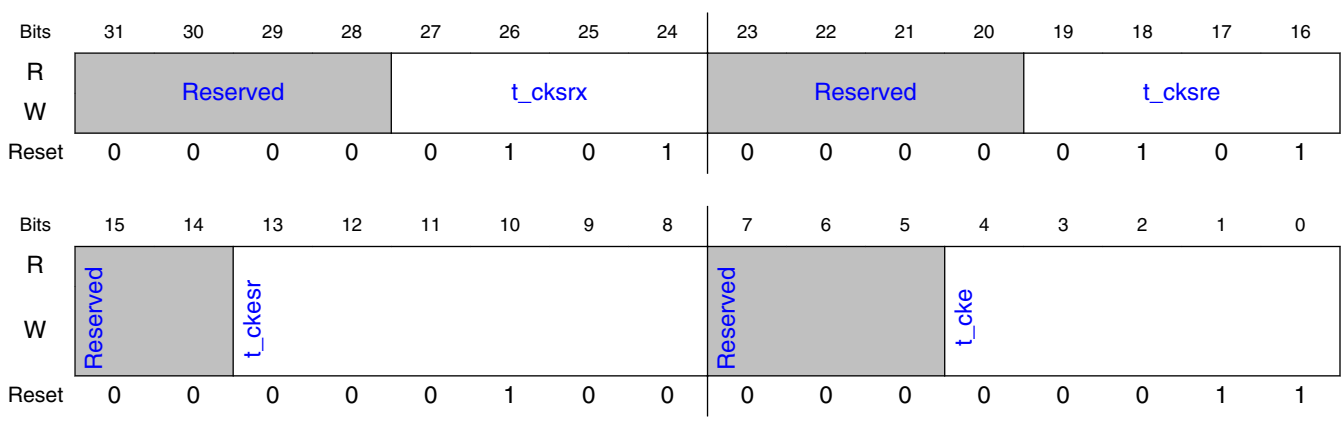
Field	Function
19-16 t_ccd	DDR4: tCCD_L: This is the minimum time between two reads or two writes for same bank group. Others: tCCD: This is the minimum time between two reads or two writes. When the controller is operating in 1:2 frequency ratio mode, program this to (tCCD_L/2 or tCCD/2) and round it up to the next integer value. Unit: clocks.
15-12 —	Reserved
11-8 t_rrd	DDR4: tRRD_L: Minimum time between activates from bank "a" to bank "b" for same bank group. Others: tRRD: Minimum time between activates from bank "a" to bank "b" When the controller is operating in 1:2 frequency ratio mode, program this to (tRRD_L/2 or tRRD/2) and round it up to the next integer value. Unit: Clocks.
7-5 —	Reserved
4-0 t_rp	tRP: Minimum time from precharge to activate of same bank. When the controller is operating in 1:1 frequency ratio mode, t_rp should be set to RoundUp(tRP/tCK). When the controller is operating in 1:2 frequency ratio mode, t_rp should be set to RoundDown(RoundUp(tRP/tCK)/2) + 1. When the controller is operating in 1:2 frequency ratio mode in LPDDR4, t_rp should be set to RoundUp(RoundUp(tRP/tCK)/2). Unit: Clocks.

9.2.3.1.111 [SHADOW] SDRAM Timing Register 5 (DRAMTMG5_SHADOW)

9.2.3.1.111.1 Offset

Register	Offset
DRAMTMG5_SHADOW	2114h

9.2.3.1.111.2 Diagram



9.2.3.1.111.3 Fields

Field	Function
31-28 —	Reserved
27-24 t_cksrx	This is the time before Self Refresh Exit that CK is maintained as a valid clock before issuing SRX. Specifies the clock stable time before SRX. Recommended settings: - mDDR: 1 - LPDDR2: 2 - LPDDR3: 2 - LPDDR4: tCKCKEH - DDR2: 1 - DDR3: tCKSRX - DDR4: tCKSRX When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer.
23-20 —	Reserved
19-16 t_cksre	This is the time after Self Refresh Down Entry that CK is maintained as a valid clock. Specifies the clock disable delay after SRE. Recommended settings: - mDDR: 0 - LPDDR2: 2 - LPDDR3: 2 - LPDDR4: tCKCKEL - DDR2: 1 - DDR3: max (10 ns, 5 tCK) - DDR4: max (10 ns, 5 tCK) (+ PL(parity latency)(*)) (*)Only if CRCPARCTL1.caparity_disable_before_sr=0, this register should be increased by PL. When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer.
15-14 —	Reserved
13-8 t_ckesr	Minimum CKE low width for Self refresh or Self refresh power down entry to exit timing in memory clock cycles. Recommended settings: - mDDR: tRFC - LPDDR2: tCKESR - LPDDR3: tCKESR - LPDDR4: max(tCKELPD, tSR) - DDR2: tCKE - DDR3: tCKE + 1 - DDR4: tCKE + 1 (+ PL(parity latency)(*)) (*)Only if CRCPARCTL1.caparity_disable_before_sr=0, this register should be increased by PL. When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer.
7-5 —	Reserved
4-0 t_cke	Minimum number of cycles of CKE HIGH/LOW during power-down and self refresh. - LPDDR2/LPDDR3 mode: Set this to the larger of tCKE or tCKESR - LPDDR4 mode: Set this to the larger of tCKE, tCKELPD or tSR. - Non-LPDDR2/non-LPDDR3/non-LPDDR4 designs: Set this to tCKE value. When the controller is operating in 1:2 frequency ratio mode, program this to (value described above)/2 and round it up to the next integer value. Unit: Clocks.

9.2.3.1.112 [SHADOW] SDRAM Timing Register 6 (DRAMTMG6_SHADOW)

9.2.3.1.112.1 Offset

Register	Offset
DRAMTMG6_SHADOW	2118h

9.2.3.1.112.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				t_ckdpde				Reserved				t_ckdpdx			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												t_ckcsx			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

9.2.3.1.112.3 Fields

Field	Function
31-28 —	Reserved
27-24 t_ckdpde	This is the time after Deep Power Down Entry that CK is maintained as a valid clock. Specifies the clock disable delay after DPDE. Recommended settings: - mDDR: 0 - LPDDR2: 2 - LPDDR3: 2 When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. This is only present for designs supporting mDDR or LPDDR2/LPDDR3 devices.
23-20 —	Reserved
19-16 t_ckdpdx	This is the time before Deep Power Down Exit that CK is maintained as a valid clock before issuing DPDX. Specifies the clock stable time before DPDX. Recommended settings: - mDDR: 1 - LPDDR2: 2 - LPDDR3: 2 When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. This is only present for designs supporting mDDR or LPDDR2 devices.
15-4 —	Reserved
3-0 t_ckcsx	This is the time before Clock Stop Exit that CK is maintained as a valid clock before issuing Clock Stop Exit. Specifies the clock stable time before next command after Clock Stop Exit. Recommended settings: - mDDR: 1 - LPDDR2: tXP + 2 - LPDDR3: tXP + 2 - LPDDR4: tXP + 2 When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. This is only present for designs supporting mDDR or LPDDR2/LPDDR3/LPDDR4 devices.

9.2.3.1.113 [SHADOW] SDRAM Timing Register 7 (DRAMTMG7_SHADOW)

9.2.3.1.113.1 Offset

Register	Offset
DRAMTMG7_SHADOW	211Ch

9.2.3.1.113.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				t_ckpde				Reserved				t_ckpdx			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

9.2.3.1.113.3 Fields

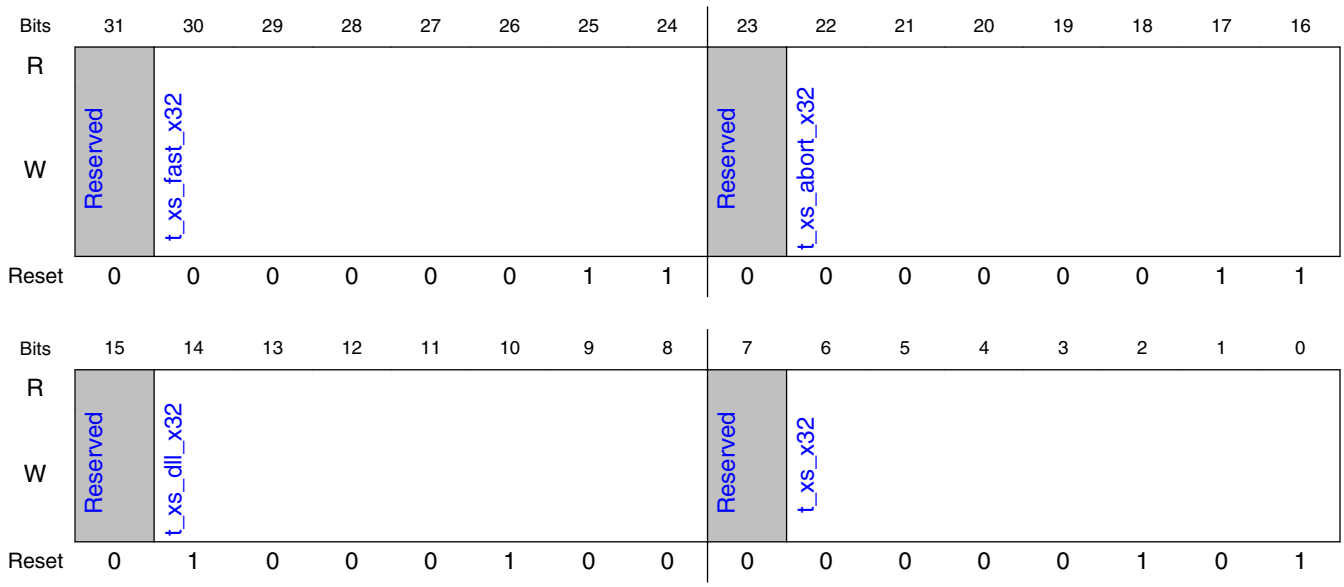
Field	Function
31-12 —	Reserved
11-8 t_ckpde	This is the time after Power Down Entry that CK is maintained as a valid clock. Specifies the clock disable delay after PDE. Recommended settings: - mDDR: 0 - LPDDR2: 2 - LPDDR3: 2 - LPDDR4: tCKCKEL When using DDR2/3/4 SDRAM, this register should be set to the same value as DRAMTMG5.t_cksre. When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. This is only present for designs supporting mDDR or LPDDR2/ LPDDR3/LPDDR4 devices.
7-4 —	Reserved
3-0 t_ckpdx	This is the time before Power Down Exit that CK is maintained as a valid clock before issuing PDX. Specifies the clock stable time before PDX. Recommended settings: - mDDR: 0 - LPDDR2: 2 - LPDDR3: 2 - LPDDR4: 2 When using DDR2/3/4 SDRAM, this register should be set to the same value as DRAMTMG5.t_cksrx. When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. This is only present for designs supporting mDDR or LPDDR2/LPDDR3/LPDDR4 devices.

9.2.3.1.114 [SHADOW] SDRAM Timing Register 8 (DRAMTMG8_SHADOW)

9.2.3.1.114.1 Offset

Register	Offset
DRAMTMG8_SHADOW	2120h

9.2.3.1.114.2 Diagram



9.2.3.1.114.3 Fields

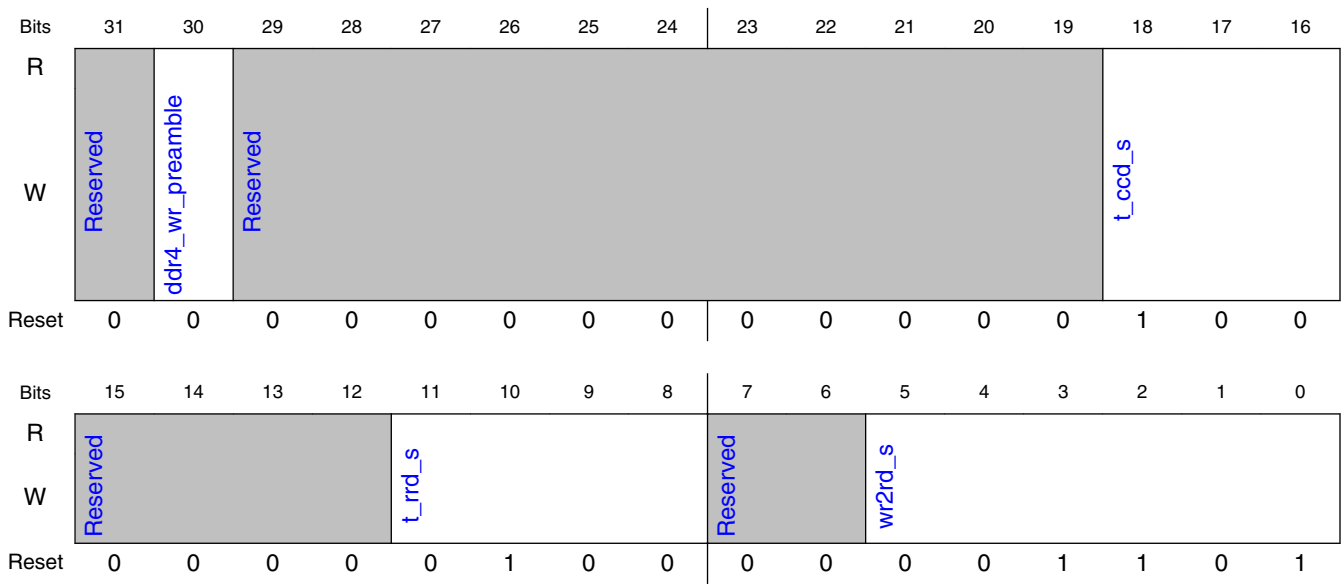
Field	Function
31 —	Reserved
30-24 t_xs_fast_x32	tXS_FAST: Exit Self Refresh to ZQCL, ZQCS and MRS (only CL, WR, RTP and Geardown mode). When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Unit: Multiples of 32 clocks. Note: This is applicable to only ZQCL/ZQCS commands. Note: Ensure this is less than or equal to t_xs_x32.
23 —	Reserved
22-16 t_xs_abort_x32	tXS_ABORT: Exit Self Refresh to commands not requiring a locked DLL in Self Refresh Abort. When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Unit: Multiples of 32 clocks. Note: Ensure this is less than or equal to t_xs_x32.
15 —	Reserved
14-8 t_xs_dll_x32	tXSDLL: Exit Self Refresh to commands requiring a locked DLL. When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Unit: Multiples of 32 clocks. Note: Used only for DDR2, DDR3 and DDR4 SDRAMs.
7 —	Reserved
6-0 t_xs_x32	tXS: Exit Self Refresh to commands not requiring a locked DLL. When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Unit: Multiples of 32 clocks. Note: Used only for DDR2, DDR3 and DDR4 SDRAMs.

9.2.3.1.115 [SHADOW] SDRAM Timing Register 9 (DRAMTMG9_SHADOW)

9.2.3.1.115.1 Offset

Register	Offset
DRAMTMG9_SHADOW	2124h

9.2.3.1.115.2 Diagram



9.2.3.1.115.3 Fields

Field	Function
31 —	Reserved
30 ddr4_wr_preamble	DDR4 Write preamble mode - 0: 1tCK preamble - 1: 2tCK preamble Present only with MEMC_FREQ_RATIO=2
29-19 —	Reserved
18-16 t_ccd_s	tCCD_S: This is the minimum time between two reads or two writes for different bank group. For bank switching (from bank "a" to bank "b"), the minimum time is this value + 1. When the controller is operating in 1:2 frequency ratio mode, program this to (tCCD_S/2) and round it up to the next integer value. Present only in designs configured to support DDR4. Unit: clocks.
15-12 —	Reserved

Table continues on the next page...

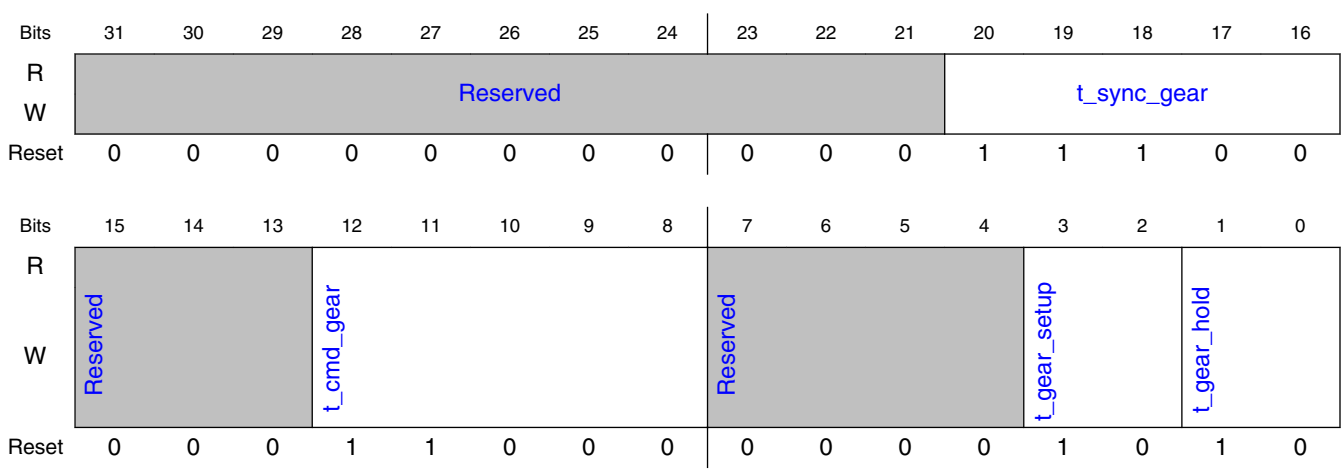
Field	Function
11-8 t_rrd_s	tRRD_S: Minimum time between activates from bank "a" to bank "b" for different bank group. When the controller is operating in 1:2 frequency ratio mode, program this to (tRRD_S/2) and round it up to the next integer value. Present only in designs configured to support DDR4. Unit: Clocks.
7-6 —	Reserved
5-0 wr2rd_s	CWL + PL + BL/2 + tWTR_S Minimum time from write command to read command for different bank group. Includes time for bus turnaround, recovery times, and all per-bank, per-rank, and global constraints. Present only in designs configured to support DDR4. Unit: Clocks. Where: - CWL = CAS write latency - PL = Parity latency - BL = burst length. This must match the value programmed in the BL bit of the mode register to the SDRAM - tWTR_S = internal write to read command delay for different bank group. This comes directly from the SDRAM specification. When the controller is operating in 1:2 mode, divide the value calculated using the above equation by 2, and round it up to next integer.

9.2.3.1.116 [SHADOW] SDRAM Timing Register 10 (DRAMTMG10_SHADOW)

9.2.3.1.116.1 Offset

Register	Offset
DRAMTMG10_SHADOW	2128h

9.2.3.1.116.2 Diagram



9.2.3.1.116.3 Fields

Field	Function
31-21	Reserved

Table continues on the next page...

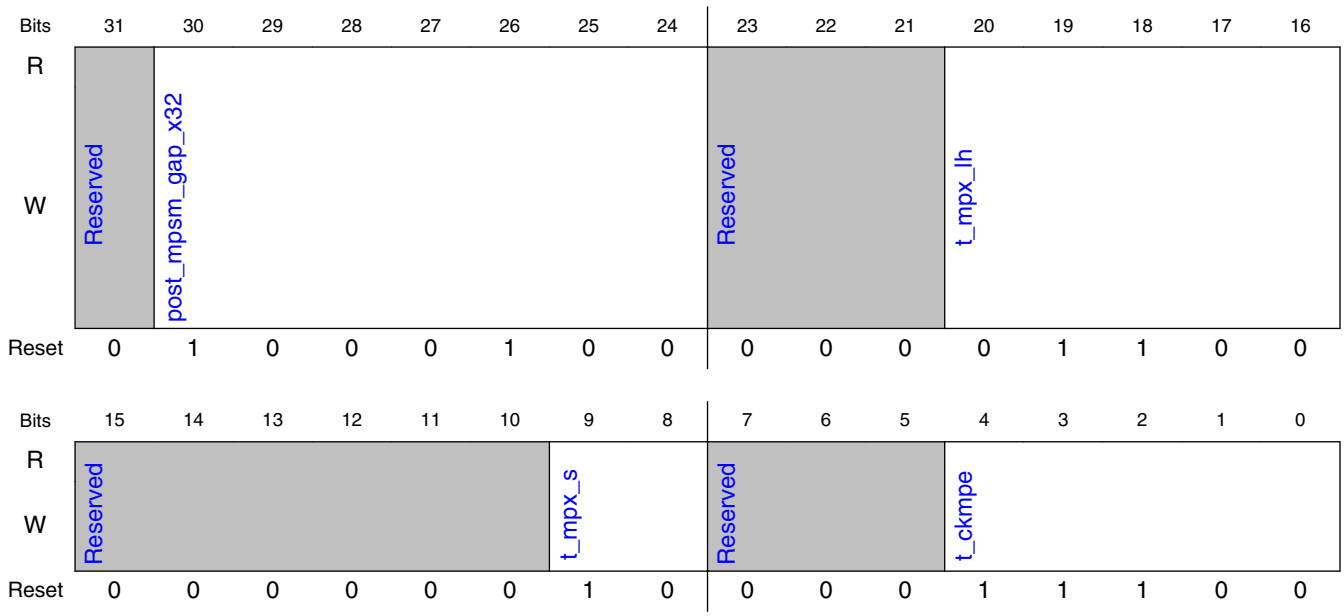
Field	Function
—	
20-16 t_sync_gear	Indicates the time between MRS command and the sync pulse time. This must be even number of clocks. For DDR4-2666 and DDR4-3200, this parameter is defined as tMOD(min)+4nCK tMOD(min) is greater of 24nCK or 15ns 15ns / .625ns = 24 Max value for this register is 24+4 = 28 When the controller is operating in 1:2 mode, program this to (tSYNC_GEAR/2) and round it up to the next integer value. Unit: Clocks
15-13 —	Reserved
12-8 t_cmd_gear	Sync pulse to first valid command. For DDR4-2666 and DDR4-3200, this parameter is defined as tMOD(min) tMOD(min) is greater of 24nCK or 15ns 15ns / .625ns = 24 Max value for this register is 24 When the controller is operating in 1:2 mode, program this to (tCMD_GEAR/2) and round it up to the next integer value. Unit: Clocks
7-4 —	Reserved
3-2 t_gear_setup	Geardown setup time. Minimum value of this register is 1. Zero is invalid. For DDR4-2666 and DDR4-3200, this parameter is defined as 2 clks When the controller is operating in 1:2 frequency ratio mode, program this to (tGEAR_setup/2) and round it up to the next integer value. Unit: Clocks
1-0 t_gear_hold	Geardown hold time. Minimum value of this register is 1. Zero is invalid. For DDR4-2666 and DDR4-3200, this parameter is defined as 2 clks When the controller is operating in 1:2 frequency ratio mode, program this to (tGEAR_hold/2) and round it up to the next integer value. Unit: Clocks

9.2.3.1.117 [SHADOW] SDRAM Timing Register 11 (DRAMTMG11_SHADOW)

9.2.3.1.117.1 Offset

Register	Offset
DRAMTMG11_SHADOW	212Ch

9.2.3.1.117.2 Diagram



9.2.3.1.117.3 Fields

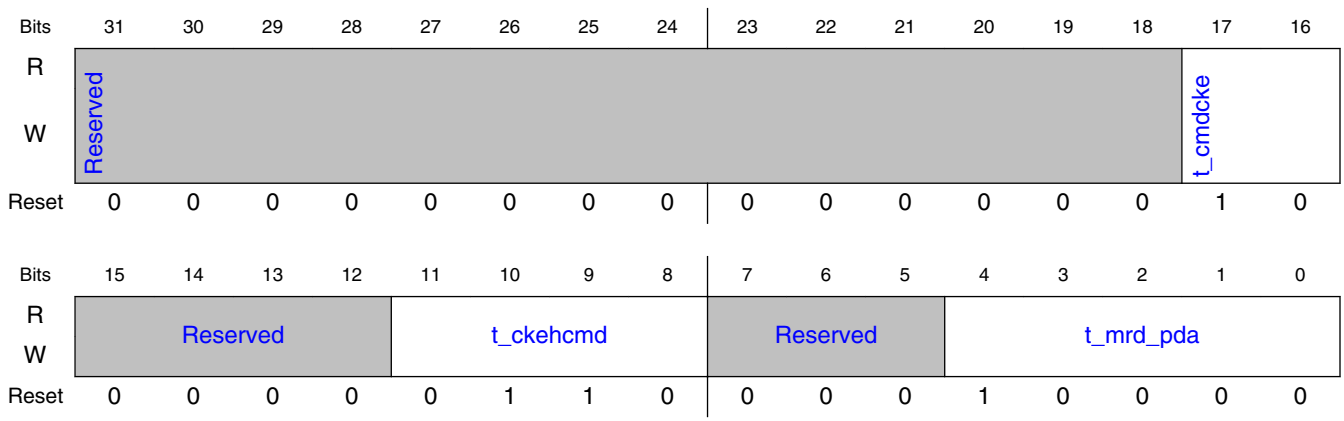
Field	Function
31 —	Reserved
30-24 post_mpsm_gap_x32	tXMPDLL: This is the minimum Exit MPSM to commands requiring a locked DLL. When the controller is operating in 1:2 frequency ratio mode, program this to (tXMPDLL/2) and round it up to the next integer value. Present only in designs configured to support DDR4. Unit: Multiples of 32 clocks.
23-21 —	Reserved
20-16 t_mpx_lh	tMPX_LH: This is the minimum CS_n Low hold time to CKE rising edge. When the controller is operating in 1:2 frequency ratio mode, program this to RoundUp(tMPX_LH/2)+1. Present only in designs configured to support DDR4. Unit: clocks.
15-10 —	Reserved
9-8 t_mpx_s	tMPX_S: Minimum time CS setup time to CKE. When the controller is operating in 1:2 frequency ratio mode, program this to (tMPX_S/2) and round it up to the next integer value. Present only in designs configured to support DDR4. Unit: Clocks.
7-5 —	Reserved
4-0 t_ckmpe	tCKMPE: Minimum valid clock requirement after MPSM entry. Present only in designs configured to support DDR4. Unit: Clocks. When the controller is operating in 1:2 frequency ratio mode, divide the value calculated using the above equation by 2, and round it up to next integer.

9.2.3.1.118 [SHADOW] SDRAM Timing Register 12 (DRAMTMG12_SHADOW)

9.2.3.1.118.1 Offset

Register	Offset
DRAMTMG12_SHADOW	2130h

9.2.3.1.118.2 Diagram



9.2.3.1.118.3 Fields

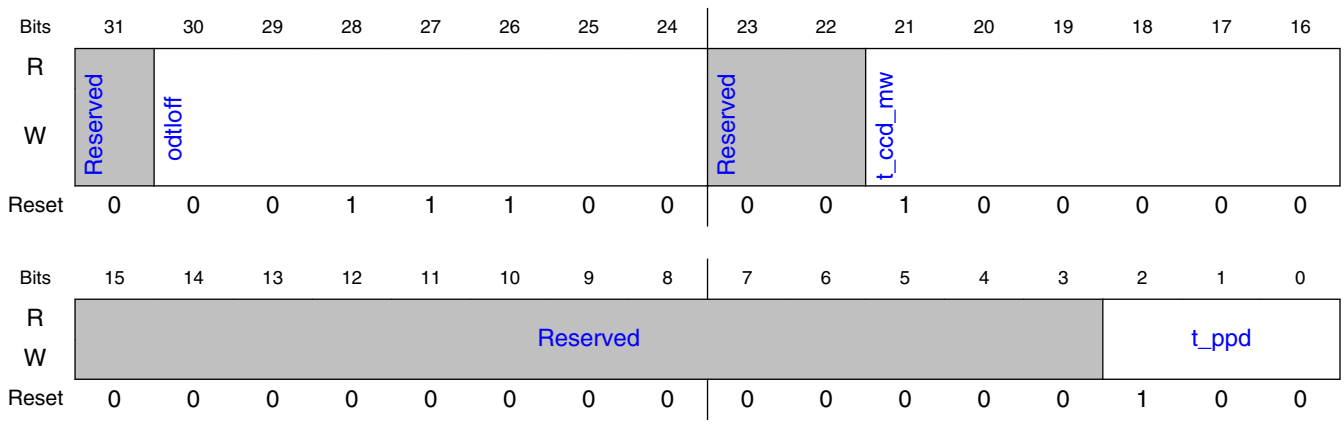
Field	Function
31-18 —	Reserved
17-16 t_cmdcke	tCMDCKE: Delay from valid command to CKE input LOW. Set this to the larger of tESCKE or tCMDCKE When the controller is operating in 1:2 frequency ratio mode, program this to (max(tESCKE, tCMDCKE)/2) and round it up to the next integer value.
15-12 —	Reserved
11-8 t_ckehcmd	tCKEHCMD: Valid command requirement after CKE input HIGH. When the controller is operating in 1:2 frequency ratio mode, program this to (tCKEHCMD/2) and round it up to the next integer value.
7-5 —	Reserved
4-0 t_mrd_pda	tMRD_PDA: This is the Mode Register Set command cycle time in PDA mode. When the controller is operating in 1:2 frequency ratio mode, program this to (tMRD_PDA/2) and round it up to the next integer value.

9.2.3.1.119 [SHADOW] SDRAM Timing Register 13 (DRAMTMG13_SHADOW)

9.2.3.1.119.1 Offset

Register	Offset
DRAMTMG13_SHADOW	2134h

9.2.3.1.119.2 Diagram



9.2.3.1.119.3 Fields

Field	Function
31 —	Reserved
30-24 odtloff	LPDDR4: tODTLoff: This is the latency from CAS-2 command to tODToff reference. When the controller is operating in 1:2 frequency ratio mode, program this to (tODTLoff/2) and round it up to the next integer value. Unit: Clocks.
23-22 —	Reserved
21-16 t_ccd_mw	LPDDR4: tCCDMW: This is the minimum time from write or masked write to masked write command for same bank. When the controller is operating in 1:2 frequency ratio mode, program this to (tCCDMW/2) and round it up to the next integer value. Unit: Clocks.
15-3 —	Reserved
2-0 t_ppd	LPDDR4: tPPD: This is the minimum time from precharge to precharge command. When the controller is operating in 1:2 frequency ratio mode, program this to (tPPD/2) and round it up to the next integer value. Unit: Clocks.

9.2.3.1.120 [SHADOW] SDRAM Timing Register 14 (DRAMTMG14_SHADOW)

9.2.3.1.120.1 Offset

Register	Offset
DRAMTMG14_SHADOW	2138h

9.2.3.1.120.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				t_xsr											
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0

9.2.3.1.120.3 Fields

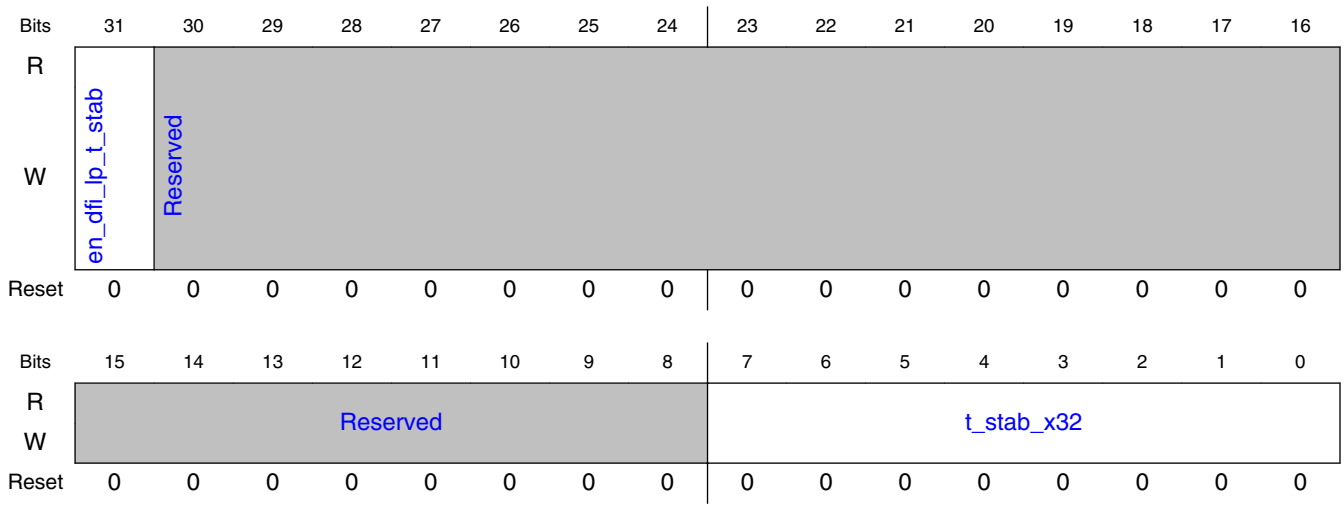
Field	Function
31-12 —	Reserved
11-0 t_xsr	tXSR: Exit Self Refresh to any command. When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Note: Used only for mDDR/LPDDR2/LPDDR3/LPDDR4 mode.

9.2.3.1.121 [SHADOW] SDRAM Timing Register 15 (DRAMTMG15_SHADOW)

9.2.3.1.121.1 Offset

Register	Offset
DRAMTMG15_SHADOW	213Ch

9.2.3.1.121.2 Diagram



9.2.3.1.121.3 Fields

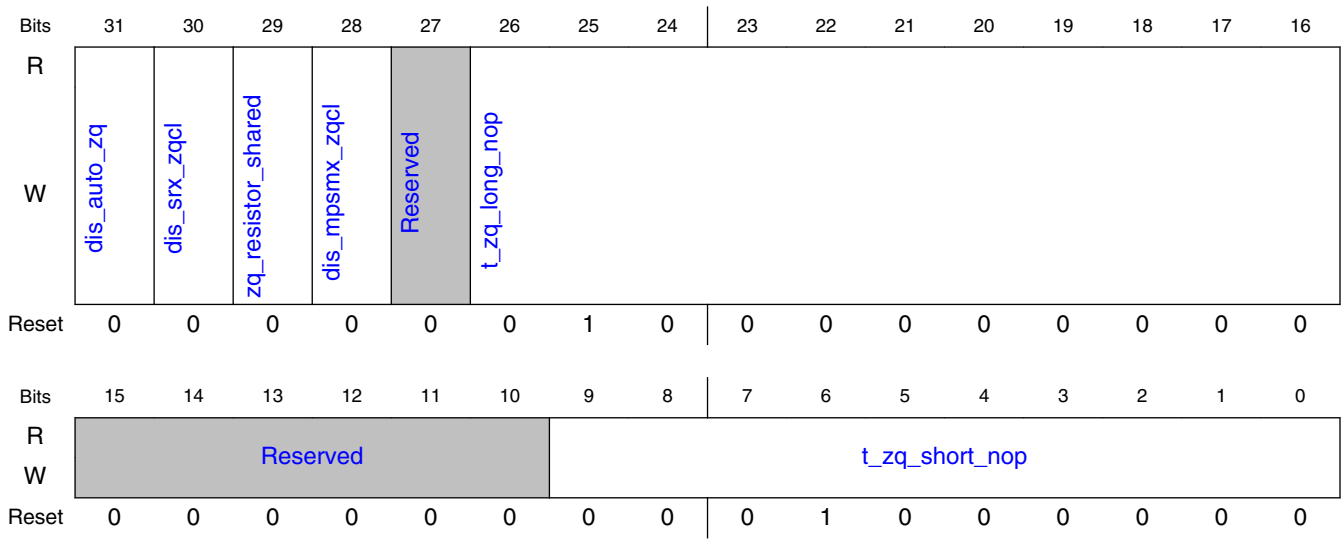
Field	Function
31 en_dfi_lp_t_stab	- 1 - Enable using tSTAB when exiting DFI LP. Needs to be set when the PHY is stopping the clock during DFI LP to save maximum power. - 0 - Disable using tSTAB when exiting DFI LP
30-8 —	Reserved
7-0 t_stab_x32	tSTAB: Stabilization time. It is required in the following two cases for DDR3/DDR4 RDIMM : - when exiting power saving mode, if the clock was stopped, after re-enabling it the clock must be stable for a time specified by tSTAB - in the case of input clock frequency change (DDR4) - after issuing control words that refers to clock timing (Specification: 6us for DDR3, 5us for DDR4) When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. Unit: Multiples of 32 clock cycles.

9.2.3.1.122 [SHADOW] ZQ Control Register 0 (ZQCTL0_SHADOW)

9.2.3.1.122.1 Offset

Register	Offset
ZQCTL0_SHADOW	2180h

9.2.3.1.122.2 Diagram



9.2.3.1.122.3 Fields

Field	Function
31 dis_auto_zq	- 1 - Disable DDRC generation of ZQCS/MPC(ZQ calibration) command. Register DBGCMD.zq_calib_short can be used instead to issue ZQ calibration request from APB module. - 0 - Internally generate ZQCS/MPC(ZQ calibration) commands based on ZQCTL1.t_zq_short_interval_x1024. This is only present for designs supporting DDR3/DDR4 or LPDDR2/LPDDR3/LPDDR4 devices.
30 dis_srx_zqcl	- 1 - Disable issuing of ZQCL/MPC(ZQ calibration) command at Self-Refresh/SR-Powerdown exit. Only applicable when run in DDR3 or DDR4 or LPDDR2 or LPDDR3 or LPDDR4 mode. - 0 - Enable issuing of ZQCL/MPC(ZQ calibration) command at Self-Refresh/SR-Powerdown exit. Only applicable when run in DDR3 or DDR4 or LPDDR2 or LPDDR3 or LPDDR4 mode. This is only present for designs supporting DDR3/DDR4 or LPDDR2/LPDDR3/LPDDR4 devices.
29 zq_resistor_shared	- 1 - Denotes that ZQ resistor is shared between ranks. Means ZQinit/ZQCL/ZQCS/MPC(ZQ calibration) commands are sent to one rank at a time with tZQinit/tZQCL/tZQCS/tZQCAL/tZQLAT timing met between commands so that commands to different ranks do not overlap. - 0 - ZQ resistor is not shared. This is only present for designs supporting DDR3/DDR4 or LPDDR2/LPDDR3/LPDDR4 devices.
28 dis_mpsmx_zqcl	- 1 - Disable issuing of ZQCL command at Maximum Power Saving Mode exit. Only applicable when run in DDR4 mode. - 0 - Enable issuing of ZQCL command at Maximum Power Saving Mode exit. Only applicable when run in DDR4 mode. This is only present for designs supporting DDR4 devices. Note: Do not issue ZQCL command at Maximum Power Save Mode exit if the DDRC_SHARED_AC configuration parameter is set. Program it to 1'b1. The software can send ZQCS after exiting MPSM mode.
27 —	Reserved
26-16 t_zq_long_nop	tZQoper for DDR3/DDR4, tZQCL for LPDDR2/LPDDR3, tZQCAL for LPDDR4: Number of DFI clock cycles of NOP required after a ZQCL (ZQ calibration long)/MPC(ZQ Start) command is issued to SDRAM. When the controller is operating in 1:2 frequency ratio mode: DDR3/DDR4: program this to tZQoper/2 and round it up to the next integer value. LPDDR2/LPDDR3: program this to tZQCL/2 and round it up to the next integer value. LPDDR4: program this to tZQCAL/2 and round it up to the next integer value. This is only present for designs supporting DDR3/DDR4 or LPDDR2/LPDDR3/LPDDR4 devices.
15-10	Reserved

Table continues on the next page...

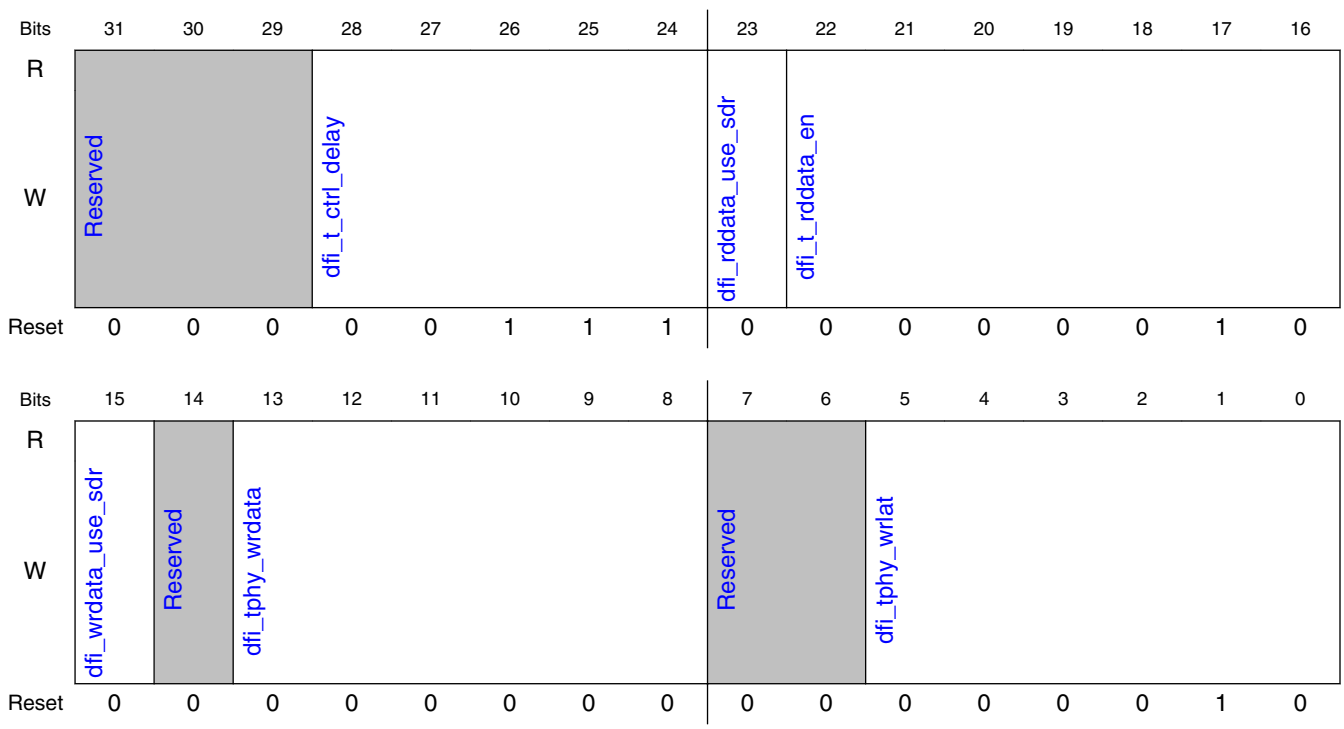
Field	Function
—	
9-0 t_zq_short_nop	tZQCS for DDR3/DD4/LPDDR2/LPDDR3, tZQLAT for LPDDR4: Number of DFI clock cycles of NOP required after a ZQCS (ZQ calibration short)/MPC(ZQ Latch) command is issued to SDRAM. When the controller is operating in 1:2 frequency ratio mode, program this to tZQCS/2 and round it up to the next integer value. This is only present for designs supporting DDR3/DDR4 or LPDDR2/LPDDR3/LPDDR4 devices.

9.2.3.1.123 [SHADOW] DFI Timing Register 0 (DFITMG0_SHADOW)

9.2.3.1.123.1 Offset

Register	Offset
DFITMG0_SHADOW	2190h

9.2.3.1.123.2 Diagram



9.2.3.1.123.3 Fields

Field	Function
31-29	Reserved

Table continues on the next page...

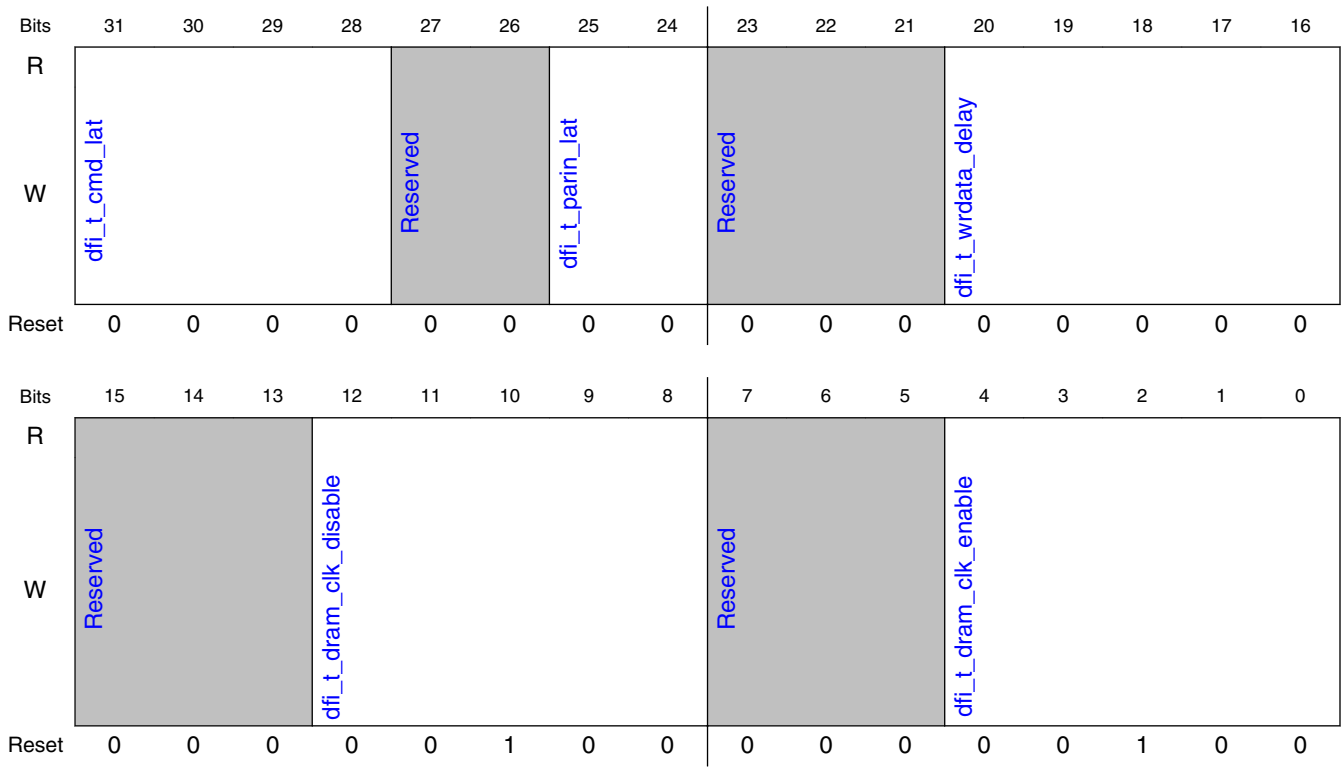
Field	Function
—	
28-24 dfi_t_ctrl_delay	Specifies the number of DFI clock cycles after an assertion or de-assertion of the DFI control signals that the control signals at the PHY-DRAM interface reflect the assertion or de-assertion. If the DFI clock and the memory clock are not phase-aligned, this timing parameter should be rounded up to the next integer value. Note that if using RDIMM/LRDIMM, it is necessary to increment this parameter by RDIMM's/ LRDIMM's extra cycle of latency in terms of DFI clock.
23 dfi_rddata_use_sdr	Defines whether dfi_rddata_en/dfi_rddata/dfi_rddata_valid is generated using HDR (DFI clock) or SDR (DFI PHY clock) values. Selects whether value in DFITMG0.dfi_t_rddata_en is in terms of HDR (DFI clock) or SDR (DFI PHY clock) cycles: - 0 in terms of HDR (DFI clock) cycles - 1 in terms of SDR (DFI PHY clock) cycles Refer to PHY specification for correct value.
22-16 dfi_t_rddata_en	Time from the assertion of a read command on the DFI interface to the assertion of the dfi_rddata_en signal. Refer to PHY specification for correct value. This corresponds to the DFI parameter trddata_en. Note that, depending on the PHY, if using RDIMM/LRDIMM, it may be necessary to use the adjusted value of CL in the calculation of trddata_en. This is to compensate for the extra cycle(s) of latency through the RDIMM/LRDIMM. Unit: DFI clock cycles or DFI PHY clock cycles, depending on DFITMG0.dfi_rddata_use_sdr.
15 dfi_wrdata_use_sdr	Defines whether dfi_wrdata_en/dfi_wrdata/dfi_wrdata_mask is generated using HDR (DFI clock) or SDR (DFI PHY clock) values Selects whether value in DFITMG0.dfi_tphy_wrlat is in terms of HDR (DFI clock) or SDR (DFI PHY clock) cycles Selects whether value in DFITMG0.dfi_tphy_wrdata is in terms of HDR (DFI clock) or SDR (DFI PHY clock) cycles - 0 in terms of HDR (DFI clock) cycles - 1 in terms of SDR (DFI PHY clock) cycles Refer to PHY specification for correct value.
14 —	Reserved
13-8 dfi_tphy_wrdata	Specifies the number of clock cycles between when dfi_wrdata_en is asserted to when the associated write data is driven on the dfi_wrdata signal. This corresponds to the DFI timing parameter tphy_wrdata. Refer to PHY specification for correct value. Note, max supported value is 8. Unit: DFI clock cycles or DFI PHY clock cycles, depending on DFITMG0.dfi_wrdata_use_sdr.
7-6 —	Reserved
5-0 dfi_tphy_wrlat	Write latency Number of clocks from the write command to write data enable (dfi_wrdata_en). This corresponds to the DFI timing parameter tphy_wrlat. Refer to PHY specification for correct value.Note that, depending on the PHY, if using RDIMM/LRDIMM, it may be necessary to use the adjusted value of CL in the calculation of tphy_wrlat. This is to compensate for the extra cycle(s) of latency through the RDIMM/LRDIMM. Unit: DFI clock cycles or DFI PHY clock cycles, depending on DFITMG0.dfi_wrdata_use_sdr.

9.2.3.1.124 [SHADOW] DFI Timing Register 1 (DFITMG1_SHADOW)

9.2.3.1.124.1 Offset

Register	Offset
DFITMG1_SHADOW	2194h

9.2.3.1.124.2 Diagram



9.2.3.1.124.3 Fields

Field	Function
31-28 dfi_t_cmd_lat	Specifies the number of DFI PHY clock cycles between when the dfi_cs signal is asserted and when the associated command is driven. This field is used for CAL mode, should be set to '0' or the value which matches the CAL mode register setting in the DRAM. If the PHY can add the latency for CAL mode, this should be set to '0'. Valid Range: 0, 3, 4, 5, 6, and 8
27-26 —	Reserved
25-24 dfi_t_parin_lat	Specifies the number of DFI PHY clock cycles between when the dfi_cs signal is asserted and when the associated dfi_parity_in signal is driven.
23-21 —	Reserved
20-16 dfi_t_wrdata_delay	Specifies the number of DFI clock cycles between when the dfi_wrdata_en signal is asserted and when the corresponding write data transfer is completed on the DRAM bus. This corresponds to the DFI timing parameter twrdata_delay. Refer to PHY specification for correct value. For DFI 3.0 PHY, set to twrdata_delay, a new timing parameter introduced in DFI 3.0. For DFI 2.1 PHY, set to tphy_wrdata + (delay of DFI write data to the DRAM). Value to be programmed is in terms of DFI clocks, not PHY clocks. In FREQ_RATIO=2, divide PHY's value by 2 and round up to next integer. If using DFITMG0.dfi_wrdata_use_sdr=1, add 1 to the value. Unit: Clocks
15-13 —	Reserved

Table continues on the next page...

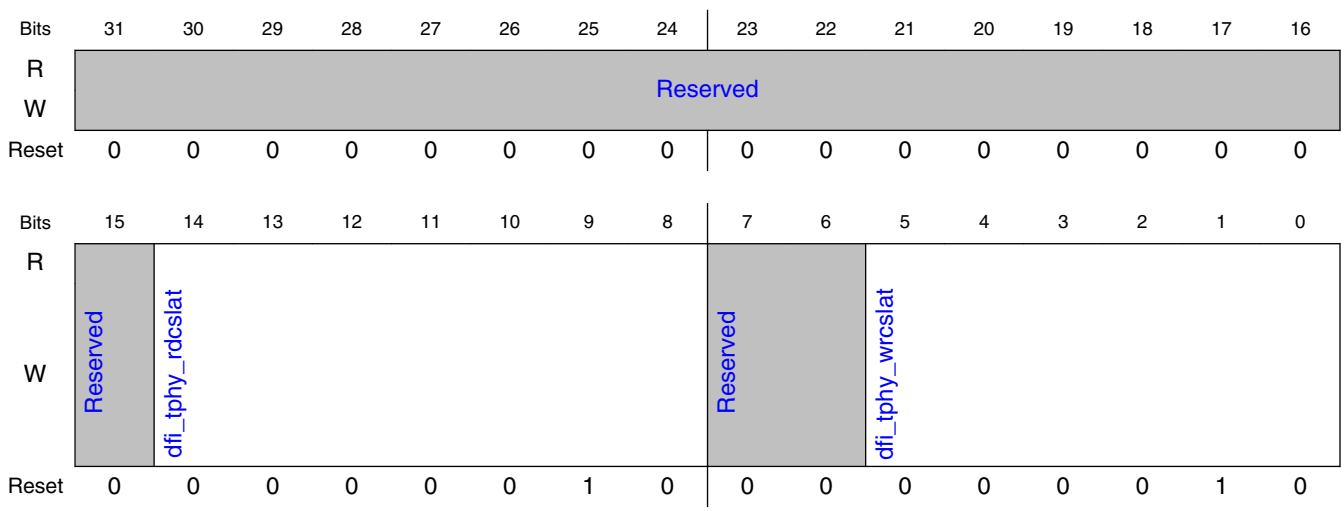
Field	Function
12-8 dfi_t_dram_clk_disable	Specifies the number of DFI clock cycles from the assertion of the dfi_dram_clk_disable signal on the DFI until the clock to the DRAM memory devices, at the PHY-DRAM boundary, maintains a low value. If the DFI clock and the memory clock are not phase aligned, this timing parameter should be rounded up to the next integer value.
7-5 —	Reserved
4-0 dfi_t_dram_clk_enable	Specifies the number of DFI clock cycles from the de-assertion of the dfi_dram_clk_disable signal on the DFI until the first valid rising edge of the clock to the DRAM memory devices, at the PHY-DRAM boundary. If the DFI clock and the memory clock are not phase aligned, this timing parameter should be rounded up to the next integer value.

9.2.3.1.125 [SHADOW] DFI Timing Register 2 (DFITMG2_SHADOW)

9.2.3.1.125.1 Offset

Register	Offset
DFITMG2_SHADOW	21B4h

9.2.3.1.125.2 Diagram



9.2.3.1.125.3 Fields

Field	Function
31-15 —	Reserved

Table continues on the next page...

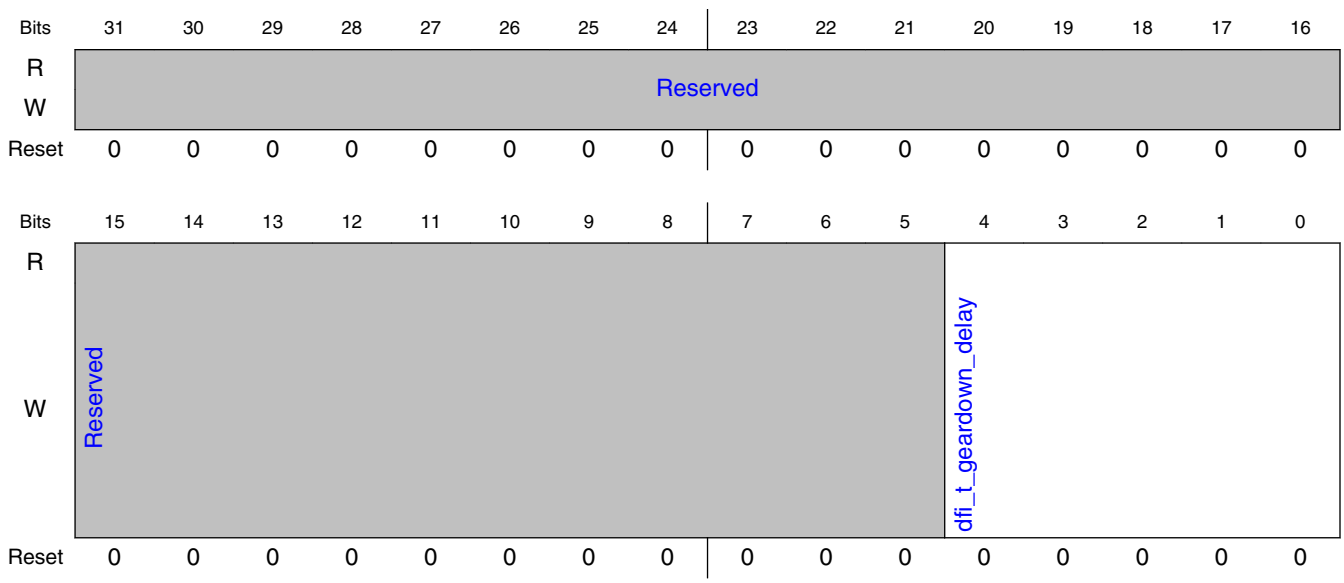
Field	Function
14-8 dfi_tphy_rdcslat	Number of DFI PHY clock cycles between when a read command is sent on the DFI control interface and when the associated dfi_rddata_cs signal is asserted. This corresponds to the DFI timing parameter tphy_rdcslat. Refer to PHY specification for correct value.
7-6 —	Reserved
5-0 dfi_tphy_wrcslat	Number of DFI PHY clock cycles between when a write command is sent on the DFI control interface and when the associated dfi_wrdata_cs signal is asserted. This corresponds to the DFI timing parameter tphy_wrcslat. Refer to PHY specification for correct value.

9.2.3.1.126 [SHADOW] DFI Timing Register 3 (DFITMG3_SHADOW)

9.2.3.1.126.1 Offset

Register	Offset
DFITMG3_SHADOW	21B8h

9.2.3.1.126.2 Diagram



9.2.3.1.126.3 Fields

Field	Function
31-5	Reserved

Table continues on the next page...

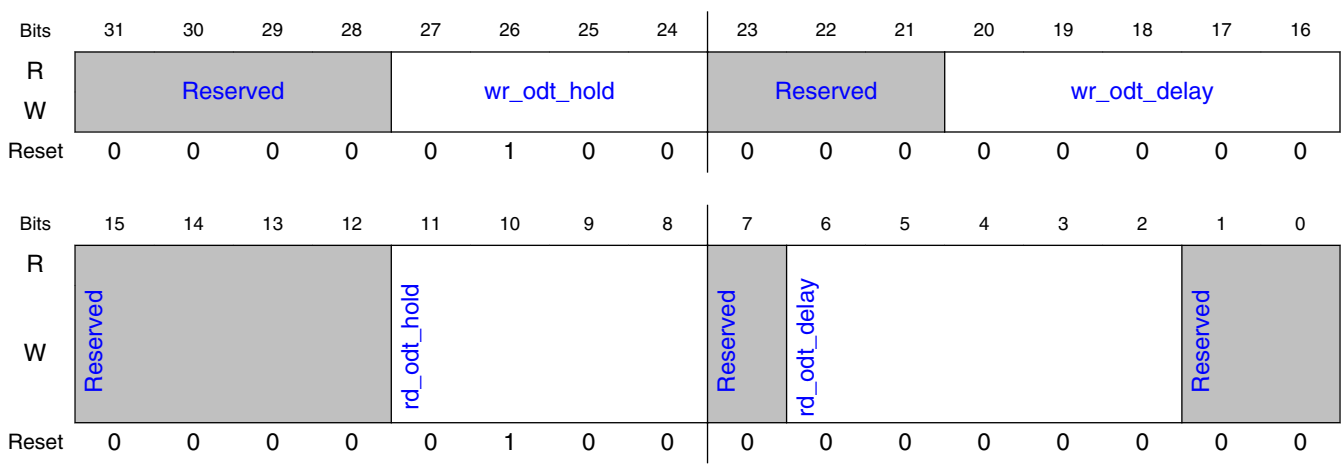
Field	Function
—	
4-0 dfi_t_geardown_delay	The delay from dfi_geardown_en assertion to the time of the PHY being ready to receive commands. Refer to PHY specification for correct value. When the controller is operating in 1:2 frequency ratio mode, program this to (tgeardown_delay/2) and round it up to the next integer value. Unit: Clocks

9.2.3.1.127 [SHADOW] ODT Configuration Register (ODTCFG_SHADOW)

9.2.3.1.127.1 Offset

Register	Offset
ODTCFG_SHADOW	2240h

9.2.3.1.127.2 Diagram



9.2.3.1.127.3 Fields

Field	Function
31-28 —	Reserved
27-24 wr_odt_hold	DFI PHY clock cycles to hold ODT for a write command. The minimum supported value is 2. Recommended values: DDR2: - BL8: 0x5 (DDR2-400/533/667), 0x6 (DDR2-800), 0x7 (DDR2-1066) - BL4: 0x3 (DDR2-400/533/667), 0x4 (DDR2-800), 0x5 (DDR2-1066) DDR3: - BL8: 0x6 DDR4: - BL8: 5 + WR_PREAMBLE + CRC_MODE WR_PREAMBLE = 1 (1tCK write preamble), 2 (2tCK write preamble) CRC_MODE = 0 (not CRC mode), 1 (CRC mode) LPDDR3: - BL8: 7 + RU(tODTon(max)/tCK)
23-21	Reserved

Table continues on the next page...

Field	Function
—	
20-16 wr_odt_delay	The delay, in DFI PHY clock cycles, from issuing a write command to setting ODT values associated with that command. ODT setting must remain constant for the entire time that DQS is driven by the DDRC. Recommended values: DDR2: - CWL + AL - 3 (DDR2-400/533/667), CWL + AL - 4 (DDR2-800), CWL + AL - 5 (DDR2-1066) If (CWL + AL - 3 < 0), DDRC does not support ODT for write operation. DDR3: - 0x0 DDR4: - DFITMG1.dfi_t_cmd_lat (to adjust for CAL mode) LPDDR3: - WL - 1 - RU(tODTon(max)/tCK))
15-12 —	Reserved
11-8 rd_odt_hold	DFI PHY clock cycles to hold ODT for a read command. The minimum supported value is 2. Recommended values: DDR2: - BL8: 0x6 (not DDR2-1066), 0x7 (DDR2-1066) - BL4: 0x4 (not DDR2-1066), 0x5 (DDR2-1066) DDR3: - BL8 - 0x6 DDR4: - BL8: 5 + RD_PREAMBLE RD_PREAMBLE = 1 (1tCK write preamble), 2 (2tCK write preamble) LPDDR3: - BL8: 5 + RU(tDQSCK(max)/tCK) - RD(tDQSCK(min)/tCK) + RU(tODTon(max)/tCK)
7 —	Reserved
6-2 rd_odt_delay	The delay, in DFI PHY clock cycles, from issuing a read command to setting ODT values associated with that command. ODT setting must remain constant for the entire time that DQS is driven by the DDRC. Recommended values: DDR2: - CL + AL - 4 (not DDR2-1066), CL + AL - 5 (DDR2-1066) If (CL + AL - 4 < 0), DDRC does not support ODT for read operation. DDR3: - CL - CWL DDR4: - CL - CWL - RD_PREAMBLE + WR_PREAMBLE + DFITMG1.dfi_t_cmd_lat (to adjust for CAL mode) WR_PREAMBLE = 1 (1tCK write preamble), 2 (2tCK write preamble) RD_PREAMBLE = 1 (1tCK write preamble), 2 (2tCK write preamble) If (CL - CWL - RD_PREAMBLE + WR_PREAMBLE) < 0, DDRC does not support ODT for read operation. LPDDR3: - RL + RD(tDQSCK(min)/tCK) - 1 - RU(tODTon(max)/tCK)
1-0 —	Reserved

9.3 DDR PHY (DDR_PHY)

9.3.1 Overview

The Double Data Rate Physical Layer (DDR PHY), provides an interface between universal controllers and external DDR3/DDR3L/DDR4/LPDDR3/LPDDR4 SDRAM devices.

NOTE

The information within this block guide is Synopsys Proprietary. Used with permission.

The block digram of the DDR PHY is shown below.

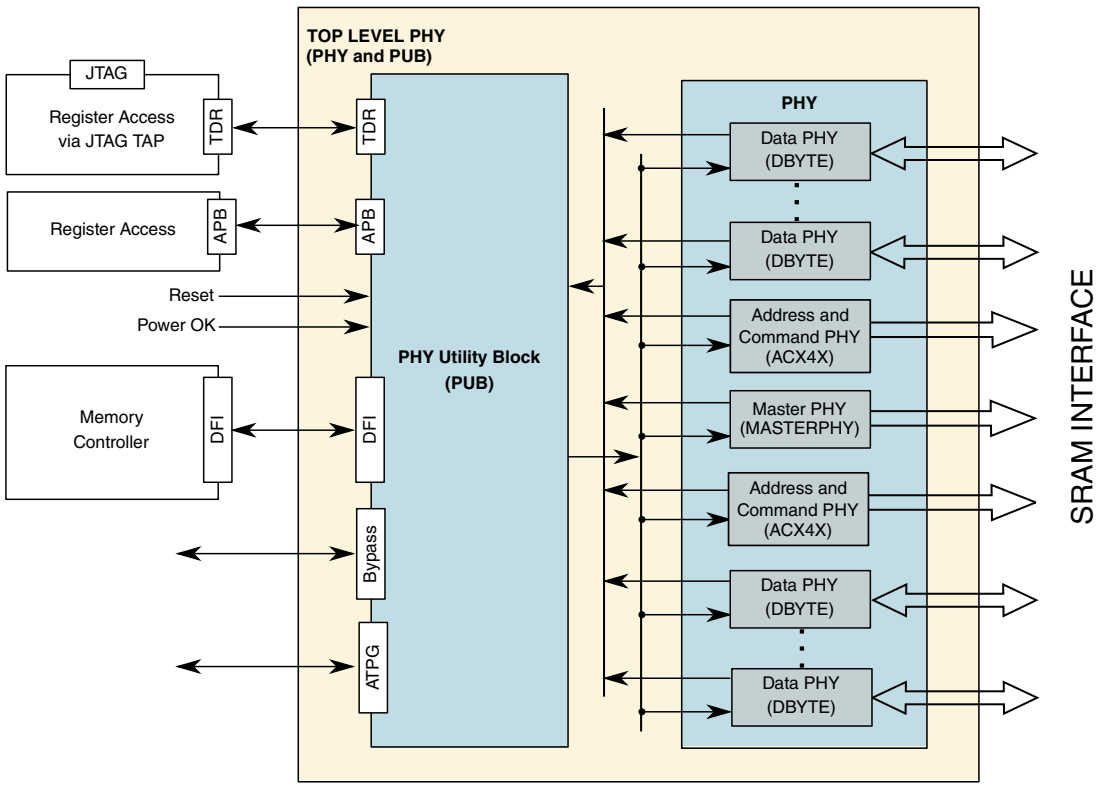


Figure 9-6. Block Diagram

9.3.1.1 Features

The following features are supported in the DDR PHY:

- LPDDR4, LPDDR3, DDR4, DDR3, and DDR3L operation
- Multiple memory rank support:
  - Four ranks supported for DDR4, DDR3, and DDR3L
  - Two ranks supported for LPDDR4 and LPDDR3
- DFI interface (DFI v4.0 Addendum 2, March 2015)
- At-speed loopback testing on both the address and data channels
- Mux-scan ATPG
- High-speed PLL used to generate the high speed clock for data transmit and high speed digital pipelines
- PHY Utility Block (PUB), which provides configuration and control
- Power on reset (POR) circuit, DRAM reset, DRAM alert and PHY debug
- A global voltage reference generation and observation circuit
- Impedance calibration circuits

9.3.2 Functional Description

The DDR PHY is composed of the following:

- DBYTE - The eight-lane data IO unit
- ACX4 - The four-lane address/command hard IP unit
- MASTERPHY - Contains the PLL, the voltage reference, thermal calibration, and reset
- PUB - PHY Utility Block

9.3.2.1 DDR Address/Command PHY (ACX4)

The DDR SDRAM address/command PHY (DDRPHYACX4) provides an address and command interface to the external SDRAM memories. A memory interface typically contains a single address/command PHY and one or more data PHYs (one for each byte lane). Figure 2-1 shows the organization of the AC component.

The AC consists of multiple design slices. The address/command (AC) slice is used for all address and command and clk/cs/odt signals. Each slice contains all logic required for both the output (address/command) and input (loopback) paths of the respective signals.

All clocking inside the AC is derived from an input clock, which is generated by the PLL in the MASTERPHY. The PLL generates the higher frequency clocks that are is used inside the macro and minimizes input clock jitter.

9.3.2.1.1 ACX4 Transmit Lanes

The ACX4 component contains 4 transmit only lanes which are grouped as follows:

- Clock group: {CK_t, CK_c}
- CA group: {ras_n, cas_n, we_n, addr…}
- CKE,CS,ODT group{CKE, CS_n, ODT…}

In DDR4 systems, 12 ACs are required to control one 72b/64b channel with four independent ranks.

In LPDDR3 and LPDDR4 systems, the 12 ACs are broken up into two independently controllable groups, enabling up to two independent 32b channels (each with a maximum of two ranks.

9.3.2.1.2 ACX4 Command Path

Each SDRAM address and command signal is driven from the memory controller to the SDRAM through an address/command slice. Controlled address/command pipeline logic inside the AC slice converts the DFI-rate signals from the controller to single data rate (SDR) Address/Command signals going to the SDRAM. The last launching register inside the address/command slice is clocked by a delay-shifted DDR clock. For SDR operation, as required by DDR3 DRAMs, all outputs remain constants for two cycles at a time.

The path for the SDRAM clock (CK) is through a standard AC slice which is configured to act as a dedicated clock (CK) slice. This shifted address/command output makes the CK centered into the data eye of the address/command signals.

All AC outputs going to the SDRAM have a LCDL, which can be used to align or shift the outputs of the data PHY. This delay element can also be used to provide delay adjustment during functional or loopback modes.

9.3.2.1.3 ACX4 Loopback/DFT Paths

For every lane there will be a loopback-only receive path. The loopback path is stimulated and checked by utilizing a FirmWare image.

9.3.2.2 DDR Data PHY (DBYTE)

The DDR SDRAM Data PHY (DBYTE) provides data interface to external SDRAM memory. Features of the DBYTE include:

- High speed digital logic pipeline for transmit datapaths to the SDRAM
- High speed digital logic pipeline for receive datapaths from the SDRAM
- Drivers and receivers for the DQ and DQS signals as well as all the necessary high voltage analog macros to interface off chip
- I/O component that includes PVT-compensated on-die termination (ODT) and output impedance
- Loop-back test mode and PRBS checkers to help verify the integrity of the internal datapath of all DBYTE I/O lanes

9.3.2.2.1 DBYTE Write Path

Each SDRAM data is driven from the PUB to the SDRAM through a data (DQ) slice. Controlled data pipeline logic inside the DBYTE slice converts the DFI-rate signals from the PUB to double-data rate (DDR) signals going to the SDRAM. The last launching register inside the data slice is clocked by the DDR clock.

The path for the SDRAM write strobe is through a dedicated data strobe (DQS) slice. The output register of the DQS slice is clocked by a delayed version of PclkIn. PclkIn drives the output register in the DQ slice is shifted using the DLL in training mode. This centers the write data strobe signal in the write data eye.

All DBYTE outputs going to the SDRAM, including write data, write data strobe read DQS gate, driver power down, receiver power down, on-die termination enable, and output enable have a DLL that is used to deskew the outputs in order to maximize the write data eye.

Apart from the DQS pipeline, the DQS slice also contains a similar pipeline for output enables, on-die termination enable, driver power down, and receiver power down. This matches the I/O control signals to the corresponding data and data strobe.

9.3.2.2.2 DBYTE Read and Loopback Path

Read data from the SDRAM is written into a four-deep asynchronous FIFO that is implemented inside the DQ slice.

NOTE

The FIFO is 16 deep in terms of the DRAM DQS writing into the FIFO; it is 8 deep on the core side when a half-speed DFICLK is employed and it is 8 deep on the core side when a full speed DFICLK is employed.

The FIFO is written using the DQS clock and read using the Controller clock. A per-bit delay line at the input of the DQ slice is used to optionally de-skew read data with respect to the read strobe in order to maximize the read data eye.

The read data strobe path and a read DQS gate path is implemented inside the DQS slice. A delay line at the input of the slice shifts the read DQS by nominal 90 degrees (with the actual value arrived at through training) so that the strobe is centered into the read data eye before clocking the data into the FIFO. A Bit- Wise Delay Line (BDL) in the read DQS path is used to match a corresponding bit de-skew BDL in the DQ slice.

The DQS slice also generates the write pointers for the data FIFO. These pointers, together with the strobe, are then distributed to all DQ slices. The PUB generates a trained read-valid signal to read the data from the PHY.

The DBI uses DQS Slice 1, configured to behave as a DQ slice in x8 mode only.

9.3.2.2.2.1 DBYTE Loopback/DFT Paths

High speed loopback is implemented on all DBYTE I/Os. The DBYTE read path also doubles as the loopback path. During loopback, the PUB enables the gating of the DQS signal in the same way as it would normally gate the DQS during reads. This allows the looped back data to be clocked into the read data FIFO using the looped back DQS signal.

9.3.2.2.2.2 Memory Controller based DBYTE Loopback

This high speed PHY loopback mode uses all the DBYTE control and data paths in mission mode like operation modes. In this mode, the memory controller issues simultaneous Read/Write commands to the PHY, resulting in transmitting and receiving the same data on each DQ pad.

9.3.2.3 Master PHY (MASTERPHY)

The Master block contains the PLL, the Primary Voltage Reference, thermal calibration, reset, and I/O cells for test and control.

9.3.2.3.1 PLL

The PLL in the MASTER is a simplified PLL that has DFICLK as an input and creates an internal clock by quadrupling the DFICLK frequency. The PLL in the MASTER can only multiply the input clock frequency to produce an output clock that is four times the frequency of the input clock. The internal clock is the output of the MASTER to the ACX4 and DBYTE.

9.3.2.3.1.1 Optimal PLL Settings for Mission Mode Frequency Bins

Table 9-4. Optimal PLL Settings for Mission Mode Frequency Bins (PLL Max Bandwidth = 2 MHz)

pllin min (MHz)	pllin max (MHz)	Input div. ratio	Fbk min (MHz)	Fbk max (MHz)	Fbk div. ratio	VCO min (MHz)	VCO max (MHz)	PllCtrl2 PllFreqSel[4:0]	PLLTestMode int_half[8] prop_half[5] vco_mode[2] overflow[1]	PllCtrl1 PllCpPropCtrl[8:5]	PllCtrl1 PllCpIntCtrl[4:0]	PllCtrl4 PllCpPropGsCtrl[8:5]	PllCtrl4 PllCpIntGsCtrl[4:0]
937.5	1067	8	117.2	133.4	4	3750	4268	11000 ’b	1011’b	0011’b	00000 ’b	1011’b	11111 ’b
625	937.5	8	78.13	117.2	4	2500	3750	11001 ’b	1011’b	0100’b	00000 ’b	1011’b	11111 ’b
468.75	625	4	117.2	156.3	8	3750	5000	01010 ’b	1011’b	0011’b	00000 ’b	1011’b	11111 ’b
312.5	468.75	4	78.13	117.2	8	2500	3750	01011 ’b	1011’b	0100’b	00000 ’b	1011’b	11111 ’b
234.4	312.5	2	117.2	156.3	16	3750	5000	00110 ’b	1011’b	0011’b	00000 ’b	1011’b	11111 ’b
166	234.4	2	83	117.2	16	2656	3750	00111 ’b	1011’b	0100’b	00000 ’b	1011’b	11111 ’b

9.3.2.3.2 Voltage Reference

This primary voltage reference generates the VREF global out used by the receivers in the ACX4 and DBYTE.

9.3.2.3.3 Calibration

The MASTER contains the analog blocks to perform output drive and input termination impedance calibration. Through the use of a digital control state machine, located in the PUB, the full mixed-signal calibration loop matches the impedance of the driver and termination with that of an external precision resistor. The resulting digital calibration code is fed to the address/command, DQ, and DQS blocks to control their impedance.

9.3.2.3.4 Reset

Reset takes the asynchronous reset bump, the PwrOk bump, and dfi_reset_n signal and combines them and synchronizes to the DFI clock. This is used by the ACX4 and DBYTE.

9.3.2.4 PHY Utility Block (PUB)

The following is an overview of the different functionality of the PHY Utility Block (PUB):

- Registers required to configure, control, train and self-test the PHY. Also contains registers required to initialize the DRAM.
- Initialization - this includes locking of the PLL, delay line calibration and calibration of I/O impedances.
- Data Training- train the PHY for optimum timing margins. This includes finding the best positioning of the DQS gating window during reads, write leveling, data bit deskew, and optimizing the read and write data eye.
- Implements a DFI-compliant interface between the external controller and the PHY.

9.3.2.4.1 DDR PHY Control and Configuration

The DDR PHY's PUB registers provide the required configuration, control, training and self-testing of the PHY. Also contains registers required to initialize the DRAM.

Refer to the DDR PHY register descriptions for more information.

9.3.2.4.2 Initialization and Training Sequence

The following outlines the general initialization and training sequence. Unless otherwise stated, these functions occur within the DDR PHY.

1. Bring up VDD, VDDQ, and VAA. This occurs outside of the DDR PHY.
2. Start the clocks and reset the PHY. This also occurs outside of the PHY.
3. Initialize PHY configuration.
4. Load the PHY training firmware into instruction memory SRAM.
5. Set the PHY input clocks to the desired frequency. This occurs outside of the PHY.

6. Write the Message Block parameters for the training firmware.There are several locations in the data memory that must be written for the training firmware to have these parameters.
7. Execute the training firmware.
8. Once the training firmware completes, the results are returned to the data memory in the message block structure. This message block can be read. The result may be that another frequency needs to be trained (return to step 5).
9. Load the PHY Initialization Engine registers with the provided initialization sequence.
10. Initialize the PHY to mission mode by performing a DFI initialization sequence per the DFI specification.
11. The PHY is now in mission mode.

9.3.2.4.3 DFI Interface

The DDR PHY and the DDR Controller both require a standard interface to transfer address, data, and control between these two blocks. These interfaces are DFI compliant. The PUB contains the DFI interface for the PHY.

For more information regarding the DFI specification, refer to the following:

- DDR PHY Interface (DFI) Specification, Revision 3.1 (Preliminary), 21, March 2014
- DDR PHY Interface (DFI) Specification Preliminary DFI 4.0 Specification - Addendum to DFI 3.1, 2 April 2014

The DFI interface is sub-divided into the following interface groups:

- Control Interface
- Write Data Interface
- Read Data Interface
- Update Interface
- Status Interface
- Training Interface (PHY-independent mode supported by the DDRC)
- Low Power Control Interface

9.3.3 Memory Map and Register Definition

This section includes the DDRC PHY module memory map and detailed descriptions of all registers.

9.3.3.1 DWC_DDRPHYA_ANIB register descriptions

9.3.3.1.1 DWC_DDRPHYA_ANIB Memory map

DDR4/3 PHY address block

DWC_DDRPHYA_ANIB0 base address: 0h

DWC_DDRPHYA_ANIB1 base address: 1000h

DWC_DDRPHYA_ANIB2 base address: 2000h

DWC_DDRPHYA_ANIB3 base address: 3000h

DWC_DDRPHYA_ANIB4 base address: 4000h

DWC_DDRPHYA_ANIB5 base address: 5000h

DWC_DDRPHYA_ANIB6 base address: 6000h

DWC_DDRPHYA_ANIB7 base address: 7000h

DWC_DDRPHYA_ANIB8 base address: 8000h

DWC_DDRPHYA_ANIB9 base address: 9000h

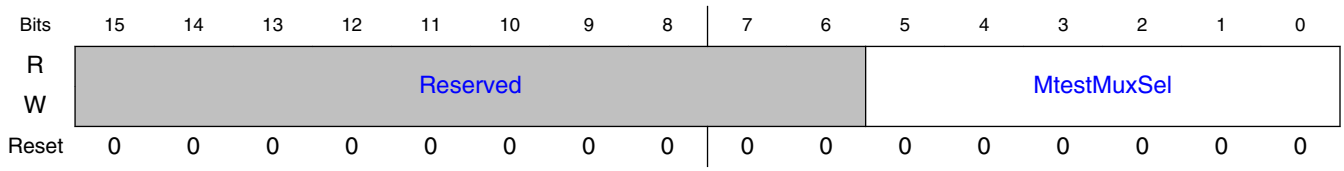
Offset	Register	Width (In bits)	Access	Reset value
34h	Digital Observation Pin control (MtestMuxSel)	16	RW	0000h
4Eh	Force Address/Command Driven (Lanes A3-A0) (AForceDrvCont)	16	RW	0000h
50h	Force Address/Command Tristate (Lanes A3-A0) (AForceTriCont)	16	RW	0000h
86h	Address TX impedance controls (ATxImpedance)	16	RW	03FFh
A6h	Address Loopback PRBS Error status for an entire ACX4 block (ATestPrbsErr)	16	RO	Table 9-4
AAh	Address TX slew rate and predriver controls (ATxSlewRate)	16	RW	07FFh
ACh	Address Loopback Test Result register (ATestPrbsErrCnt)	16	RO	Table 9-4
100h	Address/Command Delay, per pstate. (ATxDly_p0)	16	RW	0000h
20_0100h	Address/Command Delay, per pstate. (ATxDly_p1)	16	RW	0000h
40_0100h	Address/Command Delay, per pstate. (ATxDly_p2)	16	RW	0000h
60_0100h	Address/Command Delay, per pstate. (ATxDly_p3)	16	RW	0000h

9.3.3.1.2 Digital Observation Pin control (MtestMuxSel)

9.3.3.1.2.1 Offset

Register	Offset
MtestMuxSel	34h

9.3.3.1.2.2 Diagram



9.3.3.1.2.3 Fields

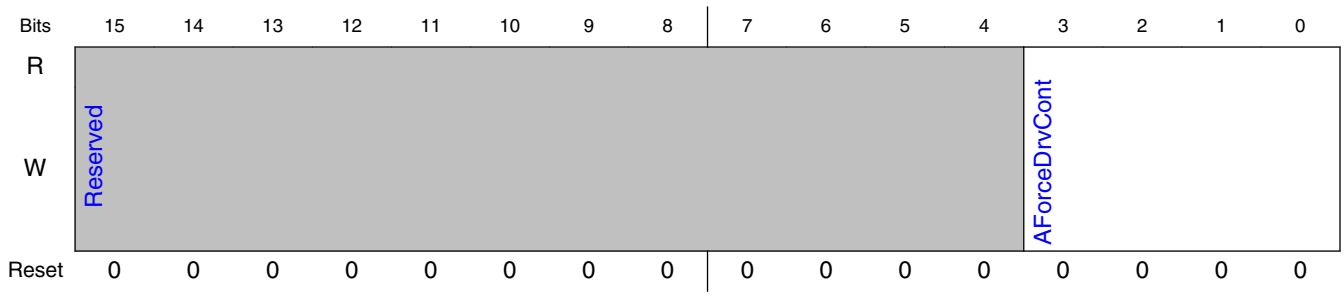
Field	Function
15-6 —	Reserved
5-0 MtestMuxSel	Controls for the 64-1 mux for asynchronous data to the Digital Observation Pin. Controls for the 64-1 mux for asynchronous data to the Digital Observation Pin. Encoding 6'h0 --> Drive 0 from this chiplet (allows flat 'OR' of pass-through information) Encoding 6'h01:1f --> Select local data from AC/DBYTE/MASTER macro Encoding 6'h20 --> Reserved (Drive 0 from macro into mux; Drive 0 from PUB synth logic path into mux) Encoding 6'h21:3f --> Select local data from PUB AC/DBYTE/MASTER synthesized logic Note: See PUB documentation for how, or if, the Digital Observation Pin is mapped to a physical bump in this configuration.

9.3.3.1.3 Force Address/Command Driven (Lanes A3-A0) (AForceDrvCont)

9.3.3.1.3.1 Offset

Register	Offset
AForceDrvCont	4Eh

9.3.3.1.3.2 Diagram



9.3.3.1.3.3 Fields

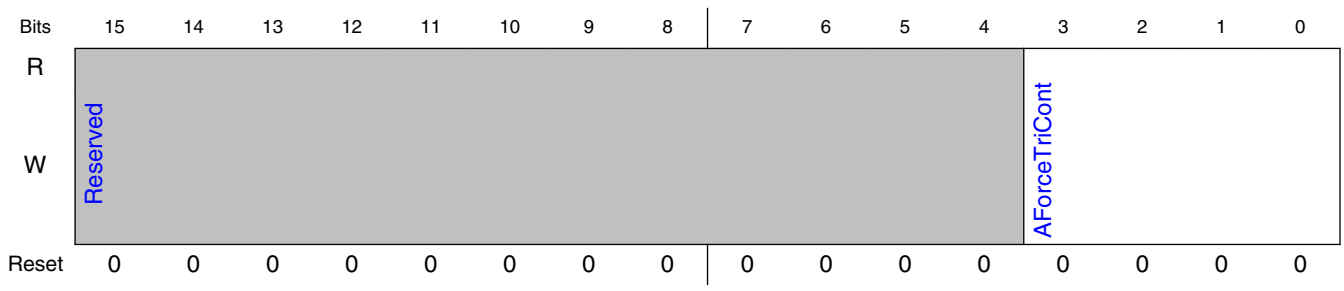
Field	Function
15-4 —	Reserved
3-0 AForceDrvCont	Force continuous drive, per-lane, of the ACX4 instance controlled by this register Setting this register will cause the PHY to drive the target lane when dfi_init_complete==1 Bit [0] = controls lane 0 of the target ACX4 block Bit [1] = controls lane 1 of the target ACX4 block Bit [2] = controls lane 2 of the target ACX4 block Bit [3] = controls lane 3 of the target ACX4 block

9.3.3.1.4 Force Address/Command Tristate (Lanes A3-A0) (AForceTriCont)

9.3.3.1.4.1 Offset

Register	Offset
AForceTriCont	50h

9.3.3.1.4.2 Diagram



9.3.3.1.4.3 Fields

Field	Function
15-4 —	Reserved
3-0 AForceTriCont	Force tristate control, per-lane, of the ACX4 instance controlled by this register Setting this register will cause the PHY to tristate the target lane when dfi_init_complete==1 Bit [0] = controls lane 0 of the target ACX4 block Bit [1] = controls lane 1 of the target ACX4 block Bit [2] = controls lane 2 of the target ACX4 block Bit [3] = controls lane 3 of the target ACX4 block

9.3.3.1.5 Address TX impedance controls (ATxImpedance)

9.3.3.1.5.1 Offset

Register	Offset
ATxImpedance	86h

9.3.3.1.5.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						ADrvStrenN					ADrvStrenP				
Reset	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

9.3.3.1.5.3 Fields

Field	Function
15-10 —	Reserved
9-5 ADrvStrenN	5 bit bus used to select the target pull down output impedance. 5 bit bus used to select the target pull down output impedance. Please Refer to Technology specific PHY DATABOOK for supported options Connects to the DrvStren pins of the driver thus: Csr_DrvStren120N[5:0] = csrADrvStrenN[4:0],1'b1 00000 = 120.0 Ohm 00001 = 60.0 Ohm 00011 = 40.0 Ohm 00111 = 30.0 Ohm

Table continues on the next page...

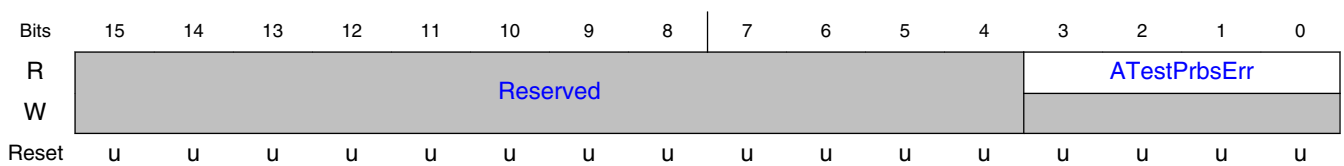
Field	Function
	01111 = 24.0 Ohm 11111 = 20.0 Ohm
4-0 ADrvStrenP	5 bit bus used to select the target pull up output impedance. 5 bit bus used to select the target pull up output impedance. Please Refer to Technology specific PHY DATABOOK for supported options Connects to the DrvStren pins of the driver thus: Csr_DrvStren120P[5:0] = csrADrvStrenP[4:0],1'b1 00000 = 120.0 Ohm 00001 = 60.0 Ohm 00011 = 40.0 Ohm 00111 = 30.0 Ohm 01111 = 24.0 Ohm 11111 = 20.0 Ohm

9.3.3.1.6 Address Loopback PRBS Error status for an entire ACX4 block (ATestPrbsErr)

9.3.3.1.6.1 Offset

Register	Offset
ATestPrbsErr	A6h

9.3.3.1.6.2 Diagram



9.3.3.1.6.3 Fields

Field	Function
15-4 —	Reserved
3-0	Overall error indicator for each prbs bump checker.

Field	Function
ATestPrbsErr	Overall error indicator for each prbs bump checker. Bit[0] = Lane0 Error Status 1 = errors found, 0 = no errors Bit[1] = Lane1 Error Status Bit[2] = Lane2 Error Status Bit[3] = Lane3 Error Status

9.3.3.1.7 Address TX slew rate and predriver controls (ATxSlewRate)

9.3.3.1.7.1 Offset

Register	Offset
ATxSlewRate	AAh

9.3.3.1.7.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					ATxPreDrvMode			ATxPreN				ATxPreP			
W																
Reset	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1

9.3.3.1.7.3 Fields

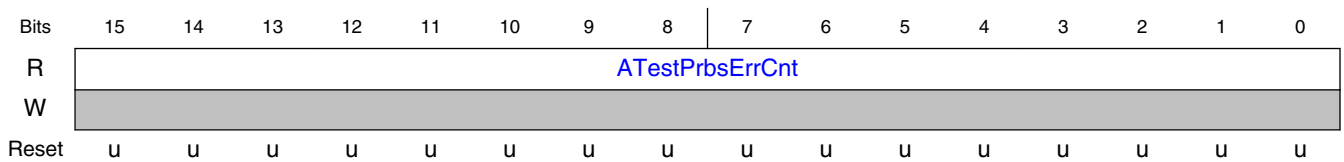
Field	Function
15-11 —	Reserved
10-8 ATxPreDrvMode	Controls predrivers to adjust timing of turn-on and turn-off of pull-up and pull-down segments.
7-4 ATxPreN	4 bit binary trim for the driver pull down slew rate. 4 bit binary trim for the driver pull down slew rate. 4'b0000 has a slower slew rate than 4'b1111
3-0 ATxPreP	4 bit binary trim for the driver pull up slew rate. 4 bit binary trim for the driver pull up slew rate. 4'b0000 has a slower slew rate than 4'b1111

9.3.3.1.8 Address Loopback Test Result register (ATestPrbsErrCnt)

9.3.3.1.8.1 Offset

Register	Offset
ATestPrbsErrCnt	ACh

9.3.3.1.8.2 Diagram



9.3.3.1.8.3 Fields

Field	Function
15-0 ATestPrbsErrCnt	Overall error indicator for each prbs bump checker. Overall error indicator for each prbs bump checker. Bit [3:0] = Lane0 Error Count Bit [7:4] = Lane1 Error Count Bit [11:8] = Lane2 Error Count Bit [15:12] = Lane3 Error Count

9.3.3.1.9 Address/Command Delay, per pstate. (ATxDly_p0)

9.3.3.1.9.1 Offset

Register	Offset
ATxDly_p0	100h

9.3.3.1.9.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								ATxDly_p0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.1.9.3 Fields

Field	Function
15-7 —	Reserved
6-0 ATxDly_p0	Trained for LPDDR3/4 to generate timed address and command signals to the DRAMs, per ACX4. Trained for LPDDR3/4 to generate timed address and command signals to the DRAMs, per ACX4. For DDR3/DDR4, it is recommended that address and command ACX4 have it ATxDly=0x00 The four signals generated by an ACX4 will have common timing. ATxDly[6] is the coarse delay, ie one unit of delay is 1 UI. ATxDly[5] is reserved. ATxDly[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. The target ACX4 used for memory clock generation, must have its ATxDly configured to 0x00.

9.3.3.1.10 Address/Command Delay, per pstate. (ATxDly_p1)

9.3.3.1.10.1 Offset

Register	Offset
ATxDly_p1	20_0100h

9.3.3.1.10.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								ATxDly_p1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.1.10.3 Fields

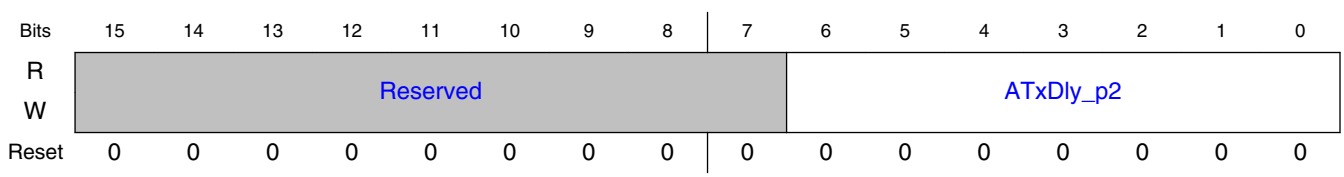
Field	Function
15-7 —	Reserved
6-0 ATxDly_p1	Trained for LPDDR3/4 to generate timed address and command signals to the DRAMs, per ACX4. Trained for LPDDR3/4 to generate timed address and command signals to the DRAMs, per ACX4. For DDR3/DDR4, it is recommended that address and command ACX4 have it ATxDly=0x00 The four signals generated by an ACX4 will have common timing. ATxDly[6] is the coarse delay, ie one unit of delay is 1 UI. ATxDly[5] is reserved. ATxDly[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. The target ACX4 used for memory clock generation, must have its ATxDly configured to 0x00.

9.3.3.1.11 Address/Command Delay, per pstate. (ATxDly_p2)

9.3.3.1.11.1 Offset

Register	Offset
ATxDly_p2	40_0100h

9.3.3.1.11.2 Diagram



9.3.3.1.11.3 Fields

Field	Function
15-7 —	Reserved
6-0 ATxDly_p2	Trained for LPDDR3/4 to generate timed address and command signals to the DRAMs, per ACX4. Trained for LPDDR3/4 to generate timed address and command signals to the DRAMs, per ACX4. For DDR3/DDR4, it is recommended that address and command ACX4 have it ATxDly=0x00 The four signals generated by an ACX4 will have common timing.

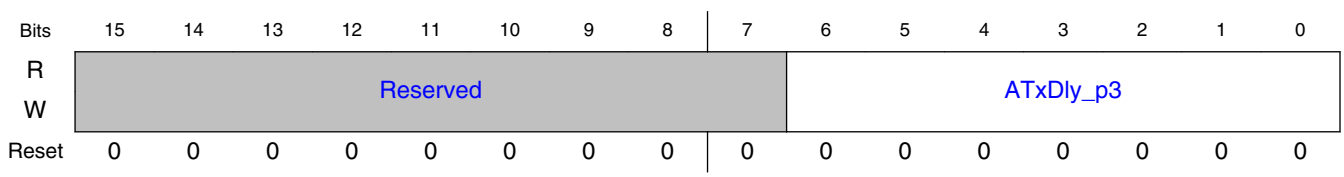
Field	Function
	ATxDly[6] is the coarse delay, ie one unit of delay is 1 UI. ATxDly[5] is reserved. ATxDly[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. The target ACX4 used for memory clock generation, must have its ATxDly configured to 0x00.

9.3.3.1.12 Address/Command Delay, per pstate. (ATxDly_p3)

9.3.3.1.12.1 Offset

Register	Offset
ATxDly_p3	60_0100h

9.3.3.1.12.2 Diagram



9.3.3.1.12.3 Fields

Field	Function
15-7 —	Reserved
6-0 ATxDly_p3	Trained for LPDDR3/4 to generate timed address and command signals to the DRAMs, per ACX4. Trained for LPDDR3/4 to generate timed address and command signals to the DRAMs, per ACX4. For DDR3/DDR4, it is recommended that address and command ACX4 have it ATxDly=0x00 The four signals generated by an ACX4 will have common timing. ATxDly[6] is the coarse delay, ie one unit of delay is 1 UI. ATxDly[5] is reserved. ATxDly[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. The target ACX4 used for memory clock generation, must have its ATxDly configured to 0x00.

9.3.3.2 DWC_DDRPHYA_APBONLY register descriptions

9.3.3.2.1 DWC_DDRPHYA_APBONLY Memory map

DDR4/3 PHY address block

DWC_DDRPHYA_APBONLY0 base address: D_0000h

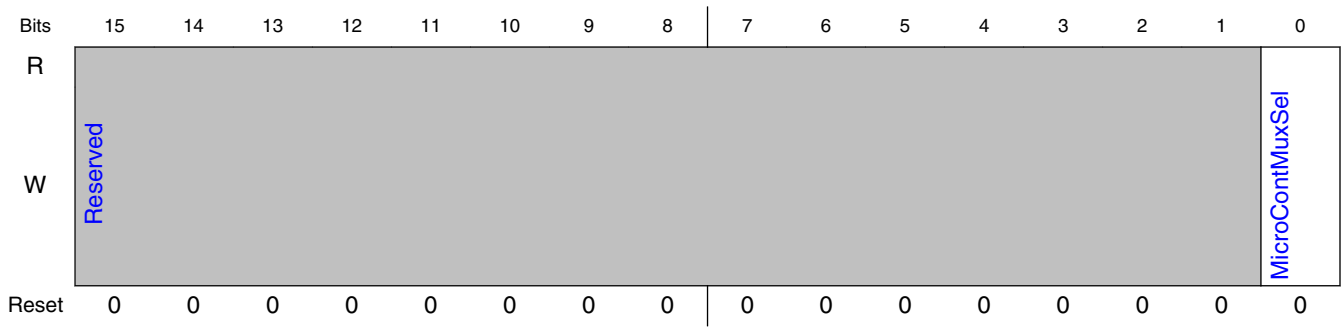
Offset	Register	Width (In bits)	Access	Reset value
0h	PMU Config Mux Select (MicroContMuxSel)	16	RW	0000h
8h	PMU/Controller Protocol - Controller Read-only Shadow (UctShadowRegs)	16	RO	Table 9-4
60h	Reserved. (DctWriteOnly)	16	RW	0000h
62h	DCT downstream mailbox protocol CSR. (DctWriteProt)	16	RW	0001h
64h	Read-only view of the csr UctDatWriteOnly (UctWriteOnlyShadow)	16	RO	Table 9-4
68h	Read-only view of the csr UctDatWriteOnly (UctDatWriteOnlyShadow)	16	RO	Table 9-4
6Eh	Number of DfiClk ticks required for valid csr Rd Data. (DfiCfgRdDataValidTicks)	16	RW	0006h
132h	Controls reset and clock shutdown on the local microcontroller (MicroReset)	16	RW	0001h
1F4h	dfi_init_complete - Controller Read-only Shadow (DfiInitCompleteShadow)	16	RO	Table 9-4

9.3.3.2.2 PMU Config Mux Select (MicroContMuxSel)

9.3.3.2.2.1 Offset

Register	Offset
MicroContMuxSel	0h

9.3.3.2.2.2 Diagram



9.3.3.2.2.3 Fields

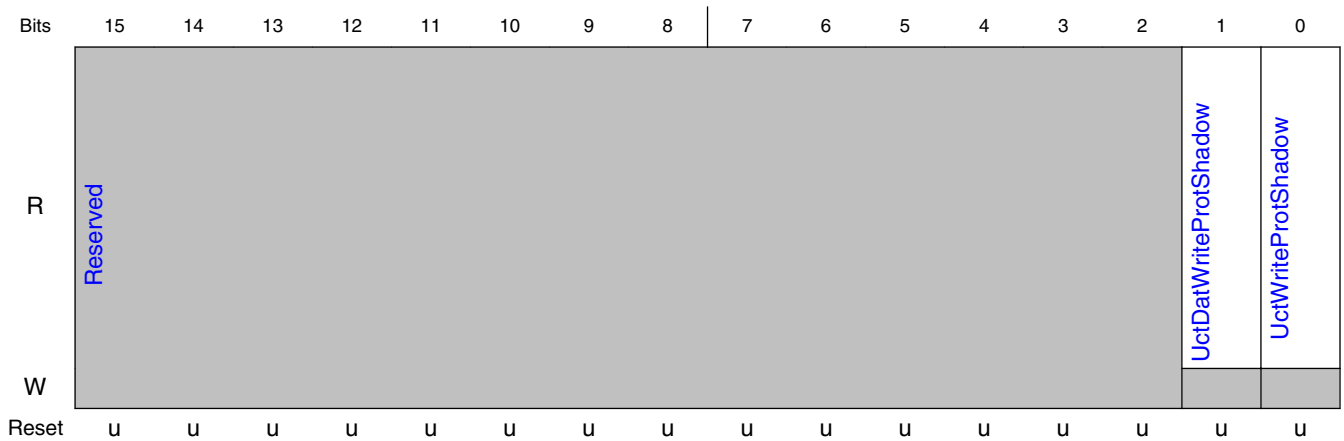
Field	Function
15-1 —	Reserved
0 MicroContMuxSel	This register controls access to the PHY configuration registers. This register controls access to the PHY configuration registers. 1 = MicroController/PHY Init Engine has control of csr bus. 0 = MicroController/PHY Init Engine csr requests are ignored.

9.3.3.2.3 PMU/Controller Protocol - Controller Read-only Shadow (UctShadowRegs)

9.3.3.2.3.1 Offset

Register	Offset
UctShadowRegs	8h

9.3.3.2.3.2 Diagram



9.3.3.2.3.3 Fields

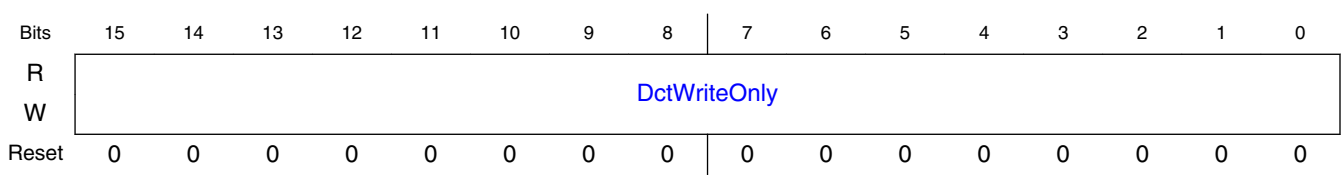
Field	Function
15-2 —	Reserved
1 UctDatWriteProtShadow	Reserved.
0 UctWriteProtShadow	When set to 0, the PMU has a message for the user

9.3.3.2.4 Reserved. (DctWriteOnly)

9.3.3.2.4.1 Offset

Register	Offset
DctWriteOnly	60h

9.3.3.2.4.2 Diagram



9.3.3.2.4.3 Fields

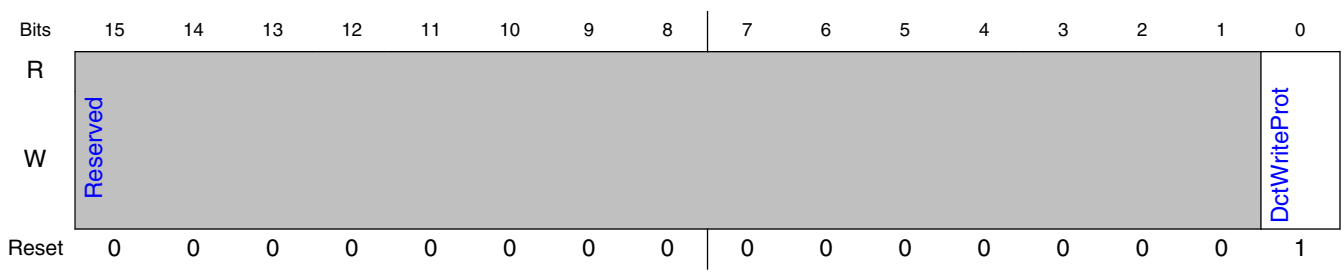
Field	Function
15-0 DctWriteOnly	Reserved.

9.3.3.2.5 DCT downstream mailbox protocol CSR. (DctWriteProt)

9.3.3.2.5.1 Offset

Register	Offset
DctWriteProt	62h

9.3.3.2.5.2 Diagram



9.3.3.2.5.3 Fields

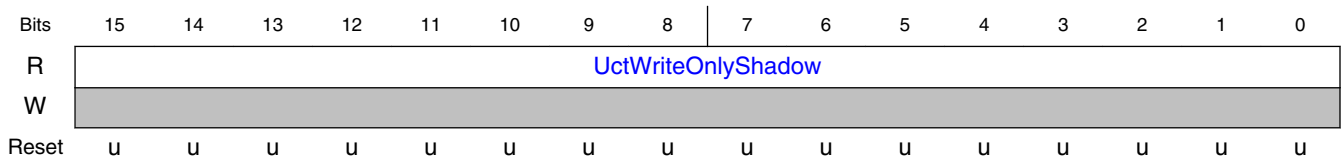
Field	Function
15-1 —	Reserved
0 DctWriteProt	By setting this register to 0, the user acknowledges the receipt of the message.

9.3.3.2.6 Read-only view of the csr UctDatWriteOnly (UctWriteOnlyShadow)

9.3.3.2.6.1 Offset

Register	Offset
UctWriteOnlyShadow	64h

9.3.3.2.6.2 Diagram



9.3.3.2.6.3 Fields

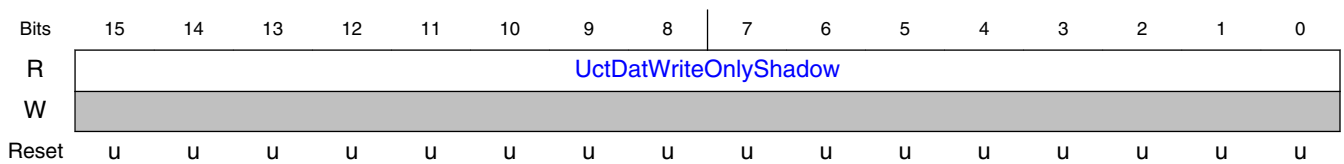
Field	Function
15-0	Used to pass the message ID for major messages.
UctWriteOnlyShadow	Used to pass the message ID for major messages. Also used to pass the lower 16 bits for streaming messages.

9.3.3.2.7 Read-only view of the csr UctDatWriteOnly (UctDatWriteOnlyShadow)

9.3.3.2.7.1 Offset

Register	Offset
UctDatWriteOnlyShadow	68h

9.3.3.2.7.2 Diagram



9.3.3.2.7.3 Fields

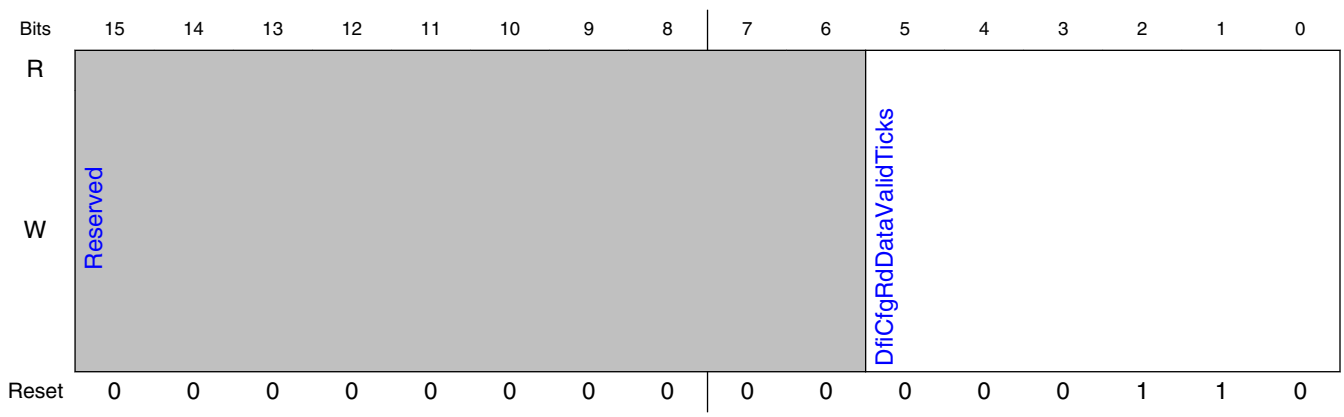
Field	Function
15-0 UctDatWriteOnlyShadow	Used to pass the upper 16 bits for streaming messages. Used to pass the upper 16 bits for streaming messages. Not used in passing major messages.

9.3.3.2.8 Number of DfiClk ticks required for valid csr Rd Data. (DfiCfgRdDataValidTicks)

9.3.3.2.8.1 Offset

Register	Offset
DfiCfgRdDataValidTicks	6Eh

9.3.3.2.8.2 Diagram



9.3.3.2.8.3 Fields

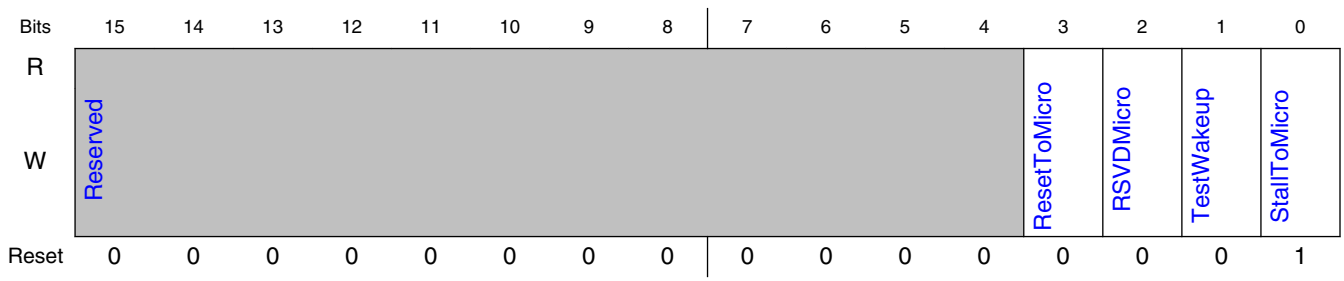
Field	Function
15-6 —	Reserved
5-0 DfiCfgRdDataValidTicks	Roundtrip delay of a register read access. Roundtrip delay of a register read access. This value must not be changed from its reset value.

9.3.3.2.9 Controls reset and clock shutdown on the local microcontroller (MicroReset)

9.3.3.2.9.1 Offset

Register	Offset
MicroReset	132h

9.3.3.2.9.2 Diagram



9.3.3.2.9.3 Fields

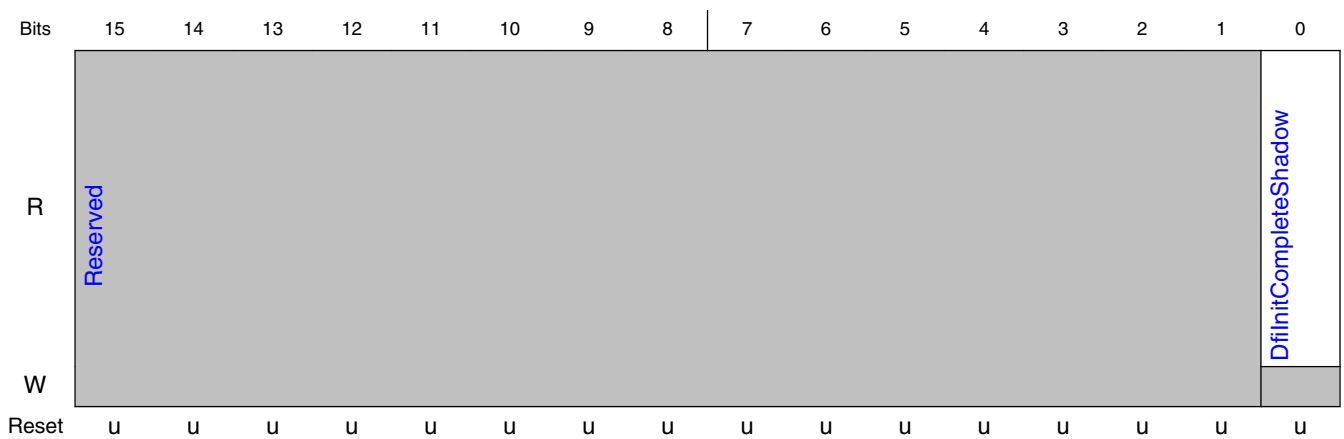
Field	Function
15-4 —	Reserved
3 ResetToMicro	Set this bit to apply synchronous reset to the microcontroller. Set this bit to apply synchronous reset to the microcontroller. Clear this bit to release reset to the microcontroller. This reset initializes the program counter to begin execution from the boot vector. This reset also clears the interrupt sticky bits.
2 RSVDMicro	RSVD
1 TestWakeup	Reserved Reserved. Must always be set to 0.
0 StallToMicro	Set this bit to stall the microcontroller by hardware. Set this bit to stall the microcontroller by hardware. Clear this bit to allow the microcontroller to continue executing from its current program counter location. Typically, this bit is used at power up to hold the program counter at the boot vector while BIOS loads the microcontroller program code. While stalled, the microcontroller clocks are gated off for power reduction.

9.3.3.2.10 dfi_init_complete - Controller Read-only Shadow (DfiInitCompleteShadow)

9.3.3.2.10.1 Offset

Register	Offset
DfiInitCompleteShadow	1F4h

9.3.3.2.10.2 Diagram



9.3.3.2.10.3 Fields

Field	Function
15-1 —	Reserved
0 DfiInitCompleteShadow	This csr presents a read-only view (a shadow) of the Register DfiInitComplete which is used by the sequencer to control the state of dfi_init_complete. This csr presents a read-only view (a shadow) of the Register DfiInitComplete which is used by the sequencer to control the state of dfi_init_complete. The value in this Register is not affected by the BlockSeq0BAck field of the SequencerOverride register. While the Register MicroContMuxSel is set, access to this Shadow register will not steal config bus bandwidth from the micro controller. That is polling won't have a performance penalty.

9.3.3.3 DWC_DDRPHYA_DBYTE register descriptions

9.3.3.3.1 DWC_DDRPHYA_DBYTE Memory map

DDR4/3 PHY address block

DWC_DDRPHYA_DBYTE0 base address: 1_0000h

DWC_DDRPHYA_DBYTE1 base address: 1_1000h

DWC_DDRPHYA_DBYTE2 base address: 1_2000h

DWC_DDRPHYA_DBYTE3 base address: 1_3000h

Offset	Register	Width (In bits)	Access	Reset value
0h	DBYTE Module Disable (DbyteMiscMode)	16	RW	Table 9-4
34h	Digital Observation Pin control (MtestMuxSel)	16	RW	0000h
40h	DFI MaxReadLatency (DFIMRL_p0)	16	RW	0006h
60h - 1060h	VrefDAC1 control for DQ Receiver (used only when DFE is enabled in DDR4) (VrefDAC1_r0 - VrefDAC1_r8)	16	RW	0000h
80h - 1080h	VrefDAC0 control for DQ Receiver (VrefDAC0_r0 - VrefDAC0_r8)	16	RW	0000h
82h - 60_0282h	Data TX impedance controls (TxImpedanceCtrl0_b0_p0 - TxImpedanceCtrl0_b1_p3)	16	RW	0FFFh
86h - 60_0286h	Dq/Dqs receiver control (DqDqsRcvCntrl_b0_p0 - DqDqsRcvCntrl_b1_p3)	16	RW	05B0h
90h - 60_0090h	Tx dq driver equalization mode controls. (TxEqualizationMode_p0 - TxEqualizationMode_p3)	16	RW	0000h
92h - 60_0292h	TX impedance controls (TxImpedanceCtrl1_b0_p0 - TxImpedanceCtrl1_b1_p3)	16	RW	0FFFh
94h	Dq/Dqs receiver control (DqDqsRcvCntrl1)	16	RW	0400h
96h - 60_0296h	TX equalization impedance controls (TxImpedanceCtrl2_b0_p0 - TxImpedanceCtrl2_b1_p3)	16	RW	0000h
98h - 60_0098h	Dq/Dqs receiver control (DqDqsRcvCntrl2_p0 - DqDqsRcvCntrl2_p3)	16	RW	0000h
9Ah - 60_029Ah	TX ODT driver strength control (TxOdtDrvStren_b0_p0 - TxOdtDrvStren_b1_p3)	16	RW	0000h
ACh	Status of RX FIFO Consistency Checks (RxFifoCheckStatus)	16	RO	Table 9-4
AEh	Contains the captured values associated with an RxFifo consistency error (RxFifoCheckErrValues)	16	RO	Table 9-4
B0h	Data Receive FIFO Pointer Values (RxFifoInfo)	16	RO	Table 9-4
B2h	RX FIFO visibility (RxFifoVisibility)	16	RW	0000h
B4h	RX FIFO contents, lane[3:0] (RxFifoContentsDQ3210)	16	RO	Table 9-4
B6h	RX FIFO contents, lane[7:4] (RxFifoContentsDQ7654)	16	RO	Table 9-4
B8h	RX FIFO contents, dbi (RxFifoContentsDBI)	16	RO	Table 9-4

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
BEh - 60_02BEh	TX slew rate controls (TxSlewRate_b0_p0 - TxSlewRate_b1_p3)	16	RW	07FFh
D0h - 10D0h	Read DQ per-bit BDL delay (Timing Group 0). (RxPBDlyTg0_r0 - RxPBDlyTg0_r8)	16	RW	0000h
D2h - 10D2h	Read DQ per-bit BDL delay (Timing Group 1). (RxPBDlyTg1_r0 - RxPBDlyTg1_r8)	16	RW	0000h
D4h - 10D4h	Read DQ per-bit BDL delay (Timing Group 2). (RxPBDlyTg2_r0 - RxPBDlyTg2_r8)	16	RW	0000h
D6h - 10D6h	Read DQ per-bit BDL delay (Timing Group 3). (RxPBDlyTg3_r0 - RxPBDlyTg3_r8)	16	RW	0000h
100h - 60_0300h	Trained Receive Enable Delay (For Timing Group 0) (RxEnDlyTg0_u0_p0 - RxEnDlyTg0_u1_p3)	16	RW	0100h
102h - 60_0302h	Trained Receive Enable Delay (For Timing Group 1) (RxEnDlyTg1_u0_p0 - RxEnDlyTg1_u1_p3)	16	RW	0100h
104h - 60_0304h	Trained Receive Enable Delay (For Timing Group 2) (RxEnDlyTg2_u0_p0 - RxEnDlyTg2_u1_p3)	16	RW	0100h
106h - 60_0306h	Trained Receive Enable Delay (For Timing Group 3) (RxEnDlyTg3_u0_p0 - RxEnDlyTg3_u1_p3)	16	RW	0100h
118h - 60_0318h	Trained Read DQS to RxClk Delay (Timing Group DEST=0). (RxClkDlyTg0_u0_p0 - RxClkDlyTg0_u1_p3)	16	RW	0010h
11Ah - 60_031Ah	Trained Read DQS to RxClk Delay (Timing Group DEST=1). (RxClkDlyTg1_u0_p0 - RxClkDlyTg1_u1_p3)	16	RW	0010h
11Ch - 60_031Ch	Trained Read DQS to RxClk Delay (Timing Group DEST=2). (RxClkDlyTg2_u0_p0 - RxClkDlyTg2_u1_p3)	16	RW	0010h
11Eh - 60_031Eh	Trained Read DQS to RxClk Delay (Timing Group DEST=3). (RxClkDlyTg3_u0_p0 - RxClkDlyTg3_u1_p3)	16	RW	0010h
120h - 60_0320h	Trained Read DQS_c to RxClkc Delay (Timing Group DEST=0). (RxClkcDlyTg0_u0_p0 - RxClkcDlyTg0_u1_p3)	16	RW	0010h
122h - 60_0322h	Trained Read DQS_c to RxClkc Delay (Timing Group DEST=0). (RxClkcDlyTg1_u0_p0 - RxClkcDlyTg1_u1_p3)	16	RW	0010h
124h - 60_0324h	Trained Read DQS_c to RxClkc Delay (Timing Group DEST=2). (RxClkcDlyTg2_u0_p0 - RxClkcDlyTg2_u1_p3)	16	RW	0010h
128h - 60_0328h	Trained Read DQS_c to RxClkc Delay (Timing Group DEST=3). (RxClkcDlyTg3_u0_p0 - RxClkcDlyTg3_u1_p3)	16	RW	0010h
140h - 14Eh	Maps Phy DQ lane to memory DQ0 (Dq0LnSel - Dq7LnSel)	16	RW	0000h
180h - 60_1180h	Write DQ Delay (Timing Group 0). (TxDqDlyTg0_r0_p0 - TxDqDlyTg0_r8_p3)	16	RW	0010h
182h - 60_1182h	Write DQ Delay (Timing Group 1). (TxDqDlyTg1_r0_p0 - TxDqDlyTg1_r8_p3)	16	RW	0010h
184h - 60_1184h	Write DQ Delay (Timing Group 2). (TxDqDlyTg2_r0_p0 - TxDqDlyTg2_r8_p3)	16	RW	0010h
186h - 60_1186h	Write DQ Delay (Timing Group 3). (TxDqDlyTg3_r0_p0 - TxDqDlyTg3_r8_p3)	16	RW	0010h
1A0h - 60_03A0h	Write DQS Delay (Timing Group DEST=0). (TxDqsDlyTg0_u0_p0 - TxDqsDlyTg0_u1_p3)	16	RW	0100h

Table continues on the next page...

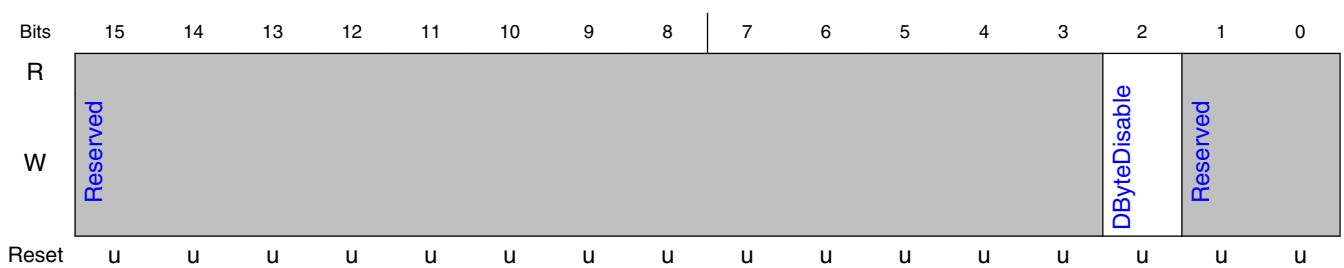
Offset	Register	Width (In bits)	Access	Reset value
1A2h - 60_03A2h	Write DQS Delay (Timing Group DEST=1). (TxDqsDlyTg1_u0_p0 - TxDqsDlyTg1_u1_p3)	16	RW	0100h
1A4h - 60_03A4h	Write DQS Delay (Timing Group DEST=2). (TxDqsDlyTg2_u0_p0 - TxDqsDlyTg2_u1_p3)	16	RW	0100h
1A6h - 60_03A6h	Write DQS Delay (Timing Group DEST=3). (TxDqsDlyTg3_u0_p0 - TxDqsDlyTg3_u1_p3)	16	RW	0100h
1C8h	Debug status of the DBYTE LCDL (DxLcdlStatus)	16	RO	Table 9-4
20_0040h	DFI MaxReadLatency (DFIMRL_p1)	16	RW	0006h
40_0040h	DFI MaxReadLatency (DFIMRL_p2)	16	RW	0006h
40_015Ch	DQS Oscillator Count inverse at time of training in LPDDR4 drift compensation (PptDqsCntInvTrnTg0_p2)	16	RW	0000h
40_015Eh	DQS Oscillator Count inverse at time of training in LPDDR4 drift compensation (PptDqsCntInvTrnTg1_p2)	16	RW	0000h
60_0040h	DFI MaxReadLatency (DFIMRL_p3)	16	RW	0006h
60_015Ch	DQS Oscillator Count inverse at time of training in LPDDR4 drift compensation (PptDqsCntInvTrnTg0_p3)	16	RW	0000h
60_015Eh	DQS Oscillator Count inverse at time of training in LPDDR4 drift compensation (PptDqsCntInvTrnTg1_p3)	16	RW	0000h

9.3.3.3.2 DBYTE Module Disable (DbyteMiscMode)

9.3.3.3.2.1 Offset

Register	Offset
DbyteMiscMode	0h

9.3.3.3.2.2 Diagram



9.3.3.3.2.3 Fields

Field	Function
15-3 —	Reserved
2 DByteDisable	Controls whether this DBYTE module is disabled. Controls whether this DBYTE module is disabled. If this DBYTE module is not enabled, it receives no clocks and remains in reset. 0 - Enable this DBYTE module 1 - Disable this DBYTE module This field should only be changed PHY initialization step C.
1-0 —	Reserved

9.3.3.3.3 Digital Observation Pin control (MtestMuxSel)

9.3.3.3.3.1 Offset

Register	Offset
MtestMuxSel	34h

9.3.3.3.3.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								MtestMuxSel							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.3.3.3 Fields

Field	Function
15-6 —	Reserved
5-0 MtestMuxSel	Controls for the 64-1 mux for asynchronous data to the Digital Observation Pin. Controls for the 64-1 mux for asynchronous data to the Digital Observation Pin. Encoding 6'h0 --> Drive 0 from this chiplet (allows flat 'OR' of pass-through information)

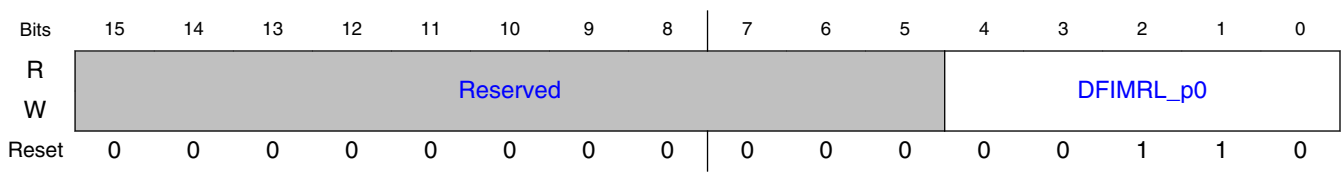
Field	Function
	Encoding 6'h01:1f --> Select local data from AC/DBYTE/MASTER macro Encoding 6'h20 --> Reserved (Drive 0 from macro into mux; Drive 0 from PUB synth logic path into mux) Encoding 6'h21:3f --> Select local data from PUB AC/DBYTE/MASTER synthesized logic Note: See PUB documentation for how, or if, the Digital Observation Pin is mapped to a physical bump in this configuration.

9.3.3.3.4 DFI MaxReadLatency (DFIMRL_p0)

9.3.3.3.4.1 Offset

Register	Offset
DFIMRL_p0	40h

9.3.3.3.4.2 Diagram



9.3.3.3.4.3 Fields

Field	Function
15-5 —	Reserved
4-0 DFIMRL_p0	This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. This is the value, in units of two mem clocks, between dfi_rddata_en and dfi_rddata_valid A unit change in the LSB is a change in MRL of two mem clocks.

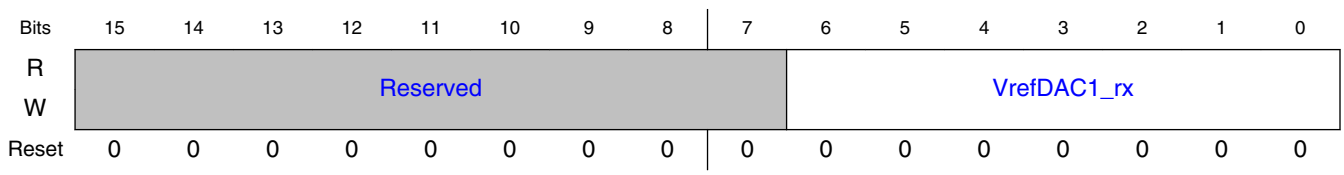
9.3.3.3.5 VrefDAC1 control for DQ Receiver (used only when DFE is enabled in DDR4) (VrefDAC1_r0 - VrefDAC1_r8)

9.3.3.3.5.1 Offset

For x = 0 to 8:

Register	Offset
VrefDAC1_rx	60h + (x × 200h)

9.3.3.3.5.2 Diagram



9.3.3.3.5.3 Fields

Field	Function
15-7 —	Reserved
6-0 VrefDAC1_rx	VrefDAC1 controls the alternate VREF setting for DFE (used only when DFE is enabled in DDR4) DAC control for rxdq cell internal VREF, trained by Firmware The VREF generators have different ranges, depending on the Mission Mode settings for DqDqsRcvCntrl::MajorMode,DqDqsRcvCntrl::ExtVrefRange 011,0 :: VREF = VDDQ*(0. VrefDAC1 controls the alternate VREF setting for DFE (used only when DFE is enabled in DDR4) DAC control for rxdq cell internal VREF, trained by Firmware The VREF generators have different ranges, depending on the Mission Mode settings for DqDqsRcvCntrl::MajorMode,DqDqsRcvCntrl::ExtVrefRange 011,0 :: VREF = VDDQ*(0.510 + VrefDAC1[6:0]*0.00345) 011,1 :: VREF = VDDQ*(0.453 + VrefDAC1[6:0]*0.00385) non-enumerated encodings are reserved Register Block Offset Address 0x030 is the VrefDAC1 for lane0. Register Block Offset Address 0x130 is the VrefDAC1 for lane1. Register Block Offset Address 0x230 is the VrefDAC1 for lane2. Register Block Offset Address 0x330 is the VrefDAC1 for lane3. Register Block Offset Address 0x430 is the VrefDAC1 for lane4. Register Block Offset Address 0x530 is the VrefDAC1 for lane5.

Field	Function
	Register Block Offset Address 0x630 is the VrefDAC1 for lane6. Register Block Offset Address 0x730 is the VrefDAC1 for lane7. Register Block Offset Address 0x830 is the VrefDAC1 for laneDBI.

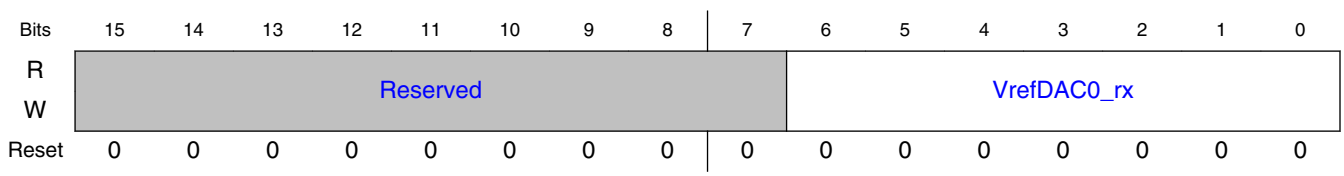
9.3.3.3.6 VrefDAC0 control for DQ Receiver (VrefDAC0_r0 - VrefDAC0_r8)

9.3.3.3.6.1 Offset

For x = 0 to 8:

Register	Offset
VrefDAC0_rx	80h + (x × 200h)

9.3.3.3.6.2 Diagram



9.3.3.3.6.3 Fields

Field	Function
15-7 —	Reserved
6-0 VrefDAC0_rx	PHY RX VREF DAC control for rxdq cell internal VREF, (used only when 2D training is enabled in LPDDR4,DDR4) DAC control for rxdq cell internal VREF, trained by Firmware The VREF generators have different ranges, depending on the Mission Mode settings for DqDqsRcvCntrl::MajorMode,DqDqsRcvCntrl::ExtVrefRange 000,0 :: VREF = VDDQ*(0. PHY RX VREF DAC control for rxdq cell internal VREF, (used only when 2D training is enabled in LPDDR4,DDR4) DAC control for rxdq cell internal VREF, trained by Firmware The VREF generators have different ranges, depending on the Mission Mode settings for DqDqsRcvCntrl::MajorMode,DqDqsRcvCntrl::ExtVrefRange 000,0 :: VREF = VDDQ*(0.287 + VrefDAC0[6:0]*0.00330) 000,1 :: VREF = VDDQ*(0.250 + VrefDAC0[6:0]*0.00385) 011,0 :: VREF = VDDQ*(0.510 + VrefDAC0[6:0]*0.00345)

Field	Function
	011,1 :: VREF = VDDQ*(0.453 + VrefDAC0[6:0]*0.00385) 010,0 :: VREF = VDDQ*(0.047 + VrefDAC0[6:0]*0.00367) non-enumerated encodings are reserved Register Block Offset Address 0x040 is the VrefDAC0 for lane0. Register Block Offset Address 0x140 is the VrefDAC0 for lane1. Register Block Offset Address 0x240 is the VrefDAC0 for lane2. Register Block Offset Address 0x340 is the VrefDAC0 for lane3. Register Block Offset Address 0x440 is the VrefDAC0 for lane4. Register Block Offset Address 0x540 is the VrefDAC0 for lane5. Register Block Offset Address 0x640 is the VrefDAC0 for lane6. Register Block Offset Address 0x740 is the VrefDAC0 for lane7. Register Block Offset Address 0x840 is the VrefDAC0 for laneDBI.

9.3.3.3.7 Data TX impedance controls (TxImpedanceCtrl0_b0_p0 - TxImpedanceCtrl0_b1_p3)

9.3.3.3.7.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
TxImpedanceCtrl0_bn_px	82h + (x × 20_0000h) + (n × 200h)

9.3.3.3.7.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				DrvStrenDqN						DrvStrenDqP					
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

9.3.3.3.7.3 Fields

Field	Function
15-12 —	Reserved

Table continues on the next page...

Field	Function
11-6 DrvStrenDqN	Please Refer to Technology specific PHY DATABOOK for supported options 6 bit bus used to select the target pull down output impedance. Please Refer to Technology specific PHY DATABOOK for supported options 6 bit bus used to select the target pull down output impedance. Connects to the DrvStren pins of the driver thus: Csr_DrvStren120N[2:0],Csr_DrvStren240N[1:0],Csr_DrvStren480N[0:0] = csrDrvStrenDqN See table of csr DrvStrenDqP for list of possible impedances.
5-0 DrvStrenDqP	Please Refer to Technology specific PHY DATABOOK for supported options 6 bit bus used to select the target pull down output impedance. Please Refer to Technology specific PHY DATABOOK for supported options 6 bit bus used to select the target pull down output impedance. Connects to the DrvStren pins of the driver thus: Csr_DrvStren120P[2:0], Csr_DrvStren240P[1:0],Csr_DrvStren480P[0:0] = csrDrvStrenDqP 000000 - HiZ. 000001 - 480.0 000010 - 240.0 000011 - 160.0 000110 - 120.0 000111 - 96.0 001010 - 80.0 001011 - 68.6 001110 - 60.0 001111 - 53.3 011010 - 48.0 011011 - 43.6 011110 - 40.0 011111 - 36.9 111010 - 34.3 111011 - 32.0 111110 - 30.0 111111 - 28.2

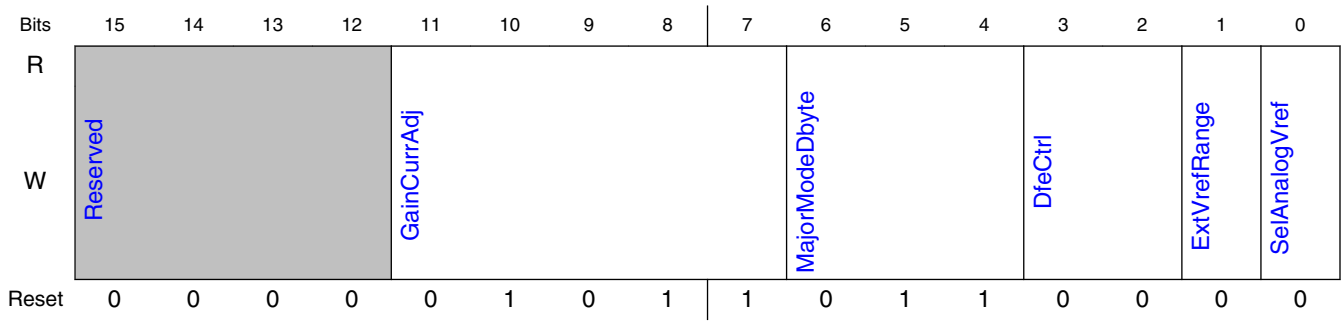
9.3.3.3.8 Dq/Dqs receiver control (DqDqsRcvCntrl_b0_p0 - DqDqsRcvCntrl_b1_p3)

9.3.3.3.8.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
DqDqsRcvCntrl_bn_px	86h + (x × 20_0000h) + (n × 200h)

9.3.3.3.8.2 Diagram



9.3.3.3.8.3 Fields

Field	Function
15-12 —	Reserved
11-7 GainCurrAdj	Adjust gain current of RX amplifier stage. Adjust gain current of RX amplifier stage. It is recommended to use default values for this CSR.
6-4 MajorModeDbyte	Selects the major mode of operation for the receiver. Selects the major mode of operation for the receiver. These settings are determined by PHY Configuration based on DRAM protocol 000 - DDR3 001 - reserved (room for LPDDR3) 010 - LPDDR4 011 - DDR4 100 - DDR3 Low-power/low-speed 101 - reserved (LPDDR3 low-power/low-speed) 110 - LPDDR4 low-power/low-speed 111 - DDR4 Low-power/low-speed
3-2 DfeCtrl	DFE may be used with MajorModeDbyte=011 only 00 - DFE off 01 - DFE on 10 - Train DFE0 Amplifier 11 - Train DFE1 Amplifier These settings are determined by PHY Training FW and should not be overridden.
1 ExtVrefRange	Extends the range available in the local per-bit VREF generator.
0 SelAnalogVref	Setting this signal high will force the local per-bit VREF generator to pass the global VREFA to the samplers.

9.3.3.3.9 Tx dq driver equalization mode controls. (TxEqualizationMode_p0 - TxEqualizationMode_p3)

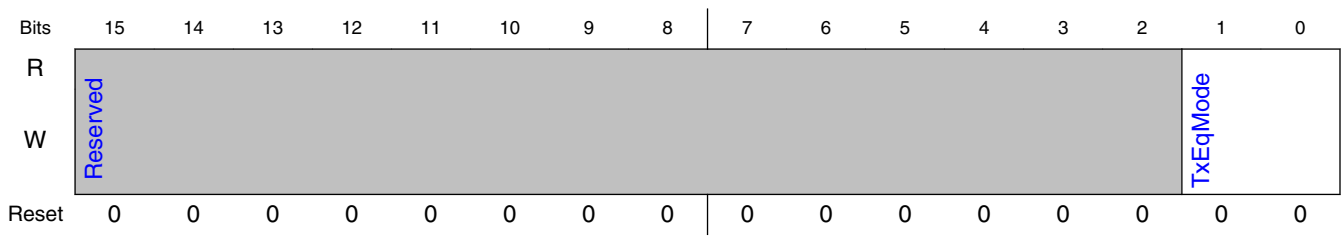
9.3.3.3.9.1 Offset

Register	Offset
TxEqualizationMode_p0	90h
TxEqualizationMode_p1	20_0090h
TxEqualizationMode_p2	40_0090h
TxEqualizationMode_p3	60_0090h

9.3.3.3.9.2 Function

Tx dq driver equalization mode controls. FFE, de-emphasis DDR43 PHY ONLY

9.3.3.3.9.3 Diagram



9.3.3.3.9.4 Fields

Field	Function
15-2 —	Reserved
1-0 TxEqMode	

9.3.3.3.10 TX impedance controls (TxImpedanceCtrl1_b0_p0 - TxImpedanceCtrl1_b1_p3)

9.3.3.3.10.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
TxImpedanceCtrl1_bn_px	92h + (x × 20_0000h) + (n × 200h)

9.3.3.3.10.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				DrvStrenFSDqN						DrvStrenFSDqP					
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

9.3.3.3.10.3 Fields

Field	Function
15-12 —	Reserved
11-6 DrvStrenFSDqN	Please Refer to Technology specific PHY DATABOOK for supported options 6 bit bus used to select the target pull up output impedance used in equalization. Please Refer to Technology specific PHY DATABOOK for supported options 6 bit bus used to select the target pull up output impedance used in equalization. Connects to the DrvStren pins of the driver thus: Csr_DrvStrenFS120N[2:0], Csr_DrvStrenFS240N[1:0], Csr_DrvStrenFS480N[0:0] = csrDrvStrenDqN See table of csr DrvStrenFSDqP for list of possible impedances.
5-0 DrvStrenFSDqP	Please Refer to Technology specific PHY DATABOOK for supported options 6 bit bus used to select the target pull up output impedance used in equalization. Please Refer to Technology specific PHY DATABOOK for supported options 6 bit bus used to select the target pull up output impedance used in equalization. Connects to the DrvStren pins of the driver thus: Csr_DrvStrenFS120P[2:0], Csr_DrvStrenFS240P[1:0], Csr_DrvStrenFS480P[0] = csrFSDrvStrenDq 000000 - HiZ. 000001 - 480.0 000010 - 240.0 000011 - 160.0 000110 - 120.0 000111 - 96.0 001010 - 80.0

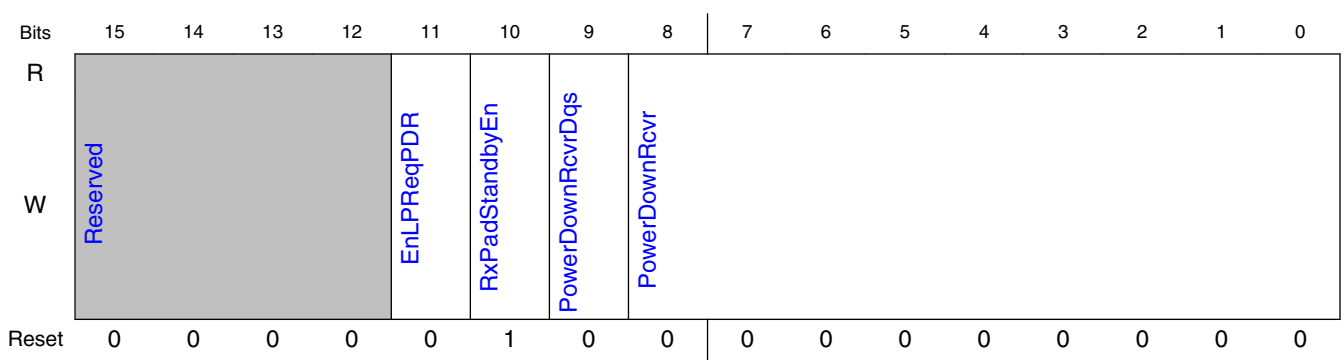
Field	Function
	001011 - 68.6 001110 - 60.0 001111 - 53.3 011010 - 48.0 011011 - 43.6 011110 - 40.0 011111 - 36.9 111010 - 34.3 111011 - 32.0 111110 - 30.0 111111 - 28.2

9.3.3.3.11 Dq/Dqs receiver control (DqDqsRcvCntrl1)

9.3.3.3.11.1 Offset

Register	Offset
DqDqsRcvCntrl1	94h

9.3.3.3.11.2 Diagram



9.3.3.3.11.3 Fields

Field	Function
15-12 —	Reserved

Table continues on the next page...

Field	Function
11 EnLPReqPDR	Reserved
10 RxPadStandbyEn	Enables the rxdq/rxdqs StandBy power savings, per pad-group.
9 PowerDownRcvrDqs	Active high signal which powers down the receiver. Active high signal which powers down the receiver. After this pin is deasserted the receiver cannot be used for a minimum of 100 ns. This control powers down both the upper and lower DQS receivers. If Register X4TG[3:0]=0000, then the upper DQS receiver is powered down.
8-0 PowerDownRcvr	Active high signal which powers down the receiver. Active high signal which powers down the receiver. [7:0 -> dq[7:0], 8 -> dbi] After this pin is deasserted the receiver cannot be used for a minimum of 100 ns.

9.3.3.3.12 TX equalization impedance controls (TxImpedanceCtrl2_b0_p0 - TxImpedanceCtrl2_b1_p3)

9.3.3.3.12.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
TxImpedanceCtrl2_bn_px	96h + (x × 20_0000h) + (n × 200h)

9.3.3.3.12.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				DrvStrenEQLoDqN						DrvStrenEQHiDqP					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.3.12.3 Fields

Field	Function
15-12 —	Reserved

Table continues on the next page...

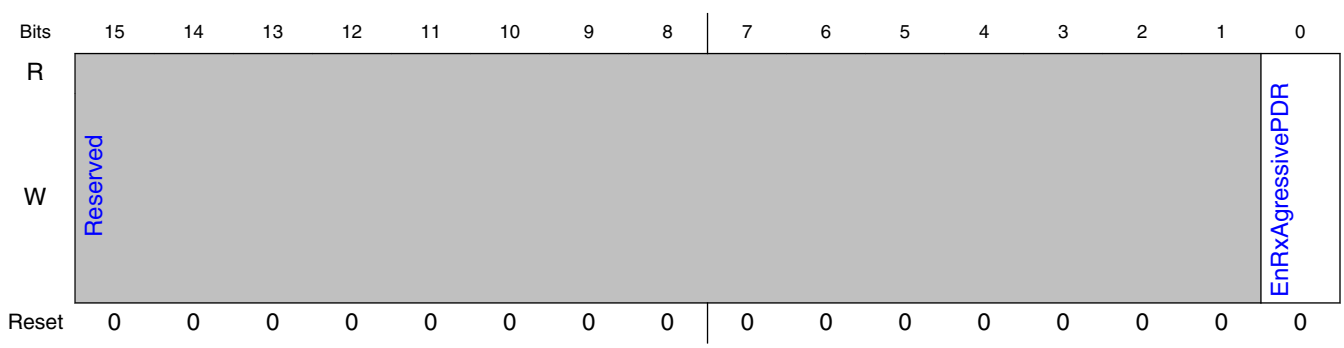
Field	Function
11-6 DrvStrenEQLoDqN	Please Refer to Technology specific PHY DATABOOK for supported options 6 bit bus used to select the target pull down output impedance used in equalization. Please Refer to Technology specific PHY DATABOOK for supported options 6 bit bus used to select the target pull down output impedance used in equalization. Connects to the DrvStren pins of the driver thus: Csr_DrvStrenEQLo120N[2:0], Csr_DrvStrenEQLo240N[1:0], Csr_DrvStrenEQLo480P[0] = csrDrvStrenEQLoDq
5-0 DrvStrenEQHiDqP	Please Refer to Technology specific PHY DATABOOK for supported options 6 bit bus used to select the target pull up output impedance used in equalization. Please Refer to Technology specific PHY DATABOOK for supported options 6 bit bus used to select the target pull up output impedance used in equalization. Connects to the DrvStren pins of the driver thus: Csr_DrvStrenEQHi120P[2:0], Csr_DrvStrenEQHi240P[1:0], Csr_DrvStrenEQHi480P[0] = csrDrvStrenEQHiDq

9.3.3.3.13 Dq/Dqs receiver control (DqDqsRcvCntrl2_p0 - DqDqsRcvCntrl2_p3)

9.3.3.3.13.1 Offset

Register	Offset
DqDqsRcvCntrl2_p0	98h
DqDqsRcvCntrl2_p1	20_0098h
DqDqsRcvCntrl2_p2	40_0098h
DqDqsRcvCntrl2_p3	60_0098h

9.3.3.3.13.2 Diagram



9.3.3.3.13.3 Fields

Field	Function
15-1 —	Reserved
0 EnRxAgressivePDR	reserved

9.3.3.3.14 TX ODT driver strength control (TxOdtDrvStren_b0_p0 - TxOdtDrvStren_b1_p3)

9.3.3.3.14.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
TxOdtDrvStren_bn_px	9Ah + (x × 20_0000h) + (n × 200h)

9.3.3.3.14.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				ODTStrenN						ODTStrenP					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.3.14.3 Fields

Field	Function
15-12 —	Reserved
11-6 ODTStrenN	Selects the ODT pull-down impedance. Selects the ODT pull-down impedance.. Please Refer to Technology specific PHY DATABOOK for supported options ODTStren120N[2:0], ODTStren120N[1:0],ODTStren480N[0:0] 000_00_0 = High Impedance 000_00_1 = 480 ohms 000_01_0 = 240 ohms

Table continues on the next page...

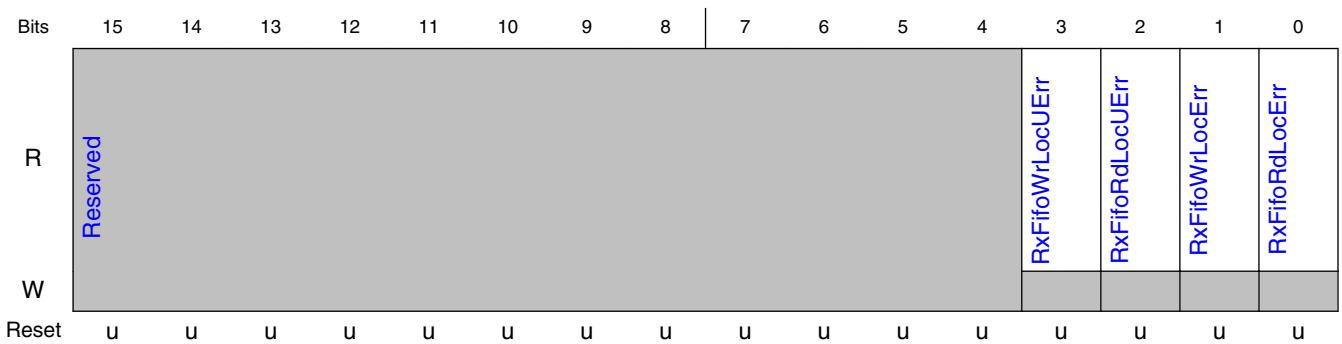
Field	Function
	000_01_1 = 160 ohms 001_00_0 = 120 ohms 001_00_1 = 96.0 ohms 001_01_0 = 80.0 ohms 001_01_1 = 68.6 ohms 011_00_0 = 60.0 ohms 011_00_1 = 53.3 ohms 011_01_0 = 48.0 ohms 011_01_1 = 43.6 ohms 111_00_0 = 40.0 ohms 111_00_1 = 36.9 ohms 111_01_0 = 34.3 ohms 111_01_1 = 32.0 ohms 111_11_0 = 30.0 ohms 111_11_1 = 28.2 ohms
5-0 ODTStrenP	Selects the ODT pull-up impedance. Selects the ODT pull-up impedance. Please Refer to Technology specific PHY DATABOOK for supported options ODTStren120P[2:0], ODTStren240P[1:0],ODTStren480P[0:0] 000_00_0 = High Impedance 000_00_1 = 480 ohms 000_01_0 = 240 ohms 000_01_1 = 160 ohms 001_00_0 = 120 ohms 001_00_1 = 96.0 ohms 001_01_0 = 80.0 ohms 001_01_1 = 68.6 ohms 011_00_0 = 60.0 ohms 011_00_1 = 53.3 ohms 011_01_0 = 48.0 ohms 011_01_1 = 43.6 ohms 111_00_0 = 40.0 ohms 111_00_1 = 36.9 ohms 111_01_0 = 34.3 ohms 111_01_1 = 32.0 ohms 111_11_0 = 30.0 ohms 111_11_1 = 28.2 ohms

9.3.3.3.15 Status of RX FIFO Consistency Checks (RxFifoCheckStatus)

9.3.3.3.15.1 Offset

Register	Offset
RxFifoCheckStatus	ACh

9.3.3.3.15.2 Diagram



9.3.3.3.15.3 Fields

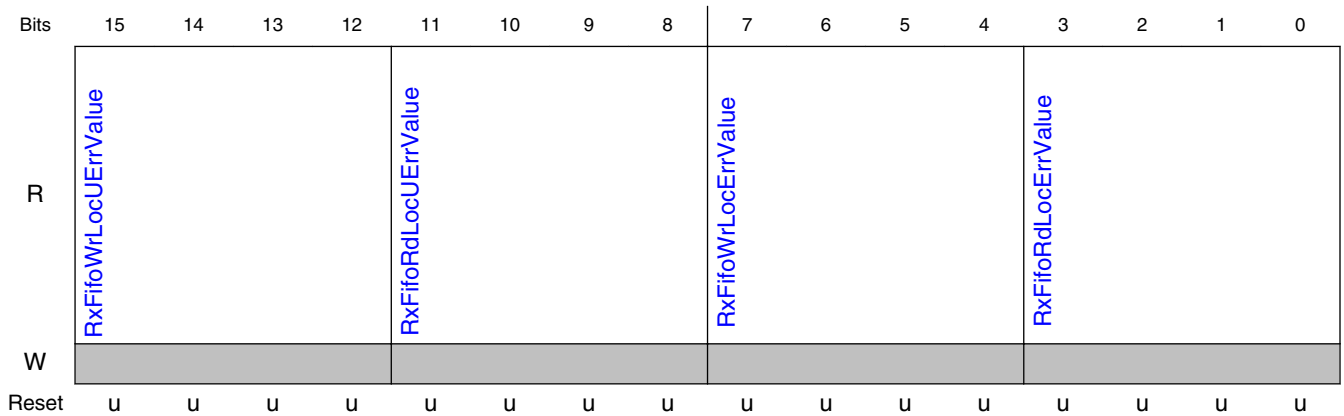
Field	Function
15-4 —	Reserved
3 RxFifoWrLocUErr	If set, the write pointer (DQS side) on the read FIFO associated with data bits [7:4] has a non-zero value at least once.
2 RxFifoRdLocUErr	If set, the read pointer (DFI side) on the read FIFO associated with data bits [7:4] has a non-zero value at least once.
1 RxFifoWrLocErr	If set, the write pointer (DQS side) on the read FIFO associated with data bits [3:0] has a non-zero value at least once.
0 RxFifoRdLocErr	If set, the read pointer (DFI side) on the read FIFO associated with data bits [3:0] had a non-zero value at least once.

9.3.3.3.16 Contains the captured values associated with an RxFifo consistency error (RxFifoCheckErrValues)

9.3.3.3.16.1 Offset

Register	Offset
RxFifoCheckErrValues	AEh

9.3.3.3.16.2 Diagram



9.3.3.3.16.3 Fields

Field	Function
15-12 RxFifoWrLocUErrValue	The first error value captured for the write pointer (DQS side) on the read FIFO associated with data bits [7:4];
11-8 RxFifoRdLocUErrValue	The first error value captured for the read pointer (DFI side) on the read FIFO associated with data bits [7:4];
7-4 RxFifoWrLocErrValue	The first error value captured for the write pointer (DQS side) on the read FIFO associated with data bits [3:0];
3-0 RxFifoRdLocErrValue	The first error value captured for the read pointer (DFI side) on the read FIFO associated with data bits [3:0];

9.3.3.3.17 Data Receive FIFO Pointer Values (RxFifoInfo)

9.3.3.3.17.1 Offset

Register	Offset
RxFifoInfo	B0h

9.3.3.3.17.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RxFifoWrLocU				RxFifoRdLocU				RxFifoWrLoc				RxFifoRdLoc			
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

9.3.3.3.17.3 Fields

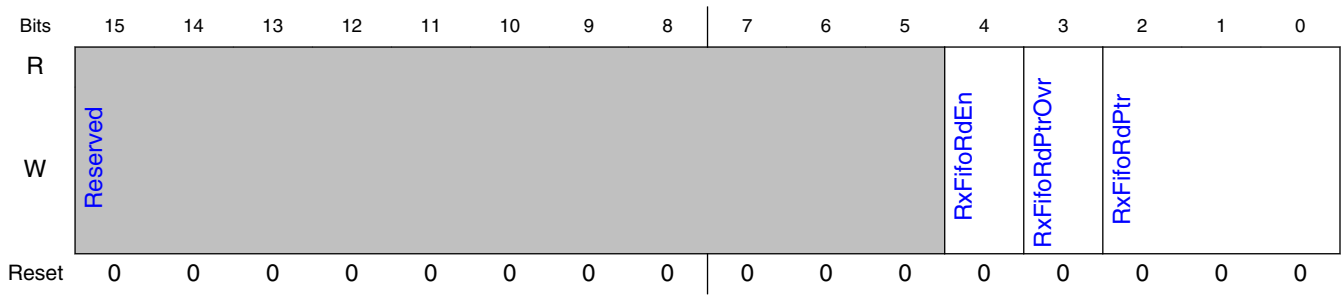
Field	Function
15-12 RxFifoWrLocU	The Mission mode write pointer of the upper-nibble Rx fifo.
11-8 RxFifoRdLocU	The Mission mode read pointer of the upper-nibble Rx fifo.
7-4 RxFifoWrLoc	The Mission mode write pointer of the lower-nibble Rx fifo.
3-0 RxFifoRdLoc	The Mission mode read pointer of the lower-nibble Rx fifo.

9.3.3.3.18 RX FIFO visibility (RxFifoVisibility)

9.3.3.3.18.1 Offset

Register	Offset
RxFifoVisibility	B2h

9.3.3.3.18.2 Diagram



9.3.3.3.18.3 Fields

Field	Function
15-5 —	Reserved
4 RxFifoRdEn	Pulse set 0-->1-->0 this bit to capture the Fifo Contents.
3 RxFifoRdPtrOvr	0 : Normal operation - mission mode read pointer is enabled 1 : Override - Control of the rx fifo read pointer is ceded to CSR RxFifoRdPtr.
2-0 RxFifoRdPtr	If CSR RxFifoRdPtrOVr is set, then this CSR selects the rxfifo entry is visible in CSR This 3b field addresses 4b units of the 8x4b (32entry) fifo; that is, rdfifo_nibble_address[2:0]=csrRxFifoRdPtr[2:0] For example, Register RxFifoRdPtr[2:0]=2 will enable reading bit-entries 11. If CSR RxFifoRdPtrOVr is set, then this CSR selects the rxfifo entry is visible in CSR This 3b field addresses 4b units of the 8x4b (32entry) fifo; that is, rdfifo_nibble_address[2:0]=csrRxFifoRdPtr[2:0] For example, Register RxFifoRdPtr[2:0]=2 will enable reading bit-entries 11..8. The exact location of read data will depend on the prior history of reads and PHY initialization.

9.3.3.3.19 RX FIFO contents, lane[3:0] (RxFifoContentsDQ3210)

9.3.3.3.19.1 Offset

Register	Offset
RxFifoContentsDQ3210	B4h

9.3.3.3.19.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RxFifoContentsDQ3210															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

9.3.3.3.19.3 Fields

Field	Function
15-0 RxFifoContents DQ3210	A window into the contents of the RxFifo, as controlled by CSR RxFifoVisibility This register reads 4b at a time from lane0. A window into the contents of the RxFifo, as controlled by CSR RxFifoVisibility This register reads 4b at a time from lane0..3 from the four fifo entries addressed by rdfifo_nibble_address[2:0]=RxFifoRdPtr[2:0] Register [15:12] = lane3_ui3,lane3_ui2,lane3_ui1,lane3_ui0 where ui0 is the first occurring bit [11: 8] = lane2_ui3,lane2_ui2,lane2_ui1,lane2_ui0 [7: 4] = lane1_ui3,lane1_ui2,lane1_ui1,lane1_ui0 [3: 0] = lane0_ui3,lane0_ui2,lane0_ui1,lane0_ui0 Note that the DBYTE lane of a given index is not the same as a memory DQ of the same index unless the csr Dq<7..0>LnSel[2:0] have their default/reset value.

9.3.3.3.20 RX FIFO contents, lane[7:4] (RxFifoContentsDQ7654)

9.3.3.3.20.1 Offset

Register	Offset
RxFifoContentsDQ7654	B6h

9.3.3.3.20.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RxFifoContentsDQ7654															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

9.3.3.3.20.3 Fields

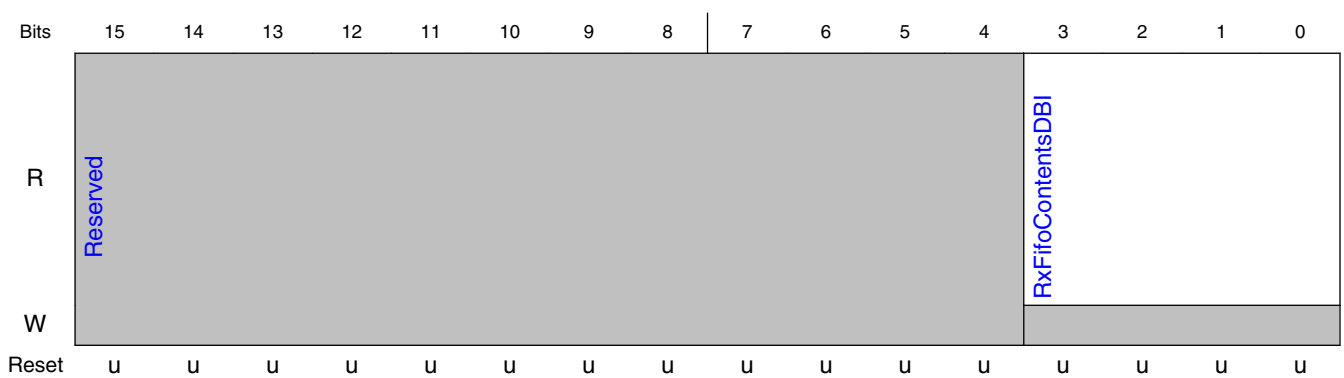
Field	Function
15-0 RxFifoContents DQ7654	A window into the contents of the RxFifo, as controlled by CSR RxFifoVisibility This register reads 4b at a time from lane4. A window into the contents of the RxFifo, as controlled by CSR RxFifoVisibility This register reads 4b at a time from lane4..7 from the four fifo entries addressed by rdfifo_nibble_address[2:0]=RxFifoRdPtr[2:0] Register [15:12] = lane7_ui3,lane7_ui2,lane7_ui1,lane7_ui0 [11: 8] = lane6_ui3,lane6_ui2,lane6_ui1,lane6_ui0 [7: 4] = lane5_ui3,lane5_ui2,lane5_ui1,lane5_ui0 [3: 0] = lane4_ui3,lane4_ui2,lane4_ui1,lane4_ui0 Note that the DBYTE lane of a given index is not the same as a memory DQ of the same index unless the csr Dq<7..0>LnSel[2:0] have their default/reset value.

9.3.3.3.21 RX FIFO contents, dbi (RxFifoContentsDBI)

9.3.3.3.21.1 Offset

Register	Offset
RxFifoContentsDBI	B8h

9.3.3.3.21.2 Diagram



9.3.3.3.21.3 Fields

Field	Function
15-4	Reserved

Table continues on the next page...

Field	Function
—	
3-0 RxFifoContents DBI	A window into the contents of the RxFifo, as controlled by CSR RxFifoVisibility This register reads 4b at a time from DBI from the four fifo entries addressed by rdfifo_nibble_address[2:0]=RxFifoRdPtr[2:0] Register [3: 0] = dbi_ui3,dbi_ui2,dbi_ui1,dbi_ui0 Note that the DBYTE DBI lane is the same as the memory DBI; it is not subject to mapping using csr Dq<7. A window into the contents of the RxFifo, as controlled by CSR RxFifoVisibility This register reads 4b at a time from DBI from the four fifo entries addressed by rdfifo_nibble_address[2:0]=RxFifoRdPtr[2:0] Register [3: 0] = dbi_ui3,dbi_ui2,dbi_ui1,dbi_ui0 Note that the DBYTE DBI lane is the same as the memory DBI; it is not subject to mapping using csr Dq<7..0>LnSel[2:0] as the DQ are.

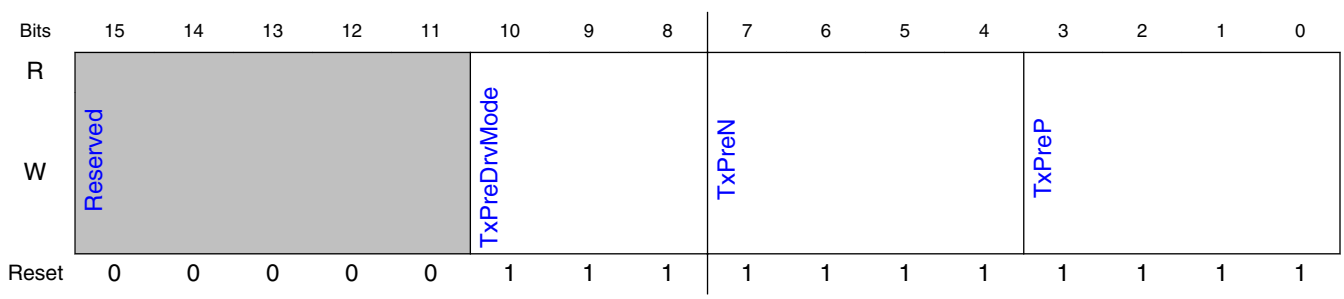
9.3.3.3.22 TX slew rate controls (TxSlewRate_b0_p0 - TxSlewRate_b1_p3)

9.3.3.3.22.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
TxSlewRate_bn_px	BEh + (x × 20_0000h) + (n × 200h)

9.3.3.3.22.2 Diagram



9.3.3.3.22.3 Fields

Field	Function
15-11 —	Reserved
10-8	Controls predrivers to adjust timing of turn-on and turn-off of pull-up and pull-down segments.

Table continues on the next page...

Field	Function
TxPreDrvMode	
7-4 TxPreN	4 bit binary trim for the driver pull down slew rate. 4 bit binary trim for the driver pull down slew rate. 4'b0000 has a slower slew rate than 4'b1111
3-0 TxPreP	4 bit binary trim for the driver pull up slew rate. 4 bit binary trim for the driver pull up slew rate. 4'b0000 has a slower slew rate than 4'b1111

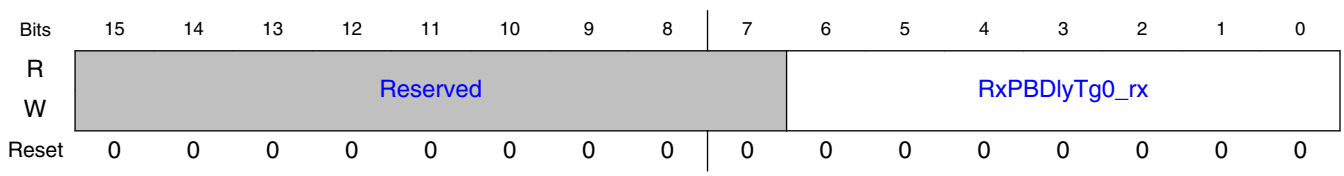
9.3.3.3.23 Read DQ per-bit BDL delay (Timing Group 0). (RxPBDlyTg0_r0 - RxPBDlyTg0_r8)

9.3.3.3.23.1 Offset

For x = 0 to 8:

Register	Offset
RxPBDlyTg0_rx	D0h + (x × 200h)

9.3.3.3.23.2 Diagram



9.3.3.3.23.3 Fields

Field	Function
15-7 —	Reserved
6-0 RxPBDlyTg0_rx	Read DQ per-bit BDL delay (Timing Group 0). Read DQ per-bit BDL delay (Timing Group 0). Trained to deskew the read DQ data that will be sampled by the incoming per-nibble read DQS, to improve the composite eye. The unit of change is a LSB in the control of the delay circuit, and is not denoninated in units of UI. Register Block Offset Address 0x068 controls the lane0 read deskew timing. (For Timing Group 0) Register Block Offset Address 0x168 controls the lane1 read deskew timing. (For Timing Group 0)

Field	Function
	Register Block Offset Address 0x268 controls the lane2 read deskew timing. (For Timing Group 0) Register Block Offset Address 0x368 controls the lane3 read deskew timing. (For Timing Group 0) Register Block Offset Address 0x468 controls the lane4 read deskew timing. (For Timing Group 0) Register Block Offset Address 0x568 controls the lane5 read deskew timing. (For Timing Group 0) Register Block Offset Address 0x668 controls the lane6 read deskew timing. (For Timing Group 0) Register Block Offset Address 0x768 controls the lane7 read deskew timing. (For Timing Group 0) Register Block Offset Address 0x868 controls the laneDBI read deskew timing. (For Timing Group 0) Note that this register is not per-pstate. The same deskew (for channel-routing differences) is used for all pstates. . Note: Timing Group 0 = RANK0 for all systems except 3DS/LRDIMM.

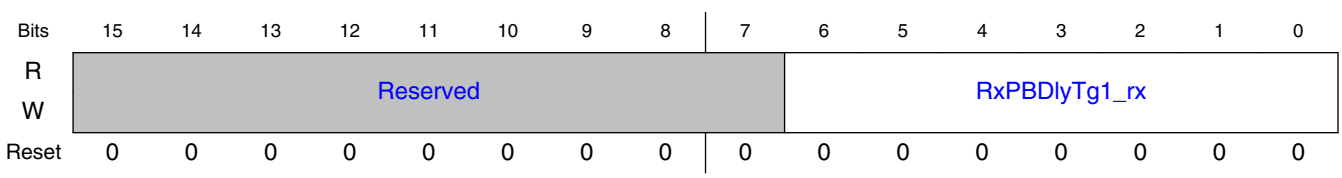
9.3.3.3.24 Read DQ per-bit BDL delay (Timing Group 1). (RxPBDlyTg1_r0 - RxPBDlyTg1_r8)

9.3.3.3.24.1 Offset

For x = 0 to 8:

Register	Offset
RxPBDlyTg1_rx	D2h + (x × 200h)

9.3.3.3.24.2 Diagram



9.3.3.3.24.3 Fields

Field	Function
15-7 —	Reserved
6-0 RxPBDlyTg1_rx	Read DQ per-bit BDL delay (Timing Group 1). Read DQ per-bit BDL delay (Timing Group 1).

Field	Function
	Trained to deskew the read DQ data that will be sampled by the incoming per-nibble read DQS, to improve the composite eye. The unit of change is a LSB in the control of the delay circuit, and is not denoninated in units of UI. Register Block Offset Address 0x069 controls the lane0 read deskew timing. (For Timing Group 1) Register Block Offset Address 0x169 controls the lane1 read deskew timing. (For Timing Group 1) Register Block Offset Address 0x269 controls the lane2 read deskew timing. (For Timing Group 1) Register Block Offset Address 0x369 controls the lane3 read deskew timing. (For Timing Group 1) Register Block Offset Address 0x469 controls the lane4 read deskew timing. (For Timing Group 1) Register Block Offset Address 0x569 controls the lane5 read deskew timing. (For Timing Group 1) Register Block Offset Address 0x669 controls the lane6 read deskew timing. (For Timing Group 1) Register Block Offset Address 0x769 controls the lane7 read deskew timing. (For Timing Group 1) Register Block Offset Address 0x869 controls the laneDBI read deskew timing. (For Timing Group 1) Note that this register is not per-pstate. The same deskew (for channel-routing differences) is used for all pstates. . Note: Timing Group 1 = RANK1 for all systems except 3DS/LRDIMM.

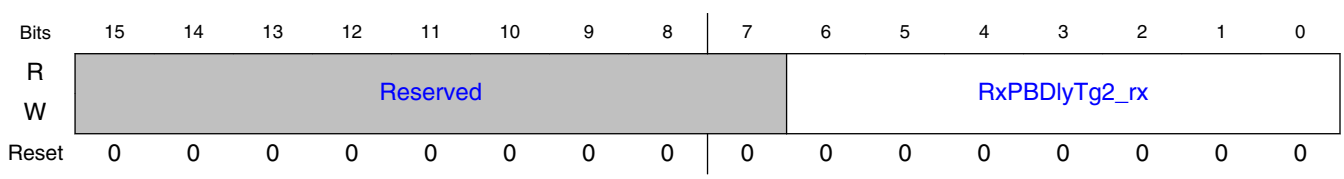
9.3.3.3.25 Read DQ per-bit BDL delay (Timing Group 2). (RxPBDlyTg2_r0 - RxPBDlyTg2_r8)

9.3.3.3.25.1 Offset

For x = 0 to 8:

Register	Offset
RxPBDlyTg2_rx	D4h + (x × 200h)

9.3.3.3.25.2 Diagram



9.3.3.3.25.3 Fields

Field	Function
15-7 —	Reserved
6-0 RxPBDlyTg2_rx	Read DQ per-bit BDL delay (Timing Group 2). Read DQ per-bit BDL delay (Timing Group 2). Trained to deskew the read DQ data that will be sampled by the incoming per-nibble read DQS, to improve the composite eye. The unit of change is a LSB in the control of the delay circuit, and is not denoninated in units of UI. Register Block Offset Address 0x06a controls the lane0 read deskew timing. (For Timing Group 2) Register Block Offset Address 0x16a controls the lane1 read deskew timing. (For Timing Group 2) Register Block Offset Address 0x26a controls the lane2 read deskew timing. (For Timing Group 2) Register Block Offset Address 0x36a controls the lane3 read deskew timing. (For Timing Group 2) Register Block Offset Address 0x46a controls the lane4 read deskew timing. (For Timing Group 2) Register Block Offset Address 0x56a controls the lane5 read deskew timing. (For Timing Group 2) Register Block Offset Address 0x66a controls the lane6 read deskew timing. (For Timing Group 2) Register Block Offset Address 0x76a controls the lane7 read deskew timing. (For Timing Group 2) Register Block Offset Address 0x86a controls the laneDBI read deskew timing. (For Timing Group 2) Note that this register is not per-pstate. The same deskew (for channel-routing differences) is used for all pstates. . Note: Timing Group 2 = RANK2 for all systems except 3DS/LRDIMM.

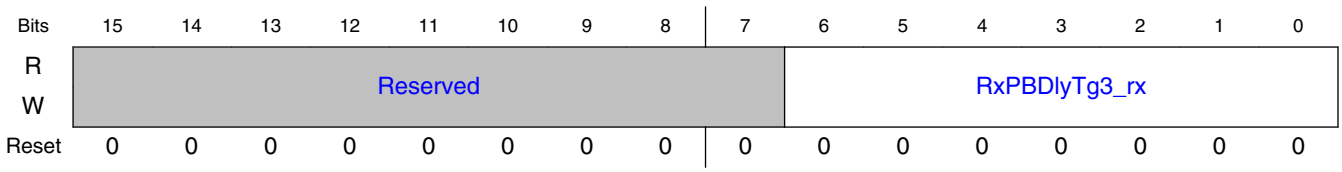
9.3.3.3.26 Read DQ per-bit BDL delay (Timing Group 3). (RxPBDlyTg3_r0 - RxPBDlyTg3_r8)

9.3.3.3.26.1 Offset

For x = 0 to 8:

Register	Offset
RxPBDlyTg3_rx	D6h + (x × 200h)

9.3.3.3.26.2 Diagram



9.3.3.3.26.3 Fields

Field	Function
15-7 —	Reserved
6-0 RxPBDlyTg3_rx	Read DQ per-bit BDL delay (Timing Group 3). Read DQ per-bit BDL delay (Timing Group 3). Trained to deskew the read DQ data that will be sampled by the incoming per-nibble read DQS, to improve the composite eye. The unit of change is a LSB in the control of the delay circuit, and is not denoninated in units of UI. Register Block Offset Address 0x06b controls the lane0 read deskew timing. (For Timing Group 3) Register Block Offset Address 0x16b controls the lane1 read deskew timing. (For Timing Group 3) Register Block Offset Address 0x26b controls the lane2 read deskew timing. (For Timing Group 3) Register Block Offset Address 0x36b controls the lane3 read deskew timing. (For Timing Group 3) Register Block Offset Address 0x46b controls the lane4 read deskew timing. (For Timing Group 3) Register Block Offset Address 0x56b controls the lane5 read deskew timing. (For Timing Group 3) Register Block Offset Address 0x66b controls the lane6 read deskew timing. (For Timing Group 3) Register Block Offset Address 0x76b controls the lane7 read deskew timing. (For Timing Group 3) Register Block Offset Address 0x86b controls the laneDBI read deskew timing. (For Timing Group 3) Note that this register is not per-pstate. The same deskew (for channel-routing differences) is used for all pstates. . Note: Timing Group 3 = RANK3 for all systems except 3DS/LRDIMM.

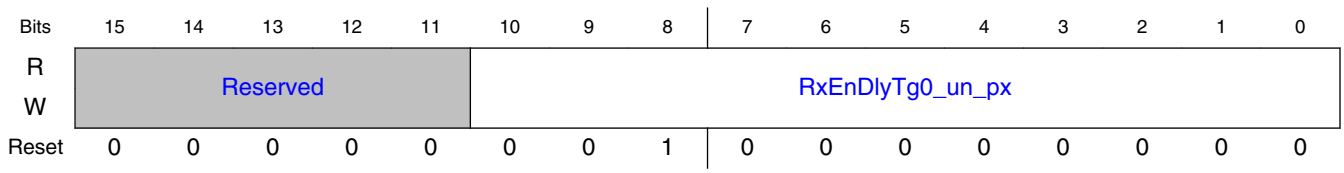
9.3.3.3.27 Trained Receive Enable Delay (For Timing Group 0) (RxEnDlyTg0_u0_p0 - RxEnDlyTg0_u1_p3)

9.3.3.3.27.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
RxEnDlyTg0_un_px	100h + (x × 20_0000h) + (n × 200h)

9.3.3.3.27.2 Diagram



9.3.3.3.27.3 Fields

Field	Function
15-11 —	Reserved
10-0 RxEnDlyTg0_un_px	Trained Receive Enable Delay (For Timing Group 0) Trained to set the delay from the memory-read command to the signal enabling the read DQS to generate read-data strobes. Trained Receive Enable Delay (For Timing Group 0) Trained to set the delay from the memory-read command to the signal enabling the read DQS to generate read-data strobes. The trained read DQS enable asserts during the read DQS preamble and deasserts at the end of the read burst. RxEnDlyTg0[10:6] is the coarse delay, ie one unit of delay is 1 UI. RxEnDlyTg0[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. RxEnDlyTg0[5] is reserved. Register Block Offset Address 0x080 controls the lower DQ nibble DqsEn. (For Timing Group 0) Register Block Offset Address 0x180 controls the upper DQ nibble DqsEn. (For Timing Group 0) These Registers are replicated per pstate. . Note: Timing Group 0 = RANK0 for all systems except 3DS/LRDIMM.

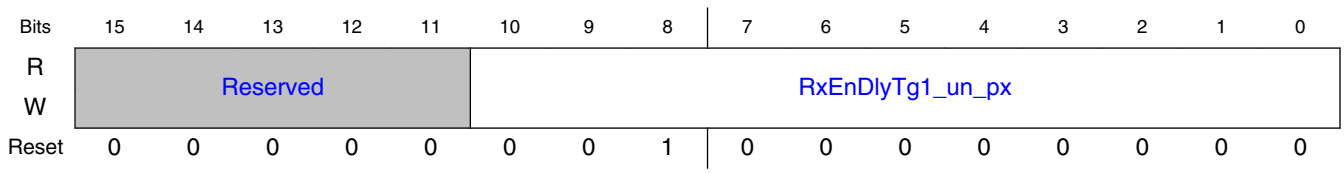
9.3.3.3.28 Trained Receive Enable Delay (For Timing Group 1) (RxEnDlyTg1_u0_p0 - RxEnDlyTg1_u1_p3)

9.3.3.3.28.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
RxEnDlyTg1_un_px	102h + (x × 20_0000h) + (n × 200h)

9.3.3.3.28.2 Diagram



9.3.3.3.28.3 Fields

Field	Function
15-11 —	Reserved
10-0 RxEnDlyTg1_un_px	Trained Receive Enable Delay (For Timing Group 1) Trained to set the delay from the memory-read command to the signal enabling the read DQS to generate read-data strobes. Trained Receive Enable Delay (For Timing Group 1) Trained to set the delay from the memory-read command to the signal enabling the read DQS to generate read-data strobes. The trained read DQS enable asserts during the read DQS preamble and deasserts at the end of the read burst. RxEnDlyTg1[10:6] is the coarse delay, ie one unit of delay is 1 UI. RxEnDlyTg1[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. RxEnDlyTg1[5] is reserved. Register Block Offset Address 0x081 controls the lower DQ nibble DqsEn. (For Timing Group 1) Register Block Offset Address 0x181 controls the upper DQ nibble DqsEn. (For Timing Group 1) These Registers are replicated per pstate. . Note: Timing Group 1 = RANK1 for all systems except 3DS/LRDIMM.

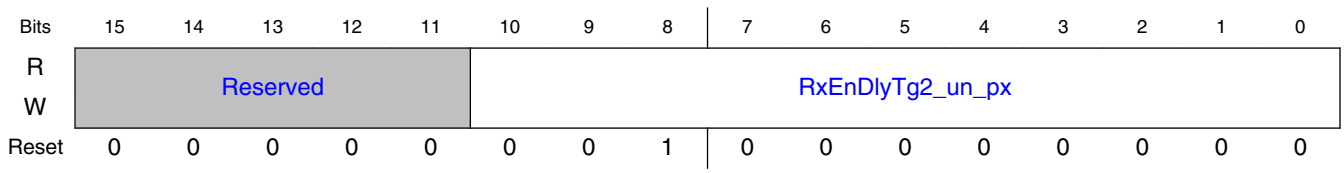
9.3.3.3.29 Trained Receive Enable Delay (For Timing Group 2) (RxEnDlyTg2_u0_p0 - RxEnDlyTg2_u1_p3)

9.3.3.3.29.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
RxEnDlyTg2_un_px	104h + (x × 20_0000h) + (n × 200h)

9.3.3.3.29.2 Diagram



9.3.3.3.29.3 Fields

Field	Function
15-11 —	Reserved
10-0 RxEnDlyTg2_un_px	Trained Receive Enable Delay (For Timing Group 2) Trained to set the delay from the memory-read command to the signal enabling the read DQS to generate read-data strobes. Trained Receive Enable Delay (For Timing Group 2) Trained to set the delay from the memory-read command to the signal enabling the read DQS to generate read-data strobes. The trained read DQS enable asserts during the read DQS preamble and deasserts at the end of the read burst. RxEnDlyTg2[10:6] is the coarse delay, ie one unit of delay is 1 UI. RxEnDlyTg2[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. RxEnDlyTg2[5] is reserved. Register Block Offset Address 0x082 controls the lower DQ nibble DqsEn. (For Timing Group 2) Register Block Offset Address 0x182 controls the upper DQ nibble DqsEn. (For Timing Group 2) These Registers are replicated per pstate. . Note: Timing Group 2 = RANK2 for all systems except 3DS/LRDIMM.

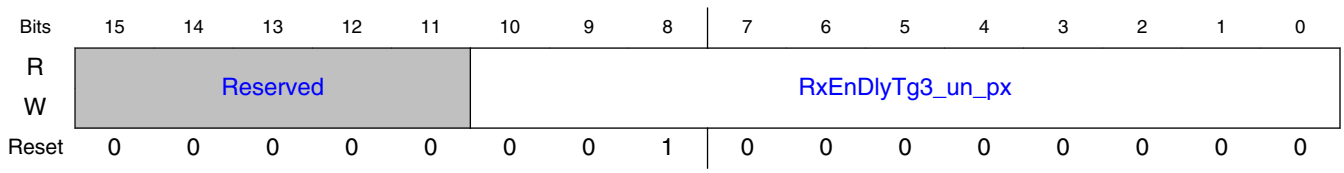
9.3.3.3.30 Trained Receive Enable Delay (For Timing Group 3) (RxEnDlyTg3_u0_p0 - RxEnDlyTg3_u1_p3)

9.3.3.3.30.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
RxEnDlyTg3_un_px	106h + (x × 20_0000h) + (n × 200h)

9.3.3.3.30.2 Diagram



9.3.3.3.30.3 Fields

Field	Function
15-11 —	Reserved
10-0 RxEnDlyTg3_un_px	Trained Receive Enable Delay (For Timing Group 3) Trained to set the delay from the memory-read command to the signal enabling the read DQS to generate read-data strobes. Trained Receive Enable Delay (For Timing Group 3) Trained to set the delay from the memory-read command to the signal enabling the read DQS to generate read-data strobes. The trained read DQS enable asserts during the read DQS preamble and deasserts at the end of the read burst. RxEnDlyTg3[10:6] is the coarse delay, ie one unit of delay is 1 UI. RxEnDlyTg3[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. RxEnDlyTg3[5] is reserved. Register Block Offset Address 0x083 controls the lower DQ nibble DqsEn. (For Timing Group 3) Register Block Offset Address 0x183 controls the upper DQ nibble DqsEn. (For Timing Group 3) These Registers are replicated per pstate. . Note: Timing Group 3 = RANK3 for all systems except 3DS/LRDIMM.

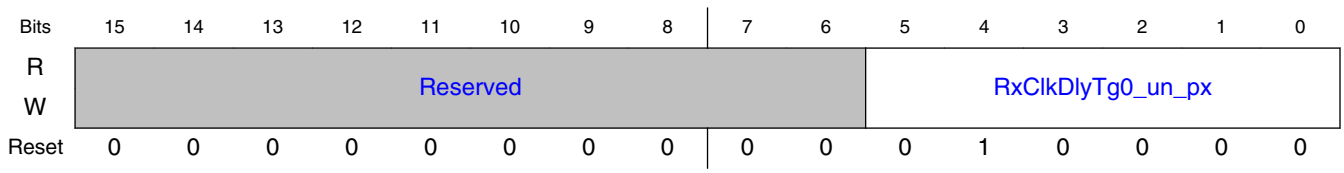
9.3.3.3.31 Trained Read DQS to RxClk Delay (Timing Group DEST=0). (RxClkDlyTg0_u0_p0 - RxClkDlyTg0_u1_p3)

9.3.3.3.31.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
RxClkDlyTg0_un_px	118h + (x × 20_0000h) + (n × 200h)

9.3.3.3.31.2 Diagram



9.3.3.3.31.3 Fields

Field	Function
15-6 —	Reserved
5-0 RxClkDlyTg0_un_px	Trained Read DQS to RxClk Delay (Timing Group DEST=0). Trained Read DQS to RxClk Delay (Timing Group DEST=0). Trained to generate read DQ receive clock from the received read DQS. RxClk Delay (6 bits, fine delay only); (For Timing Group 0) RxClkDlyTg0[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. RxClkDlyTg0[5] is reserved. There is no coarse delay field, ie the max configurable delay is 31/32 UI. Register Block Offset Address 0x08c controls the lower DQ nibble RxClk timing. (For Timing Group 0) Register Block Offset Address 0x18c controls the upper DQ nibble RxClk timing. (For Timing Group 0) These Registers are replicated per pstate. . Note: Timing Group 0 = RANK0 for all systems except 3DS/LRDIMM.

9.3.3.3.32 Trained Read DQS to RxClk Delay (Timing Group DEST=1). (RxClkDlyTg1_u0_p0 - RxClkDlyTg1_u1_p3)

9.3.3.3.32.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
RxClkDlyTg1_un_px	11Ah + (x × 20_0000h) + (n × 200h)

9.3.3.3.32.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										RxClkDlyTg1_un_px					
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

9.3.3.3.32.3 Fields

Field	Function
15-6 —	Reserved
5-0 RxClkDlyTg1_un_px	Trained Read DQS to RxClk Delay (Timing Group DEST=1). Trained Read DQS to RxClk Delay (Timing Group DEST=1). Trained to generate read DQ receive clock from the received read DQS. RxClk Delay (6 bits, fine delay only); (For Timing Group 1) RxClkDlyTg1[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. RxClkDlyTg1[5] is reserved. There is no coarse delay field, ie the max configurable delay is 31/32 UI. Register Block Offset Address 0x08d controls the lower DQ nibble RxClk timing. (For Timing Group 1) Register Block Offset Address 0x18d controls the upper DQ nibble RxClk timing. (For Timing Group 1) These Registers are replicated per pstate. . Note: Timing Group 1 = RANK1 for all systems except 3DS/LRDIMM.

9.3.3.3.33 Trained Read DQS to RxClk Delay (Timing Group DEST=2). (RxClkDlyTg2_u0_p0 - RxClkDlyTg2_u1_p3)

9.3.3.3.33.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
RxClkDlyTg2_un_px	11Ch + (x × 20_0000h) + (n × 200h)

9.3.3.3.33.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										RxClkDlyTg2_un_px					
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

9.3.3.3.33.3 Fields

Field	Function
15-6 —	Reserved
5-0 RxClkDlyTg2_un_px	Trained Read DQS to RxClk Delay (Timing Group DEST=2). Trained Read DQS to RxClk Delay (Timing Group DEST=2). Trained to generate read DQ receive clock from the received read DQS. RxClk Delay (6 bits, fine delay only); (For Timing Group 2) RxClkDlyTg2[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. RxClkDlyTg2[5] is reserved. There is no coarse delay field, ie the max configurable delay is 31/32 UI. Register Block Offset Address 0x08e controls the lower DQ nibble RxClk timing. (For Timing Group 2) Register Block Offset Address 0x18e controls the upper DQ nibble RxClk timing. (For Timing Group 2) These Registers are replicated per pstate. . Note: Timing Group 2 = RANK2 for all systems except 3DS/LRDIMM.

9.3.3.3.34 Trained Read DQS to RxClk Delay (Timing Group DEST=3). (RxClkDlyTg3_u0_p0 - RxClkDlyTg3_u1_p3)

9.3.3.3.34.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
RxClkDlyTg3_un_px	11Eh + (x × 20_0000h) + (n × 200h)

9.3.3.3.34.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										RxClkDlyTg3_un_px					
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

9.3.3.3.34.3 Fields

Field	Function
15-6 —	Reserved
5-0 RxClkDlyTg3_un_px	Trained Read DQS to RxClk Delay (Timing Group DEST=3). Trained Read DQS to RxClk Delay (Timing Group DEST=3). Trained to generate read DQ receive clock from the received read DQS. RxClk Delay (6 bits, fine delay only); (For Timing Group 3) RxClkDlyTg3[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. RxClkDlyTg3[5] is reserved. There is no coarse delay field, ie the max configurable delay is 31/32 UI. Register Block Offset Address 0x08f controls the lower DQ nibble RxClk timing. (For Timing Group 3) Register Block Offset Address 0x18f controls the upper DQ nibble RxClk timing. (For Timing Group 3) These Registers are replicated per pstate. . Note: Timing Group 3 = RANK3 for all systems except 3DS/LRDIMM.

9.3.3.3.35 Trained Read DQS_c to RxClkc Delay (Timing Group DEST=0). (RxClkcDlyTg0_u0_p0 - RxClkcDlyTg0_u1_p3)

9.3.3.3.35.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
RxClkcDlyTg0_un_px	120h + (x × 20_0000h) + (n × 200h)

9.3.3.3.35.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										RxClkcDlyTg0_un_px					
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

9.3.3.3.35.3 Fields

Field	Function
15-6 —	Reserved
5-0 RxClkcDlyTg0_un_px	Trained Read DQS_c to RxClkc Delay (Timing Group DEST=0). Trained Read DQS_c to RxClkc Delay (Timing Group DEST=0). Trained to generate read DQ receive clock from the received read DQS_c. RxClkc Delay (6 bits, fine delay only); (For Timing Group 0) RxClkcDlyTg0[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. RxClkcDlyTg0[5] is reserved. There is no coarse delay field, ie the max configurable delay is 31/32 UI. Register Block Offset Address 0x090 controls the lower DQ nibble RxClkc timing. (For Timing Group 0) Register Block Offset Address 0x190 controls the upper DQ nibble RxClkc timing. (For Timing Group 0) These Registers are replicated per pstate. Note: Timing Group 0 = RANK0.

9.3.3.3.36 Trained Read DQS_c to RxClkc Delay (Timing Group DEST=0). (RxClkcDlyTg1_u0_p0 - RxClkcDlyTg1_u1_p3)

9.3.3.3.36.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
RxClkcDlyTg1_un_px	122h + (x × 20_0000h) + (n × 200h)

9.3.3.3.36.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										RxClkcDlyTg1_un_px					
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

9.3.3.3.36.3 Fields

Field	Function
15-6 —	Reserved
5-0 RxClkcDlyTg1_un_px	Trained Read DQS_c to RxClkc Delay (Timing Group DEST=1). Trained Read DQS_c to RxClkc Delay (Timing Group DEST=1). Trained to generate read DQ receive clock from the received read DQS_c. RxClkc Delay (6 bits, fine delay only); (For Timing Group 1) RxClkcDlyTg1[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. RxClkcDlyTg1[5] is reserved. There is no coarse delay field, ie the max configurable delay is 31/32 UI. Register Block Offset Address 0x091 controls the lower DQ nibble RxClkc timing. (For Timing Group 1) Register Block Offset Address 0x191 controls the upper DQ nibble RxClkc timing. (For Timing Group 1) These Registers are replicated per pstate. Note: Timing Group 1 = RANK1.

9.3.3.3.37 Trained Read DQS_c to RxClkc Delay (Timing Group DEST=2). (RxClkcDlyTg2_u0_p0 - RxClkcDlyTg2_u1_p3)

9.3.3.3.37.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
RxClkcDlyTg2_un_px	124h + (x × 20_0000h) + (n × 200h)

9.3.3.3.37.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										RxClkcDlyTg2_un_px					
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

9.3.3.3.37.3 Fields

Field	Function
15-6 —	Reserved
5-0 RxClkcDlyTg2_un_px	Trained Read DQS_c to RxClkc Delay (Timing Group DEST=2). Trained Read DQS_c to RxClkc Delay (Timing Group DEST=2). Trained to generate read DQ receive clock from the received read DQS_c. RxClkc Delay (6 bits, fine delay only); (For Timing Group 2) RxClkcDlyTg2[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. RxClkcDlyTg2[5] is reserved. There is no coarse delay field, ie the max configurable delay is 31/32 UI. Register Block Offset Address 0x092 controls the lower DQ nibble RxClkc timing. (For Timing Group 2) Register Block Offset Address 0x192 controls the upper DQ nibble RxClkc timing. (For Timing Group 2) These Registers are replicated per pstate. Note: Timing Group 2 = RANK2.

9.3.3.3.38 Trained Read DQS_c to RxClkc Delay (Timing Group DEST=3). (RxClkcDlyTg3_u0_p0 - RxClkcDlyTg3_u1_p3)

9.3.3.3.38.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
RxClkcDlyTg3_un_px	128h + (x × 20_0000h) + (n × 200h)

9.3.3.3.38.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										RxClkcDlyTg3_un_px					
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

9.3.3.3.38.3 Fields

Field	Function
15-6 —	Reserved
5-0 RxClkcDlyTg3_un_px	Trained Read DQS_c to RxClkc Delay (Timing Group DEST=3). Trained Read DQS_c to RxClkc Delay (Timing Group DEST=3). Trained to generate read DQ receive clock from the received read DQS_c. RxClkc Delay (6 bits, fine delay only); (For Timing Group 3) RxClkcDlyTg3[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. RxClkcDlyTg3[5] is reserved. There is no coarse delay field, ie the max configurable delay is 31/32 UI. Register Block Offset Address 0x094 controls the lower DQ nibble RxClkc timing. (For Timing Group 3) Register Block Offset Address 0x194 controls the upper DQ nibble RxClkc timing. (For Timing Group 3) These Registers are replicated per pstate. Note: Timing Group 3 = RANK3.

9.3.3.3.39 Maps Phy DQ lane to memory DQ0 (Dq0LnSel - Dq7LnSel)

9.3.3.3.39.1 Offset

For n = 0 to 7:

Register	Offset
DqnLnSel	140h + (n × 2h)

9.3.3.3.39.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved													DqLnSel		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.3.39.3 Fields

Field	Function
15-3 —	Reserved
2-0 DqLnSel	Supports mapping of PHY dq to dram dq within a byte (swizzle). Supports mapping of PHY dq to dram dq within a byte (swizzle). Intended to undo the swizzle of board-level phy-to-dram connections. such that MRR operations of binary counters may be return correct values. Note that this register is per-dbyte such that the sizzle may be different per dbyte. Each register in a byte's set of DqLnSel must have a unique value within the set. For example, if, on this dbyte, PHY lane 3 is connected to memory dq0 (on this dbyte), then Register Dq0LnSel for this dbyte should be 3.

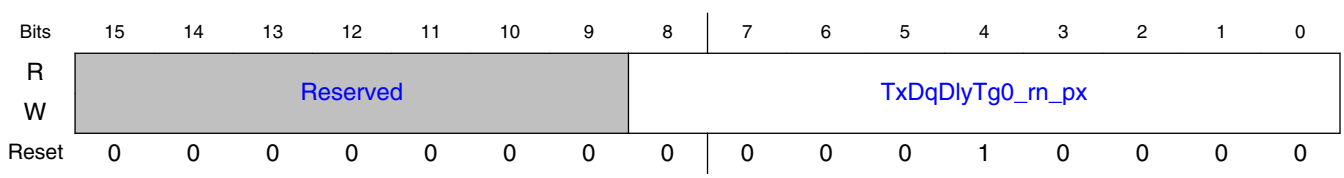
9.3.3.3.40 Write DQ Delay (Timing Group 0). (TxDqDlyTg0_r0_p0 - TxDqDlyTg0_r8_p3)

9.3.3.3.40.1 Offset

For n = 0 to 8; x = 0 to 3:

Register	Offset
TxDqDlyTg0_rn_px	180h + (x × 20_0000h) + (n × 200h)

9.3.3.3.40.2 Diagram



9.3.3.3.40.3 Fields

Field	Function
15-9 —	Reserved

Table continues on the next page...

Field	Function
8-0 TxDqDlyTg0_rn_px	Write DQ Delay (Timing Group 0). Write DQ Delay (Timing Group 0). Trained to center the delay between the write DQ to the write DQS. Tx DQ Delay (9 bits, includes 3b coarse and 6b fine delays); (For Timing Group 0) TxDqDlyTg0[8:6] is the coarse delay, ie one unit of delay is 1 UI. TxDqDlyTg0[4:0] is the fine (fractional UI) delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. TxDqDlyTg0[5] is reserved. Register Block Offset Address 0x0C0 controls the lane0 write DQ timing. (For Timing Group 0) Register Block Offset Address 0x1C0 controls the lane1 write DQ timing. (For Timing Group 0) Register Block Offset Address 0x2C0 controls the lane2 write DQ timing. (For Timing Group 0) Register Block Offset Address 0x3C0 controls the lane3 write DQ timing. (For Timing Group 0) Register Block Offset Address 0x4C0 controls the lane4 write DQ timing. (For Timing Group 0) Register Block Offset Address 0x5C0 controls the lane5 write DQ timing. (For Timing Group 0) Register Block Offset Address 0x6C0 controls the lane6 write DQ timing. (For Timing Group 0) Register Block Offset Address 0x7C0 controls the lane7 write DQ timing. (For Timing Group 0) Register Block Offset Address 0x8C0 controls the laneDMDBI write DQ timing. (For Timing Group 0) These Registers are replicated per pstate. . Note: Timing Group 0 = RANK0 for all systems except 3DS/LRDIMM.

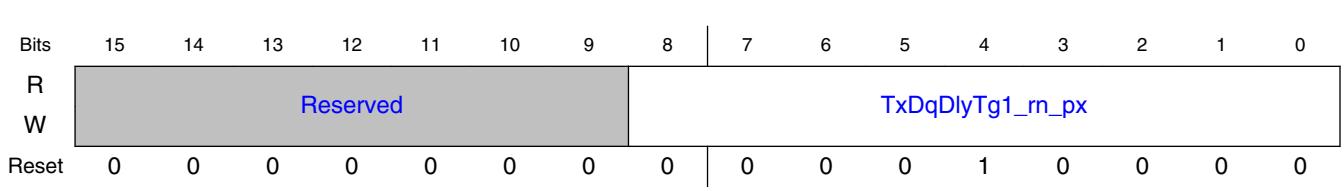
9.3.3.3.41 Write DQ Delay (Timing Group 1). (TxDqDlyTg1_r0_p0 - TxDq DlyTg1_r8_p3)

9.3.3.3.41.1 Offset

For n = 0 to 8; x = 0 to 3:

Register	Offset
TxDqDlyTg1_rn_px	182h + (x × 20_0000h) + (n × 200h)

9.3.3.3.41.2 Diagram



9.3.3.3.41.3 Fields

Field	Function
15-9 —	Reserved
8-0 TxDqDlyTg1_rn_px	Write DQ Delay (Timing Group 1). Write DQ Delay (Timing Group 1). Trained to center the delay between the write DQ to the write DQS. Tx DQ Delay (9 bits, includes 3b coarse and 6b fine delays); (For Timing Group 1) TxDqDlyTg1[8:6] is the coarse delay, ie one unit of delay is 1 UI. TxDqDlyTg1[4:0] is the fine (fractional UI) delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. TxDqDlyTg1[5] is reserved. Register Block Offset Address 0x0C1 controls the lane0 write DQ timing. (For Timing Group 1) Register Block Offset Address 0x1C1 controls the lane1 write DQ timing. (For Timing Group 1) Register Block Offset Address 0x2C1 controls the lane2 write DQ timing. (For Timing Group 1) Register Block Offset Address 0x3C1 controls the lane3 write DQ timing. (For Timing Group 1) Register Block Offset Address 0x4C1 controls the lane4 write DQ timing. (For Timing Group 1) Register Block Offset Address 0x5C1 controls the lane5 write DQ timing. (For Timing Group 1) Register Block Offset Address 0x6C1 controls the lane6 write DQ timing. (For Timing Group 1) Register Block Offset Address 0x7C1 controls the lane7 write DQ timing. (For Timing Group 1) Register Block Offset Address 0x8C1 controls the laneDMDBI write DQ timing. (For Timing Group 1) These Registers are replicated per pstate. . Note: Timing Group 1 = RANK1 for all systems except 3DS/LRDIMM.

9.3.3.3.42 Write DQ Delay (Timing Group 2). (TxDqDlyTg2_r0_p0 - TxDq DlyTg2_r8_p3)

9.3.3.3.42.1 Offset

For n = 0 to 8; x = 0 to 3:

Register	Offset
TxDqDlyTg2_rn_px	184h + (x × 20_0000h) + (n × 200h)

9.3.3.3.42.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								TxDqDlyTg2_rn_px							
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

9.3.3.3.42.3 Fields

Field	Function
15-9 —	Reserved
8-0 TxDqDlyTg2_rn_px	Write DQ Delay (Timing Group 2). Write DQ Delay (Timing Group 2). Trained to center the delay between the write DQ to the write DQS. Tx DQ Delay (9 bits, includes 3b coarse and 6b fine delays); (For Timing Group 2) TxDqDlyTg2[8:6] is the coarse delay, ie one unit of delay is 1 UI. TxDqDlyTg2[4:0] is the fine (fractional UI) delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. TxDqDlyTg2[5] is reserved. Register Block Offset Address 0x0C2 controls the lane0 write DQ timing. (For Timing Group 2) Register Block Offset Address 0x1C2 controls the lane1 write DQ timing. (For Timing Group 2) Register Block Offset Address 0x2C2 controls the lane2 write DQ timing. (For Timing Group 2) Register Block Offset Address 0x3C2 controls the lane3 write DQ timing. (For Timing Group 2) Register Block Offset Address 0x4C2 controls the lane4 write DQ timing. (For Timing Group 2) Register Block Offset Address 0x5C2 controls the lane5 write DQ timing. (For Timing Group 2) Register Block Offset Address 0x6C2 controls the lane6 write DQ timing. (For Timing Group 2) Register Block Offset Address 0x7C2 controls the lane7 write DQ timing. (For Timing Group 2) Register Block Offset Address 0x8C2 controls the laneDMDBI write DQ timing. (For Timing Group 2) These Registers are replicated per pstate. . Note: Timing Group 2 = RANK2 for all systems except 3DS/LRDIMM.

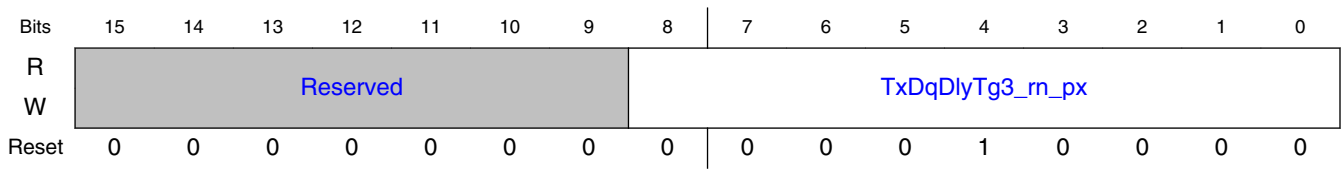
9.3.3.3.43 Write DQ Delay (Timing Group 3). (TxDqDlyTg3_r0_p0 - TxDq DlyTg3_r8_p3)

9.3.3.3.43.1 Offset

For n = 0 to 8; x = 0 to 3:

Register	Offset
TxDqDlyTg3_rn_px	186h + (x × 20_0000h) + (n × 200h)

9.3.3.3.43.2 Diagram



9.3.3.3.43.3 Fields

Field	Function
15-9 —	Reserved
8-0 TxDqDlyTg3_rn_px	Write DQ Delay (Timing Group 3). Write DQ Delay (Timing Group 3). Trained to center the delay between the write DQ to the write DQS. Tx DQ Delay (9 bits, includes 3b coarse and 6b fine delays); (For Timing Group 3) TxDqDlyTg3[8:6] is the coarse delay, ie one unit of delay is 1 UI. TxDqDlyTg3[4:0] is the fine (fractional UI) delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. TxDqDlyTg3[5] is reserved. Register Block Offset Address 0x0C3 controls the lane0 write DQ timing. (For Timing Group 3) Register Block Offset Address 0x1C3 controls the lane1 write DQ timing. (For Timing Group 3) Register Block Offset Address 0x2C3 controls the lane2 write DQ timing. (For Timing Group 3) Register Block Offset Address 0x3C3 controls the lane3 write DQ timing. (For Timing Group 3) Register Block Offset Address 0x4C3 controls the lane4 write DQ timing. (For Timing Group 3) Register Block Offset Address 0x5C3 controls the lane5 write DQ timing. (For Timing Group 3) Register Block Offset Address 0x6C3 controls the lane6 write DQ timing. (For Timing Group 3) Register Block Offset Address 0x7C3 controls the lane7 write DQ timing. (For Timing Group 3) Register Block Offset Address 0x8C3 controls the laneDMDBI write DQ timing. (For Timing Group 3) These Registers are replicated per pstate. . Note: Timing Group 3 = RANK3 for all systems except 3DS/LRDIMM.

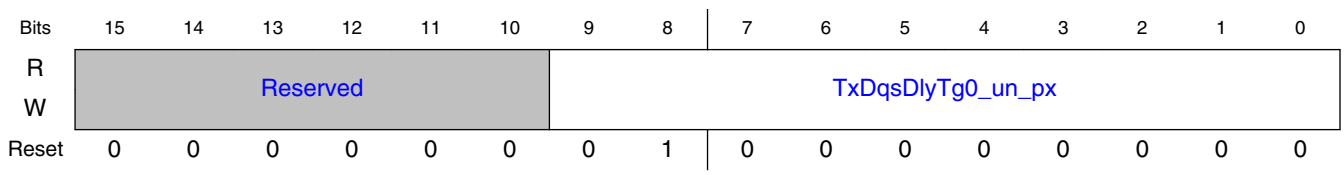
9.3.3.3.44 Write DQS Delay (Timing Group DEST=0). (TxDqsDlyTg0_u0_p0 - TxDqsDlyTg0_u1_p3)

9.3.3.3.44.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
TxDqsDlyTg0_un_px	1A0h + (x × 20_0000h) + (n × 200h)

9.3.3.3.44.2 Diagram



9.3.3.3.44.3 Fields

Field	Function
15-10 —	Reserved
9-0 TxDqsDlyTg0_un_px	Write DQS Delay (Timing Group DEST=0). Write DQS Delay (Timing Group DEST=0). Trained to set the delay from the memory-write command to the signal driving the write DQS Tx DQS Delay (10 bits, includes coarse and fine delays); (For Timing Group 0) TxDqsDlyTg0[9:6] is the coarse delay, ie one unit of delay is 1 UI. TxDqsDlyTg0[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. TxDqsDlyTg0[5] is reserved. Register is per nibble, per timing-group, per pstate. Register Block Offset Address 0x0D0 controls the lower DQ nibble write DQS timing. (For Timing Group 0) Register Block Offset Address 0x1D0 controls the upper DQ nibble write DQS timing. (For Timing Group 0) . Note: Timing Group 0 = RANK0 for all systems except 3DS/LRDIMM.

9.3.3.3.45 Write DQS Delay (Timing Group DEST=1). (TxDqsDlyTg1_u0_p0 - TxDqsDlyTg1_u1_p3)

9.3.3.3.45.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
TxDqsDlyTg1_un_px	1A2h + (x × 20_0000h) + (n × 200h)

9.3.3.3.45.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						TxDqsDlyTg1_un_px									
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

9.3.3.3.45.3 Fields

Field	Function
15-10 —	Reserved
9-0 TxDqsDlyTg1_un_px	Write DQS Delay (Timing Group DEST=1). Write DQS Delay (Timing Group DEST=1). Trained to set the delay from the memory-write command to the signal driving the write DQS Tx DQS Delay (10 bits, includes coarse and fine delays); (For Timing Group 1) TxDqsDlyTg1[9:6] is the coarse delay, ie one unit of delay is 1 UI. TxDqsDlyTg1[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. TxDqsDlyTg1[5] is reserved. Register is per nibble, per timing-group, per pstate. Register Block Offset Address 0x0D1 controls the lower DQ nibble write DQS timing. (For Timing Group 1) Register Block Offset Address 0x1D1 controls the upper DQ nibble write DQS timing. (For Timing Group 1) . Note: Timing Group 1 = RANK1 for all systems except 3DS/LRDIMM.

9.3.3.3.46 Write DQS Delay (Timing Group DEST=2). (TxDqsDlyTg2_u0_p0 - TxDqsDlyTg2_u1_p3)

9.3.3.3.46.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
TxDqsDlyTg2_un_px	1A4h + (x × 20_0000h) + (n × 200h)

9.3.3.3.46.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						TxDqsDlyTg2_un_px									
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

9.3.3.3.46.3 Fields

Field	Function
15-10 —	Reserved
9-0 TxDqsDlyTg2_un_px	Write DQS Delay (Timing Group DEST=2). Write DQS Delay (Timing Group DEST=2). Trained to set the delay from the memory-write command to the signal driving the write DQS Tx DQS Delay (10 bits, includes coarse and fine delays); (For Timing Group 2) TxDqsDlyTg2[9:6] is the coarse delay, ie one unit of delay is 1 UI. TxDqsDlyTg2[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. TxDqsDlyTg2[5] is reserved. Register is per nibble, per timing-group, per pstate. Register Block Offset Address 0x0D2 controls the lower DQ nibble write DQS timing. (For Timing Group 2) Register Block Offset Address 0x1D2 controls the upper DQ nibble write DQS timing. (For Timing Group 2) . Note: Timing Group 2 = RANK2 for all systems except 3DS/LRDIMM.

9.3.3.3.47 Write DQS Delay (Timing Group DEST=3). (TxDqsDlyTg3_u0_p0 - TxDqsDlyTg3_u1_p3)

9.3.3.3.47.1 Offset

For n = 0 to 1; x = 0 to 3:

Register	Offset
TxDqsDlyTg3_un_px	1A6h + (x × 20_0000h) + (n × 200h)

9.3.3.3.47.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						TxDqsDlyTg3_un_px									
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

9.3.3.3.47.3 Fields

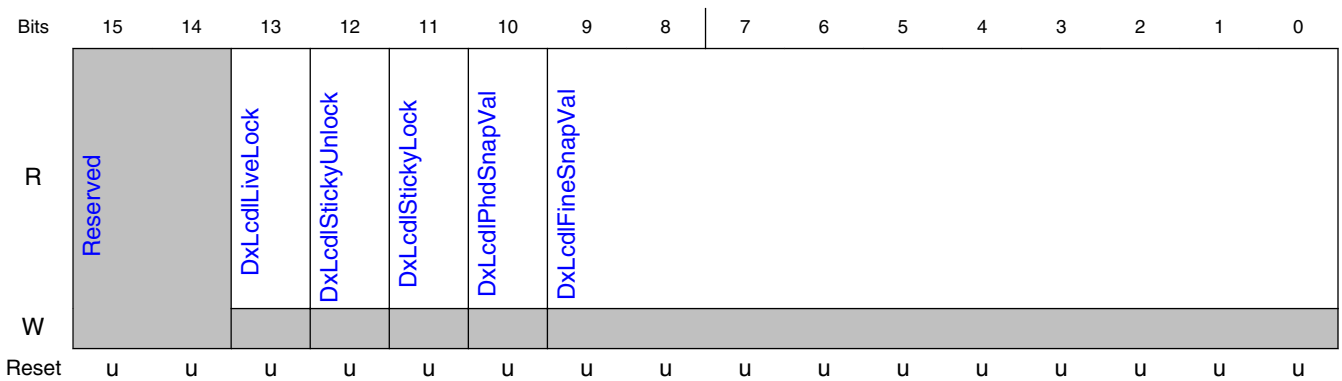
Field	Function
15-10 —	Reserved
9-0 TxDqsDlyTg3_un_px	Write DQS Delay (Timing Group DEST=3). Write DQS Delay (Timing Group DEST=3). Trained to set the delay from the memory-write command to the signal driving the write DQS Tx DQS Delay (10 bits, includes coarse and fine delays); (For Timing Group 3) TxDqsDlyTg3[9:6] is the coarse delay, ie one unit of delay is 1 UI. TxDqsDlyTg3[4:0] is the fine delay, ie one unit of delay is one-thirtysecond of a UI = UI/32. TxDqsDlyTg3[5] is reserved. Register is per nibble, per timing-group, per pstate. Register Block Offset Address 0x0D3 controls the lower DQ nibble write DQS timing. (For Timing Group 3) Register Block Offset Address 0x1D3 controls the upper DQ nibble write DQS timing. (For Timing Group 3) . Note: Timing Group 3 = RANK3 for all systems except 3DS/LRDIMM.

9.3.3.3.48 Debug status of the DBYTE LCDL (DxLcdlStatus)

9.3.3.3.48.1 Offset

Register	Offset
DxLcdlStatus	1C8h

9.3.3.3.48.2 Diagram



9.3.3.3.48.3 Fields

Field	Function
15-14 —	Reserved
13 DxLcdlLiveLock	present value of whether the LCDL is locked, valid when LcdlTstEnable=1.
12 DxLcdlStickyUnlock	latched value of whether the LCDL ever lost lock after the assertion of LcdlTstEnable.
11 DxLcdlStickyLock	latched value of whether the LCDL ever achieved lock after the assertion of LcdlTstEnable.
10 DxLcdlPhdSnapVal	Value of the LCDL phase-detector output, latched by pulse on csr LcdlFineSnap. Value of the LCDL phase-detector output, latched by pulse on csr LcdlFineSnap. while csr LcdlTstEnable=1.
9-0 DxLcdlFineSnapVal	Value of the LCDL 1UI estimate code, latched by pulse on csr LcdlFineSnap while csr LcdlTstEnable=1. Value of the LCDL 1UI estimate code, latched by pulse on csr LcdlFineSnap while csr LcdlTstEnable=1. Index 9 is reserved for growth.

9.3.3.3.49 DFI MaxReadLatency (DFIMRL_p1)

9.3.3.3.49.1 Offset

Register	Offset
DFIMRL_p1	20_0040h

9.3.3.3.49.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											DFIMRL_p1				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

9.3.3.3.49.3 Fields

Field	Function
15-5 —	Reserved
4-0 DFIMRL_p1	This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. This is the value, in units of two mem clocks, between dfi_rddata_en and dfi_rddata_valid A unit change in the LSB is a change in MRL of two mem clocks.

9.3.3.3.50 DFI MaxReadLatency (DFIMRL_p2)

9.3.3.3.50.1 Offset

Register	Offset
DFIMRL_p2	40_0040h

9.3.3.3.50.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											DFIMRL_p2				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

9.3.3.3.50.3 Fields

Field	Function
15-5 —	Reserved
4-0 DFIMRL_p2	This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. This is the value, in units of two mem clocks, between dfi_rddata_en and dfi_rddata_valid A unit change in the LSB is a change in MRL of two mem clocks.

9.3.3.3.51 DQS Oscillator Count inverse at time of training in LPDDR4 drift compensation (PptDqsCntInvTrnTg0_p2)

9.3.3.3.51.1 Offset

Register	Offset
PptDqsCntInvTrnTg0_p2	40_015Ch

9.3.3.3.51.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PptDqsCntInvTrnTg0_p2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.3.51.3 Fields

Field	Function
15-0 PptDqsCntInvTrnTg0_p2	Programmed by PHY training firmware to support LPDDR3/LPDDR4 DRAM drift compensation.

9.3.3.3.52 DQS Oscillator Count inverse at time of training in LPDDR4 drift compensation (PptDqsCntInvTrnTg1_p2)

9.3.3.3.52.1 Offset

Register	Offset
PptDqsCntInvTrnTg1_p2	40_015Eh

9.3.3.3.52.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PptDqsCntInvTrnTg1_p2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.3.52.3 Fields

Field	Function
15-0 PptDqsCntInvTrnTg1_p2	Programmed by PHY training firmware to support LPDDR3/LPDDR4 DRAM drift compensation.

9.3.3.3.53 DFI MaxReadLatency (DFIMRL_p3)

9.3.3.3.53.1 Offset

Register	Offset
DFIMRL_p3	60_0040h

9.3.3.3.53.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											DFIMRL_p3				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

9.3.3.3.53.3 Fields

Field	Function
15-5 —	Reserved
4-0 DFIMRL_p3	This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. This is the value, in units of two mem clocks, between dfi_rddata_en and dfi_rddata_valid A unit change in the LSB is a change in MRL of two mem clocks.

9.3.3.3.54 DQS Oscillator Count inverse at time of training in LPDDR4 drift compensation (PptDqsCntInvTrnTg0_p3)

9.3.3.3.54.1 Offset

Register	Offset
PptDqsCntInvTrnTg0_p3	60_015Ch

9.3.3.3.54.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PptDqsCntInvTrnTg0_p3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.3.54.3 Fields

Field	Function
15-0 PptDqsCntInvTrnTg0_p3	Programmed by PHY training firmware to support LPDDR3/LPDDR4 DRAM drift compensation.

9.3.3.3.55 DQS Oscillator Count inverse at time of training in LPDDR4 drift compensation (PptDqsCntInvTrnTg1_p3)

9.3.3.3.55.1 Offset

Register	Offset
PptDqsCntInvTrnTg1_p3	60_015Eh

9.3.3.3.55.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PptDqsCntInvTrnTg1_p3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.3.55.3 Fields

Field	Function
15-0 PptDqsCntInvTrnTg1_p3	Programmed by PHY training firmware to support LPDDR3/LPDDR4 DRAM drift compensation.

9.3.3.4 DWC_DDRPHYA_DRTUB register descriptions

9.3.3.4.1 DWC_DDRPHYA_DRTUB Memory map

DDR4/3 PHY address block

DWC_DDRPHYA_DRTUB0 base address: C_0000h

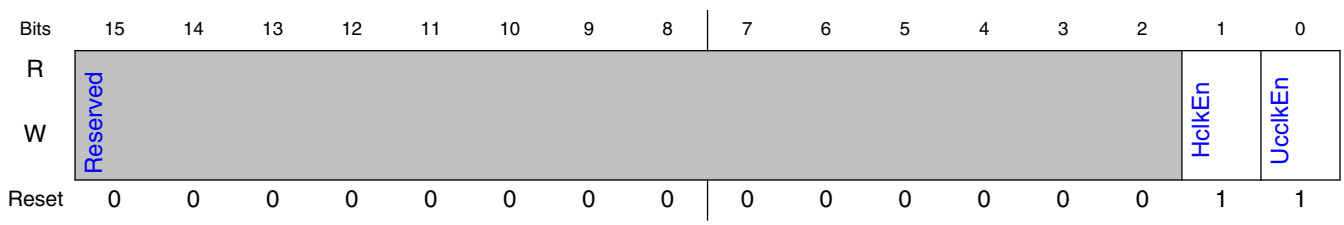
Offset	Register	Width (In bits)	Access	Reset value
100h	Ucclk and Hclk enables (UcclkHclkEnables)	16	RW	0003h
102h	PIE current Pstate value (CurPstate0b)	16	RW	0000h
1DAh	Customer settable by the customer (CUSTPUBREV)	16	RO	Table 9-4
1DCh	The hardware version of this PUB, excluding the PHY (PUBREV)	16	RO	1160h

9.3.3.4.2 Ucclk and Hclk enables (UcclkHclkEnables)

9.3.3.4.2.1 Offset

Register	Offset
UcclkHclkEnables	100h

9.3.3.4.2.2 Diagram



9.3.3.4.2.3 Fields

Field	Function
15-2 —	Reserved
1 HclkEn	When training has completed (and assuming no further need for the training hardware), the enable should be set to 0 to reduce power.
0 UcclkEn	When training has completed (and assuming no further need for the microcontroller), the enable should be set to 0 to reduce power.

9.3.3.4.3 PIE current Pstate value (CurPstate0b)

9.3.3.4.3.1 Offset

Register	Offset
CurPstate0b	102h

9.3.3.4.3.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												CurPstate0b			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.4.3.3 Fields

Field	Function
15-4 —	Reserved
3-0 CurPstate0b	PIE current Pstate value This register is used to select values for writing by the Pstate sequencer and is written in the beginning of the Pstate switch. PIE current Pstate value This register is used to select values for writing by the Pstate sequencer and is written in the beginning of the Pstate switch. It can also be used to reference which Pstate the sequencer thinks it's in.

9.3.3.4.4 Customer settable by the customer (CUSTPUBREV)

9.3.3.4.4.1 Offset

Register	Offset
CUSTPUBREV	1DAh

9.3.3.4.4.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										CUSTPUBREV					
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

9.3.3.4.4.3 Fields

Field	Function
15-6 —	Reserved
5-0 CUSTPUBREV	The customer settable PUB version number. The customer settable PUB version number. The value is derived from the `define DWC_DDRPHY_CUST_PUBREV

9.3.3.4.5 The hardware version of this PUB, excluding the PHY (PUBREV)

9.3.3.4.5.1 Offset

Register	Offset
PUBREV	1DCh

9.3.3.4.5.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUBMJR								PUBMDR				PUBMNR			
W																
Reset	0	0	0	1	0	0	0	1	0	1	1	0	0	0	0	0

9.3.3.4.5.3 Fields

Field	Function
15-8 PUBMJR	Indicates major revision of the PUB.
7-4 PUBMDR	Indicates moderate revision of the PUB.
3-0 PUBMNR	Indicates minor update of the PUB.

9.3.3.5 DWC_DDRPHYA_INITENG register descriptions

9.3.3.5.1 DWC_DDRPHYA_INITENG Memory map

DesignWare Cores DDR4/3 PHY address block

DWC_DDRPHYA_INITENG0 base address: 9_0000h

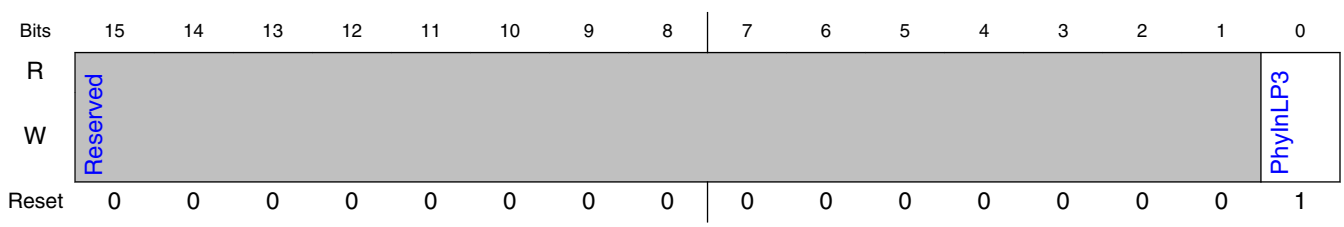
Offset	Register	Width (In bits)	Access	Reset value
50h	Indicator for PIE Lower Power 3 (LP3) Status (PhyInLP3)	16	RW	0001h

9.3.3.5.2 Indicator for PIE Lower Power 3 (LP3) Status (PhyInLP3)

9.3.3.5.2.1 Offset

Register	Offset
PhyInLP3	50h

9.3.3.5.2.2 Diagram



9.3.3.5.2.3 Fields

Field	Function
15-1 —	Reserved
0 PhyInLP3	Read Only. Read Only. Set to 1 by the PIE once completed LP3 Entry sequence; Cleared during LP3 Exit sequence. System software can read this csr for the status of PIE engine.

9.3.3.6 DWC_DDRPHYA_MASTER register descriptions

9.3.3.6.1 DWC_DDRPHYA_MASTER Memory map

DDR4/3 PHY address block

DWC_DDRPHYA_MASTER0 base address: 2_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Rx FIFO pointer initialization control (RxFifoInit)	16	RW	0000h
2h	Clock gating control (ForceClkDisable)	16	RW	0000h
6h	This Register used by Training Firmware to force an internal PHY Update Event. (ForceInternalUpdate)	16	RW	0000h
8h	Read Only displays PHY Configuration. (PhyConfig)	16	RO	Table 9-4
Ah	PHY General Configuration Register(PGCR). (PGCR)	16	RW	0000h
Eh	Test Bump Control1 (TestBumpCntrl1)	16	RW	0B03h
10h	Impedance Calibration Clock Ratio (CalUclkInfo_p0)	16	RW	0320h
14h	Test Bump Control (TestBumpCntrl)	16	RW	0000h
16h	PHY Initialization Engine (PIE) Delay Register 0 (Seq0BDLY0_p0)	16	RW	0000h
18h	PHY Initialization Engine (PIE) Delay Register 1 (Seq0BDLY1_p0)	16	RW	0000h
1Ah	PHY Initialization Engine (PIE) Delay Register 2 (Seq0BDLY2_p0)	16	RW	0000h
1Ch	PHY Initialization Engine (PIE) Delay Register 3 (Seq0BDLY3_p0)	16	RW	0000h
1Eh	PHY Alert status bit (PhyAlertStatus)	16	RO	Table 9-4
20h	Setup Intervals for DFI PHY Master operations (PPTTrainSetup_p0)	16	RW	0000h
24h	ATestMode control (ATestMode)	16	RW	0000h
28h	TX P Impedance Calibration observation (TxCalBinP)	16	RO	Table 9-4
2Ah	TX N Impedance Calibration observation (TxCalBinN)	16	RO	Table 9-4
2Ch	TX P Impedance Calibration override (TxCalPOvr)	16	RW	0000h
2Eh	TX N Impedance Calibration override (TxCalNOvr)	16	RW	0000h
30h	Enables for update and low-power interfaces for DFI0 and DFI1 (DfiMode)	16	RW	0005h
32h	Mode select register for MEMCLK/Address/Command Tristates (TristateModeCA_p0)	16	RW	0000h
34h	Digital Observation Pin control (MtestMuxSel)	16	RW	0000h
36h	Digital Observation Pin program info for debug (MtestPgmInfo)	16	RW	0000h
38h	Dynaimc Power Up/Down control (DynPwrDnUp)	16	RW	0000h
3Ch	PHY Technology ID Register (PhyTID)	16	RW	0000h
40h	HWT MaxReadLatency. (HwtMRL_p0)	16	RW	0006h
42h	DFI PhyUpdate Request time counter (in MEMCLKs) (DFIPHYUPD)	16	RW	FF07h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
44h	Controls the write DQ generation for Per-Dram-Addressing of MRS (PdaMrsWriteMode)	16	RW	0000h
46h	Controls whether dfi_geardown_en will cause CS and CKE timing to change. (DFIGEARDOWNCTL)	16	RW	0000h
48h	Control the PHY logic related to the read and write DQS preamble (DqsPreambleControl_p0)	16	RW	0008h
4Ah	DBYTE module controls to select X4 Dram device mode (MasterX4 Config)	16	RW	0000h
4Ch	Write level feedback DQ observability select. (WrLevBits)	16	RW	0098h
4Eh	In DDR4 Mode , this controls whether CS_N[3:2] should be multicast on CID[1:0] (EnableCsMulticast)	16	RW	0000h
50h	Drives cs_n[0] onto cs_n[1] during training (HwtLpCsMultiCast)	16	RW	0000h
58h	Disable for unused ACX Nibbles (Acx4AnibDis)	16	RW	0000h
5Ah	This Register is used to enable the Read-DBI function in each DBYTE (DMIPinPresent_p0)	16	RW	0000h
5Ch	Address/Command FIFO ReadPointer Initial Value (ARdPtrInitVal_p0)	16	RW	0000h
74h	DLL Mode control CSR for DBYTEs (DbyteDllModeCntrl)	16	RW	Table 9-4
8Ah	Impedance Calibration offsets control (CalOffsets)	16	RW	0000h
8Eh	Sar Init Vals (SarInitVals)	16	RW	0127h
92h	Impedance Calibration PExt Override control (CalPExtOvr)	16	RW	0000h
94h	Impedance Calibration Cmpr 50 control (CalCmpr5Ovr)	16	RW	0000h
96h	Impedance Calibration NInt Override control (CalNIntOvr)	16	RW	0000h
A0h	Impedance Calibration driver strength control (CalDrvStr0)	16	RW	0000h
ACh	READ DATA On-Die Termination Timing Control (by PHY) (ProcOdtTimeCtl_p0)	16	RW	0006h
B6h	This Register is used to configure the MemAlert Receiver (MemAlert Control)	16	RW	4000h
B8h	This Register is used to configure the MemAlert Receiver (MemAlert Control2)	16	RW	0000h
C0h	Protection and control of BP_MemReset_L (MemResetL)	16	RW	0000h
DAh	Drive CS_N 3:0 onto CS_N 7:4 (DriveCSLowOntoHigh)	16	RW	0000h
DCh	PUBMODE - HWT Mux Select (PUBMODE)	16	RW	0000h
DEh	Misc PHY status bits (MiscPhyStatus)	16	RO	Table 9-4
E0h	Controls whether the loopback path bypasses the final PAD node. (CoreLoopbackSel)	16	RW	0000h
E2h	DLL Various Training Parameters (DllTrainParam)	16	RW	0002h
E8h	CSn Disable Bypass for LPDDR3/4 (HwtLpCsEnBypass)	16	RW	0000h
EAh	Dfi Command/Address Mode (DfiCAMode)	16	RW	0000h
F0h	DLL Lock State machine control register (DllControl)	16	RW	0003h
F2h	DLL update phase control (PulseDllUpdatePhase)	16	RW	0000h
F8h	DLL gain control (DllGainCtl_p0)	16	RW	00A1h
110h	Impedance Calibration Control (CalRate)	16	RW	0009h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
112h	Impedance Calibration Zap/Reset (CalZap)	16	RW	0000h
116h	PSTATE Selection (PState)	16	RW	000Ch
11Ah	PLL Output Control (PllOutGateControl)	16	RW	0000h
120h	PMU Power-on Reset Control (PLL/DLL Lock Done) (PorControl)	16	RW	0000h
12Eh	Impedance Calibration Busy Status (CalBusy)	16	RO	Table 9-4
130h	Miscellaneous impedance calibration controls. (CalMisc2)	16	RW	0004h
134h	Controls for disabling the impedance calibration of certain targets. (CalMisc)	16	RW	0000h
136h	(CalVRefs)	16	RO	0002h
138h	Impedance Calibration Cmpr control (CalCmpr5)	16	RO	Table 9-4
13Ah	Impedance Calibration NInt control (CalNInt)	16	RO	Table 9-4
13Ch	Impedance Calibration PExt control (CalPExt)	16	RO	Table 9-4
150h	Impedance Calibration Cmp Invert control (CalCmpInvert)	16	RW	0003h
15Ch	Impedance Calibration Cmpana control (CalCmpanaCntrl)	16	RW	0112h
160h	Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiRdDataCsDestMap_p0)	16	RW	00E4h
164h	PHY Global Vref Controls (VrefInGlobal_p0)	16	RW	0200h
168h	Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiWrDataCsDestMap_p0)	16	RW	00E4h
16Ah	Counts successful PHY Master Interface Updates (PPTs) (MasUpdGoodCtr)	16	RO	Table 9-4
16Ch	Counts successful PHY-initiated DFI0 Interface Updates (PhyUpd0GoodCtr)	16	RO	Table 9-4
16Eh	Counts successful PHY-initiated DFI1 Interface Updates (PhyUpd1GoodCtr)	16	RO	Table 9-4
170h	Counts successful Memory Controller DFI0 Interface Updates (CtlUpd0GoodCtr)	16	RO	Table 9-4
172h	Counts successful Memory Controller DFI1 Interface Updates (CtlUpd1GoodCtr)	16	RO	Table 9-4
174h	Counts unsuccessful PHY Master Interface Updates (MasUpdFailCtr)	16	RO	Table 9-4
176h	Counts unsuccessful PHY-initiated DFI0 Interface Updates (PhyUpd0FailCtr)	16	RO	Table 9-4
178h	Counts unsuccessful PHY-initiated DFI1 Interface Updates (PhyUpd1FailCtr)	16	RO	Table 9-4
17Ah	Enables for Performance Counters (PhyPerfCtrEnable)	16	RW	0000h
186h	PLL Power Down (PllPwrDn)	16	RW	0001h
188h	PLL Reset (PllReset)	16	RW	0001h
18Ah	PState dependent PLL Control Register 2 (PllCtrl2_p0)	16	RW	0019h
18Ch	PLL Control Register 0 (PllCtrl0)	16	RW	2008h
18Eh	PState dependent PLL Control Register 1 (PllCtrl1_p0)	16	RW	0020h
190h	PLL Testing Control Register (PllTst)	16	RW	0000h
192h	PLL's pll_lock pin output (PllLockStatus)	16	RO	Table 9-4

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
194h	Additional controls for PLL CP/VCO modes of operation (PllTestMode_p0)	16	RW	0124h
196h	PLL Control Register 3 (PllCtrl3)	16	RW	01F0h
198h	PState dependent PLL Control Register 4 (PllCtrl4_p0)	16	RW	017Fh
19Ah	PLL's eoc (end of calibration) output (PllEndofCal)	16	RO	Table 9-4
19Ch	PLL's standby_eff (effective standby) output (PllStandbyEff)	16	RO	Table 9-4
19Eh	PLL's Dacval_out output (PllDacValOut)	16	RO	Table 9-4
1C6h	Controls for use in observing and testing the LCDLs. (LcdlDbgCntl)	16	RW	0000h
1C8h	Debug status of the DBYTE LCDL (AcLcdlStatus)	16	RO	Table 9-4
1DAh	Customer settable by the customer (CUSTPHYREV)	16	RO	Table 9-4
1DCh	The hardware version of this PHY, excluding the PUB (PHYREV)	16	RO	0100h
1DEh	Start vector value to be used for LP3-exit or Init PIE Sequence (LP3ExitSeq0BStartVector)	16	RW	00DEh
1E0h	DFI Frequency Translation Register 0 (DfiFreqXlat0)	16	RW	0000h
1E2h	DFI Frequency Translation Register 1 (DfiFreqXlat1)	16	RW	0000h
1E4h	DFI Frequency Translation Register 2 (DfiFreqXlat2)	16	RW	0000h
1E6h	DFI Frequency Translation Register 3 (DfiFreqXlat3)	16	RW	0000h
1E8h	DFI Frequency Translation Register 4 (DfiFreqXlat4)	16	RW	0000h
1EAh	DFI Frequency Translation Register 5 (DfiFreqXlat5)	16	RW	0000h
1ECh	DFI Frequency Translation Register 6 (DfiFreqXlat6)	16	RW	0000h
1EEh	DFI Frequency Translation Register 7 (DfiFreqXlat7)	16	RW	0000h
1F0h	TxRdPtrInit control register (TxRdPtrInit)	16	RW	0001h
1F2h	DFI Init Complete control (DfiInitComplete)	16	RW	0000h
1F4h	DFI Frequency Ratio (DfiFreqRatio_p0)	16	RW	0001h
1F6h	Enable more frequent consistency checks of the RX FIFOs (RxFifoChecks)	16	RW	0000h
1FEh	(MTestDtoCtrl)	16	RW	0000h
200h	Maps PHY CAA lane 0 from dfi0_address of the index of the register contents (MapCAA0toDfi)	16	RW	0000h
202h	Maps PHY CAA lane 1 from dfi0_address of the index of the register contents (MapCAA1toDfi)	16	RW	0001h
204h	Maps PHY CAA lane 2 from dfi0_address of the index of the register contents (MapCAA2toDfi)	16	RW	0002h
206h	Maps PHY CAA lane 3 from dfi0_address of the index of the register contents (MapCAA3toDfi)	16	RW	0003h
208h	Maps PHY CAA lane 4 from dfi0_address of the index of the register contents (MapCAA4toDfi)	16	RW	0004h
20Ah	Maps PHY CAA lane 5 from dfi0_address of the index of the register contents (MapCAA5toDfi)	16	RW	0005h
20Ch	Maps PHY CAA lane 6 from dfi0_address of the index of the register contents (MapCAA6toDfi)	16	RW	0006h
20Eh	Maps PHY CAA lane 7 from dfi0_address of the index of the register contents (MapCAA7toDfi)	16	RW	0007h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
210h	Maps PHY CAA lane 8 from dfi0_address of the index of the register contents (MapCAA8toDfi)	16	RW	0008h
212h	Maps PHY CAA lane 9 from dfi0_address of the index of the register contents (MapCAA9toDfi)	16	RW	0009h
220h	Maps PHY CAB lane 0 from dfi1_address of the index of the register contents (MapCAB0toDfi)	16	RW	0000h
222h	Maps PHY CAB lane 1 from dfi1_address of the index of the register contents (MapCAB1toDfi)	16	RW	0001h
224h	Maps PHY CAB lane 2 from dfi1_address of the index of the register contents (MapCAB2toDfi)	16	RW	0002h
226h	Maps PHY CAB lane 3 from dfi1_address of the index of the register contents (MapCAB3toDfi)	16	RW	0003h
228h	Maps PHY CAB lane 4 from dfi1_address of the index of the register contents (MapCAB4toDfi)	16	RW	0004h
22Ah	Maps PHY CAB lane 5 from dfi1_address of the index of the register contents (MapCAB5toDfi)	16	RW	0005h
22Ch	Maps PHY CAB lane 6 from dfi1_address of the index of the register contents (MapCAB6toDfi)	16	RW	0006h
22Eh	Maps PHY CAB lane 7 from dfi1_address of the index of the register contents (MapCAB7toDfi)	16	RW	0007h
230h	Maps PHY CAB lane 8 from dfi1_address of the index of the register contents (MapCAB8toDfi)	16	RW	0008h
232h	Maps PHY CAB lane 9 from dfi1_address of the index of the register contents (MapCAB9toDfi)	16	RW	0009h
236h	Interrupt Enable Bits (PhyInterruptEnable)	16	RW	0000h
238h	Interrupt Firmware Control Bits (PhyInterruptFWControl)	16	RW	0000h
23Ah	Interrupt Mask Bits (PhyInterruptMask)	16	RW	0000h
23Ch	Interrupt Clear Bits (PhyInterruptClear)	16	RW	0000h
23Eh	Interrupt Status Bits (PhyInterruptStatus)	16	RO	Table 9-4
240h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress0)	16	RW	0000h
242h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress1)	16	RW	0000h
244h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress2)	16	RW	0000h
246h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress3)	16	RW	0000h
248h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress4)	16	RW	0000h
24Ah	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress5)	16	RW	0000h
24Ch	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress6)	16	RW	0000h
24Eh	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress7)	16	RW	0000h
250h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress8)	16	RW	0000h
252h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress9)	16	RW	0000h
254h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress10)	16	RW	0000h
256h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress11)	16	RW	0000h
258h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress12)	16	RW	0000h
25Ah	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress13)	16	RW	0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
25Ch	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress14)	16	RW	0000h
25Eh	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress15)	16	RW	0000h
260h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress17)	16	RW	0000h
262h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtActN)	16	RW	0000h
264h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtBank0)	16	RW	0000h
266h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtBank1)	16	RW	0000h
268h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtBank2)	16	RW	0000h
26Ah	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtBg0)	16	RW	0000h
26Ch	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtBg1)	16	RW	0000h
26Eh	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtCasN)	16	RW	0000h
270h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtRasN)	16	RW	0000h
272h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtWeN)	16	RW	0000h
274h	Signal swizzle selection for HWT swizzle (HwtSwizzleHwtParityIn)	16	RW	0000h
278h	Add assertion/deassertion delays on handshake signals Logic assumes that dfi signal assertions exceed the programmed delays (DfiHandshakeDelays0)	16	RW	0000h
27Ah	Add assertion/deassertion delays on handshake signals Logic assumes that dfi signal assertions exceed the programmed delays (DfiHandshakeDelays1)	16	RW	0000h
20_0010h	Impedance Calibration Clock Ratio (CalUclkInfo_p1)	16	RW	0320h
20_0016h	PHY Initialization Engine (PIE) Delay Register 0 (Seq0BDLY0_p1)	16	RW	0000h
20_0018h	PHY Initialization Engine (PIE) Delay Register 1 (Seq0BDLY1_p1)	16	RW	0000h
20_001Ah	PHY Initialization Engine (PIE) Delay Register 2 (Seq0BDLY2_p1)	16	RW	0000h
20_001Ch	PHY Initialization Engine (PIE) Delay Register 3 (Seq0BDLY3_p1)	16	RW	0000h
20_0020h	Setup Intervals for DFI PHY Master operations (PPTTrainSetup_p1)	16	RW	0000h
20_0032h	Mode select register for MEMCLK/Address/Command Tristates (TristateModeCA_p1)	16	RW	0000h
20_0040h	HWT MaxReadLatency. (HwtMRL_p1)	16	RW	0006h
20_0048h	Control the PHY logic related to the read and write DQS preamble (DqsPreambleControl_p1)	16	RW	0008h
20_005Ah	This Register is used to enable the Read-DBI function in each DBYTE (DMIPinPresent_p1)	16	RW	0000h
20_005Ch	Address/Command FIFO ReadPointer Initial Value (ARdPtrInitVal_p1)	16	RW	0000h
20_00ACh	READ DATA On-Die Termination Timing Control (by PHY) (ProcOdtTimeCtl_p1)	16	RW	0006h
20_00F8h	DLL gain control (DllGainCtl_p1)	16	RW	00A1h
20_0160h	Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiRdDataCsDestMap_p1)	16	RW	00E4h
20_0164h	PHY Global Vref Controls (VrefInGlobal_p1)	16	RW	0200h
20_0168h	Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiWrDataCsDestMap_p1)	16	RW	00E4h
20_018Ah	PState dependent PLL Control Register 2 (PllCtrl2_p1)	16	RW	0019h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
20_018Eh	PState dependent PLL Control Register 1 (PllCtrl1_p1)	16	RW	0020h
20_0194h	Additional controls for PLL CP/VCO modes of operation (PllTestMode_p1)	16	RW	0124h
20_0198h	PState dependent PLL Control Register 4 (PllCtrl4_p1)	16	RW	017Fh
20_01F4h	DFI Frequency Ratio (DfiFreqRatio_p1)	16	RW	0001h
40_0010h	Impedance Calibration Clock Ratio (CalUclkInfo_p2)	16	RW	0320h
40_0016h	PHY Initialization Engine (PIE) Delay Register 0 (Seq0BDLY0_p2)	16	RW	0000h
40_0018h	PHY Initialization Engine (PIE) Delay Register 1 (Seq0BDLY1_p2)	16	RW	0000h
40_001Ah	PHY Initialization Engine (PIE) Delay Register 2 (Seq0BDLY2_p2)	16	RW	0000h
40_001Ch	PHY Initialization Engine (PIE) Delay Register 3 (Seq0BDLY3_p2)	16	RW	0000h
40_0020h	Setup Intervals for DFI PHY Master operations (PPTTrainSetup_p2)	16	RW	0000h
40_0032h	Mode select register for MEMCLK/Address/Command Tristates (TristateModeCA_p2)	16	RW	0000h
40_0040h	HWT MaxReadLatency. (HwtMRL_p2)	16	RW	0006h
40_0048h	Control the PHY logic related to the read and write DQS preamble (DqsPreambleControl_p2)	16	RW	0008h
40_005Ah	This Register is used to enable the Read-DBI function in each DBYTE (DMIPinPresent_p2)	16	RW	0000h
40_005Ch	Address/Command FIFO ReadPointer Initial Value (ARdPtrInitVal_p2)	16	RW	0000h
40_00ACh	READ DATA On-Die Termination Timing Control (by PHY) (ProcOdtTimeCtl_p2)	16	RW	0006h
40_00F8h	DLL gain control (DllGainCtl_p2)	16	RW	00A1h
40_0160h	Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiRdDataCsDestMap_p2)	16	RW	00E4h
40_0164h	PHY Global Vref Controls (VrefInGlobal_p2)	16	RW	0200h
40_0168h	Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiWrDataCsDestMap_p2)	16	RW	00E4h
40_018Ah	PState dependent PLL Control Register 2 (PllCtrl2_p2)	16	RW	0019h
40_018Eh	PState dependent PLL Control Register 1 (PllCtrl1_p2)	16	RW	0020h
40_0194h	Additional controls for PLL CP/VCO modes of operation (PllTestMode_p2)	16	RW	0124h
40_0198h	PState dependent PLL Control Register 4 (PllCtrl4_p2)	16	RW	017Fh
40_01F4h	DFI Frequency Ratio (DfiFreqRatio_p2)	16	RW	0001h
60_0010h	Impedance Calibration Clock Ratio (CalUclkInfo_p3)	16	RW	0320h
60_0016h	PHY Initialization Engine (PIE) Delay Register 0 (Seq0BDLY0_p3)	16	RW	0000h
60_0018h	PHY Initialization Engine (PIE) Delay Register 1 (Seq0BDLY1_p3)	16	RW	0000h
60_001Ah	PHY Initialization Engine (PIE) Delay Register 2 (Seq0BDLY2_p3)	16	RW	0000h
60_001Ch	PHY Initialization Engine (PIE) Delay Register 3 (Seq0BDLY3_p3)	16	RW	0000h
60_0020h	Setup Intervals for DFI PHY Master operations (PPTTrainSetup_p3)	16	RW	0000h
60_0032h	Mode select register for MEMCLK/Address/Command Tristates (TristateModeCA_p3)	16	RW	0000h

Table continues on the next page...

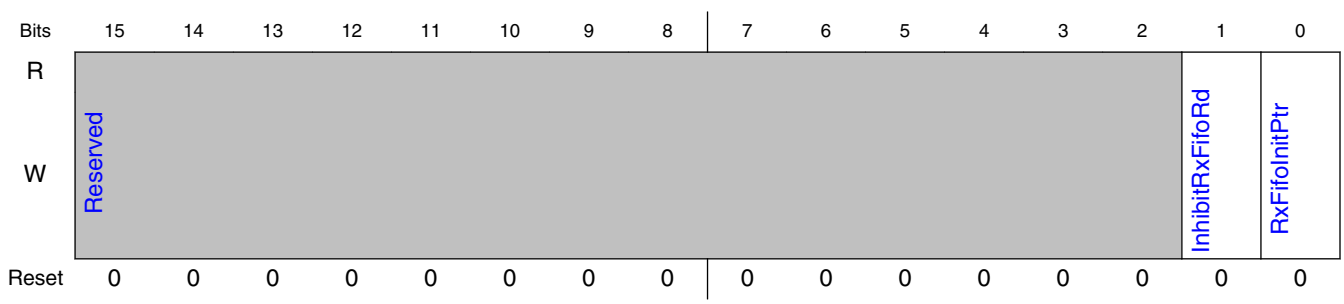
Offset	Register	Width (In bits)	Access	Reset value
60_0040h	HWT MaxReadLatency. (HwtMRL_p3)	16	RW	0006h
60_0048h	Control the PHY logic related to the read and write DQS preamble (DqsPreambleControl_p3)	16	RW	0008h
60_005Ah	This Register is used to enable the Read-DBI function in each DBYTE (DMIPinPresent_p3)	16	RW	0000h
60_005Ch	Address/Command FIFO ReadPointer Initial Value (ARdPtrInitVal_p3)	16	RW	0000h
60_00ACh	READ DATA On-Die Termination Timing Control (by PHY) (ProcOdtTimeCtl_p3)	16	RW	0006h
60_00F8h	DLL gain control (DllGainCtl_p3)	16	RW	00A1h
60_0160h	Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiRdDataCsDestMap_p3)	16	RW	00E4h
60_0164h	PHY Global Vref Controls (VrefInGlobal_p3)	16	RW	0200h
60_0168h	Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiWrDataCsDestMap_p3)	16	RW	00E4h
60_018Ah	PState dependent PLL Control Register 2 (PllCtrl2_p3)	16	RW	0019h
60_018Eh	PState dependent PLL Control Register 1 (PllCtrl1_p3)	16	RW	0020h
60_0194h	Additional controls for PLL CP/VCO modes of operation (PllTestMode_p3)	16	RW	0124h
60_0198h	PState dependent PLL Control Register 4 (PllCtrl4_p3)	16	RW	017Fh
60_01F4h	DFI Frequency Ratio (DfiFreqRatio_p3)	16	RW	0001h

9.3.3.6.2 Rx FIFO pointer initialization control (RxFifoInit)

9.3.3.6.2.1 Offset

Register	Offset
RxFifoInit	0h

9.3.3.6.2.2 Diagram



9.3.3.6.2.3 Fields

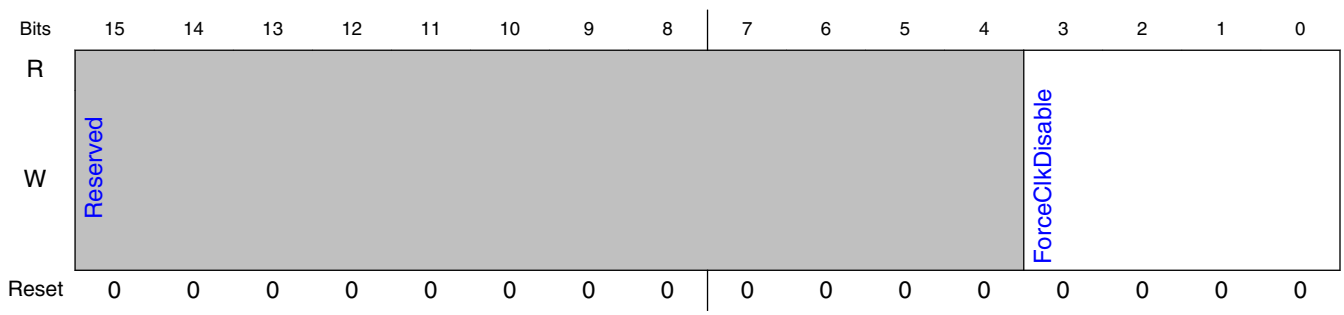
Field	Function
15-2 —	Reserved
1 InhibitRxFifoRd	This field is reserved for training FW use. This field is reserved for training FW use. Setting this inhibits reads of the PHYRXDATAFIFO.
0 RxFifoInitPtr	Setting this bit will reset the PHY RXDATAFIFO read and write pointers. Setting this bit will reset the PHY RXDATAFIFO read and write pointers. This bit must be Cleared to return to normal operation. This is not required in normal operation, but can be used instead of a dfi_ctrlupdate (for controllers that don't support ctrlupdate)

9.3.3.6.3 Clock gating control (ForceClkDisable)

9.3.3.6.3.1 Offset

Register	Offset
ForceClkDisable	2h

9.3.3.6.3.2 Diagram



9.3.3.6.3.3 Fields

Field	Function
15-4 —	Reserved
3-0 ForceClkDisable	This CSR forces the gating of MEMCLKs driven from the PHY ForceClkDisable[0] - controls CLK_H/L0 ForceClkDisable[1] - controls CLK_H/L1 (if present) ForceClkDisable[2] - controls CLK_H/L2 (if present) ForceClkDisable[3] - controls CLK_H/L3 (if present)

9.3.3.6.4 This Register used by Training Firmware to force an internal PHY Update Event. (ForceInternalUpdate)

9.3.3.6.4.1 Offset

Register	Offset
ForceInternalUpdate	6h

9.3.3.6.4.2 Diagram

9.3.3.6.4.3 Fields

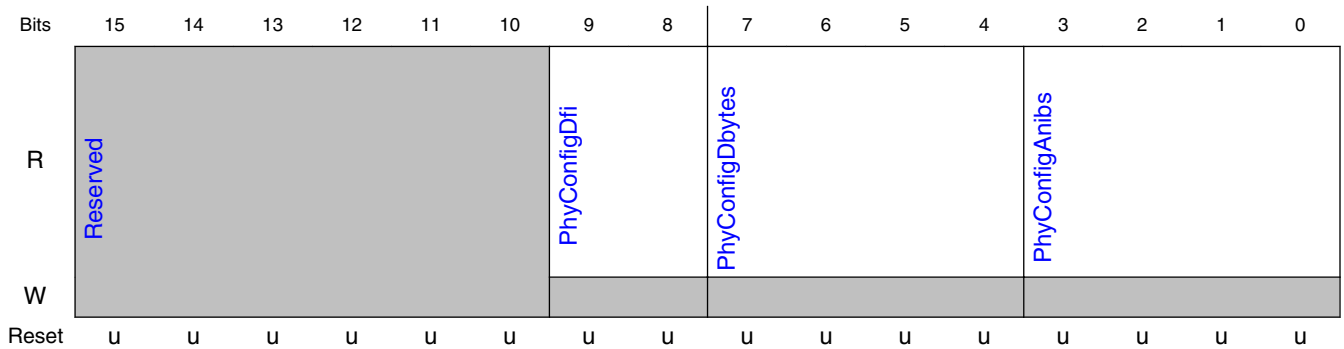
Field	Function
15-1 —	Reserved
0 ForceInternalUpdate	This Register is used by Training Firmware to force an internal PHY Update Event. This Register is used by Training Firmware to force an internal PHY Update Event. Optionally can be used by System Software to issue a PHY update Event. The register may be written to 1'b1 only when all of the following are true: [+] The DFI interface is offline. [+] The Training Hardware is sending only DES and all transactions have retired. Additionally, this Register must stay asserted for at least 32 DFICLKs before clearing. Prematurely clearing this register may result in an incomplete update.

9.3.3.6.5 Read Only displays PHY Configuration. (PhyConfig)

9.3.3.6.5.1 Offset

Register	Offset
PhyConfig	8h

9.3.3.6.5.2 Diagram



9.3.3.6.5.3 Fields

Field	Function
15-10 —	Reserved
9-8 PhyConfigDfi	Returns the following value . Returns the following value ... depending on the define "0x1" if DWC_DDRPHY_DFI1_EXISTS is not defined "0x2" if DWC_DDRPHY_DFI1_EXISTS is defined
7-4 PhyConfigDbytes	Returns the following value . Returns the following value ... depending on the chosen define "0x1" if DWC_DDRPHY_NUM_DBYTES_1 "0x2" if DWC_DDRPHY_NUM_DBYTES_2 "0x3" if DWC_DDRPHY_NUM_DBYTES_3 "0x4" if DWC_DDRPHY_NUM_DBYTES_4 "0x5" if DWC_DDRPHY_NUM_DBYTES_5 "0x6" if DWC_DDRPHY_NUM_DBYTES_6 "0x7" if DWC_DDRPHY_NUM_DBYTES_7 "0x8" if DWC_DDRPHY_NUM_DBYTES_8 "0x9" if DWC_DDRPHY_NUM_DBYTES_9 "0xA" if DWC_DDRPHY_NUM_DBYTES_10
3-0	Returns the following value .

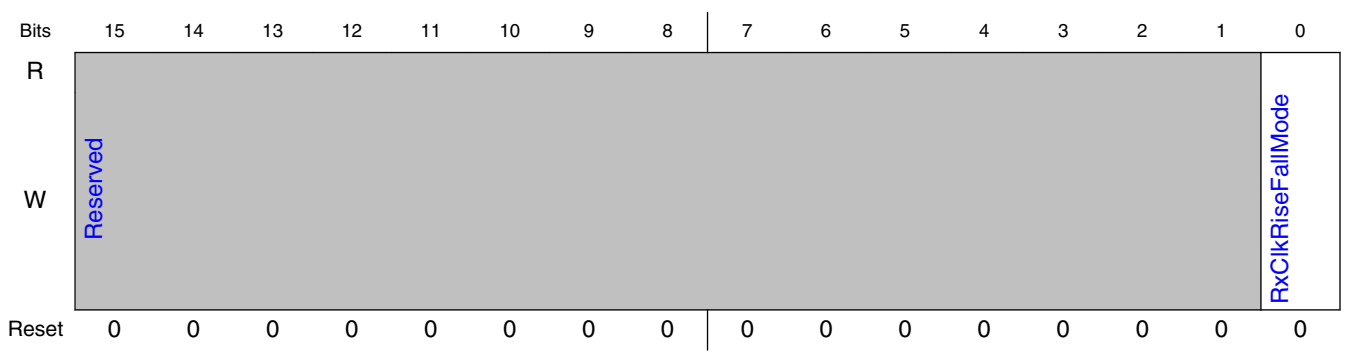
Field	Function
PhyConfigAnibs	Returns the following value ... depending on the chosen define "0x3" if DWC_DDRPHY_NUM_ANIBS_3 "0x6" if DWC_DDRPHY_NUM_ANIBS_6 "0xA" if DWC_DDRPHY_NUM_ANIBS_10 "0xC" if DWC_DDRPHY_NUM_ANIBS_12

9.3.3.6.6 PHY General Configuration Register(PGCR). (PGCR)

9.3.3.6.6.1 Offset

Register	Offset
PGCR	Ah

9.3.3.6.6.2 Diagram



9.3.3.6.6.3 Fields

Field	Function
15-1 —	Reserved
0 RxClkRiseFallMode	This register field controls independent training for RxClk_c and RxClk_t. This register field controls independent training for RxClk_c and RxClk_t. 0: Independent training for RxClk_c and RxClk_t is disabled. 1: Independent training for RxClk_c and RxClk_t is enabled. Default Value is 0.

9.3.3.6.7 Test Bump Control1 (TestBumpCntrl1)

9.3.3.6.7.1 Offset

Register	Offset
TestBumpCntrl1	Eh

9.3.3.6.7.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	TestPowerGateEn	TestExtVrefRange	TestSelExternalVref	TestGainCurrAdj					TestAnalogOutCtrl				TestBiasBypassEn	TestMajorMode		
Reset	0	0	0	0	1	0	1	1	0	0	0	0	0	0	1	1

9.3.3.6.7.3 Fields

Field	Function
15 TestPowerGateEn	Do not use, for debug only
14 TestExtVrefRange	Setting this bit will extend the VREF DAC range for debug. Setting this bit will extend the VREF DAC range for debug. MUST BE SET TO ZERO in mission mode when MemAlert is enabled
13 TestSelExternalVref	Do not use, for debug only
12-8 TestGainCurrAdj	Adjust gain and current of analog observe RX amplifier stage at analog test point Recommended mission mode default = 5'b01011
7-4 TestAnalogOutCtrl	Select receiver internal analog signals to monitor at analog test point 0xxx: AnalogTestOut=HiZ 1000: AnalogTestOut=VSS 1001: AnalogTestOut=vref_dfe0 -- observe by sweeping MALERTVrefLevel 1010: AnalogTestOut=vref_dfe1 -- observe by sweeping MALERTVrefLevel 1011: AnalogTestOut=VSS 1100: AnalogTestOut=vstg2 1101: AnalogTestOut=vcasc_cs1 1110: AnalogTestOut=vbias_cs1 Recommended mission mode default = 4'b0000
3 TestBiasBypassEn	Do not use, for debug only

Table continues on the next page...

Field	Function
2-0 TestMajorMode	Selects the major mode of operation for the receiver. Selects the major mode of operation for the receiver. These modes are mutually exclusive. 000 - setting for DDR3-RDIMM systems to receive ERROUT_n 001 - reserved 010 - reserved 011 - setting for DDR4 systems to receive Alert_n 100 - reserved for debug 101 - reserved 110 - reserved 111 - reserved for debug

9.3.3.6.8 Impedance Calibration Clock Ratio (CalUclkInfo_p0)

9.3.3.6.8.1 Offset

Register	Offset
CalUclkInfo_p0	10h

9.3.3.6.8.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								CalUClkTicksPer1uS							
Reset	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0

9.3.3.6.8.3 Fields

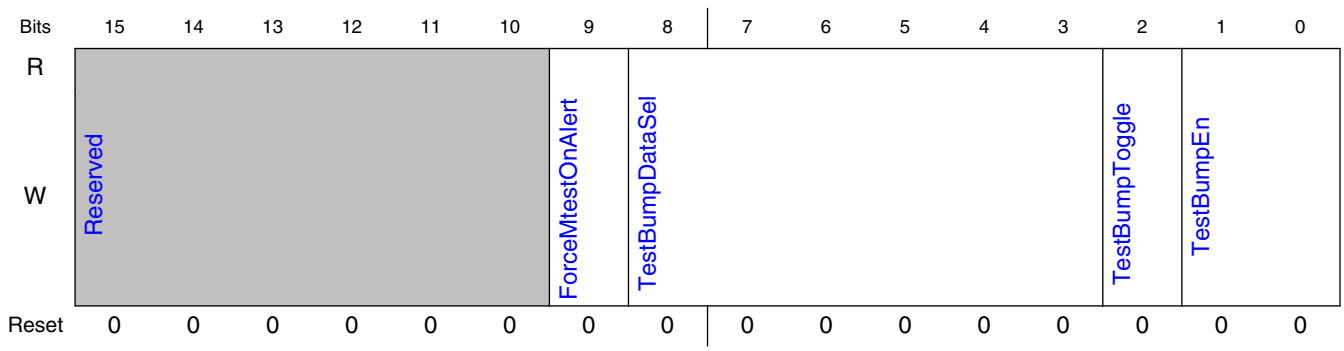
Field	Function
15-10 —	Reserved
9-0 CalUClkTicksPer1uS	Must be programmed to the number of DfiClks in 1us (rounded up), with minimum value of 24. Must be programmed to the number of DfiClks in 1us (rounded up), with minimum value of 24. if (DfiClk < 24MHz) CalUclkInfo = 24 else CalUclkInfo = (number of DfiClks in 1us)

9.3.3.6.9 Test Bump Control (TestBumpCntrl)

9.3.3.6.9.1 Offset

Register	Offset
TestBumpCntrl	14h

9.3.3.6.9.2 Diagram



9.3.3.6.9.3 Fields

Field	Function
15-10 —	Reserved
9 ForceMtestOnAlert	When set, causes the Digital Observation output pin to be driven onto BP_ALERT_N
8-3 TestBumpDataSel	RVSD.
2 TestBumpToggle	This field controls the output function of the signal Digital Observation Pin, if available in the configuration of the PHY. This field controls the output function of the signal Digital Observation Pin, if available in the configuration of the PHY. When set to 1'b0 -- the output will be tristated When set to 1'b1 -- the output will be selected by MtestMuxSel
1-0 TestBumpEn	Field TestBumpEn[1:0] controls the output function of: the signal BP_ALERT_N. Field TestBumpEn[1:0] controls the output function of: the signal BP_ALERT_N. TestBumpEn Value on BP_ALERT_N output

Field	Function
	---------- ------------------------- 2'b00 high-z, 2'b01 PllOut-divided-by-2, works up to full data rate 2'b10 PllDigTst[1], works up to 400MHz 2'b11 PllDigTst[1], works up to 400MHz

9.3.3.6.10 PHY Initialization Engine (PIE) Delay Register 0 (Seq0BDLY0_p0)

9.3.3.6.10.1 Offset

Register	Offset
Seq0BDLY0_p0	16h

9.3.3.6.10.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Seq0BDLY0_p0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.10.3 Fields

Field	Function
15-0 Seq0BDLY0_p0	PHY Initialization Engine (PIE) Delay Register 0 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 0 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: -------------------------------- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it

Field	Function
	should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --------- ---------- ---------- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.11 PHY Initialization Engine (PIE) Delay Register 1 (Seq0BDLY1_p0)

9.3.3.6.11.1 Offset

Register	Offset
Seq0BDLY1_p0	18h

9.3.3.6.11.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Seq0BDLY1_p0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.11.3 Fields

Field	Function
15-0 Seq0BDLY1_p0	PHY Initialization Engine (PIE) Delay Register 1 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 1 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --------------------------------

Field	Function
	These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --------- ---------- ---------- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.12 PHY Initialization Engine (PIE) Delay Register 2 (Seq0BDLY2_p0)

9.3.3.6.12.1 Offset

Register	Offset
Seq0BDLY2_p0	1Ah

9.3.3.6.12.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Seq0BDLY2_p0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.12.3 Fields

Field	Function
15-0 Seq0BDLY2_p0	PHY Initialization Engine (PIE) Delay Register 2 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 2 This register is available for selection by the NOP and WAIT instructions

Field	Function
	in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.13 PHY Initialization Engine (PIE) Delay Register 3 (Seq0BDLY3_p0)

9.3.3.6.13.1 Offset

Register	Offset
Seq0BDLY3_p0	1Ch

9.3.3.6.13.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Seq0BDLY3_p0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.13.3 Fields

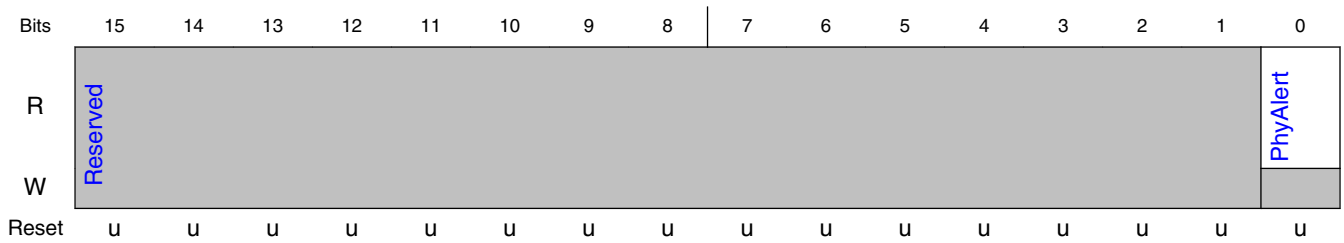
Field	Function
15-0 Seq0BDLY3_p0	PHY Initialization Engine (PIE) Delay Register 3 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 3 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.14 PHY Alert status bit (PhyAlertStatus)

9.3.3.6.14.1 Offset

Register	Offset
PhyAlertStatus	1Eh

9.3.3.6.14.2 Diagram



9.3.3.6.14.3 Fields

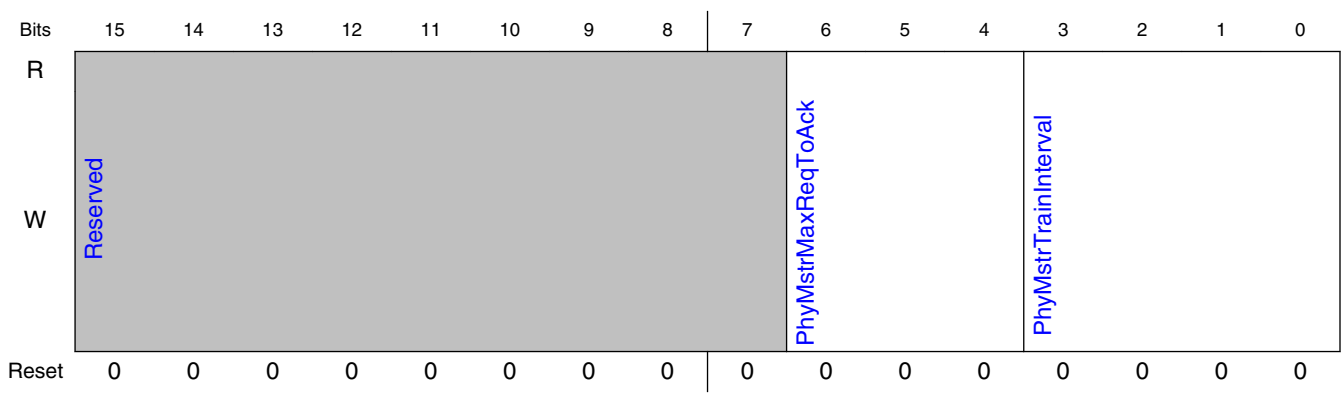
Field	Function
15-1 —	Reserved
0 PhyAlert	Current state of ALERT_N.

9.3.3.6.15 Setup Intervals for DFI PHY Master operations (PPTTrainSetup_p0)

9.3.3.6.15.1 Offset

Register	Offset
PPTTrainSetup_p0	20h

9.3.3.6.15.2 Diagram



9.3.3.6.15.3 Fields

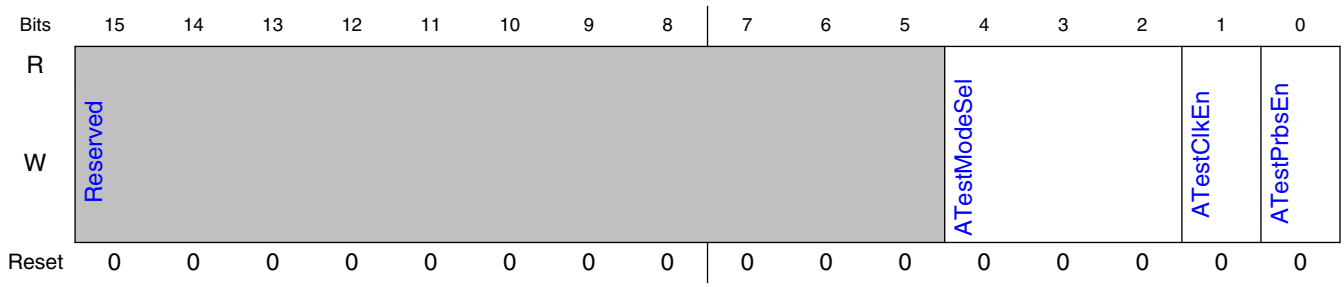
Field	Function
15-7 —	Reserved
6-4 PhyMstrMaxReqToAck	Bits 6:4 of this register specify the max time from tdfi_phymstr_req asserted to tdfi_phymstr_ack asserted. Bits 6:4 of this register specify the max time from tdfi_phymstr_req asserted to tdfi_phymstr_ack asserted. 3'b000 Disable PHY Master Interface 3'b001 set tPHYMSTR_resp. = 512 MEMCLKs 3'b010 set tPHYMSTR_resp. = 1024 MEMCLKs 3'b011 set tPHYMSTR_resp. = 2048 MEMCLKs 3'b100 set tPHYMSTR_resp. = 4096 MEMCLKs 3'b101 set tPHYMSTR_resp. = 8192 MEMCLKs 3'b110 set tPHYMSTR_resp. = 32768 MEMCLKs 3'b111 set tPHYMSTR_resp. = undefined
3-0 PhyMstrTrainInterval	Bits 3:0 of this register specifies the time between the end of one training and the start of the next. Bits 3:0 of this register specifies the time between the end of one training and the start of the next. (As per section 11 of the Preliminary DFI 4.0 Specification, V2, it is the max expected time from dfi_init_complete asserted to tdfi_phymstr_ack asserted). 4'b0000 Disable PHY Master Interface 4'b0001 PHY MASTER Request Interval = 524288 MEMCLKs 4'b0010 PHY MASTER Request Interval = 1048576 MEMCLKs 4'b0011 PHY MASTER Request Interval = 2097152 MEMCLKs 4'b0100 PHY MASTER Request Interval = 4194304 MEMCLKs 4'b0101 PHY MASTER Request Interval = 8388608 MEMCLKs 4'b0110 PHY MASTER Request Interval = 16777216 MEMCLKs 4'b0111 PHY MASTER Request Interval = 33554432 MEMCLKs 4'b1000 PHY MASTER Request Interval = 67108864 MEMCLKs 4'b1001 PHY MASTER Request Interval = 134217728 MEMCLKs 4'b1010 PHY MASTER Request Interval = 268435456 MEMCLKs 4'b1011 - 4'b1111 PHY MASTER Request Interval = undefined

9.3.3.6.16 ATestMode control (ATestMode)

9.3.3.6.16.1 Offset

Register	Offset
ATestMode	24h

9.3.3.6.16.2 Diagram



9.3.3.6.16.3 Fields

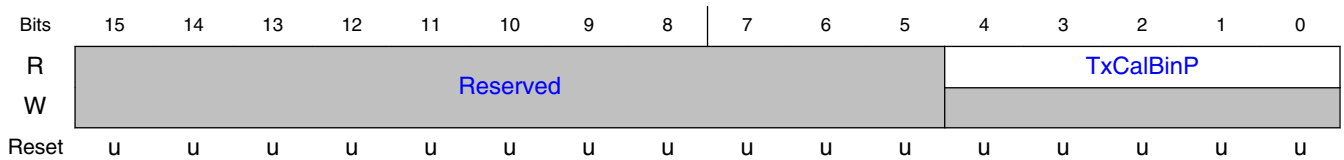
Field	Function
15-5 —	Reserved
4-2 ATestModeSel	Master Mode select for ATest (Loopback) 000 - Mission mode, all ATest disabled, loopback receivers powered down 001 - External Loopback mode [Single data rate pattern - dfi_cas sent to all lanes] 010 - Internal Loopback mode [Single data rate pattern] 011 - Internal Loopback mode [Double data rate pattern] 100 - External Loopback mode [Single data rate pattern - corresponding DFI signal sent to each lane]
1 ATestClkEn	Enables the clock for loopback PRBS7 testing for all BP_A* pins.
0 ATestPrbsEn	Enables loopback PRBS7 testing of all the DDR output pins in this chiplet. Enables loopback PRBS7 testing of all the DDR output pins in this chiplet. The asserting edge resets all the individual TestPrbsErrCnt & consequently TestPrbsErr; there is no other reset for the TestPrbsErrCnt. The ATestClkEn CSR below must be enabled first prior to enabling this bit.

9.3.3.6.17 TX P Impedance Calibration observation (TxCalBinP)

9.3.3.6.17.1 Offset

Register	Offset
TxCalBinP	28h

9.3.3.6.17.2 Diagram



9.3.3.6.17.3 Fields

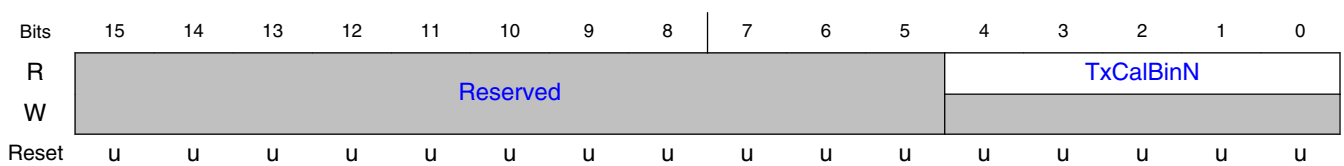
Field	Function
15-5 —	Reserved
4-0 TxCalBinP	This csr holds the binary result of the 31 bit thermometer pullup code.

9.3.3.6.18 TX N Impedance Calibration observation (TxCalBinN)

9.3.3.6.18.1 Offset

Register	Offset
TxCalBinN	2Ah

9.3.3.6.18.2 Diagram



9.3.3.6.18.3 Fields

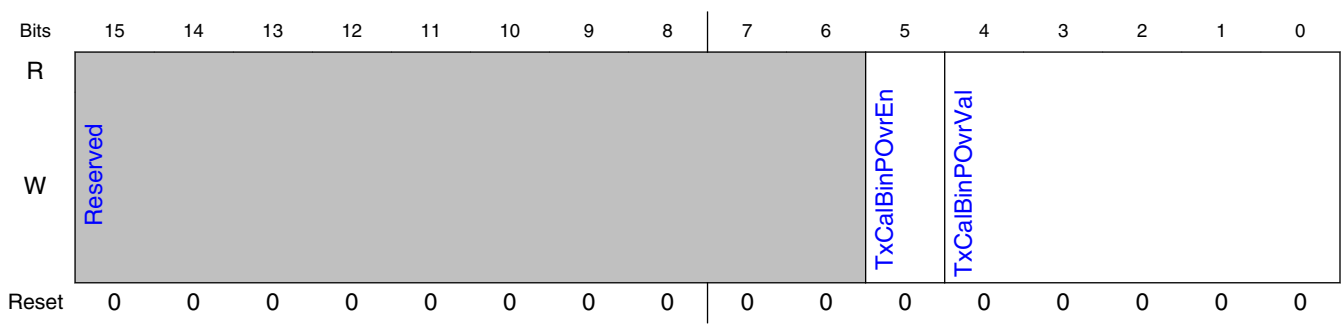
Field	Function
15-5 —	Reserved
4-0 TxCalBinN	This csr holds the binary result of the 31 bit thermometer pulldown code.

9.3.3.6.19 TX P Impedance Calibration override (TxCalPOvr)

9.3.3.6.19.1 Offset

Register	Offset
TxCalPOvr	2Ch

9.3.3.6.19.2 Diagram



9.3.3.6.19.3 Fields

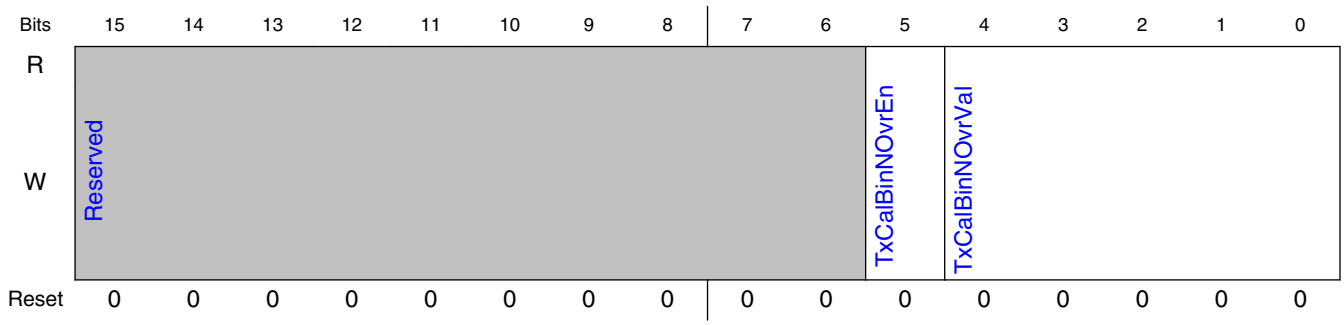
Field	Function
15-6 —	Reserved
5 TxCalBinPOvrEn	1 = use the override value present in Register TxCalBinPOvrVal 0 = don't.
4-0 TxCalBinPOvrVal	The binary value which can overide the Register TxCalBinP calibrator results if Register TxCalBinPOvrEn is set.

9.3.3.6.20 TX N Impedance Calibration override (TxCalNOvr)

9.3.3.6.20.1 Offset

Register	Offset
TxCalNOvr	2Eh

9.3.3.6.20.2 Diagram



9.3.3.6.20.3 Fields

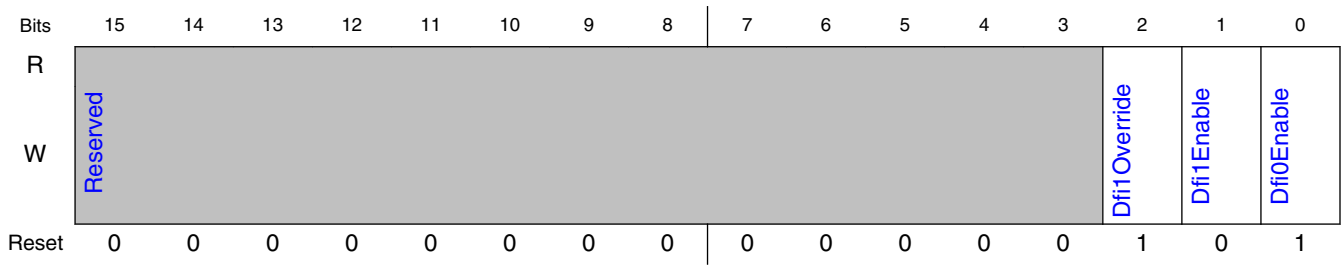
Field	Function
15-6 —	Reserved
5 TxCalBinNOvrEn	1 = use the override value present in Register TxCalBinNOvrVal 0 = don't.
4-0 TxCalBinNOvrVal	The binary value which can overide the Register TxCalBinN calibrator results if Register TxCalBinPOvrEn is set.

9.3.3.6.21 Enables for update and low-power interfaces for DFI0 and DFI1 (DfiMode)

9.3.3.6.21.1 Offset

Register	Offset
DfiMode	30h

9.3.3.6.21.2 Diagram



9.3.3.6.21.3 Fields

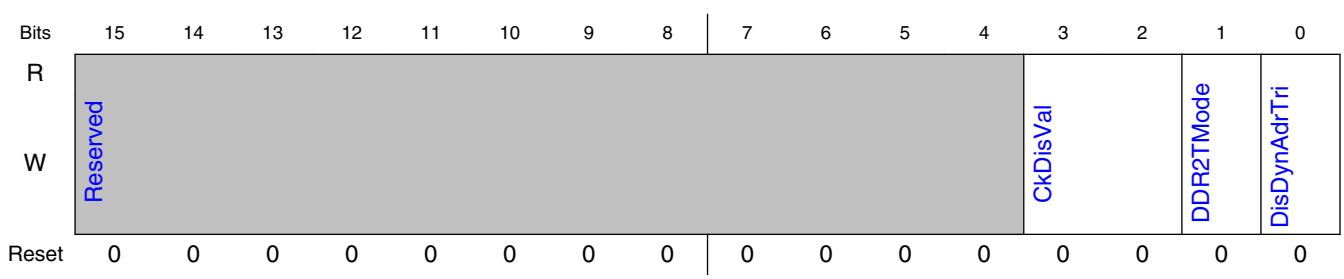
Field	Function
15-3 —	Reserved
2 Dfi1Override	DFI0 is used to control the PHY logic associated with both DFI0 and DFI1
1 Dfi1Enable	Enables operation for the PHY logic associated with DFI1
0 Dfi0Enable	Enables operation for the PHY logic associated with DFI0

9.3.3.6.22 Mode select register for MEMCLK/Address/Command Tristates (TristateModeCA_p0)

9.3.3.6.22.1 Offset

Register	Offset
TristateModeCA_p0	32h

9.3.3.6.22.2 Diagram



9.3.3.6.22.3 Fields

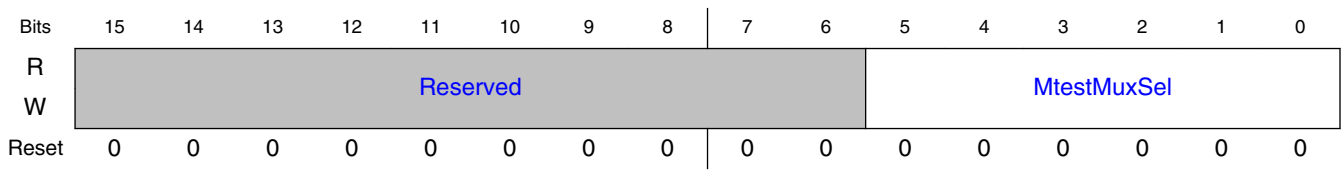
Field	Function
15-4 —	Reserved
3-2 CkDisVal	The PHY provides 4 memory clocks, n=0. The PHY provides 4 memory clocks, n=0..3. When the toggling of memory clock CK_t[n] is disabled with dfi_clk_disable[n]=1, the memory clock CK_t[n] is driven with Register CkDisVal[1]. When the toggling of memory clock CK_c[n] is disabled with dfi_clk_disable[n]=1, the memory clock CK_c[n] is driven with ~(Register CkDisVal[0]) (ie inverted). Note that the non-toggling differential memory clock CK_t[n],CK_c[n] may be driven with any combination, LL,LH,HL,HH; the default CK_t[n],CK_c[n]=LH.
1 DDR2TMode	Must be set to 1 for Dynamic Tristate to work when CA bus is 2T or Geardown mode. Must be set to 1 for Dynamic Tristate to work when CA bus is 2T or Geardown mode. This CSR bit has no effect when DisDynAdrTri==1
0 DisDynAdrTri	When DisDynAdrTri=1, Dynamic Tristating is disabled. When DisDynAdrTri=1, Dynamic Tristating is disabled. Dynamic Tristating is on by default. . Dynamic Tristating should be disabled (DisDynAdrTri=0) for these modes: 1. DDR3/2T if the controller cannot follow the 2T PHY tristate protocol. 2. DDR4/2T/2N if the controller cannot follow the 2T PHY tristate protocol. 3. LPDDR4 . When DisDynAdrTri=0 (default), In DDR3 mode, The following SDRAM pins can be dynamically tristated: A,BA,RAS_n,CAS_n,WE_n In DDR4 mode, The following SDRAM pins can be dynamically tristated: A,BA,BG,ACT_n In LPDDR3 mode, The following SDRAM pins can be dynamically tristated: CA[*] . In 1T mode, the PHY will tristate the relevant pins when all ranks are DESelected . In 2T mode (or geardown), the PHY will tristate the relevant pins when: D3 -- All ranks DESelected (all CS_L==1) AND RAS_n,CAS_n,WE_n,BA0 = 1,1,1,0 D4 -- All ranks DESelected (all CS_L==1) AND RAS_n,CAS_n,WE_n,BA0,ACT_n = 1,1,1,0,1 When in this mode, the controller should avoid sending the above patterns with any rank selected. [e.g. NOPs sent by the controller should have BA0 set to a non-zero value]

9.3.3.6.23 Digital Observation Pin control (MtestMuxSel)

9.3.3.6.23.1 Offset

Register	Offset
MtestMuxSel	34h

9.3.3.6.23.2 Diagram



9.3.3.6.23.3 Fields

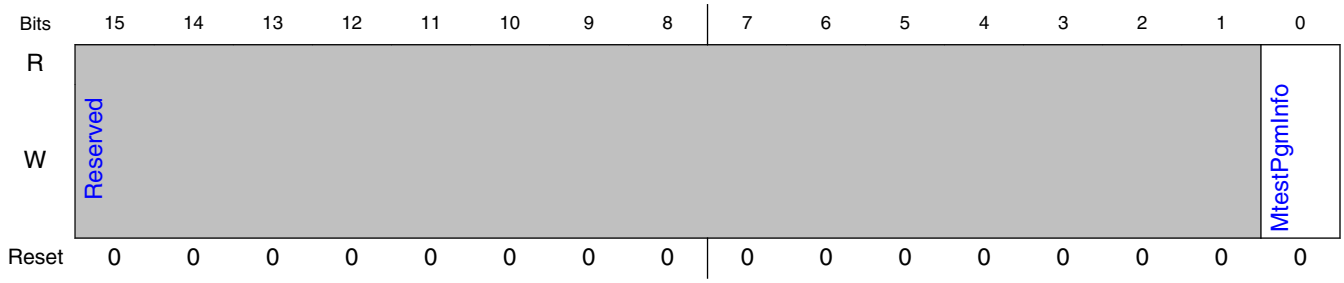
Field	Function
15-6 —	Reserved
5-0 MtestMuxSel	Controls for the 64-1 mux for asynchronous data to the Digital Observation Pin. Controls for the 64-1 mux for asynchronous data to the Digital Observation Pin. Encoding 6'h0 --> Drive 0 from this chiplet (allows flat 'OR' of pass-through information) Encoding 6'h01:1f --> Select local data from AC/DBYTE/MASTER macro Encoding 6'h20 --> Reserved (Drive 0 from macro into mux; Drive 0 from PUB synth logic path into mux) Encoding 6'h21:3f --> Select local data from PUB AC/DBYTE/MASTER synthesized logic Note: See PUB documentation for how, or if, the Digital Observation Pin is mapped to a physical bump in this configuration.

9.3.3.6.24 Digital Observation Pin program info for debug (MtestPgmInfo)

9.3.3.6.24.1 Offset

Register	Offset
MtestPgmInfo	36h

9.3.3.6.24.2 Diagram



9.3.3.6.24.3 Fields

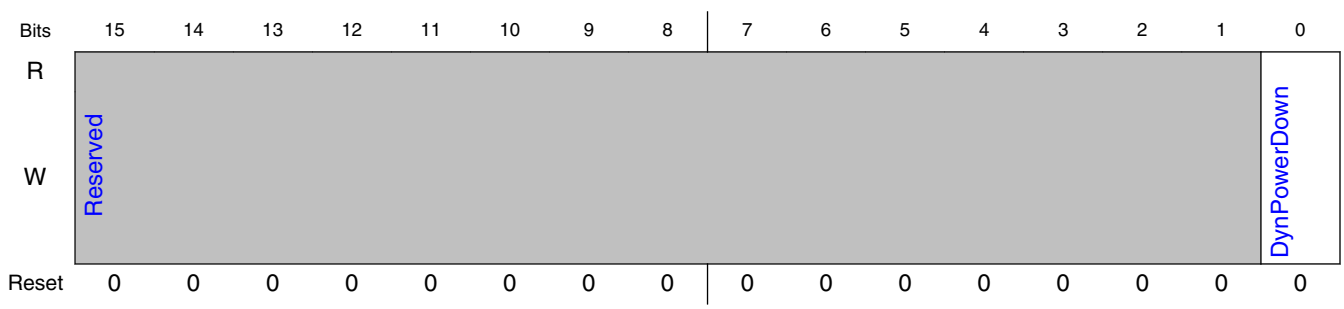
Field	Function
15-1 —	Reserved
0 MtestPgmInfo	The value of this csr may be driven onto the Digital Observation Pin. The value of this csr may be driven onto the Digital Observation Pin. It has no other hardware effect.

9.3.3.6.25 Dynaimc Power Up/Down control (DynPwrDnUp)

9.3.3.6.25.1 Offset

Register	Offset
DynPwrDnUp	38h

9.3.3.6.25.2 Diagram



9.3.3.6.25.3 Fields

Field	Function
15-1 —	Reserved
0 DynPowerDown	1 - analog circuitry (voltage dacs, bias gen) is turned off. 1 - analog circuitry (voltage dacs, bias gen) is turned off. 0 - normal operation.

9.3.3.6.26 PHY Technology ID Register (PhyTID)

9.3.3.6.26.1 Offset

Register	Offset
PhyTID	3Ch

9.3.3.6.26.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PhyTID															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.26.3 Fields

Field	Function
15-0 PhyTID	This register is a placeholder to store technology-specific information

9.3.3.6.27 HWT MaxReadLatency. (HwtMRL_p0)

9.3.3.6.27.1 Offset

Register	Offset
HwtMRL_p0	40h

9.3.3.6.27.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											HwtMRL_p0				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

9.3.3.6.27.3 Fields

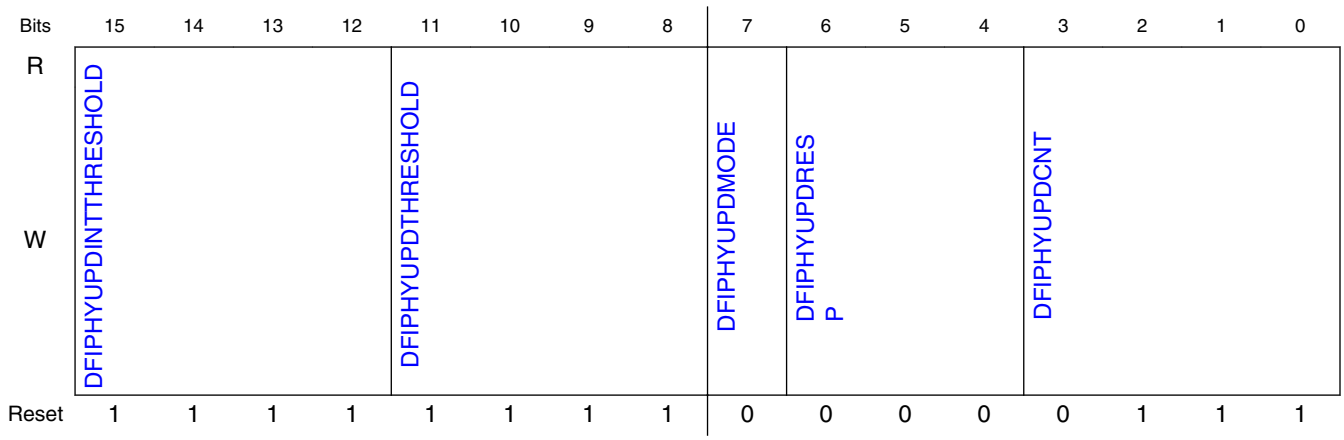
Field	Function
15-5 —	Reserved
4-0 HwtMRL_p0	This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. The CSR is in units of two mem clocks; that is, a unit change in the LSB is a change in MRL of two mem clocks. This MASTER copy of MRL is used by the PHY training hardware only.

9.3.3.6.28 DFI PhyUpdate Request time counter (in MEMCLKs) (DFIPHYUPD)

9.3.3.6.28.1 Offset

Register	Offset
DFIPHYUPD	42h

9.3.3.6.28.2 Diagram



9.3.3.6.28.3 Fields

Field	Function
15-12 DFIPHYUPDINTTHRESHOLD	This subfield is similar to DFIPHYUPDTHRESHOLD except that rather than affecting the Phy Update request, it affects only the threshold used to generate the VT Drift Alarm Interrupt. This subfield is similar to DFIPHYUPDTHRESHOLD except that rather than affecting the Phy Update request, it affects only the threshold used to generate the VT Drift Alarm Interrupt. Further, the actual threshold is twice what is specified in the field.
11-8 DFIPHYUPDTHRESHOLD	4'h0 Disable Threshold-based Phy Update Requests when DFIPHYUPDMODE==1'b1 Nonzero codes are the threshold value for the change in the master LCDL 1UI phase code since the last Phy Update Request that will trigger a new Phy Update Request; If (current_1UI_phase - last_1UI_phase) > DFIPHYUPDTHRESHOLD, then a Phy Update will be requested. 4'h0 Disable Threshold-based Phy Update Requests when DFIPHYUPDMODE==1'b1 Nonzero codes are the threshold value for the change in the master LCDL 1UI phase code since the last Phy Update Request that will trigger a new Phy Update Request; If (current_1UI_phase - last_1UI_phase) > DFIPHYUPDTHRESHOLD, then a Phy Update will be requested. In units of native-control phase units, Minimum value is 1. With DFIPHYUPDTHRESHOLD=1, a master LCDL dithering between two values will not cause a Phy Update Request.
7 DFIPHYUPDMODE	1'b0 [Default] deterministic timer-based Phy Update Requests; enables multi-channel/multi-phy lockstep operation. 1'b0 [Default] deterministic timer-based Phy Update Requests; enables multi-channel/multi-phy lockstep operation. 1'b1 Non-deterministic threshold-based Phy Update Requests; multi-channel/multi-phy operation not permitted.
6-4 DFIPHYUPDRESP	Enforces the t_phyupd_resp time, the maximum time that is allowed to controller to respond to the request for a PHY update. Enforces the t_phyupd_resp time, the maximum time that is allowed to controller

Table continues on the next page...

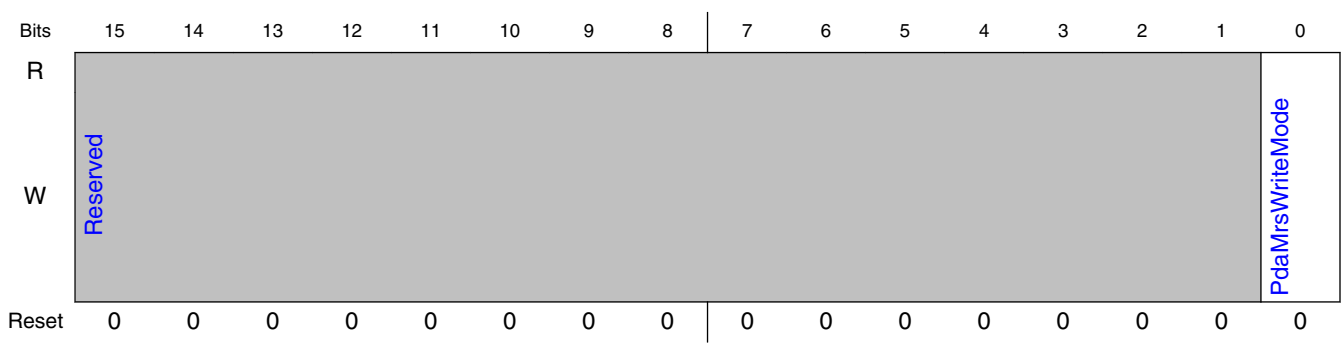
Field	Function
	to respond to the request for a PHY update. A dfi_error will be signaled if there is no acknowledgement of the update request within nMEMCLKs_phyupd_resp. 3'b000 nMEMCLKs_phyupd_resp= 1024 MEMCLKs, default value 3'b001 nMEMCLKs_phyupd_resp= 2048 MEMCLKs 3'b010 nMEMCLKs_phyupd_resp= 4096 MEMCLKs 3'b011 nMEMCLKs_phyupd_resp= 8192 MEMCLKs, [not allowed in LPDDR3, LPDDR4 modes] Any code not enumerated above is RSVD and should not be set.
3-0 DFIPHYUPDCNT	This controls the interval between the end of a phyupdate transaction and a subsequent request. This controls the interval between the end of a phyupdate transaction and a subsequent request. 4'b0000 - Disable timer-based Phy Update when DFIPHYUPDMODE==1'b0 4'b0001 - 16K MEMCLKs minus nMEMCLKs_phyupd_resp 4'b0011 - 32K MEMCLKs minus nMEMCLKs_phyupd_resp 4'b0111 - 64K MEMCLKs minus nMEMCLKs_phyupd_resp 4'b1111 - 128K MEMCLKs minus nMEMCLKs_phyupd_resp Any code not enumerated above is RSVD and should not be set.

9.3.3.6.29 Controls the write DQ generation for Per-Dram-Addressing of MRS (PdaMrsWriteMode)

9.3.3.6.29.1 Offset

Register	Offset
PdaMrsWriteMode	44h

9.3.3.6.29.2 Diagram



9.3.3.6.29.3 Fields

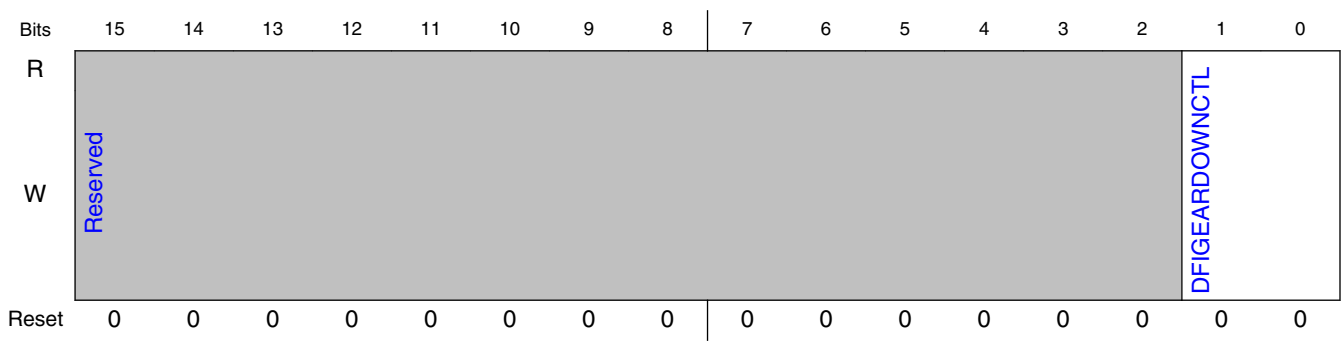
Field	Function
15-1 —	Reserved
0 PdaMrsWriteMode	Controls the write DQ generation per the timing requirements on the DQ signals used for Per-Dram-Addressing mode of MRS commands. Controls the write DQ generation per the timing requirements on the DQ signals used for Per-Dram-Addressing mode of MRS commands. When asserted, the DQ are driven for an additional UI before the logical burst and for an additonal UI after the logical burst. The early UI has the value of first logical UI, and the late UI has the value of the last logical UI. The timing of the logical burst is unchanged; no modification to csr TxDqDly is required or performed. Strongly recommended to be cleared after completing the MRS that clears MR3 A4 and for mission-mode writes, else required bubbles will be larger and the power consumed will be greater. See JEDEC MR3 A4, Per DRAM Addresssability.

9.3.3.6.30 Controls whether dfi_geardown_en will cause CS and CKE timing to change. (DFIGEARDOWNCTL)

9.3.3.6.30.1 Offset

Register	Offset
DFIGEARDOWNCTL	46h

9.3.3.6.30.2 Diagram



9.3.3.6.30.3 Fields

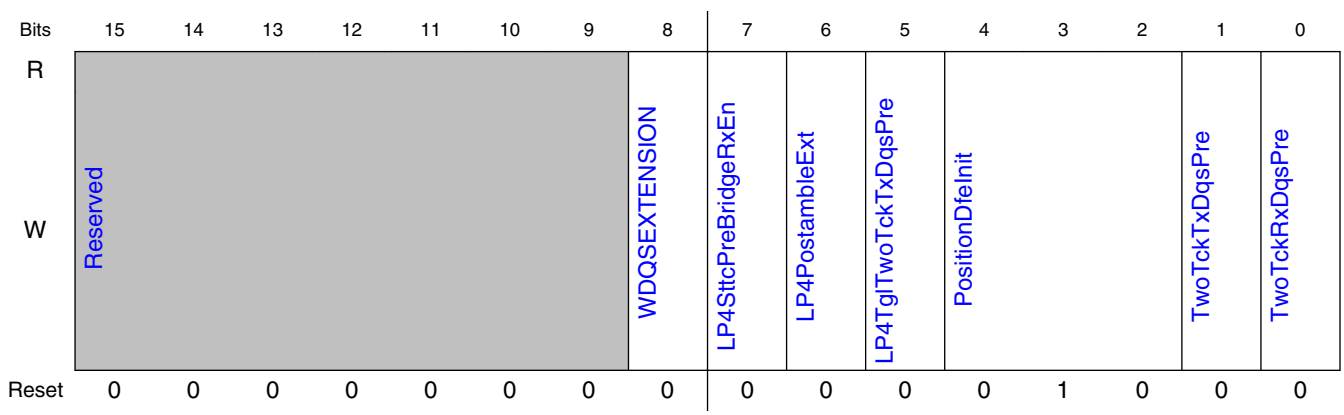
Field	Function
15-2 —	Reserved
1-0 DFIGEARDOWNCTL	DFIGEARDOWNCTL[0] controls whether dfi_geardown_en will cause chip-select (CS) timing to change. DFIGEARDOWNCTL[0] controls whether dfi_geardown_en will cause chip-select (CS) timing to change. 0 - CS timing will not be dynamically modified wtih dfi_geardown_en 1 - CS will be delayed by an additional 1UI dynamically when dfi_geardown_en==1 Note, this CSR must be set to zero if the PHY is not in DDR4 Mode DFIGEARDOWNCTL[1] controls whether dfi_geardown_en will cause clock-enable (CKE) timing to change. 0 - CKE timing will not be dynamically modified with dfi_geardown_en 1 - CKE will be delayed by an additional 1UI dynamically when dfi_geardown_en==1 Note, this CSR must be set to zero if the PHY is not in DDR4 Mode

9.3.3.6.31 Control the PHY logic related to the read and write DQS preamble (DqsPreambleControl_p0)

9.3.3.6.31.1 Offset

Register	Offset
DqsPreambleControl_p0	48h

9.3.3.6.31.2 Diagram



9.3.3.6.31.3 Fields

Field	Function
15-9 —	Reserved
8 WDQSEXTENSION	When set, DQS_T and DQS_C will be driven differentially to 0 and 1, respectively, before and after a write burst, except during a memory read transaction. When set, DQS_T and DQS_C will be driven differentially to 0 and 1, respectively, before and after a write burst, except during a memory read transaction. See DesignWare Cores LPDDR4 MultiPHY: WDQS Extension Application Note. Use of WDQSEXTENSION requires POdtTailWidth=3 and POdtStartDelay=00
7 LP4SttcPreBridgeRxEn	Used in LPDDR4 static-preamble mode to bridge the RxEn between two reads to the same timing group when the bubble is 1 memclk. Used in LPDDR4 static-preamble mode to bridge the RxEn between two reads to the same timing group when the bubble is 1 memclk. LPDDR4 static-preamble mode is set in the DRAMs with MR1, OP[3]=0.
6 LP4PostambleExt	In LPDDR4 mode must be set to extend the write postamble. In LPDDR4 mode must be set to extend the write postamble. 0: half-memclk postamble, as in D4 1: one-and-one-half-memclk postamble; see LPDDR4 Spec MR3, OP[1] WR PST, vendor-specific function.
5 LP4TglTwoTckTxDqsPre	Used in LPDDR4 mode to modify the early preamble when Register TwoTckTxDqsPre=1 0: level first-memclk preamble 1: toggling first-memclk preamble
4-2 PositionDfeInit	For DDR4 phy only when receive DFE is enabled. For DDR4 phy only when receive DFE is enabled. PositionDfeInit[2]=1 causes DfeInit to be asserted late PositionDfeInit[1]=1 causes DfeInit to be asserted at the nominal time PositionDfeInit[0]=1 causes DfeInit to be asserted early PositionDfeInit=3'b010 is the nominal, recommended value for leading edge used by receiver.
1 TwoTckTxDqsPre	0: Standard 1tck TxDqs Preamble 1: Enable Optional D4 2tck TxDqs Preamble The DDR4 MR4 A12 is Write Preamble, 1=2nCK, 0=1nCK.
0 TwoTckRxDqsPre	Widens the RxDqsEn window to allow larger drift in the incoming read DQS to take advantage of the larger/wider preamble generated by the DRAMSs when the D4 DRAMS are configured with DDR4 MR4 A11 Read Preamble=1 for causing a 2nCK read preamble. Widens the RxDqsEn window to allow larger drift in the incoming read DQS to take advantage of the larger/wider preamble generated by the DRAMSs when the D4 DRAMS are configured with DDR4 MR4 A11 Read Preamble=1 for causing a 2nCK read preamble. For LPDDR4, all read operations are 2nCK such that this control must be set to 1. Note the nominal trained-to-center point will be earlier relative to the first strobing edge of DQS than when not in this mode.

9.3.3.6.32 DBYTE module controls to select X4 Dram device mode (MasterX4Config)

9.3.3.6.32.1 Offset

Register	Offset
MasterX4Config	4Ah

9.3.3.6.32.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												X4TG			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.32.3 Fields

Field	Function
15-4 —	Reserved
3-0 X4TG	Set to 1 if this Timing Group/Rank is x4 (as opposed to x8) memory. Set to 1 if this Timing Group/Rank is x4 (as opposed to x8) memory. XXX[1|0] : TxReqDest/RxReqDest 0 X4/X8. XX[1|0]X : TxReqDest/RxReqDest 1 X4/X8. X[1|0]XX : TxReqDest/RxReqDest 2 X4/X8. [1|0]XXX : TxReqDest/RxReqDest 3 X4/X8. This CSR must be written before dfi_init_start is asserted.

9.3.3.6.33 Write level feedback DQ observability select. (WrLevBits)

9.3.3.6.33.1 Offset

Register	Offset
WrLevBits	4Ch

9.3.3.6.33.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								WrLevForDQSU				WrLevForDQSL			
Reset	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0

9.3.3.6.33.3 Fields

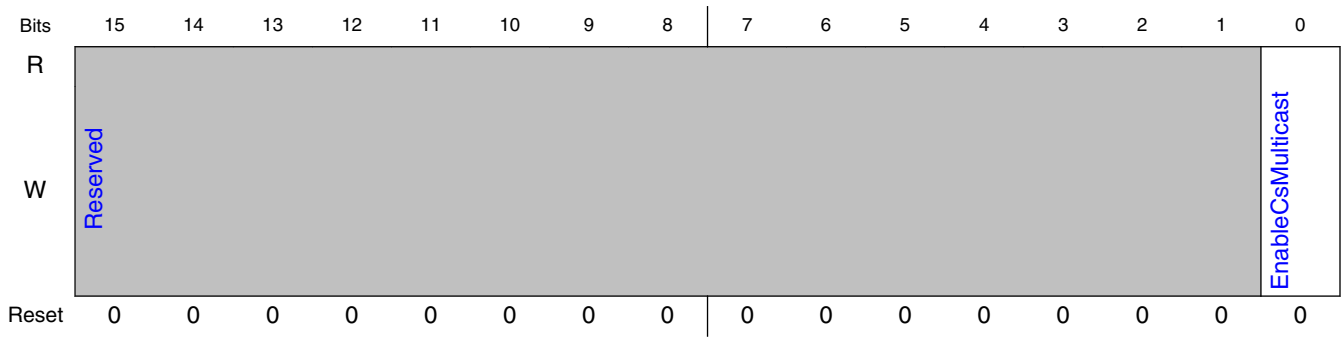
Field	Function
15-8 —	Reserved
7-4 WrLevForDQSU	Indicates which DQ bit is used for Write Levelization. Indicates which DQ bit is used for Write Levelization. 0000 - use DQ0 ... 0111 - use DQ7 1000 - use lDQ[3:0] 1001 - use lDQ[7:4] 1010 - use lDQ[7:0]
3-0 WrLevForDQSL	Indicates which DQ bit is used for Write Levelization. Indicates which DQ bit is used for Write Levelization. 0000 - use DQ0 ... 0111 - use DQ7 1000 - use lDQ[3:0] 1001 - use lDQ[7:4] 1010 - use lDQ[7:0]

9.3.3.6.34 In DDR4 Mode , this controls whether CS_N[3:2] should be multicast on CID[1:0] (EnableCsMulticast)

9.3.3.6.34.1 Offset

Register	Offset
EnableCsMulticast	4Eh

9.3.3.6.34.2 Diagram



9.3.3.6.34.3 Fields

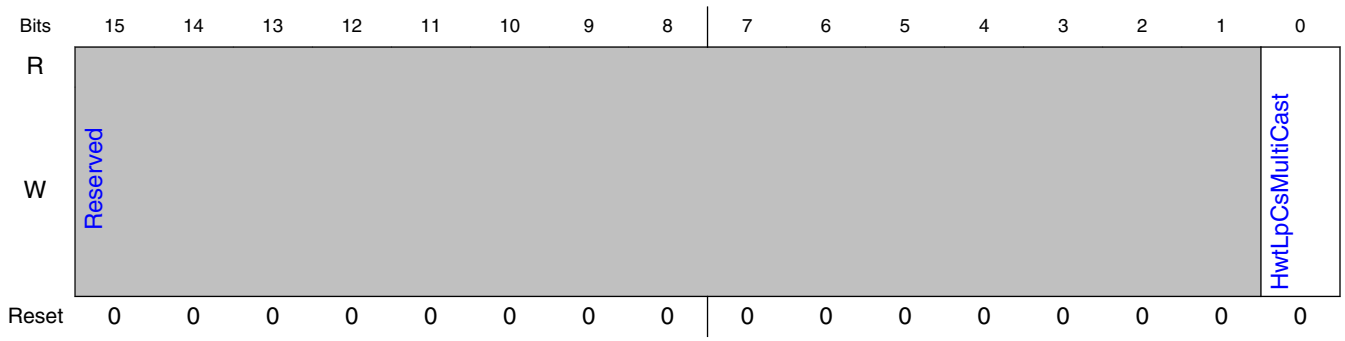
Field	Function
15-1 —	Reserved
0 EnableCsMulticast	In DDR4 Mode , this controls whether CS_N[3:2] should be multicast on CID[1:0] 0 - Do not override pins corresponding to cid[1:0] (dfi_cid[1:0] will connect to the pads) 1 - Overrirde pins corresponding to cid[1:0] with dfi_cs[3:2]. In DDR4 Mode , this controls whether CS_N[3:2] should be multicast on CID[1:0] 0 - Do not override pins corresponding to cid[1:0] (dfi_cid[1:0] will connect to the pads) 1 - Overrirde pins corresponding to cid[1:0] with dfi_cs[3:2]. (cs[3:2] will be multicast on cs[3:2] and cid[1:0])

9.3.3.6.35 Drives cs_n[0] onto cs_n[1] during training (HwtLpCsMultiCast)

9.3.3.6.35.1 Offset

Register	Offset
HwtLpCsMultiCast	50h

9.3.3.6.35.2 Diagram



9.3.3.6.35.3 Fields

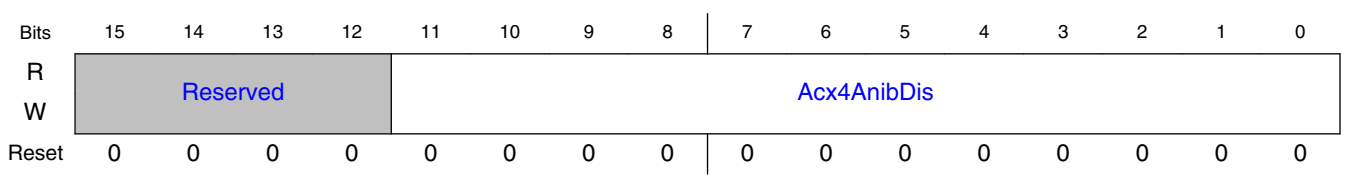
Field	Function
15-1 —	Reserved
0 HwtLpCsMultiCast	When set, drives cs_n[0] onto cs_n[1] during training

9.3.3.6.36 Disable for unused ACX Nibbles (Acx4AnibDis)

9.3.3.6.36.1 Offset

Register	Offset
Acx4AnibDis	58h

9.3.3.6.36.2 Diagram



9.3.3.6.36.3 Fields

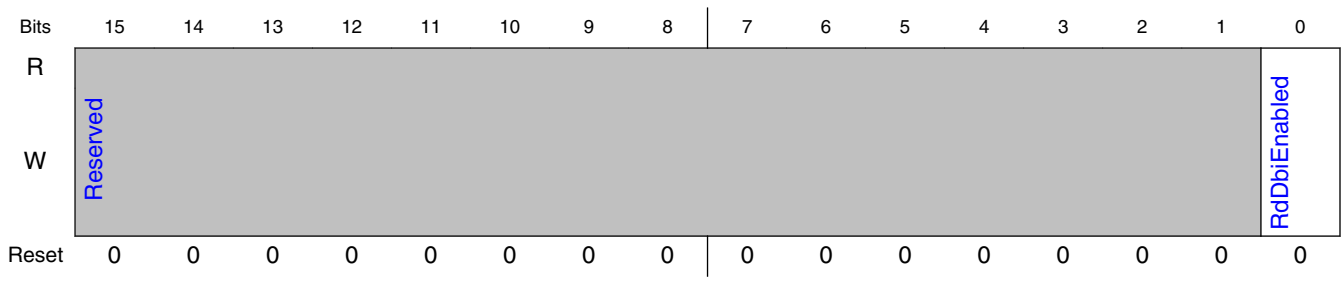
Field	Function
15-12 —	Reserved
11-0 Acx4AnibDis	When a bit is set, the corresponding ACX nibble is disabled (specifically, the I/O OE is disabled, as is the Dfi-side FIFO clock

9.3.3.6.37 This Register is used to enable the Read-DBI function in each DBYTE (DMIPinPresent_p0)

9.3.3.6.37.1 Offset

Register	Offset
DMIPinPresent_p0	5Ah

9.3.3.6.37.2 Diagram



9.3.3.6.37.3 Fields

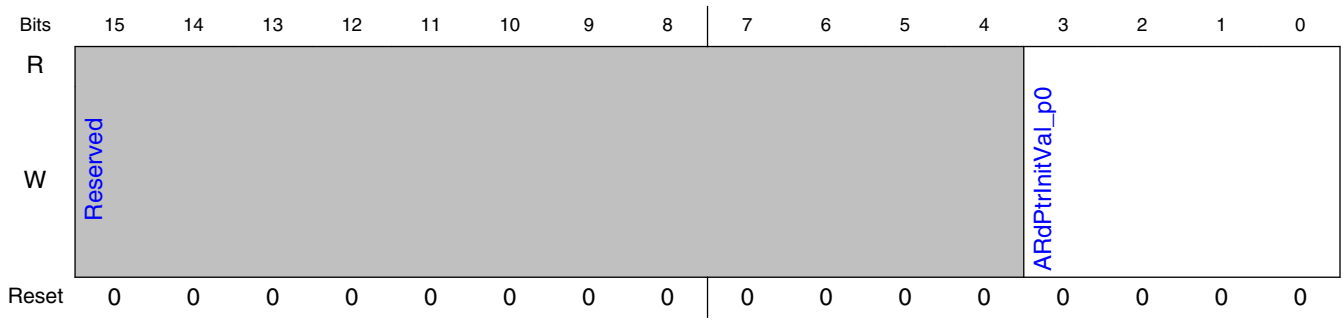
Field	Function
15-1 —	Reserved
0 RdDbiEnabled	This bit must be set to 1'b1 if Read-DBI is enabled in a connected DDR4 or LPDDR4 device. This bit must be set to 1'b1 if Read-DBI is enabled in a connected DDR4 or LPDDR4 device. If set, the following DRAM MR should also be set [DDR4.MR5.A12=1 or LPDDR4.MR3.OP[7]=1]

9.3.3.6.38 Address/Command FIFO ReadPointer Initial Value (ARdPtrInitVal_p0)

9.3.3.6.38.1 Offset

Register	Offset
ARdPtrInitVal_p0	5Ch

9.3.3.6.38.2 Diagram



9.3.3.6.38.3 Fields

Field	Function
15-4 —	Reserved
3-0 ARdPtrInitVal_p0	This is the initial Pointer Offset for the free-running FIFOs in the DBYTE and ACX4 hardips. This is the initial Pointer Offset for the free-running FIFOs in the DBYTE and ACX4 hardips. The units of this register are in UI. The pointer separation should be chosen to compensate for all sources of skew and drift of the PHY DFICLK and PCLK networks. Please see PUB databook section 8.1.1 This CSR must be programmed in Step C of the PHY Initialization sequence

9.3.3.6.39 DLL Mode control CSR for DBYTEs (DbyteDllModeCntrl)

9.3.3.6.39.1 Offset

Register	Offset
DbyteDllModeCntrl	74h

9.3.3.6.39.2 Diagram

9.3.3.6.39.3 Fields

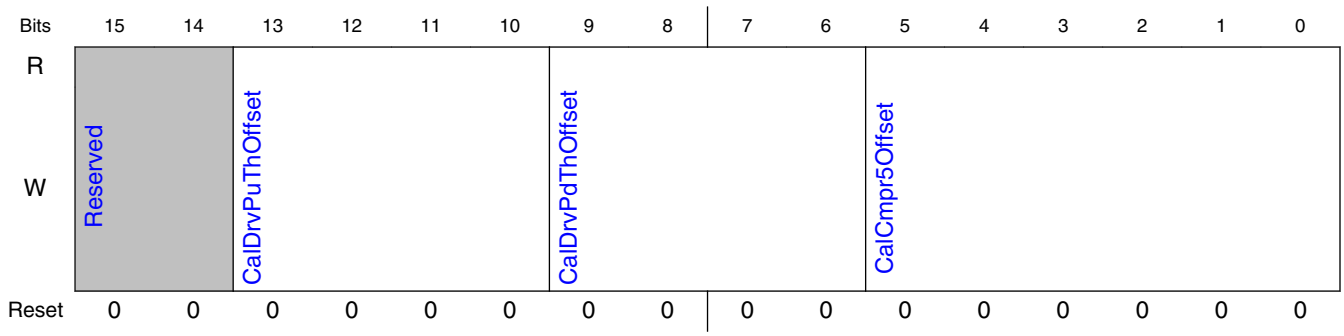
Field	Function
15-2 —	Reserved
1 DllRxPreambleMode	Must be set to 1 if read DQS preamble contains a toggle, for example DDR4 or LPDDR4 read toggling preambe mode
0 —	Reserved

9.3.3.6.40 Impedance Calibration offsets control (CalOffsets)

9.3.3.6.40.1 Offset

Register	Offset
CalOffsets	8Ah

9.3.3.6.40.2 Diagram



9.3.3.6.40.3 Fields

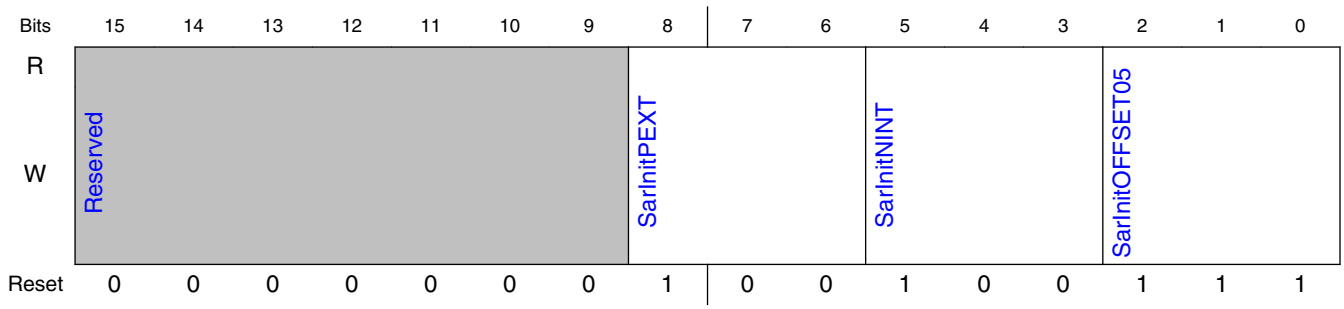
Field	Function
15-14 —	Reserved
13-10 CalDrvPuThOffset	This value adjusts the driver pullup calibration code
9-6 CalDrvPdThOffset	This value adjusts the driver pulldown calibration code
5-0 CalCmpr5Offset	This value adjusts the offset-compensated DAC code for the cmpana circuit at VRef == 0. This value adjusts the offset-compensated DAC code for the cmpana circuit at VRef == 0.5 * VDDQ.

9.3.3.6.41 Sar Init Vals (SarInitVals)

9.3.3.6.41.1 Offset

Register	Offset
SarInitVals	8Eh

9.3.3.6.41.2 Diagram



9.3.3.6.41.3 Fields

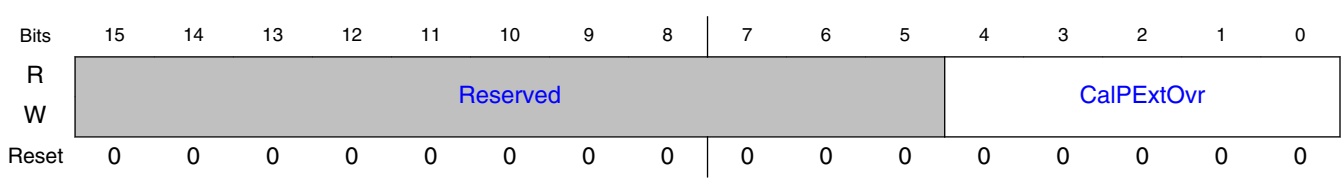
Field	Function
15-9 —	Reserved
8-6 SarInitPEXT	Specify the SAR starting value for PEXT calibration. Specify the SAR starting value for PEXT calibration. This is a 5 bit wide target.
5-3 SarInitNINT	Specify the SAR starting value for NINT calibration. Specify the SAR starting value for NINT calibration. This is a 5 bit wide target.
2-0 SarInitOFFSET05	Specify the SAR starting value for OFFSET05 calibration. Specify the SAR starting value for OFFSET05 calibration. This is an 8 bit wide target.

9.3.3.6.42 Impedance Calibration PExt Override control (CalPExtOvr)

9.3.3.6.42.1 Offset

Register	Offset
CalPExtOvr	92h

9.3.3.6.42.2 Diagram



9.3.3.6.42.3 Fields

Field	Function
15-5 —	Reserved
4-0 CalPExtOvr	If the CSR CalPExtDis is set then the value provided here by software will be used instead of the automatically generated value which is visible via CSR CalPExt This CSR may only be written when the calibrator is not running.

9.3.3.6.43 Impedance Calibration Cmpr 50 control (CalCmpr5Ovr)

9.3.3.6.43.1 Offset

Register	Offset
CalCmpr5Ovr	94h

9.3.3.6.43.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								CalCmpr5Ovr							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.43.3 Fields

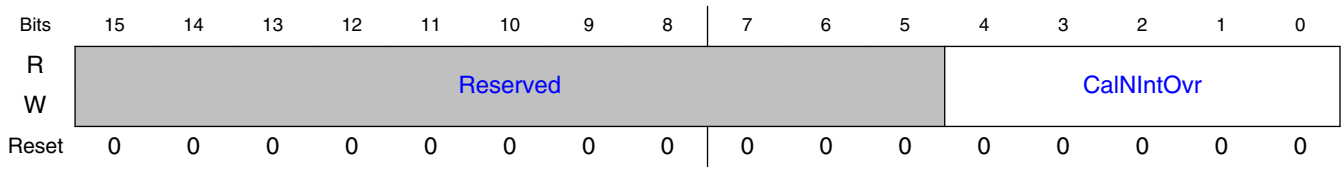
Field	Function
15-8 —	Reserved
7-0 CalCmpr5Ovr	If the CSR CalCmpr5Dis is set then the value provided here by software will be used instead of the automatically generated value which is visible via CSR CalCmpr5 This CSR may only be written when the calibrator is not running.

9.3.3.6.44 Impedance Calibration NInt Override control (CalNIntOvr)

9.3.3.6.44.1 Offset

Register	Offset
CalNIntOvr	96h

9.3.3.6.44.2 Diagram



9.3.3.6.44.3 Fields

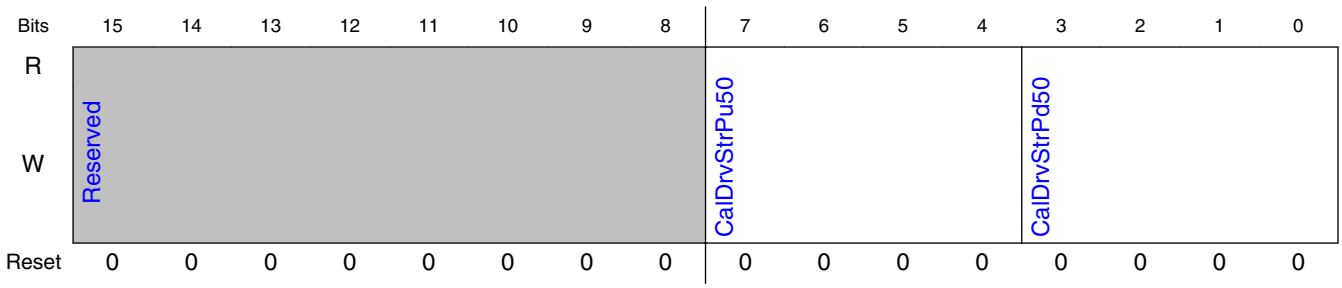
Field	Function
15-5 —	Reserved
4-0 CalNIntOvr	If the CSR CalNIntDis is set then the value provided here by software will be used instead of the automatically generated value which is visible via CSR CalNInt. If the CSR CalNIntDis is set then the value provided here by software will be used instead of the automatically generated value which is visible via CSR CalNInt. This CSR may only be written when the calibrator is not running.

9.3.3.6.45 Impedance Calibration driver strength control (CalDrvStr0)

9.3.3.6.45.1 Offset

Register	Offset
CalDrvStr0	A0h

9.3.3.6.45.2 Diagram



9.3.3.6.45.3 Fields

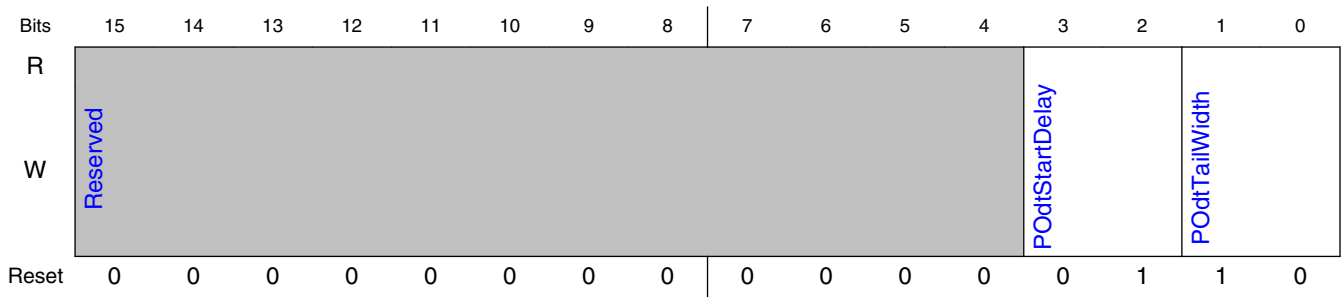
Field	Function
15-8 —	Reserved
7-4 CalDrvStrPu50	3. 3..15 = Reserved 2 - 40.0R 1 - 120.0R 0 - 240.0R (Default)
3-0 CalDrvStrPd50	3. 3..15 = Reserved 2 - 40.0R 1 - 120.0R 0 - 240.0R (Default)

9.3.3.6.46 READ DATA On-Die Termination Timing Control (by PHY) (Proc OdtTimeCtl_p0)

9.3.3.6.46.1 Offset

Register	Offset
ProcOdtTimeCtl_p0	ACh

9.3.3.6.46.2 Diagram



9.3.3.6.46.3 Fields

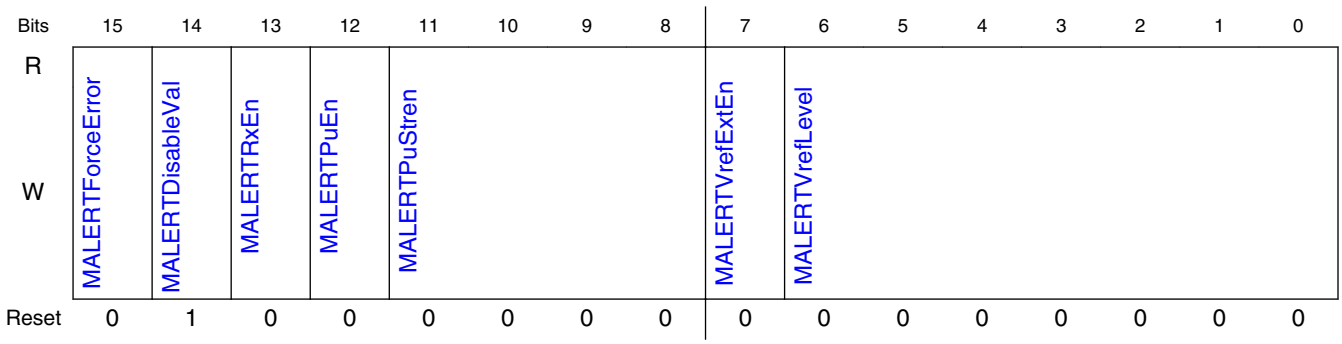
Field	Function
15-4 —	Reserved
3-2 POdtStartDelay	controls the start of ProcOdt, units of UI 3 delay start 2 UI, maximum delay of start of ProcOdt 2 delay start 1 UI, 1 delay start 0 UI, default 0 early by 1 UI, The time from ProcODT assertion to opening the window to receive DQS is (10 - POdtStartDelay) UI.
1-0 POdtTailWidth	controls the length of the tail of ProcOdt, units of UI 3 tail 3UI more than for Register POdtTailWidth=0, maximum 2 tail 2UI more than for Register POdtTailWidth=0, default 1 tail 1UI more than for Register POdtTailWidth=0 0 minimum length tail The time from ProcODT to closing the window to receive DQS to ProcODT POdtTailWidth is (2 + POdtTailWidth) UI

9.3.3.6.47 This Register is used to configure the MemAlert Receiver (MemAlertControl)

9.3.3.6.47.1 Offset

Register	Offset
MemAlertControl	B6h

9.3.3.6.47.2 Diagram



9.3.3.6.47.3 Fields

Field	Function
15 MALERTForceError	When MALERTForceError is set, this CSR state is used to force parity error to memory.
14 MALERTDisableVal	When MALERTRxEn is not set, this CSR state is used to drive dfi_alert_n.
13 MALERTRxEn	1 - Enables receiver and received data is forwared to dfi_alert_n. 1 - Enables receiver and received data is forwared to dfi_alert_n. 0 - Disable the receiver and send MALERTDisableVal to dfi_alert_n
12 MALERTPuEn	When set, enables the Pull-up termination on MALERT
11-8 MALERTPuStren	Controls the Pull-up termination on MALERT ========== bit[8] - controls a 240 Ohm Pull-up leg bit[9] - controls a 240 Ohm Pull-up leg bit[10] - controls a 120 Ohm Pull-up leg bit[11] - controls a 120 Ohm Pull-up leg ========== 0000 - No PullUp Strength 0001 - 240 Ohm PullUp Strength 0010 - 240 Ohm PullUp Strength 0011 - 120 Ohm PullUp Strength 0100 - 120 Ohm PullUp Strength 0101 - 80 Ohm PullUp Strength 0110 - 80 Ohm PullUp Strength 0111 - 60 Ohm PullUp Strength 1000 - 120 Ohm PullUp Strength 1001 - 80 Ohm PullUp Strength 1010 - 80 Ohm PullUp Strength 1011 - 60 Ohm PullUp Strength 1100 - 60 Ohm PullUp Strength 1101 - 48 Ohm PullUp Strength 1110 - 48 Ohm PullUp Strength 1111 - 40 Ohm PullUp Strength
7 MALERTVrefExtEn	When set for test/debug, selects external Vref source, This should not be set in mission mode.
6-0 MALERTVrefLevel	Sets the vref level of internal VREF DAC. Sets the vref level of internal VREF DAC. Assuming the following Mission mode settings: TestExtVrefRange,TestMajorMode = 0011 or 0000) The VREF level chosen by the receiver follows the equation:: VREF = VDDQ*(0.51 + MALERTVrefLevel[6:0]*0.00345)

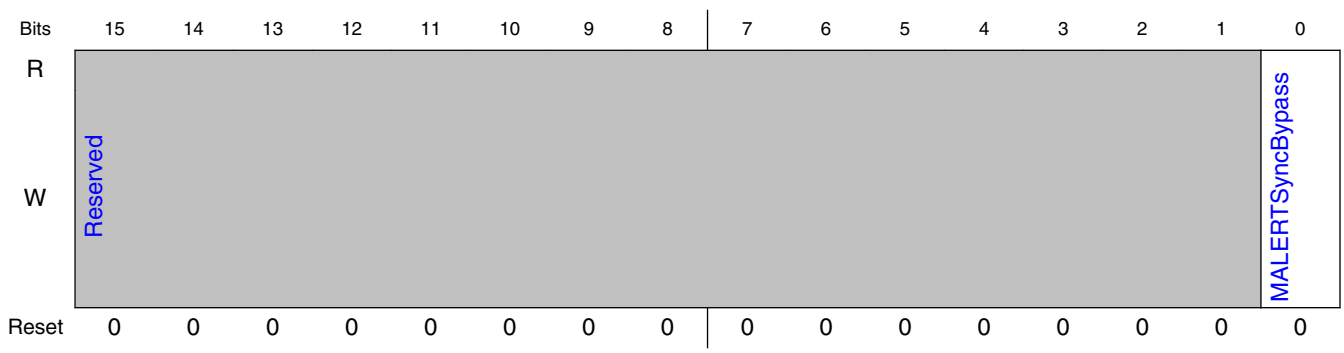
Field	Function
	The min/max values possible are: VREF_min = VDDQ*(0.51) ; VREF_max = VDDQ*(0.948)

9.3.3.6.48 This Register is used to configure the MemAlert Receiver (MemAlertControl2)

9.3.3.6.48.1 Offset

Register	Offset
MemAlertControl2	B8h

9.3.3.6.48.2 Diagram



9.3.3.6.48.3 Fields

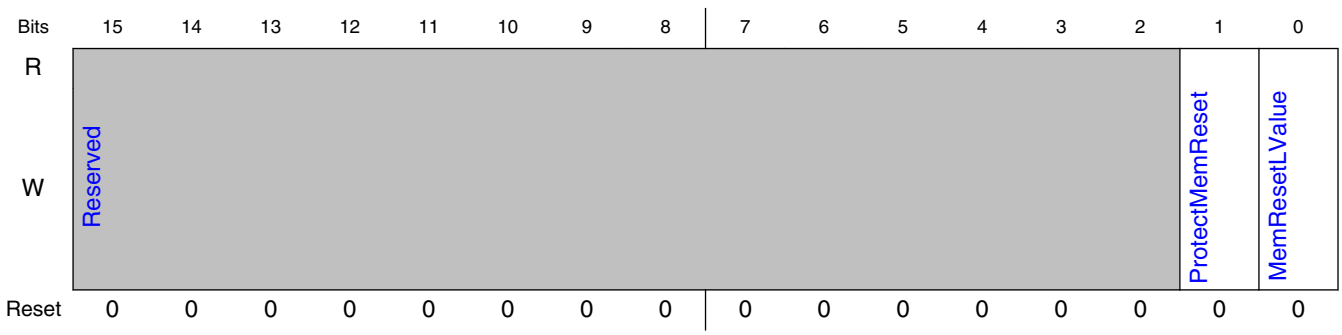
Field	Function
15-1 —	Reserved
0 MALERTSyncBypass	MALERTSyncBypass==[0], the phy will drive dfi_alert_n with a synchronized value of the ALERT_N receiver. MALERTSyncBypass==[0], the phy will drive dfi_alert_n with a synchronized value of the ALERT_N receiver. MALERTSyncBypass==[1], the phy will drive dfi_alert_n with the asynchronous value of the ALERT_N receiver.

9.3.3.6.49 Protection and control of BP_MemReset_L (MemResetL)

9.3.3.6.49.1 Offset

Register	Offset
MemResetL	C0h

9.3.3.6.49.2 Diagram



9.3.3.6.49.3 Fields

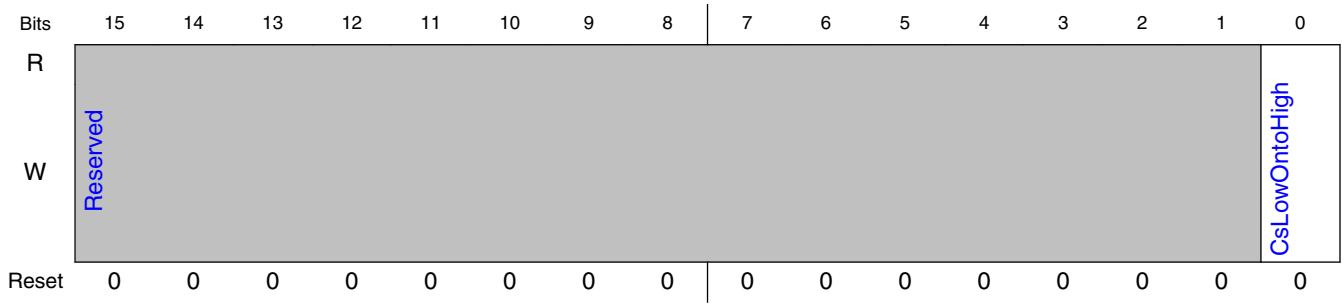
Field	Function
15-2 —	Reserved
1 ProtectMemReset	Control the MemResetL output of the PHY. Control the MemResetL output of the PHY. The state of the MemResetL field is transfered to the PHY BP_MemReset_L output when ProtectMemReset is set.
0 MemResetLValue	Control the MemResetL output of the PHY. Control the MemResetL output of the PHY. The state of this field is transfered to the PHY BP_MemReset_L output when ProtectMemReset is set. BP_MemReset_L cannot reset LPDDR3.

9.3.3.6.50 Drive CS_N 3:0 onto CS_N 7:4 (DriveCSLowOntoHigh)

9.3.3.6.50.1 Offset

Register	Offset
DriveCSLowOntoHigh	DAh

9.3.3.6.50.2 Diagram



9.3.3.6.50.3 Fields

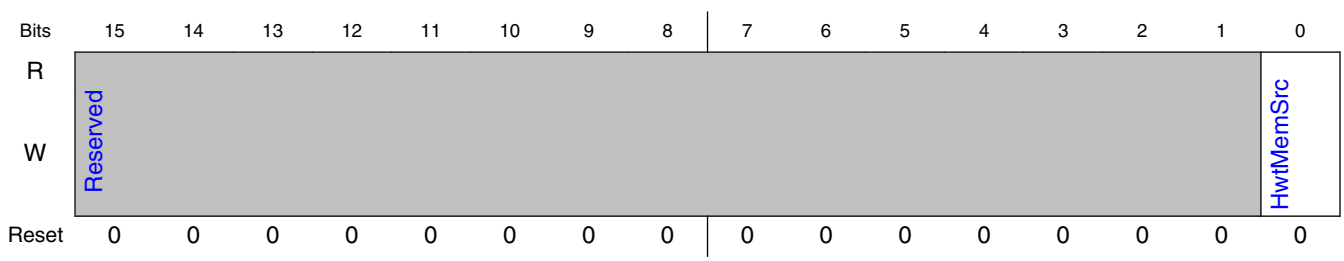
Field	Function
15-1 —	Reserved
0 CsLowOntoHigh	When this is set to a 1, CS[3:0] from the ACSM are driven to CS[7:4] pins and CS[3:0] are deasserted. When this is set to a 1, CS[3:0] from the ACSM are driven to CS[7:4] pins and CS[3:0] are deasserted. When this is set to a 0, CS[3:0] from the ACSM are driven to CS[3:0] pins and CS[7:4] are deasserted. This is used only in DDR4 DirectQuadCS ACX13 mode

9.3.3.6.51 PUBMODE - HWT Mux Select (PUBMODE)

9.3.3.6.51.1 Offset

Register	Offset
PUBMODE	DCh

9.3.3.6.51.2 Diagram



9.3.3.6.51.3 Fields

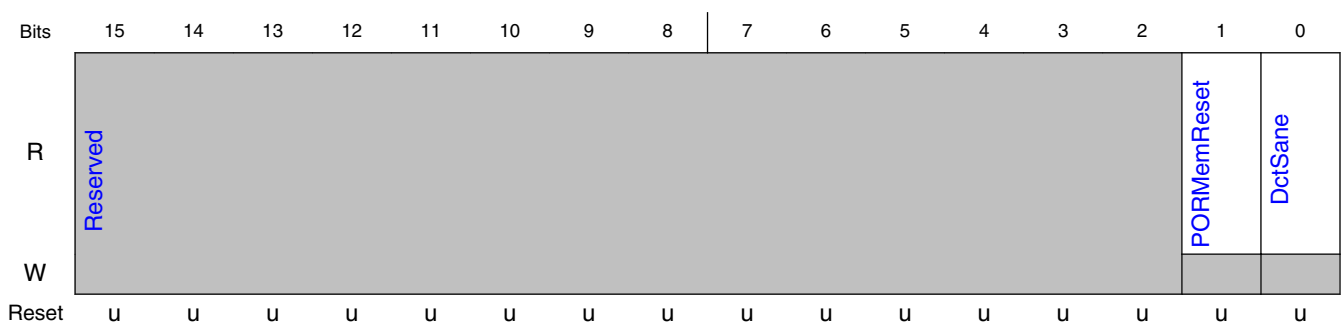
Field	Function
15-1 —	Reserved
0 HwtMemSrc	When this is set to a 1, the mux that switches between DCT and HWT for the source of memory transactions is switched to HWT. When this is set to a 1, the mux that switches between DCT and HWT for the source of memory transactions is switched to HWT. Setting this bit relinquishes control to the HWT. Clearing this bit relinquishes control to the DCT. The transition can only be made when the FIFOs are all in PtrInit (i.e. all PtrInit signals are high).

9.3.3.6.52 Misc PHY status bits (MiscPhyStatus)

9.3.3.6.52.1 Offset

Register	Offset
MiscPhyStatus	DEh

9.3.3.6.52.2 Diagram



9.3.3.6.52.3 Fields

Field	Function
15-2	Reserved

Table continues on the next page...

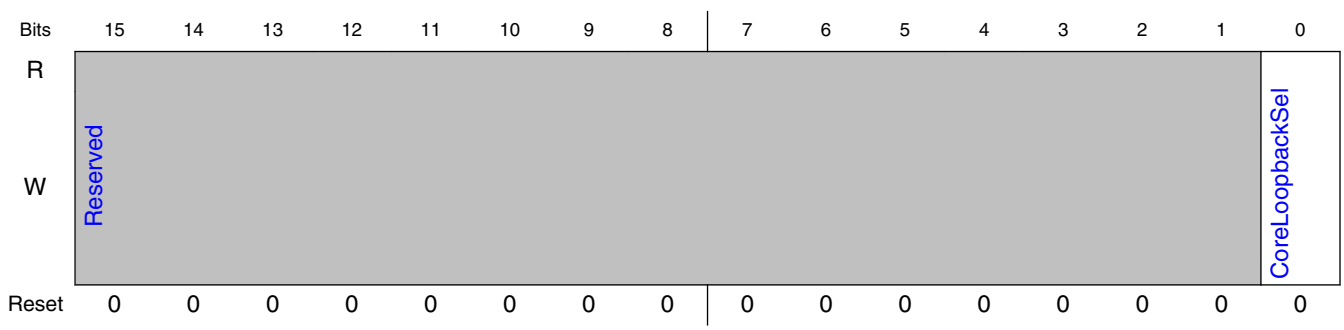
Field	Function
—	
1 PORMemReset	Returns the active-high value used by the custom circuit which drives the memory RESET signal. Returns the active-high value used by the custom circuit which drives the memory RESET signal. To comply with JEDEC specifications for memory reset, this signal powers on 1'b1. This signal is reset during the PHY Initialization sequence.
0 DctSane	Returns the status of the custom circuit which protects the MemResetL output of the PHY on initial power-on or reset. Returns the status of the custom circuit which protects the MemResetL output of the PHY on initial power-on or reset. This circuit resets to 0 meaning that hardware has control of the MemResetL output. If set to 1, memory RESET is being controlled by Registers MemResetLVaue and ProtectMemReset or dfi_reset_n.

9.3.3.6.53 Controls whether the loopback path bypasses the final PAD node. (CoreLoopbackSel)

9.3.3.6.53.1 Offset

Register	Offset
CoreLoopbackSel	E0h

9.3.3.6.53.2 Diagram



9.3.3.6.53.3 Fields

Field	Function
15-1	Reserved

Table continues on the next page...

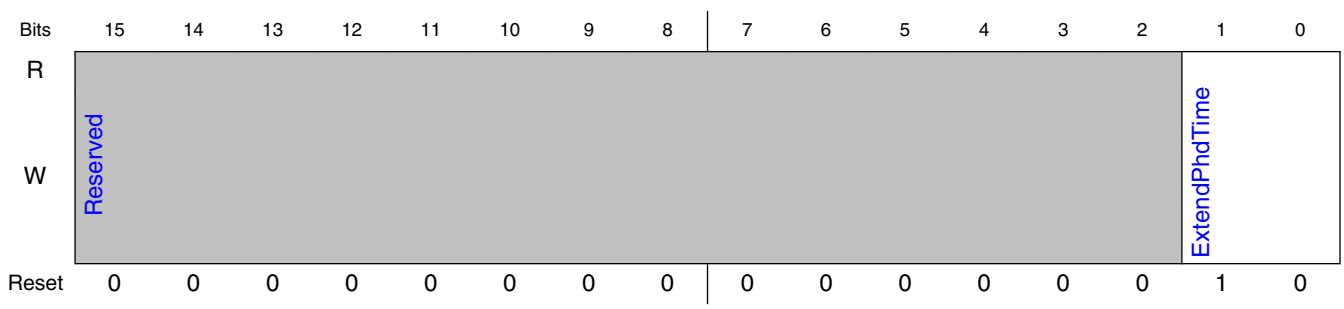
Field	Function
—	
0 CoreLoopbackSel	This register is controlled by the PHY test firmware This register enables Core-Side loopback operation of the PHY. This register is controlled by the PHY test firmware This register enables Core-Side loopback operation of the PHY. When set, the PHY PAD node will be bypassed, and pre-pad node will be selected for sampling by the PHY RX. Setting this register enables full-speed loopback operation of the PHY regardless of the loading of the pad. This register must be set to 1'b0 for mission mode operation (or pad-side loopback).

9.3.3.6.54 DLL Various Training Parameters (DllTrainParam)

9.3.3.6.54.1 Offset

Register	Offset
DllTrainParam	E2h

9.3.3.6.54.2 Diagram



9.3.3.6.54.3 Fields

Field	Function
15-2 —	Reserved
1-0 ExtendPhdTime	Used by the PHY firmware locking the LCDL delay cells. Used by the PHY firmware locking the LCDL delay cells. The number of extra DfiClk periods (= 4*UI) for the DLL phase-detector outputs to become not-metastable

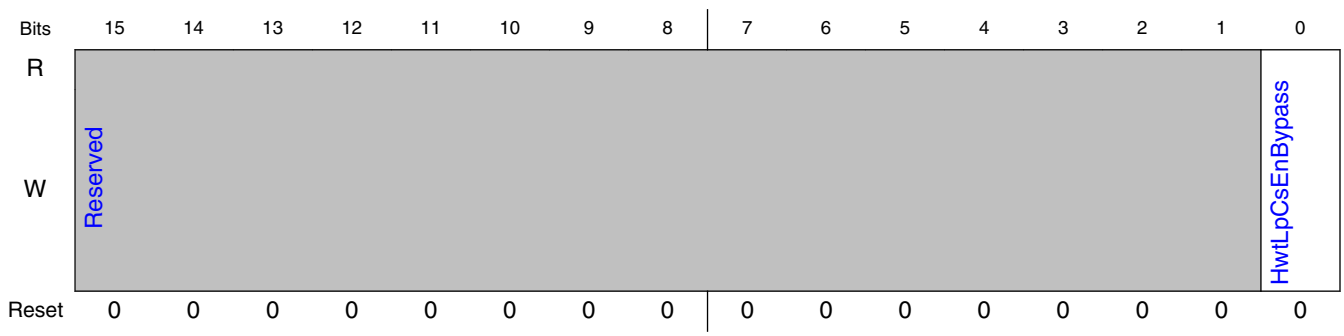
Field	Function
	Recommend to set to 3 if mem CK frequency ge 1800MHz Recommend to set to 2 if mem CK frequency lt 1800MHz and ge 1600MHz Recommend to set to 1 if mem CK frequency lt 1600MHz and ge 1000MHz Recommend to set to 0 if mem CK frequency lt 1000MHz

9.3.3.6.55 CSn Disable Bypass for LPDDR3/4 (HwtLpCsEnBypass)

9.3.3.6.55.1 Offset

Register	Offset
HwtLpCsEnBypass	E8h

9.3.3.6.55.2 Diagram



9.3.3.6.55.3 Fields

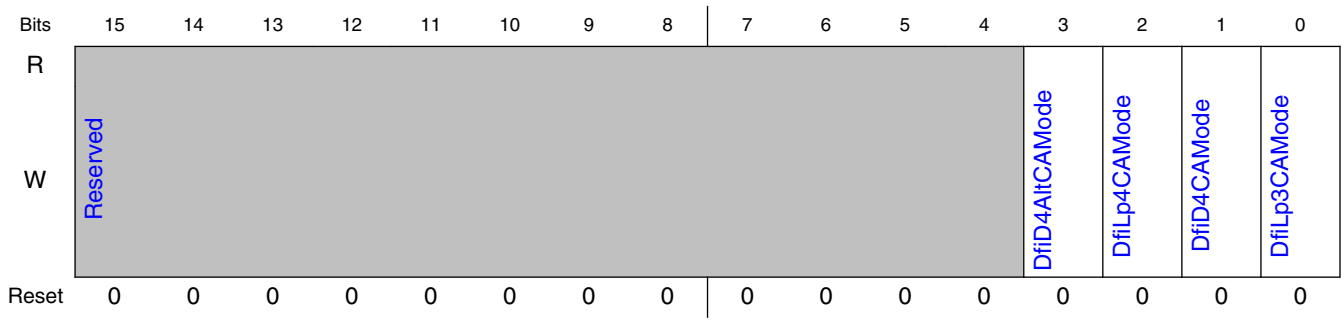
Field	Function
15-1 —	Reserved
0 HwtLpCsEnBypass	When set, these bits disable LpCsEn function for LPDDR3/4

9.3.3.6.56 Dfi Command/Address Mode (DfiCAMode)

9.3.3.6.56.1 Offset

Register	Offset
DfiCAMode	EAh

9.3.3.6.56.2 Diagram



9.3.3.6.56.3 Fields

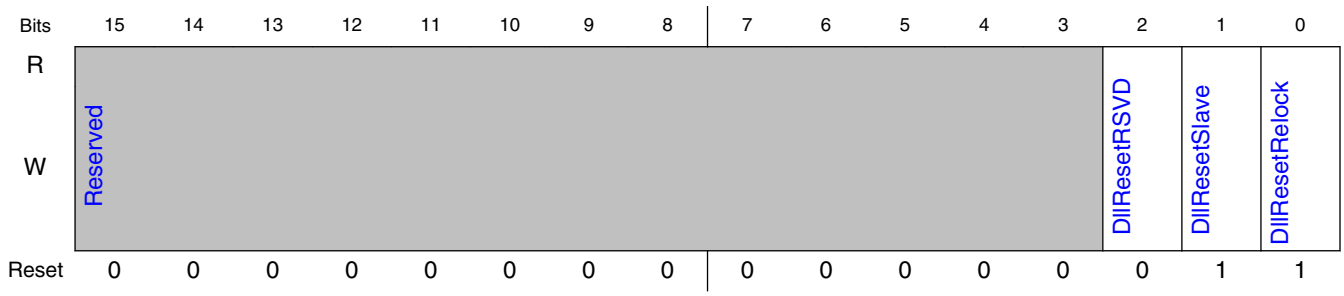
Field	Function
15-4 —	Reserved
3 DfiD4AltCAMode	Enable D4-Alt Mode 0: D4-Altmode disabled 1: D4-Altmode enabled
2 DfiLp4CAMode	Enable LP4 Mode 0: LP4 mode disabled 1: LP4 mode enabled
1 DfiD4CAMode	Enable D4 Mode 0: D4 mode disabled 1: D4 mode enabled
0 DfiLp3CAMode	Controls the output data-rate of the AC module Command/Address pins 0: LP3 DDR address mode disabled 1: LP3 DDR address mode enabled

9.3.3.6.57 DLL Lock State machine control register (DllControl)

9.3.3.6.57.1 Offset

Register	Offset
DllControl	F0h

9.3.3.6.57.2 Diagram



9.3.3.6.57.3 Fields

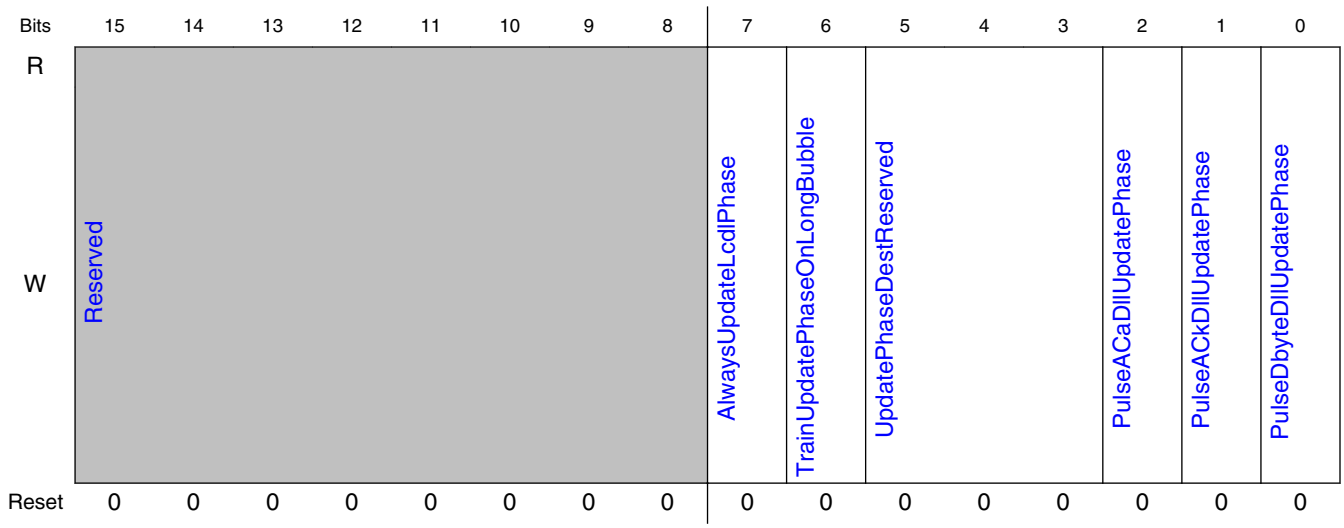
Field	Function
15-3 —	Reserved
2 DllResetRSVD	Reserved
1 DllResetSlave	Reserved
0 DllResetRelock	Used to reset the DDL/LCDL lock state machine Deasserting starts locking sequence. Used to reset the DDL/LCDL lock state machine Deasserting starts locking sequence. This must be done anytime the DLLs are to be relocked (e.g. frequency change).

9.3.3.6.58 DLL update phase control (PulseDllUpdatePhase)

9.3.3.6.58.1 Offset

Register	Offset
PulseDllUpdatePhase	F2h

9.3.3.6.58.2 Diagram



9.3.3.6.58.3 Fields

Field	Function
15-8 —	Reserved
7 AlwaysUpdateLcdlPhase	Causes each new operation to reload the LcdlPhase; will increase bubbles.
6 TrainUpdatePhaseOnLongBubble	Causes LongBubble to update the dbyte & anib LDCL Phase. Causes LongBubble to update the dbyte & anib LDCL Phase. May be used for training, but not to be used for mission mode.
5-3 UpdatePhaseDestReserved	reserved, not used
2 PulseACaDllUpdatePhase	Causes an AC module CA (command/address/cke/odt) DLL phase update.
1 PulseACkDllUpdatePhase	Causes an AC module CK (memck) DLL phase update. Causes an AC module CK (memck) DLL phase update. This should be written with 1 only when the memory CKE=0 or memory RST=0, or when there is no memory, as in tester mode.
0 PulseDbyteDllUpdatePhase	Causes a LongBubble to the DBYTE modules, which causes a update of the DBYTE module DLLs (tx,rxen,rxclk). Causes a LongBubble to the DBYTE modules, which causes a update of the DBYTE module DLLs (tx,rxen,rxclk). The value of Register UpdatePhaseDest must be setup prior to this pulse.

9.3.3.6.59 DLL gain control (DllGainCtl_p0)

9.3.3.6.59.1 Offset

Register	Offset
DllGainCtl_p0	F8h

9.3.3.6.59.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				DllSeedSel				DllGainTV				DllGainIV			
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	1

9.3.3.6.59.3 Fields

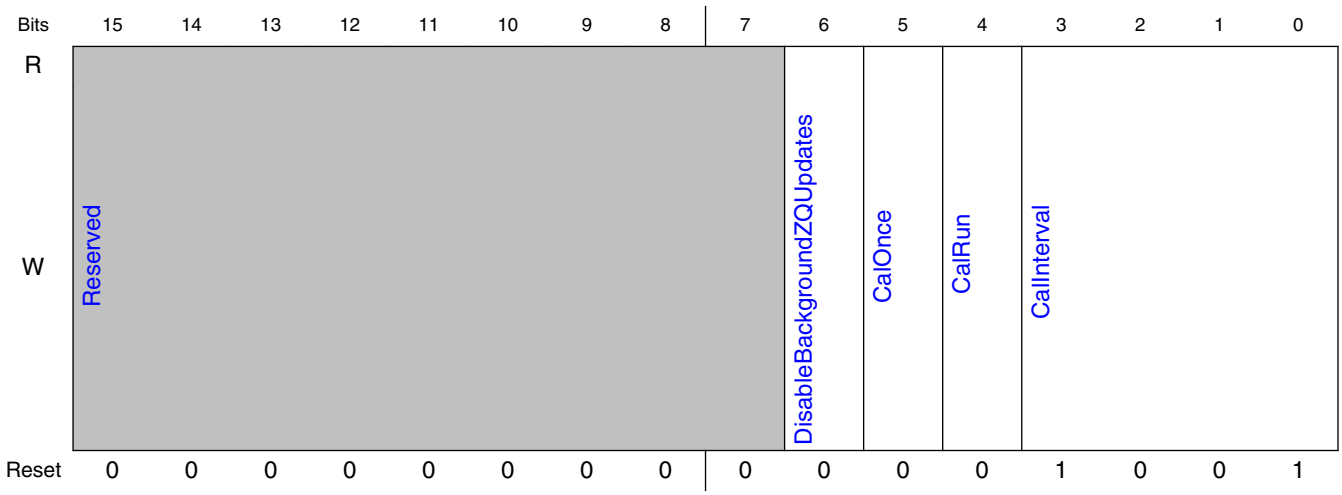
Field	Function
15-12 —	Reserved
11-8 DllSeedSel	Reserved, must be configured to be 0.
7-4 DllGainTV	Terminal value of DllGain, ie the value in effect when locking is done and the value used for maintaining lock, ie tracking pclk variation. Terminal value of DllGain, ie the value in effect when locking is done and the value used for maintaining lock, ie tracking pclk variation. DllGainTV must be greater than or equal to DllGainIV. The maximum value is 10. The minimum value is 6.
3-0 DllGainIV	Initial value of DllGain.

9.3.3.6.60 Impedance Calibration Control (CalRate)

9.3.3.6.60.1 Offset

Register	Offset
CalRate	110h

9.3.3.6.60.2 Diagram



9.3.3.6.60.3 Fields

Field	Function
15-7 —	Reserved
6 DisableBackgroundZQUpdates	1: Instead of having the driver compensation codes go asynchronously out to all IO, hold until for any of PHYUPD ACK, CTRLUPD ACK, PHYMSTR ACK) 0: Calibrated ZQ Updates to IO aren't gated. 1: Instead of having the driver compensation codes go asynchronously out to all IO, hold until for any of PHYUPD ACK, CTRLUPD ACK, PHYMSTR ACK) 0: Calibrated ZQ Updates to IO aren't gated. The default value of this field is 0 because the minor Impedance corrections (Converted from binary to thermometer) is applied over multiple cycles and the impact caused is minor.
5 CalOnce	The setting of this CSR changes the behaviour of CSR CalRun. The setting of this CSR changes the behaviour of CSR CalRun. 1: The 0->1 transition of CSR CalRun causes a single iteration of the calibration sequence to occur. Once the calibration is complete, the calibration engine will remain IDLE until another CalRun Trigger. 0: Calibration will proceed at the rate determined by Register CalInterval.

Table continues on the next page...

Field	Function
	This field should only be changed while the calibrator is idle. ie before csr CalRun is set.
4 CalRun	1: A calibration sequence will be triggered by the 0->1 transition of this bit, as determined by CSR CalOnce. 1: A calibration sequence will be triggered by the 0->1 transition of this bit, as determined by CSR CalOnce. 0: Calibrator will not run. If already active then it will complete the current sequence before quiescing. The default value of this field is 0 because the calibrator needs to know whether it is in D3 mode before it starts up. This csr won't do anything while CalZap is asserted.
3-0 CalInterval	This CSR specifies the interval between successive calibrations, in mS. This CSR specifies the interval between successive calibrations, in mS. csrValue : Interval (mS) 0 : 0 (continuous) 1 : 0.013 2 : 0.10 3 : 1 4 : 2 5 : 3 6 : 4 7 : 8 8 : 10 9 : 20 10-15 : Reserved. This field should only be changed while the calibrator is idle. ie before csr CalRun is set.

9.3.3.6.61 Impedance Calibration Zap/Reset (CalZap)

9.3.3.6.61.1 Offset

Register	Offset
CalZap	112h

9.3.3.6.61.2 Diagram

9.3.3.6.61.3 Fields

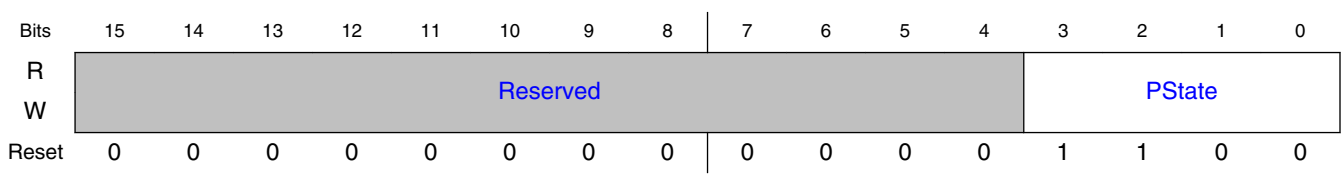
Field	Function
15-1 —	Reserved
0 CalZap	NOTE : This CSR is written by PHY Initialization Engine (PIE) and the data in here will be overwritten. NOTE : This CSR is written by PHY Initialization Engine (PIE) and the data in here will be overwritten. Setting this csr resets the calibrator to its idle state. Must be cleared in order for the calibration engine to run again.

9.3.3.6.62 PSTATE Selection (PState)

9.3.3.6.62.1 Offset

Register	Offset
PState	116h

9.3.3.6.62.2 Diagram



9.3.3.6.62.3 Fields

Field	Function
15-4	Reserved

Table continues on the next page...

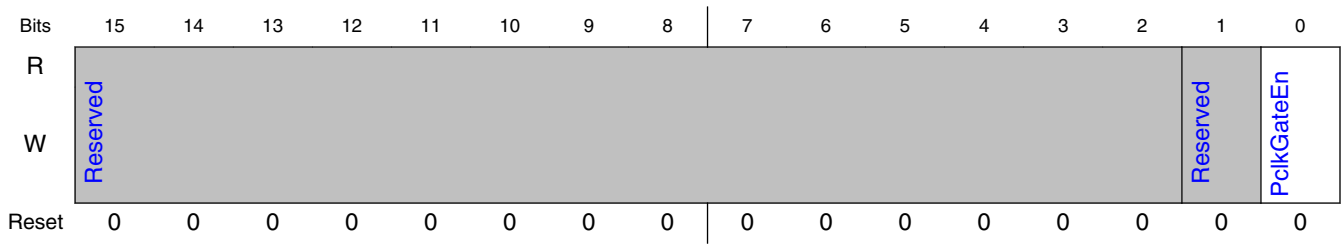
Field	Function
—	
3-0 PState	NOTE : This CSR is written by PHY Initialization Engine (PIE) and the data in here will be overwritten. NOTE : This CSR is written by PHY Initialization Engine (PIE) and the data in here will be overwritten. The current PState inferred by the PIE from handshake requests on DFI status interface. 0000 - PState P0 0001 - PState P1 0010 - PState P2 0011 - PState P3 0100 - Reserved 0101 - Reserved 0110 - Reserved 0111 - Reserved 1000 - Reserved 1001 - Reserved 1010 - Reserved 1011 - Reserved 1100 - LP2 1101 - Reserved 1110 - Reserved 1111 - LP3 The lower two bits of this CSR are used to select the PHY's current Pstate for all Pstateable registers. Note: It is a don't care what value is selected in either LP2 or LP3 because no transactions will be taking place when in those states.

9.3.3.6.63 PLL Output Control (PllOutGateControl)

9.3.3.6.63.1 Offset

Register	Offset
PllOutGateControl	11Ah

9.3.3.6.63.2 Diagram



9.3.3.6.63.3 Fields

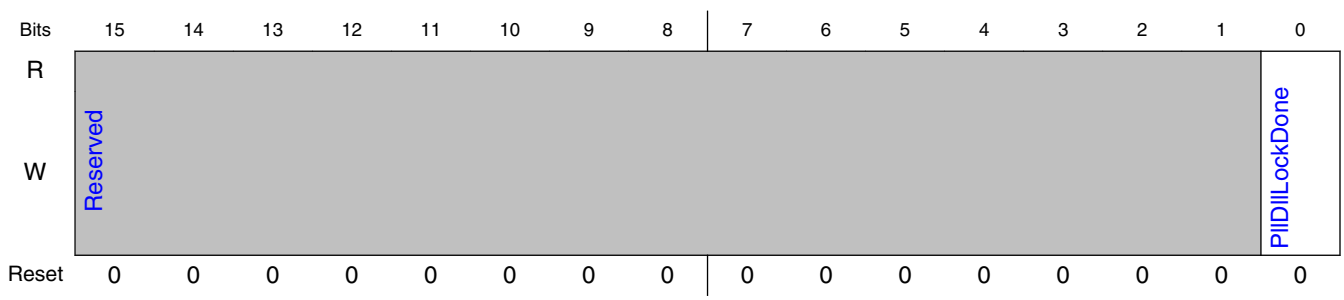
Field	Function
15-2 —	Reserved
1 —	Reserved
0 PclkGateEn	Reserved

9.3.3.6.64 PMU Power-on Reset Control (PLL/DLL Lock Done) (PorControl)

9.3.3.6.64.1 Offset

Register	Offset
PorControl	120h

9.3.3.6.64.2 Diagram



9.3.3.6.64.3 Fields

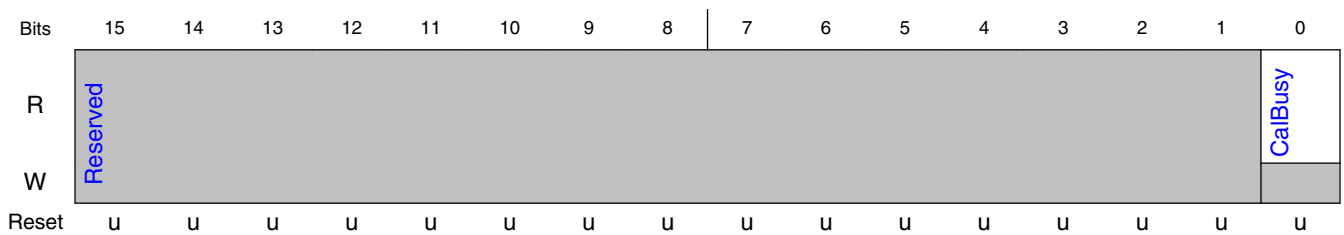
Field	Function
15-1 —	Reserved
0 PllDllLockDone	Set by the PIE to 1 after it has finished the PLL/DLL lock sequence. Set by the PIE to 1 after it has finished the PLL/DLL lock sequence. It is only cleared to 0 when PHY is Reset or exiting LP3 state.

9.3.3.6.65 Impedance Calibration Busy Status (CalBusy)

9.3.3.6.65.1 Offset

Register	Offset
CalBusy	12Eh

9.3.3.6.65.2 Diagram



9.3.3.6.65.3 Fields

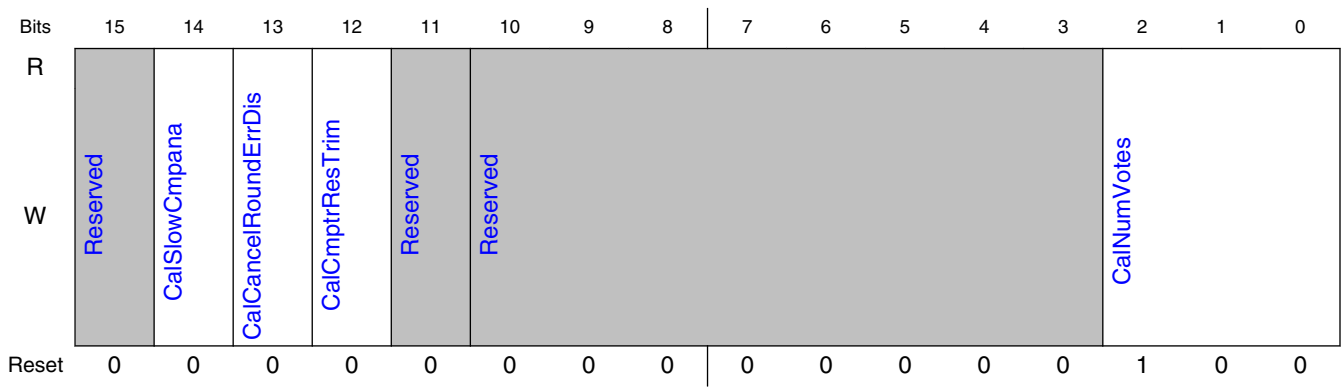
Field	Function
15-1 —	Reserved
0 CalBusy	Read 1 if the calibrator is actively calibrating. Read 1 if the calibrator is actively calibrating. Any changes to calibrator-related CSRs may only be made if the calibrator is disabled (via CSR CalRun) and this CSR reads 0.

9.3.3.6.66 Miscellaneous impedance calibration controls. (CalMisc2)

9.3.3.6.66.1 Offset

Register	Offset
CalMisc2	130h

9.3.3.6.66.2 Diagram



9.3.3.6.66.3 Fields

Field	Function
15 —	Miscellaneous impedance calibration controls.
14 CalSlowCmpana	When set, this CSR increases the time allowed for the cmpana cell to settle, by 50%. When set, this CSR increases the time allowed for the cmpana cell to settle, by 50%. It changes from 3 to 6 refClk ticks.
13 CalCancelRoundErrDis	The PEXT calibration result and NINT calibration results naturally include a rounding error which manifests as a change of impedance at the pad. The PEXT calibration result and NINT calibration results naturally include a rounding error which manifests as a change of impedance at the pad. Where both of these errors are in the same direction the PEXT calibration result will be adjusted by 1 to offset NINT error. Setting thie csr to 1 prevents this adjustment.
12 CalCmptrResTrim	Reserved

Table continues on the next page...

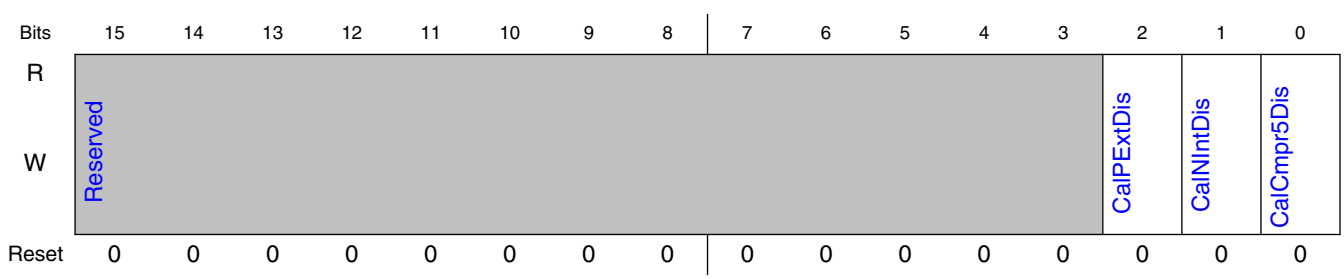
Field	Function
11 —	Reserved Formerly used to Skip ODT calibration if required.
10-3 —	Reserved
2-0 CalNumVotes	This CSR controls the number of consecutive comparator output bits over which majority voting is done. This CSR controls the number of consecutive comparator output bits over which majority voting is done. The result of the voting is fed into the SAR. The voting is initiated after the time configured via csrs CalSlowCmpana has elapsed. CSR Value : number of Votes 0 : 1 1 : 9 2 : 17 3 : 33 4 : 65 (defaults) 5 : 129 6 : 193 7 : 255

9.3.3.6.67 Controls for disabling the impedance calibration of certain targets. (CalMisc)

9.3.3.6.67.1 Offset

Register	Offset
CalMisc	134h

9.3.3.6.67.2 Diagram



9.3.3.6.67.3 Fields

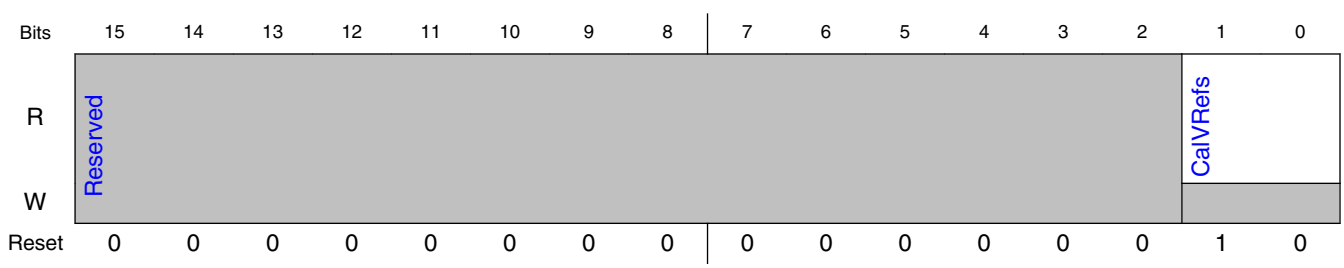
Field	Function
15-3 —	Reserved
2 CalPExtDis	Setting this CSR prevents the calibration engine from overwriting the CSRs TxCalBinP and TxCalThP with an automatically generated value, in which case a value must be supplied by software.
1 CalNIntDis	Setting this CSR prevents the calibration engine from overwriting the CSRs TxCalBinN and TxCalThN with an automatically generated value, in which case a value must be supplied by software.
0 CalCmpr5Dis	Setting this CSR prevents the calibration engine from using the result from the CalCmpr5 stage of calibration. Setting this CSR prevents the calibration engine from using the result from the CalCmpr5 stage of calibration. Software must supply a replacement value via CSR CalCmpr5Ovr.

9.3.3.6.68 (CalVRefs)

9.3.3.6.68.1 Offset

Register	Offset
CalVRefs	136h

9.3.3.6.68.2 Diagram



9.3.3.6.68.3 Fields

Field	Function
15-2 —	Reserved
1-0	This CSR drives the Cmpdig_CalRef pin of the cmpana cell at various stages of calibration.

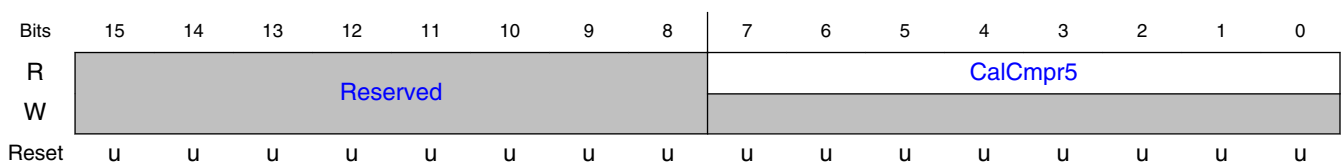
Field	Function
CalVRefs	Following are the encoding values: 2'b00 - 40% Vref 2'b01 - 41.34% Vref 2'b10 - 50% Vref (Default) 2'b11 - 60% Vref

9.3.3.6.69 Impedance Calibration Cmpr control (CalCmpr5)

9.3.3.6.69.1 Offset

Register	Offset
CalCmpr5	138h

9.3.3.6.69.2 Diagram



9.3.3.6.69.3 Fields

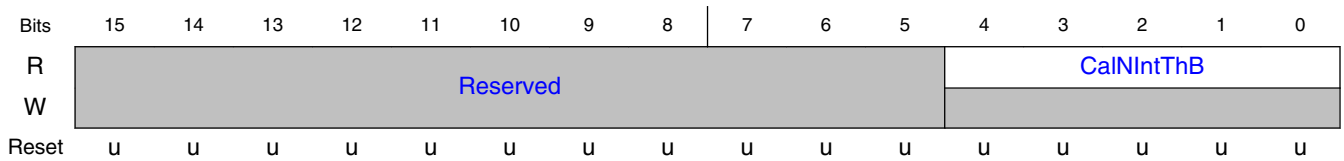
Field	Function
15-8 —	Reserved
7-0 CalCmpr5	Returns the offset-compensated DAC code for the cmpana circuit at VRef == 0. Returns the offset-compensated DAC code for the cmpana circuit at VRef == 0.5 * VDDQ.

9.3.3.6.70 Impedance Calibration NInt control (CalNInt)

9.3.3.6.70.1 Offset

Register	Offset
CalNInt	13Ah

9.3.3.6.70.2 Diagram



9.3.3.6.70.3 Fields

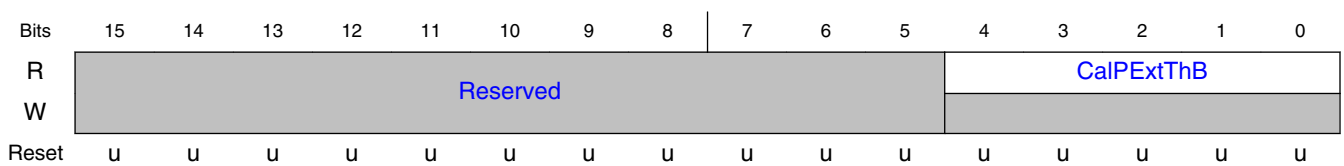
Field	Function
15-5 —	Reserved
4-0 CalNIntThB	The value here is the number of thermometer bits which are set.

9.3.3.6.71 Impedance Calibration PExt control (CalPExt)

9.3.3.6.71.1 Offset

Register	Offset
CalPExt	13Ch

9.3.3.6.71.2 Diagram



9.3.3.6.71.3 Fields

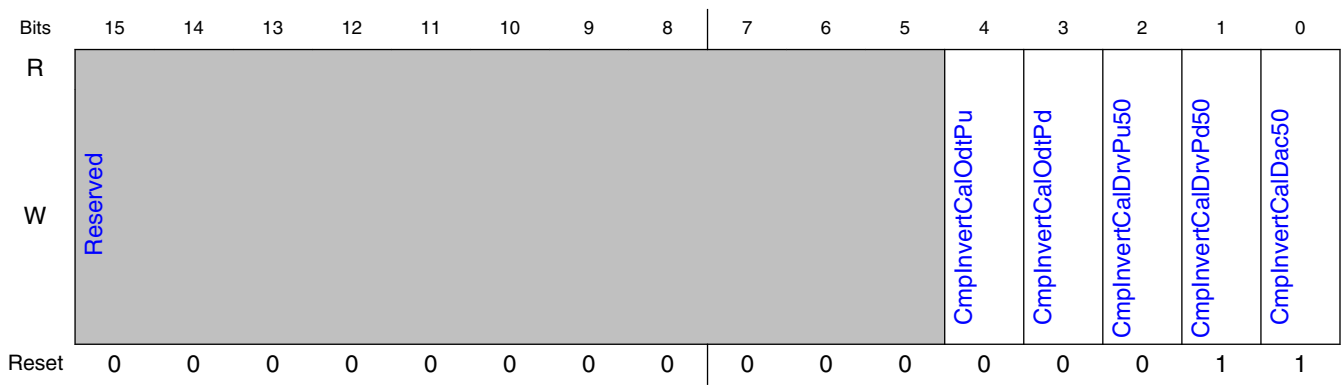
Field	Function
15-5 —	Reserved
4-0 CalPExtThB	The value here is the number of thermometer bits which are set.

9.3.3.6.72 Impedance Calibration Cmp Invert control (CalCmpInvert)

9.3.3.6.72.1 Offset

Register	Offset
CalCmpInvert	150h

9.3.3.6.72.2 Diagram



9.3.3.6.72.3 Fields

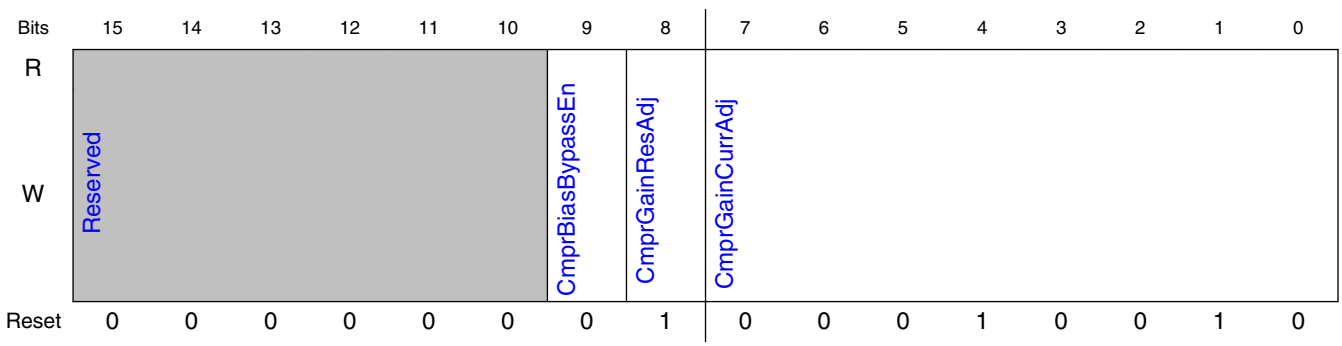
Field	Function
15-5 —	Reserved
4 CmpInvertCalOdtPu	Impedance Calibration Cmp Invert control
3 CmpInvertCalOdtPd	Impedance Calibration Cmp Invert control
2 CmpInvertCalDrvPu50	Impedance Calibration Cmp Invert control
1 CmpInvertCalDrvPd50	Impedance Calibration Cmp Invert control
0 CmpInvertCalDac50	Impedance Calibration Cmp Invert control

9.3.3.6.73 Impedance Calibration Cmpana control (CalCmpanaCntrl)

9.3.3.6.73.1 Offset

Register	Offset
CalCmpanaCntrl	15Ch

9.3.3.6.73.2 Diagram



9.3.3.6.73.3 Fields

Field	Function
15-10 —	Reserved
9 CmprBiasBypassEn	Impedance Calibration Cmpana control
8 CmprGainResAdj	Impedance Calibration Cmpana control
7-0 CmprGainCurrAdj	Impedance Calibration Cmpana control

9.3.3.6.74 Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiRdDataCsDestMap_p0)

9.3.3.6.74.1 Offset

Register	Offset
DfiRdDataCsDestMap_p0	160h

9.3.3.6.74.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								DfiRdDestm3		DfiRdDestm2		DfiRdDestm1		DfiRdDestm0	
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	0

9.3.3.6.74.3 Fields

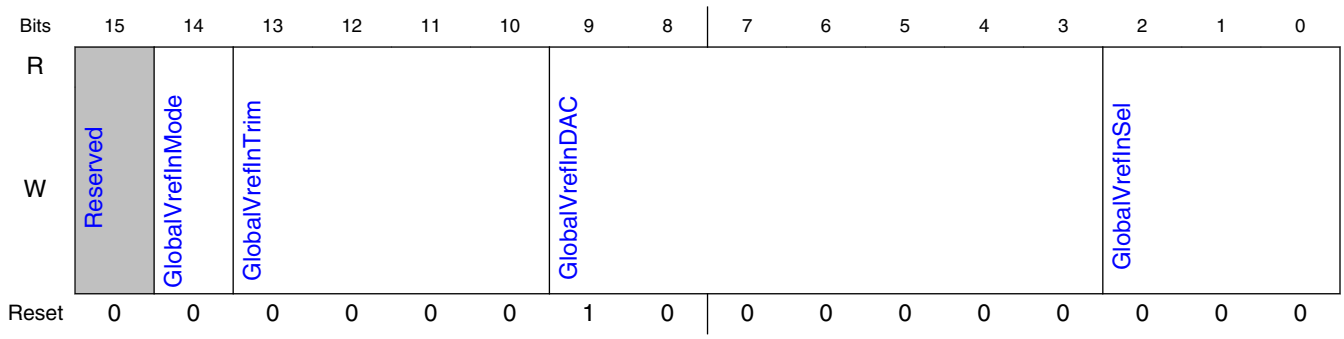
Field	Function
15-8 —	Reserved
7-6 DfiRdDestm3	Maps dfi_rddata_cs_n_p0[3] to dest DfiRdDestm3 timing For example, if 3 dfi_rddata_cs_n_p0[3] will use Register RxEn,ClkDlyTg3 timing.
5-4 DfiRdDestm2	Maps dfi_rddata_cs_n_p0[2] to dest DfiRdDestm2 timing For example, if 2 dfi_rddata_cs_n_p0[2] will use Register RxEn,ClkDlyTg2 timing.
3-2 DfiRdDestm1	Maps dfi_rddata_cs_n_p0[1] to dest DfiRdDestm1 timing For example, if 1 dfi_rddata_cs_n[_p01] will use Register RxEn,ClkDlyTg1 timing.
1-0 DfiRdDestm0	Maps dfi_rddata_cs_n_p0[0] to dest DfiRdDestm0 timing For example, if 0 dfi_rddata_cs_n_p0[0] will use Register RxEn,ClkDlyTg0 timing.

9.3.3.6.75 PHY Global Vref Controls (VrefInGlobal_p0)

9.3.3.6.75.1 Offset

Register	Offset
VrefInGlobal_p0	164h

9.3.3.6.75.2 Diagram



9.3.3.6.75.3 Fields

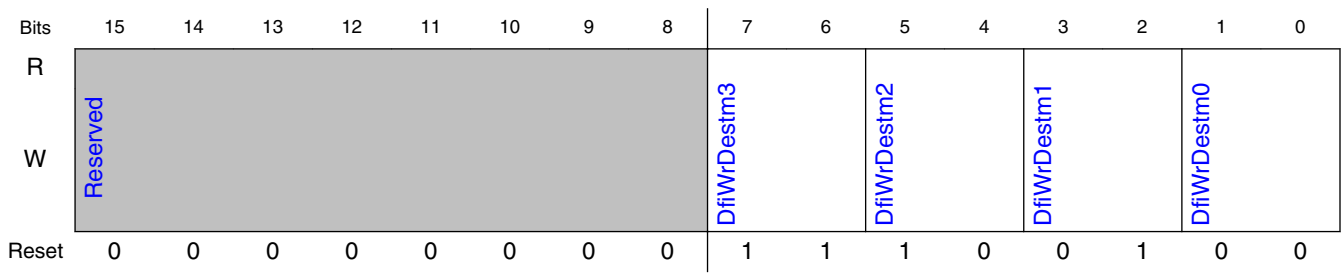
Field	Function
15 —	Reserved
14 GlobalVrefInMode	RSVD
13-10 GlobalVrefInTrim	RSVD
9-3 GlobalVrefInDAC	DAC code for internal Vref generation The DAC has two ranges; the range is set by GlobalVrefInSel[2] ========== RANGE0 : DDR3,DDR4,LPDDR3 [GlobalVrefInSel[2] = 0] DAC Output Voltage = GlobalVrefInDAC == 6'h00 ? Hi-Z : 0. DAC code for internal Vref generation The DAC has two ranges; the range is set by GlobalVrefInSel[2] ========== RANGE0 : DDR3,DDR4,LPDDR3 [GlobalVrefInSel[2] = 0] DAC Output Voltage = GlobalVrefInDAC == 6'h00 ? Hi-Z : 0.345*VDDQ + (0.005*GlobalVrefInDAC)*VDDQ ========== RANGE1 : LPDDR4 [GlobalVrefInSel[2] = 1] DAC Output Voltage = GlobalVrefInDAC == 6'h00 ? Hi-Z : (0.005*(GlobalVrefInDAC-1))*VDDQ
2-0 GlobalVrefInSel	GlobalVrefInSel[1:0] controls the mode of the PHY VREF DAC and the BP_VREF pin ========== 2'b00 - PHY Vref DAC Range0 -- BP_VREF = Hi-Z 2'b01 - Reserved Encoding 2'b10 - PHY Vref DAC Range0 -- BP_VREF connected to PLL Analog Bus 2'b11 - PHY Vref DAC Range0 -- BP_VREF connected to PHY Vref DAC ========== GlobalVrefInSel[2] shall be set according to Dram Protocol: Protocol GlobalVrefInSel[2] ---------- DDR3 1'b0 DDR4 1'b0 LPDDR3 1'b0 LPDDR4 1'b1

9.3.3.6.76 Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiWrDataCsDestMap_p0)

9.3.3.6.76.1 Offset

Register	Offset
DfiWrDataCsDestMap_p0	168h

9.3.3.6.76.2 Diagram



9.3.3.6.76.3 Fields

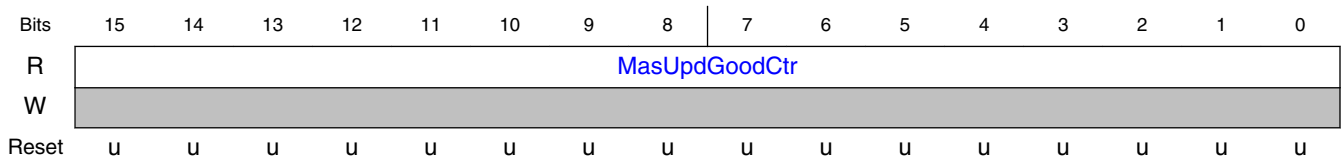
Field	Function
15-8 —	Reserved
7-6 DfiWrDestm3	Maps dfi_wrdata_cs_n_p0[3] to dest DfiWrDestm3 timing (use Register TxDq,DqsDlyTg3) For example, if 3 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg3 timing.
5-4 DfiWrDestm2	Maps dfi_wrdata_cs_n_p0[2] to dest DfiWrDestm2 timing For example, if 2 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg2 timing.
3-2 DfiWrDestm1	Maps dfi_wrdata_cs_n_p0[1] to dest DfiWrDestm1 timing For example, if 1 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg1 timing.
1-0 DfiWrDestm0	Maps dfi_wrdata_cs_n_p0[0] to dest DfiWrDestm0 timing For example, if 0 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg0 timing.

9.3.3.6.77 Counts successful PHY Master Interface Updates (PPTs) (MasUpdGoodCtr)

9.3.3.6.77.1 Offset

Register	Offset
MasUpdGoodCtr	16Ah

9.3.3.6.77.2 Diagram



9.3.3.6.77.3 Fields

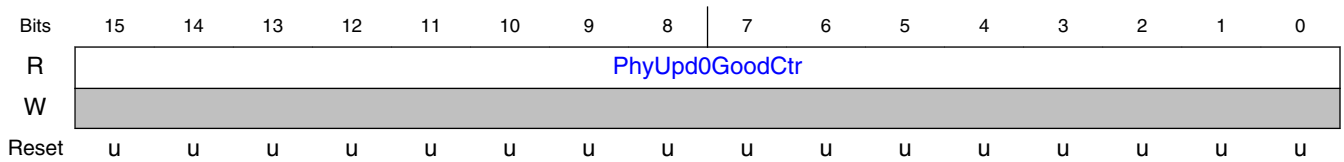
Field	Function
15-0 MasUpdGoodCtr	This register increments whenever the Memory Controller acknowledges a PHY Master Interface request (i. This register increments whenever the Memory Controller acknowledges a PHY Master Interface request (i.e., a request for the PHY to take over the DFI for Periodic Phase Training (PPT)). (As per section 11 of the Preliminary DFI 4.0 Specification Rev. 2) In a 2-channel system, both MC channels must ack. The counter saturates at max value 65535.

9.3.3.6.78 Counts successful PHY-initiated DFI0 Interface Updates (PhyUpd0GoodCtr)

9.3.3.6.78.1 Offset

Register	Offset
PhyUpd0GoodCtr	16Ch

9.3.3.6.78.2 Diagram



9.3.3.6.78.3 Fields

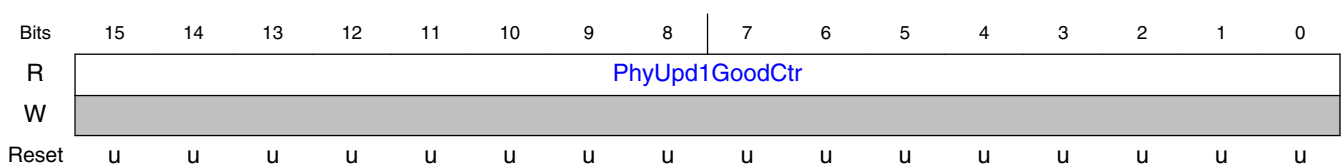
Field	Function
15-0 PhyUpd0GoodCtr	This register increments whenever the Memory Controller acknowledges a PHY-initiated DFI0 interface update request. This register increments whenever the Memory Controller acknowledges a PHY-initiated DFI0 interface update request. The counter saturates at max value 65535.

9.3.3.6.79 Counts successful PHY-initiated DFI1 Interface Updates (PhyUpd1GoodCtr)

9.3.3.6.79.1 Offset

Register	Offset
PhyUpd1GoodCtr	16Eh

9.3.3.6.79.2 Diagram



9.3.3.6.79.3 Fields

Field	Function
15-0 PhyUpd1GoodCtr	This register increments whenever the Memory Controller acknowledges a PHY-initiated DFI1 interface update request. This register increments whenever the Memory Controller acknowledges

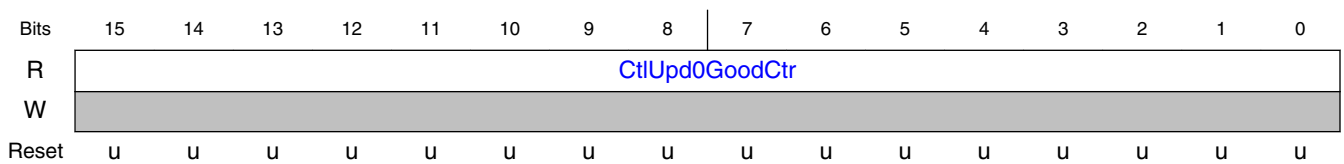
Field	Function
	a PHY-initiated DFI1 interface update request. The counter saturates at max value 65535.

9.3.3.6.80 Counts successful Memory Controller DFI0 Interface Updates (CtlUpd0GoodCtr)

9.3.3.6.80.1 Offset

Register	Offset
CtlUpd0GoodCtr	170h

9.3.3.6.80.2 Diagram



9.3.3.6.80.3 Fields

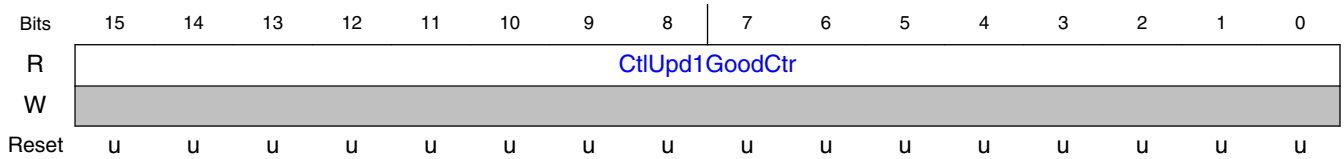
Field	Function
15-0 CtlUpd0GoodCtr	This register increments whenever the PHY acknowledges a Memory Controller-initiated DFI0 interface update request. This register increments whenever the PHY acknowledges a Memory Controller-initiated DFI0 interface update request. The counter saturates at max value 65535.

9.3.3.6.81 Counts successful Memory Controller DFI1 Interface Updates (CtlUpd1GoodCtr)

9.3.3.6.81.1 Offset

Register	Offset
CtlUpd1GoodCtr	172h

9.3.3.6.81.2 Diagram



9.3.3.6.81.3 Fields

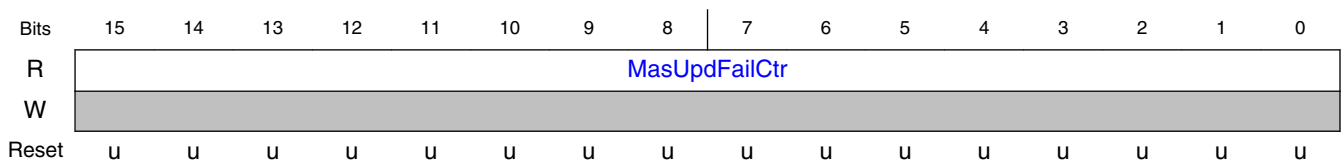
Field	Function
15-0 CtlUpd1GoodCtr	This register increments whenever the PHY acknowledges a Memory Controller-initiated DFI1 interface update request. This register increments whenever the PHY acknowledges a Memory Controller-initiated DFI1 interface update request. The counter saturates at max value 65535.

9.3.3.6.82 Counts unsuccessful PHY Master Interface Updates (MasUpdFailCtr)

9.3.3.6.82.1 Offset

Register	Offset
MasUpdFailCtr	174h

9.3.3.6.82.2 Diagram



9.3.3.6.82.3 Fields

Field	Function
15-0 MasUpdFailCtr	This register increments whenever the PHY asserts a PHY Master Interface request, but the Memory Controller doesn't acknowledge the request within the allowed interval.

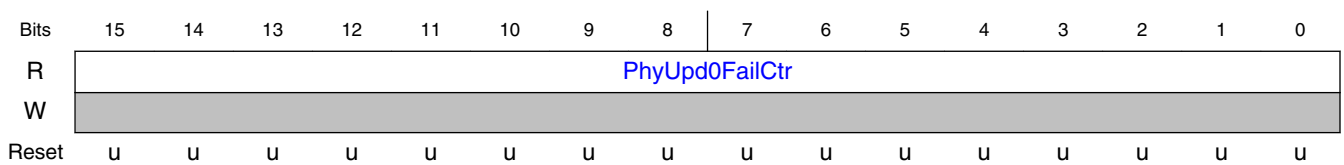
Field	Function
	This register increments whenever the PHY asserts a PHY Master Interface request, but the Memory Controller doesn't acknowledge the request within the allowed interval. (As per section 11 of the Preliminary DFI 4.0 Specification Rev. 2) The counter saturates at max value 65535. In a 2-channel system, a failure occurs if either MC channels fails to ack.

9.3.3.6.83 Counts unsuccessful PHY-initiated DFI0 Interface Updates (PhyUpd0FailCtr)

9.3.3.6.83.1 Offset

Register	Offset
PhyUpd0FailCtr	176h

9.3.3.6.83.2 Diagram



9.3.3.6.83.3 Fields

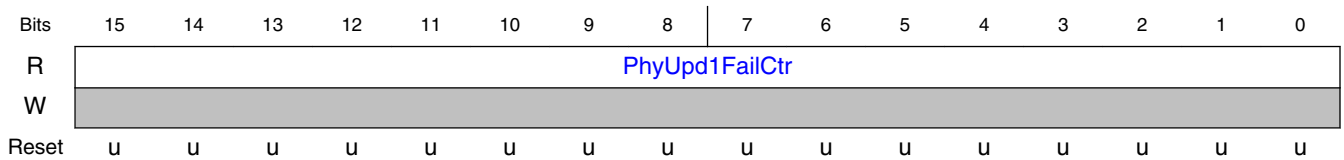
Field	Function
15-0 PhyUpd0FailCtr	This register increments whenever the PHY asserts a DFI0 Interface update request, but the Memory Controller doesn't acknowledge the request within the allowed interval. This register increments whenever the PHY asserts a DFI0 Interface update request, but the Memory Controller doesn't acknowledge the request within the allowed interval. (As per section 11 of the Preliminary DFI 4.0 Specification Rev. 2) The counter saturates at max value 65535.

9.3.3.6.84 Counts unsuccessful PHY-initiated DFI1 Interface Updates (PhyUpd1FailCtr)

9.3.3.6.84.1 Offset

Register	Offset
PhyUpd1FailCtr	178h

9.3.3.6.84.2 Diagram



9.3.3.6.84.3 Fields

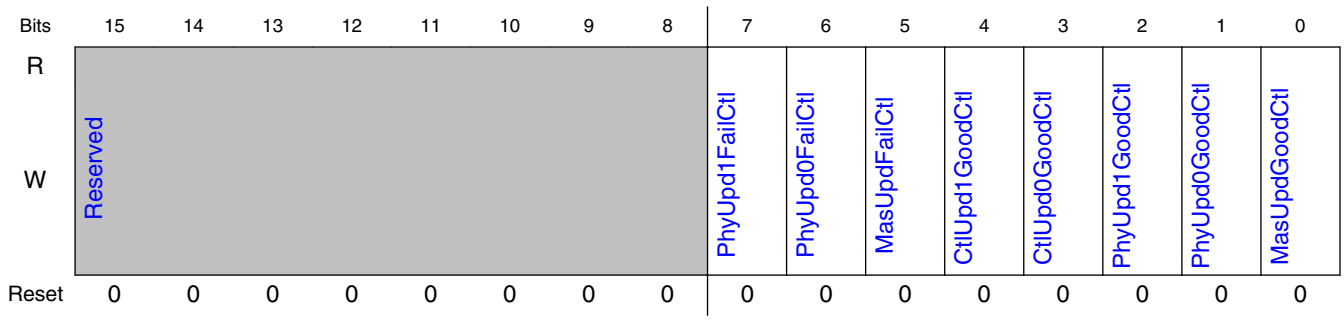
Field	Function
15-0 PhyUpd1FailCtr	This register increments whenever the PHY asserts a DFI1 Interface update request, but the Memory Controller doesn't acknowledge the request within the allowed interval. This register increments whenever the PHY asserts a DFI1 Interface update request, but the Memory Controller doesn't acknowledge the request within the allowed interval. (As per section 11 of the Preliminary DFI 4.0 Specification Rev. 2) The counter saturates at max value 65535.

9.3.3.6.85 Enables for Performance Counters (PhyPerfCtrEnable)

9.3.3.6.85.1 Offset

Register	Offset
PhyPerfCtrEnable	17Ah

9.3.3.6.85.2 Diagram



9.3.3.6.85.3 Fields

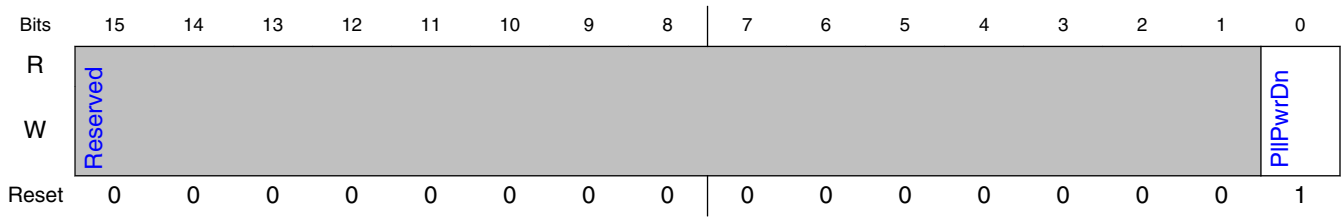
Field	Function
15-8 —	Reserved
7 PhyUpd1FailCtl	Enables PhyUpd1FailCtr
6 PhyUpd0FailCtl	Enables PhyUpd0FailCtr
5 MasUpdFailCtl	Enables MasUpdFailCtr
4 CtlUpd1GoodCtl	Enables CtlUpd1GoodCtr
3 CtlUpd0GoodCtl	Enables CtlUpd0GoodCtr
2 PhyUpd1GoodCtl	Enables PhyUpd1GoodCtr
1 PhyUpd0GoodCtl	Enables PhyUpd0GoodCtr
0 MasUpdGoodCtl	Enables MasUpdGoodCtr

9.3.3.6.86 PLL Power Down (PllPwrDn)

9.3.3.6.86.1 Offset

Register	Offset
PllPwrDn	186h

9.3.3.6.86.2 Diagram



9.3.3.6.86.3 Fields

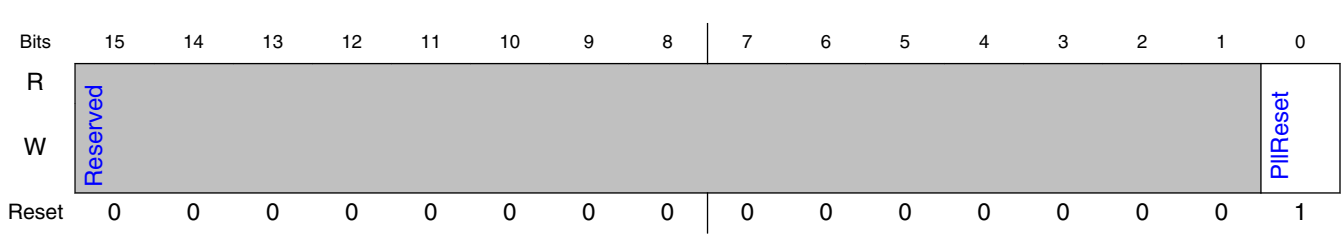
Field	Function
15-1 —	Reserved
0 PllPwrDn	NOTE : This CSR is written by PHY Initialization Engine (PIE) and the data in here will be overwritten. NOTE : This CSR is written by PHY Initialization Engine (PIE) and the data in here will be overwritten. Places the PLL in Power Down when asserted.

9.3.3.6.87 PLL Reset (PllReset)

9.3.3.6.87.1 Offset

Register	Offset
PllReset	188h

9.3.3.6.87.2 Diagram



9.3.3.6.87.3 Fields

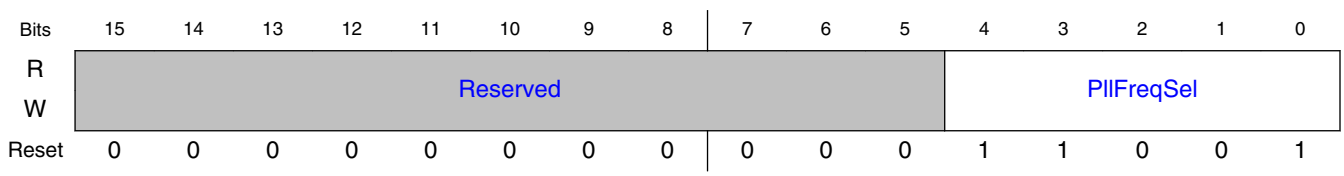
Field	Function
15-1 —	Reserved
0 PllReset	Reserved

9.3.3.6.88 PState dependent PLL Control Register 2 (PllCtrl2_p0)

9.3.3.6.88.1 Offset

Register	Offset
PllCtrl2_p0	18Ah

9.3.3.6.88.2 Diagram



9.3.3.6.88.3 Fields

Field	Function
15-5 —	Reserved
4-0 PllFreqSel	Adjusts the loop parameters to compensate for different VCO bias points, and input/output clock division ratios. Adjusts the loop parameters to compensate for different VCO bias points, and input/output clock division ratios. Program based on reference clock freqency (DfiClk) with this table.

9.3.3.6.89 PLL Control Register 0 (PllCtrl0)

9.3.3.6.89.1 Offset

Register	Offset
PllCtrl0	18Ch

9.3.3.6.89.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PllSpareCtrl0	PllLockPhSel		PllLockCntSel	PllGearShift	PllReserved10x9		PllSyncBusByp	PllSyncBusFlush	PllSelDfiFreqRatio	PllBypassMode	PllPreset	PllOutBypEn	PllX2Mode	PllBypSel	PllStandby
Reset	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0

9.3.3.6.89.3 Fields

Field	Function
15 PllSpareCtrl0	Spare bits for PLL control.
14-13 PllLockPhSel	Lock detect phase selection. Lock detect phase selection. 0: Test mode reduced lock window 1: Default operation phase lock 2: Test mode (mismatch check) 3: Test mode (+80ps window)
12 PllLockCntSel	Lock detect counter selection. Lock detect counter selection. 0: Normal lock detector behavior phase lock 1: Test mode (Shorten lock cycle count 16->8)
11 PllGearShift	Puts PLL in fast re-locking mode 0: default, normal mode 1: fast relock gear
10-9 PllReserved10x9	Reserved.
8 PllSyncBusByp	When asserted bypasses the Pll SyncPulse and uses a synchronizer of the same latency.
7	Used to flush the syncbus logic of the PLL during PHY initialization or LP3 Exit sequence.

Table continues on the next page...

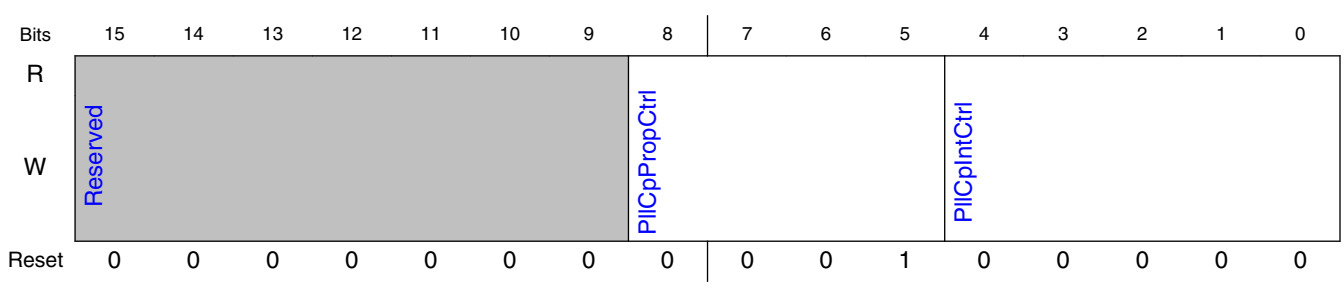
Field	Function
PllSyncBusFlush	Used to flush the syncbus logic of the PLL during PHY initialization or LP3 Exit sequence. Should Only be toggled when synbus is not used and Register PllOutBypEn is asserted.
6 PllSelDfiFreqRatio	reserved.
5 PllBypassMode	PLL Bypass clock mux control. PLL Bypass clock mux control. If set, BypassPclk input will be used as byp_pll1x_in clock for PLL. Default 0.
4 PllPreset	Put PLL in preset mode.
3 PllOutBypEn	Controls the antiglitch mux on the pllout_x1x2x4 path 1: pllout_x1x2x4 = byp_pllin_x1 0: pllout_x1x2x4 = VCO (SCD) (selected by x2_mode)
2 PllX2Mode	conects to x2_mode pins of PLL. conects to x2_mode pins of PLL. pllout_x4x2 output frequency selection. 0: PLOUT_X4X2 output = 4*pllin_x1 frequency 1: PLOUT_X4X2 output = 2*pllin_x1 frequency
1 PllBypSel	Reserved
0 PllStandby	Connects directly to standby pin of PLL. Connects directly to standby pin of PLL. Puts PLL into standbymode.

9.3.3.6.90 PState dependent PLL Control Register 1 (PllCtrl1_p0)

9.3.3.6.90.1 Offset

Register	Offset
PllCtrl1_p0	18Eh

9.3.3.6.90.2 Diagram



9.3.3.6.90.3 Fields

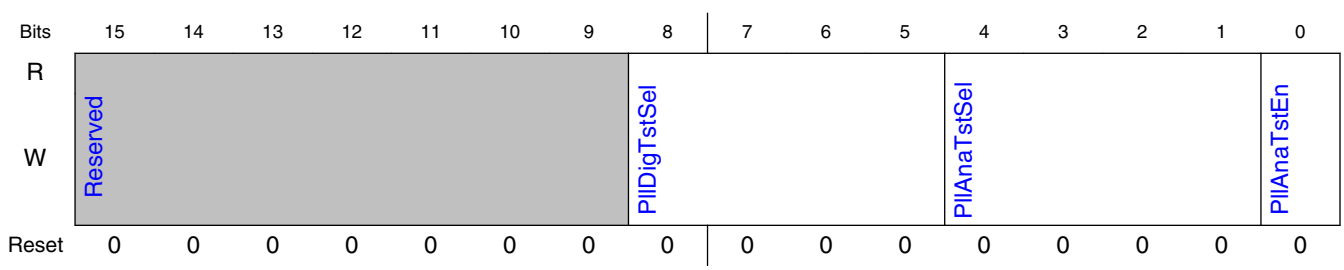
Field	Function
15-9 —	Reserved
8-5 PllCpPropCtrl	connects directly to cp_prop_cntrl<3:0> of PLL. connects directly to cp_prop_cntrl<3:0> of PLL. Charge pump proportional current control.
4-0 PllCpIntCtrl	connects directly to cp_int_cntrl<1:0> in PLL. connects directly to cp_int_cntrl<1:0> in PLL. Charge pump integrating current control.

9.3.3.6.91 PLL Testing Control Register (PllTst)

9.3.3.6.91.1 Offset

Register	Offset
PllTst	190h

9.3.3.6.91.2 Diagram



9.3.3.6.91.3 Fields

Field	Function
15-9 —	Reserved
8-5 PllDigTstSel	Connects directly to pll_dig_test_sel<2:0> of PLL. Connects directly to pll_dig_test_sel<2:0> of PLL. Digital test mux select. Control bits are decoded to enable various digital

Table continues on the next page...

Field	Function
	test signals to be brought out via digital test pin, see decoder table for detail. PllDigTstOut[1] is observable on BP_ALERT and Digital Observation Pin via MtestMuxSel logic. (if enabled in this configuration) PllDigTstOut[0] is observable on Digital Observation Pin via MtestMuxSel logic only. During mission mode PllDigTstSel should be set to 3'h0. Byp = PllBypassMode CSR value Sel = PllDigTstSel CSR value |---------- | | | PllDigTstOut | |Byp| Sel | [1] [0] | Description | ========== ===== | 0 | 000 | 0 | 0 | This code should be set during normal use | | | | | to minimize clock jitter | 0 | 001 | fbk_clk_ldet | pllin_x1 | Buffered version of fbk_clk at PFD input | 0 | 010 | ref_clk_ldet | pllin_x1 | Buffered version of ref_clk at PFD input | 0 | 011 | pllin_x1 | pllin_x1 | For calibrating delay skew between two | | | | | test outputs - input reference clock | 0 | 100 | clk_fbk | pllin_x1 | Buffered version of pllout_x1 | 0 | 101 | pllout_amux_x1x2x4 | pllin_x1 | Syncbus clock output (Fout<Fmax_dto only) | 0 | 110 | en_count | pll_lock | Lock detector state test | 0 | 111 | standby_eff | eoc | Calibration state test | 1 | 000 | 0 | 0 | This code should be set during normal use | | | | | to minimize clock jitter | 1 | 001 | fbk_clk_ldet | pllin_x1 | Buffered version of fbk_clk at PFD input | | | | | if PLL in mission/byp 1 modes | 1 | 010 | ref_clk_ldet | pllin_x1 | Buffered version of ref_clk at PFD input | | | | | if PLL in mission/byp 1 modes | 1 | 011 | pllin_x1 | pllin_x1 | For calibrating delay skew between two | | | | | test outputs - input reference clock | 1 | 100 | clk_fbk | pllin_x1 | Buffered version of pllout_x1 if PLL in | | | | | mission/byp 1 modes | 1 | 101 | pllout_amux_x1x2x4 | pllin_x1 | Syncbus clock output (Fout<Fmax_dto only) | 1 | 110 | en_count | pll_lock | Lock detector test if PLL in mission/byp | | | | | 1 modes | 1 | 111 | standby_eff | eoc | Calibration state test if PLL in | | | | | mission/byp 1 modes
4-1	Connects directly to pll_ana_test_sel<3:0> of PLL.

Table continues on the next page...

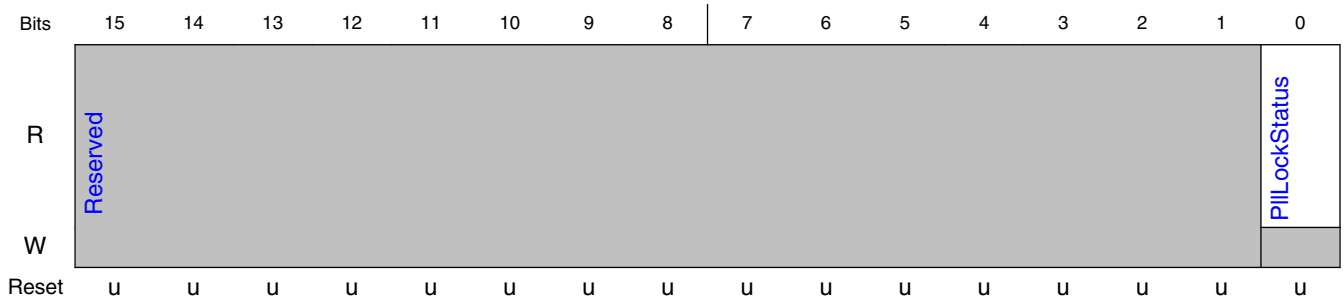
Field	Function
PllAnaTstSel	Connects directly to pll_ana_test_sel<3:0> of PLL. Control bits are decoded to enable various analog test signals to be brought out via analog test pin (pll_analog_test). Table below provides decode detail. |PllAnaTstSel | Pll Analog | Description | [4:1] | Pin Function | |========== | 0000 | Hi-Z | This code should be set when the PLL | | | is in normal use or when the digital | | | test port is being used for accurate | | | timing measurements | 0001 | core | Override vmarg_ext with analog pin | 0010 | core | Override regulator input with analog pin | 0011 | core | Observe gd voltage on analog test pin | 0100 | core | Observe vp voltage on analog test pin | 0101 | core | Observe vph voltage on analog test pin | 0110 | core | Observe vreg_cp voltage on analog test pin | 0111 | core | Observe v2i current (~10uA) on analog test pin | 1000 | core | Observe vreg_v2i voltage on analog test pin | 1001 | core | Observe vbp voltage from current source on | | | analog pin | 1010 | core | Observe vbn voltage from current source on | | | analog tpin | 1011 | core | NC | 1100 | core | NC | 1101 | VSS | 0v probe ir drop test | 1110 | VP | Core supply probe SCD IR drop test | 1111 | VPH | Core Supply probe Refdiv IR drop test
0 PllAnaTstEn	Connects directly to pll_ana_test_en of PLL. Connects directly to pll_ana_test_en of PLL. Analog test port enable 0 = analog test port disabled (hi-Z) 1 = analog test port enabled

9.3.3.6.92 PLL's pll_lock pin output (PllLockStatus)

9.3.3.6.92.1 Offset

Register	Offset
PllLockStatus	192h

9.3.3.6.92.2 Diagram



9.3.3.6.92.3 Fields

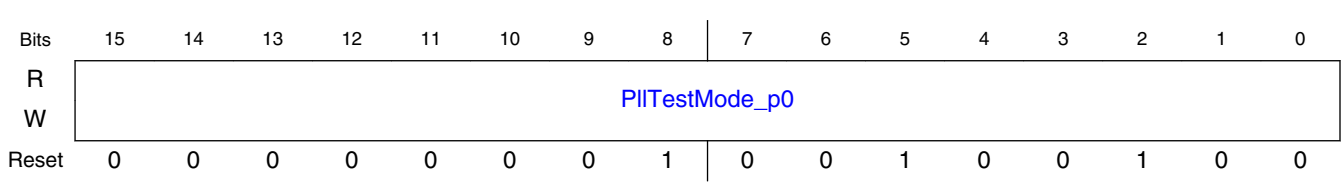
Field	Function
15-1 —	Reserved
0 PllLockStatus	Directly connected to the pll_Lock output. Directly connected to the pll_Lock output. When asserted Pll macro has asserted the Lock status. To avoid metastability issues, this register should perform a majority-vote after reading this register several times.

9.3.3.6.93 Additional controls for PLL CP/VCO modes of operation (PllTestMode_p0)

9.3.3.6.93.1 Offset

Register	Offset
PllTestMode_p0	194h

9.3.3.6.93.2 Diagram



9.3.3.6.93.3 Fields

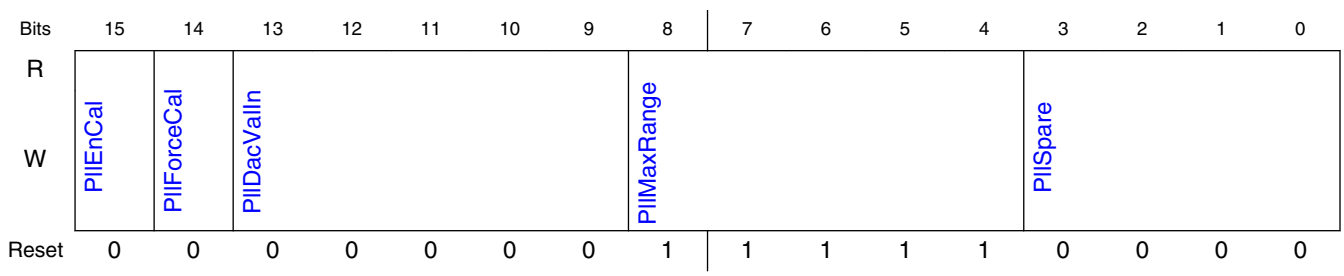
Field	Function
15-0 PllTestMode_p0	It is required to use default values for this CSR. Directly connected to testmode[15:0] pin of PLL

9.3.3.6.94 PLL Control Register 3 (PllCtrl3)

9.3.3.6.94.1 Offset

Register	Offset
PllCtrl3	196h

9.3.3.6.94.2 Diagram



9.3.3.6.94.3 Fields

Field	Function
15 PllEnCal	Calibration will run at standby rising edge if en_cal=1 if en_cal=0 calibration will not run
14 PllForceCal	connects directly to force_cal of PLL. connects directly to force_cal of PLL. Forces Calibration code dacval_in<4:0> to be used at preset mode if force_cal=1 if force_cal=0 previous stored value will be used
13-9 PllDacValIn	connects directly to dacval_in<4:0> of PLL. connects directly to dacval_in<4:0> of PLL. Calibration DAC input current setting, can be forced at rising edge of preset if force_cal =1

Table continues on the next page...

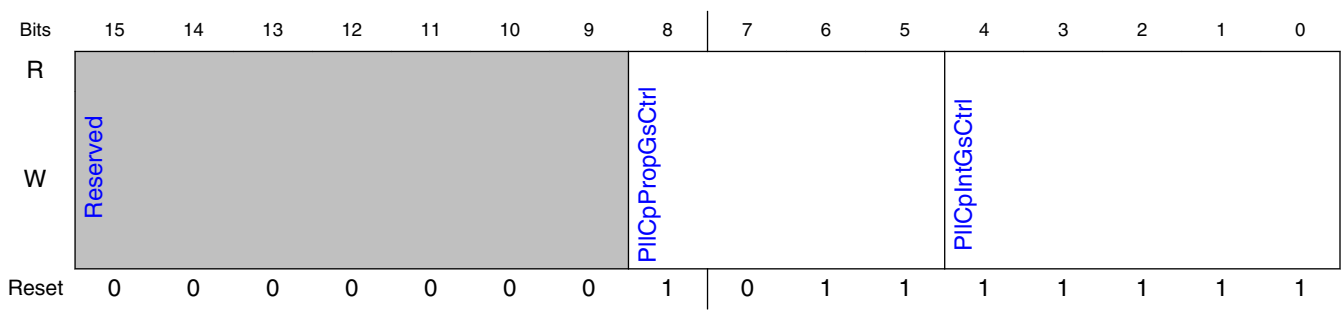
Field	Function
8-4 PllMaxRange	connects directly to maxrange of PLL. connects directly to maxrange of PLL. Setting for Saturation control of PLL.
3-0 PllSpare	Reserved

9.3.3.6.95 PState dependent PLL Control Register 4 (PllCtrl4_p0)

9.3.3.6.95.1 Offset

Register	Offset
PllCtrl4_p0	198h

9.3.3.6.95.2 Diagram



9.3.3.6.95.3 Fields

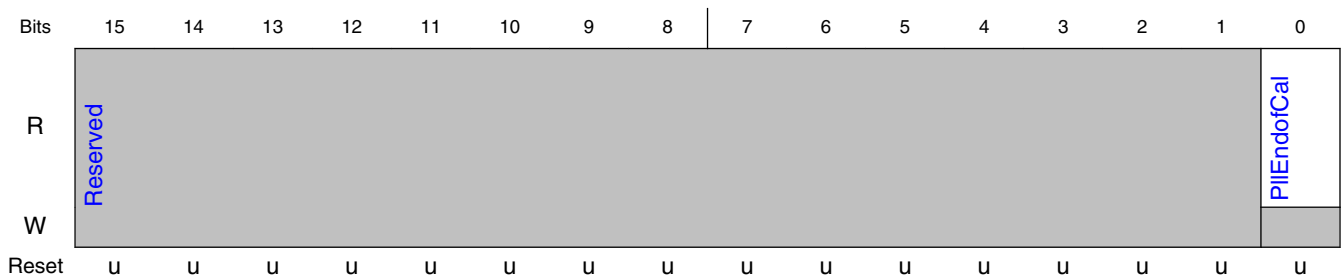
Field	Function
15-9 —	Reserved
8-5 PllCpPropGsCtrl	connects directly to cp_prop_gs_cntrl<3:0> of PLL. connects directly to cp_prop_gs_cntrl<3:0> of PLL. Charge pump proportional current control for fast relock and gearshift.
4-0 PllCpIntGsCtrl	connects directly to cp_int_gs_cntrl<4:0> in PLL. connects directly to cp_int_gs_cntrl<4:0> in PLL. Charge pump integrating current control for fast relock and gearshift.

9.3.3.6.96 PLL's eoc (end of calibration) output (PllEndofCal)

9.3.3.6.96.1 Offset

Register	Offset
PllEndofCal	19Ah

9.3.3.6.96.2 Diagram



9.3.3.6.96.3 Fields

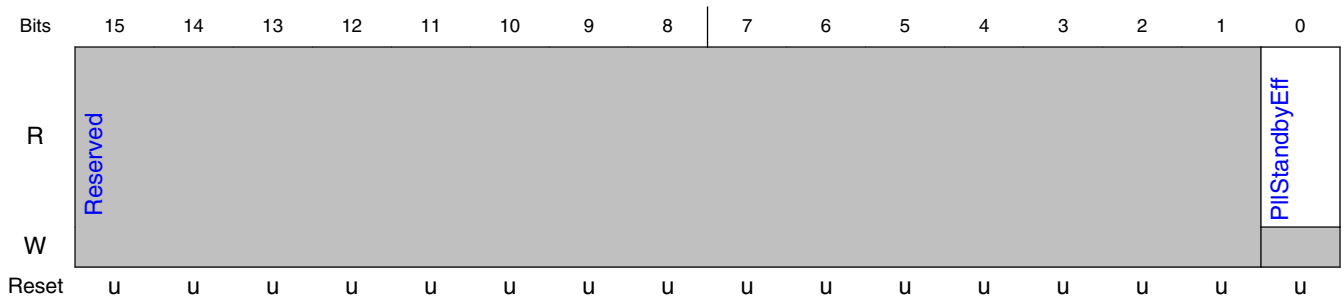
Field	Function
15-1 —	Reserved
0 PllEndofCal	Directly connected to the pll's eoc output.

9.3.3.6.97 PLL's standby_eff (effective standby) output (PllStandbyEff)

9.3.3.6.97.1 Offset

Register	Offset
PllStandbyEff	19Ch

9.3.3.6.97.2 Diagram



9.3.3.6.97.3 Fields

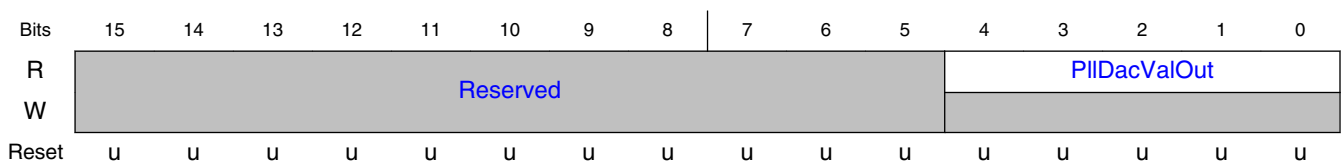
Field	Function
15-1 —	Reserved
0 PllStandbyEff	Returns state off PLL standby. Returns state off PLL standby. PLL will be in standby in PHY LP2 states.

9.3.3.6.98 PLL's Dacval_out output (PllDacValOut)

9.3.3.6.98.1 Offset

Register	Offset
PllDacValOut	19Eh

9.3.3.6.98.2 Diagram



9.3.3.6.98.3 Fields

Field	Function
15-5	Reserved

Table continues on the next page...

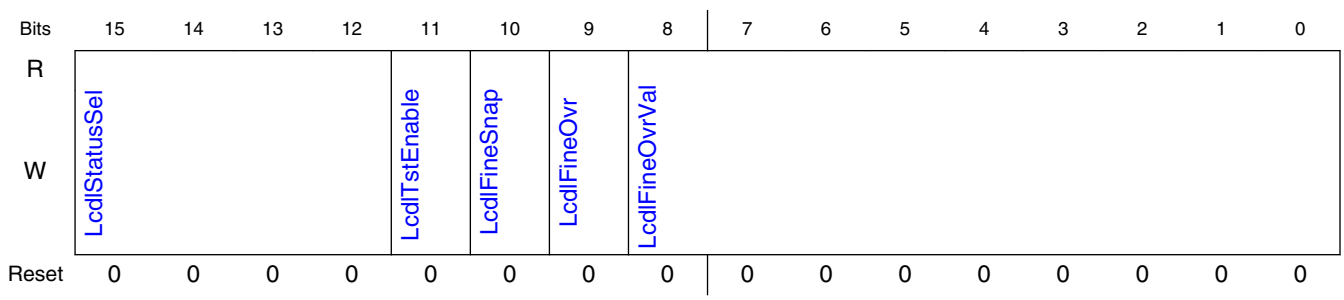
Field	Function
—	
4-0 PllDacValOut	Directly connected to the pll's dacval_out output. Directly connected to the pll's dacval_out output. Used for observation of PLL DAC code.

9.3.3.6.99 Controls for use in observing and testing the LCDLs. (LcdlDbgCntl)

9.3.3.6.99.1 Offset

Register	Offset
LcdlDbgCntl	1C6h

9.3.3.6.99.2 Diagram



9.3.3.6.99.3 Fields

Field	Function
15-12 LcdlStatusSel	Selects the LCDL status, from among the status for the 16 LCDLs in the DBYTE, for reading via Register DxLcdlStatus and an LCDL from among the LCDLs in the ANIB for reading via Register AcLcdlStatus LcdlStatusSel source for DxLcdlStatus source for AcLcdlStatus 15 lcdl_rxclk1t reserved 14 lcdl_rxclk0t reserved 13 lcdl_rxclk1c reserved 12 lcdl_rxclk0c reserved 11 lcdl_rxen1 anib11-tx 10 lcdl_rxen0 anib10-tx 9 lcdl_txln9 (dqs-lower) anib9-tx 8 lcdl_txln8 (dm/dqs-upper) anib8-tx 7 lcdl_txln7 (dq7) anib7-tx 6 lcdl_txln6 (dq6) anib6-tx 5 lcdl_txln5 (dq5) anib5-tx 4 lcdl_txln4 (dq4) anib4-tx 3 lcdl_txln3 (dq3) anib3-tx 2 lcdl_txln2 (dq2) anib2-tx 1 lcdl_txln1 (dq1) anib1-tx 0 lcdl_txln0 (dq0) anib0-tx
11 LcdlTstEnable	Enables the debug/test operations and status Ovr,Snap,StickyLock,StickyUnlock, and LiveLock.
10 LcdlFineSnap	Latch enable for reading the present LCDL 1UI estimate code in LcdlFineSnapVal and the present phase-detector value in LcdlPhdSnapVal
9 LcdlFineOvr	Forces the value of the present LCDL 1UI estimate code to be LcdlFineOvrVal for all LCDLs.

Table continues on the next page...

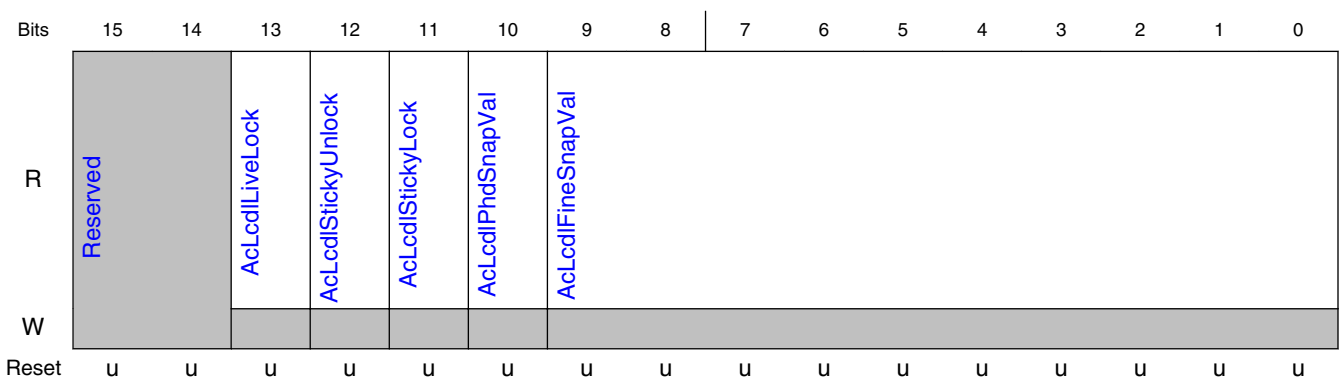
Field	Function
8-0 LcdlFineOvrVal	Value forced as the initial value while LcdlTstEnable=1 and LcdlFineOvr. Value forced as the initial value while LcdlTstEnable=1 and LcdlFineOvr. After the deassertion of LcdlFineOvr, the locking state-machine will attempt locking.

9.3.3.6.100 Debug status of the DBYTE LCDL (AcLcdlStatus)

9.3.3.6.100.1 Offset

Register	Offset
AcLcdlStatus	1C8h

9.3.3.6.100.2 Diagram



9.3.3.6.100.3 Fields

Field	Function
15-14 —	Reserved
13 AcLcdlLiveLock	present value of whether the LCDL is locked, valid when LcdlTstEnable=1.
12 AcLcdlStickyUnlock	latched value of whether the LCDL ever lost lock after the assertion of LcdlTstEnable.
11 AcLcdlStickyLock	latched value of whether the LCDL ever achieved lock after the assertion of LcdlTstEnable.

Table continues on the next page...

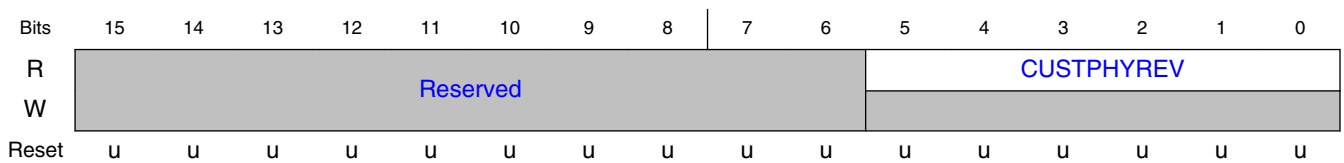
Field	Function
10 AcLcdlPhdSnapVal	Value of the LCDL phase-detector output, latched by pulse on LcdlFineSnap while csr LcdlTstEnable=1.
9-0 AcLcdlFineSnapVal	Value of the LCDL 1UI estimate code, latched by pulse on csrLcdlFineSnap while csr LcdlTstEnable=1. Value of the LCDL 1UI estimate code, latched by pulse on csrLcdlFineSnap while csr LcdlTstEnable=1. Index 9 is reserved for growth.

9.3.3.6.101 Customer settable by the customer (CUSTPHYREV)

9.3.3.6.101.1 Offset

Register	Offset
CUSTPHYREV	1DAh

9.3.3.6.101.2 Diagram



9.3.3.6.101.3 Fields

Field	Function
15-6 —	Reserved
5-0 CUSTPHYREV	The customer settable PHY version number. The customer settable PHY version number. The value is derived from the `define DWC_DDRPHY_CUST_PHYREV

9.3.3.6.102 The hardware version of this PHY, excluding the PUB (PHYREV)

9.3.3.6.102.1 Offset

Register	Offset
PHYREV	1DCh

9.3.3.6.102.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PHYMJR								PHYMDR				PHYMNR			
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

9.3.3.6.102.3 Fields

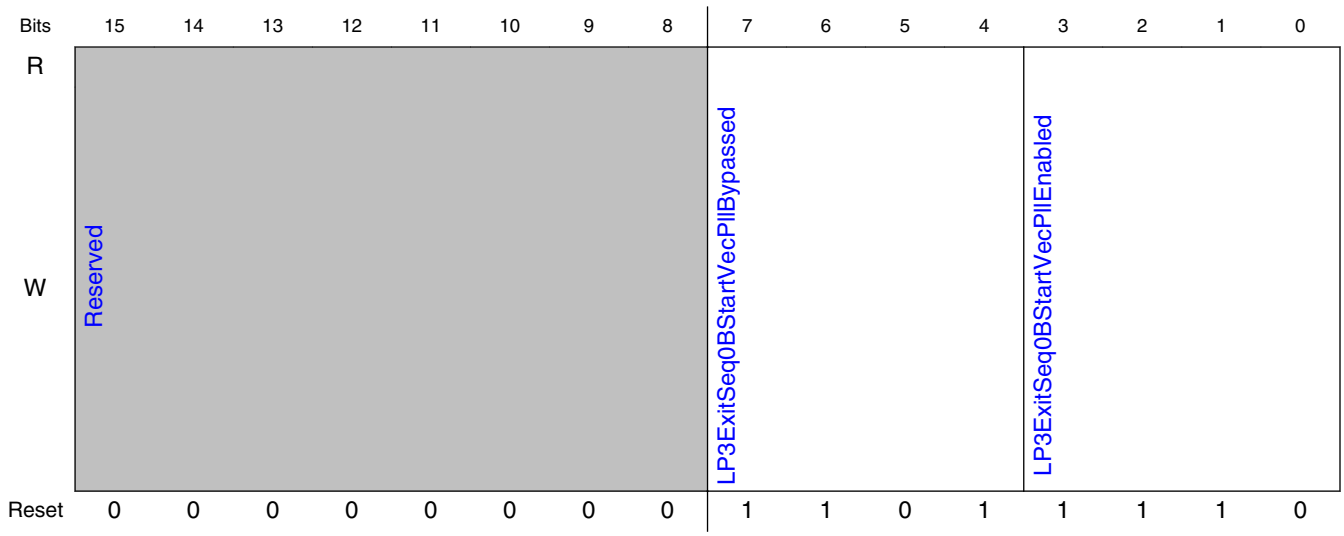
Field	Function
15-8 PHYMJR	Indicates major revision of the PHY.
7-4 PHYMDR	Indicates moderate revision of the PHY.
3-0 PHYMNR	Indicates minor update of the PHY.

9.3.3.6.103 Start vector value to be used for LP3-exit or Init PIE Sequence (LP3ExitSeq0BStartVector)

9.3.3.6.103.1 Offset

Register	Offset
LP3ExitSeq0BStartVector	1DEh

9.3.3.6.103.2 Diagram



9.3.3.6.103.3 Fields

Field	Function
15-8 —	Reserved
7-4 LP3ExitSeq0BStartVecPllBypassed	PIE Start Vector value to be used for LP3-exit or Init and target P-state has PLL bypassed
3-0 LP3ExitSeq0BStartVecPllEnabled	PIE Start Vector value to be used for LP3-exit or Init and target P-state has PLL enabled

9.3.3.6.104 DFI Frequency Translation Register 0 (DfiFreqXlat0)

9.3.3.6.104.1 Offset

Register	Offset
DfiFreqXlat0	1E0h

9.3.3.6.104.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DfiFreqXlatVal3				DfiFreqXlatVal2				DfiFreqXlatVal1				DfiFreqXlatVal0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.104.3 Fields

Field	Function
15-12 DfiFreqXlatVal3	The sequencer start vector used when dfi_freq value is 3.
11-8 DfiFreqXlatVal2	The sequencer start vector used when dfi_freq value is 2.
7-4 DfiFreqXlatVal1	The sequencer start vector used when dfi_freq value is 1.
3-0 DfiFreqXlatVal0	The sequencer start vector used when dfi_freq value is 0.

9.3.3.6.105 DFI Frequency Translation Register 1 (DfiFreqXlat1)

9.3.3.6.105.1 Offset

Register	Offset
DfiFreqXlat1	1E2h

9.3.3.6.105.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DfiFreqXlatVal7				DfiFreqXlatVal6				DfiFreqXlatVal5				DfiFreqXlatVal4			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.105.3 Fields

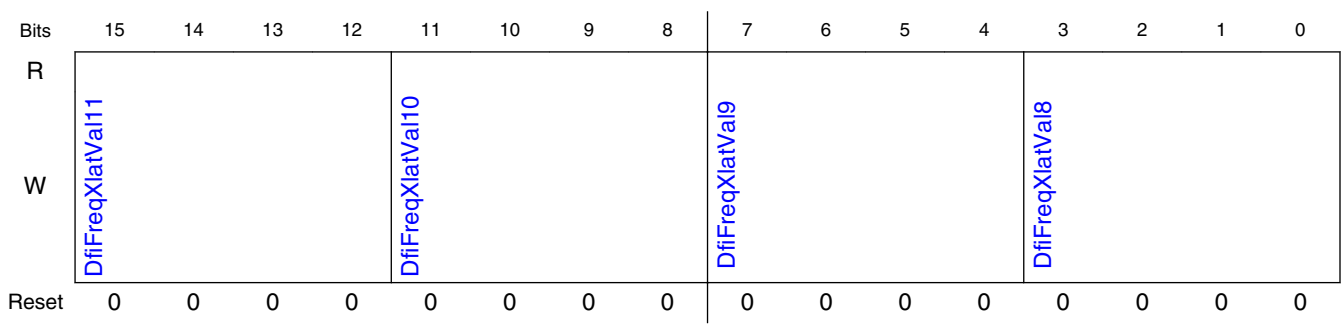
Field	Function
15-12 DfiFreqXlatVal7	The sequencer start vector used when dfi_freq value is 7.
11-8 DfiFreqXlatVal6	The sequencer start vector used when dfi_freq value is 6.
7-4 DfiFreqXlatVal5	The sequencer start vector used when dfi_freq value is 5.
3-0 DfiFreqXlatVal4	The sequencer start vector used when dfi_freq value is 4.

9.3.3.6.106 DFI Frequency Translation Register 2 (DfiFreqXlat2)

9.3.3.6.106.1 Offset

Register	Offset
DfiFreqXlat2	1E4h

9.3.3.6.106.2 Diagram



9.3.3.6.106.3 Fields

Field	Function
15-12 DfiFreqXlatVal11	The sequencer start vector used when dfi_freq value is 11.

Table continues on the next page...

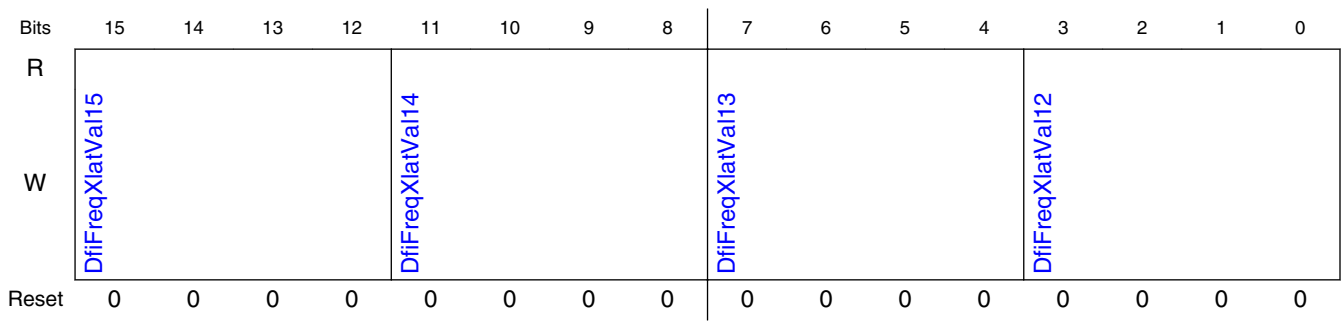
Field	Function
11-8 DfiFreqXlatVal10	The sequencer start vector used when dfi_freq value is 10.
7-4 DfiFreqXlatVal9	The sequencer start vector used when dfi_freq value is 9.
3-0 DfiFreqXlatVal8	The sequencer start vector used when dfi_freq value is 8.

9.3.3.6.107 DFI Frequency Translation Register 3 (DfiFreqXlat3)

9.3.3.6.107.1 Offset

Register	Offset
DfiFreqXlat3	1E6h

9.3.3.6.107.2 Diagram



9.3.3.6.107.3 Fields

Field	Function
15-12 DfiFreqXlatVal15	The sequencer start vector used when dfi_freq value is 15.
11-8 DfiFreqXlatVal14	The sequencer start vector used when dfi_freq value is 14.
7-4	The sequencer start vector used when dfi_freq value is 13.

Table continues on the next page...

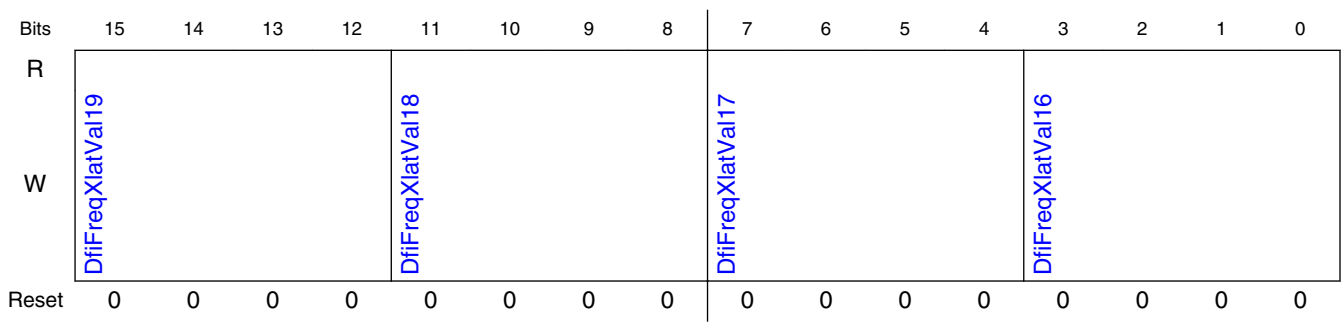
Field	Function
DfiFreqXlatVal13	
3-0 DfiFreqXlatVal12	The sequencer start vector used when dfi_freq value is 12.

9.3.3.6.108 DFI Frequency Translation Register 4 (DfiFreqXlat4)

9.3.3.6.108.1 Offset

Register	Offset
DfiFreqXlat4	1E8h

9.3.3.6.108.2 Diagram



9.3.3.6.108.3 Fields

Field	Function
15-12 DfiFreqXlatVal19	The sequencer start vector used when dfi_freq value is 19.
11-8 DfiFreqXlatVal18	The sequencer start vector used when dfi_freq value is 18.
7-4 DfiFreqXlatVal17	The sequencer start vector used when dfi_freq value is 17.
3-0 DfiFreqXlatVal16	The sequencer start vector used when dfi_freq value is 16.

9.3.3.6.109 DFI Frequency Translation Register 5 (DfiFreqXlat5)

9.3.3.6.109.1 Offset

Register	Offset
DfiFreqXlat5	1EAh

9.3.3.6.109.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DfiFreqXlatVal23				DfiFreqXlatVal22				DfiFreqXlatVal21				DfiFreqXlatVal20			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.109.3 Fields

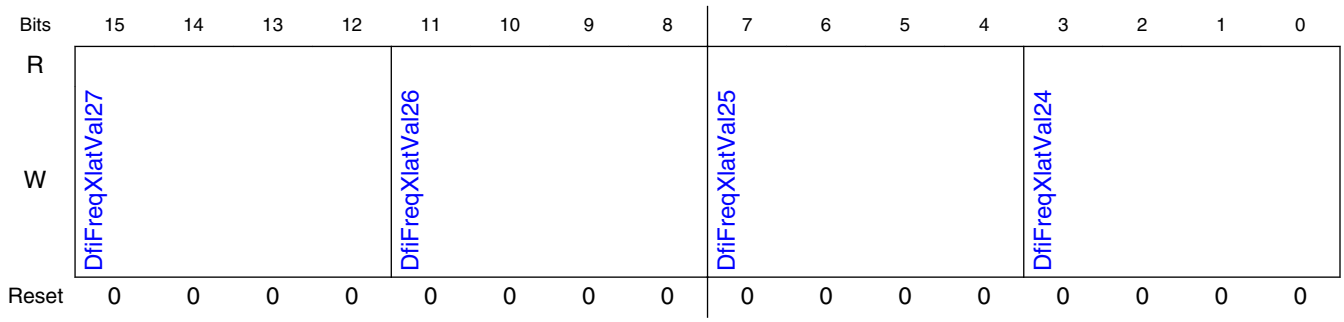
Field	Function
15-12 DfiFreqXlatVal23	The sequencer start vector used when dfi_freq value is 23.
11-8 DfiFreqXlatVal22	The sequencer start vector used when dfi_freq value is 22.
7-4 DfiFreqXlatVal21	The sequencer start vector used when dfi_freq value is 21.
3-0 DfiFreqXlatVal20	The sequencer start vector used when dfi_freq value is 20.

9.3.3.6.110 DFI Frequency Translation Register 6 (DfiFreqXlat6)

9.3.3.6.110.1 Offset

Register	Offset
DfiFreqXlat6	1ECh

9.3.3.6.110.2 Diagram



9.3.3.6.110.3 Fields

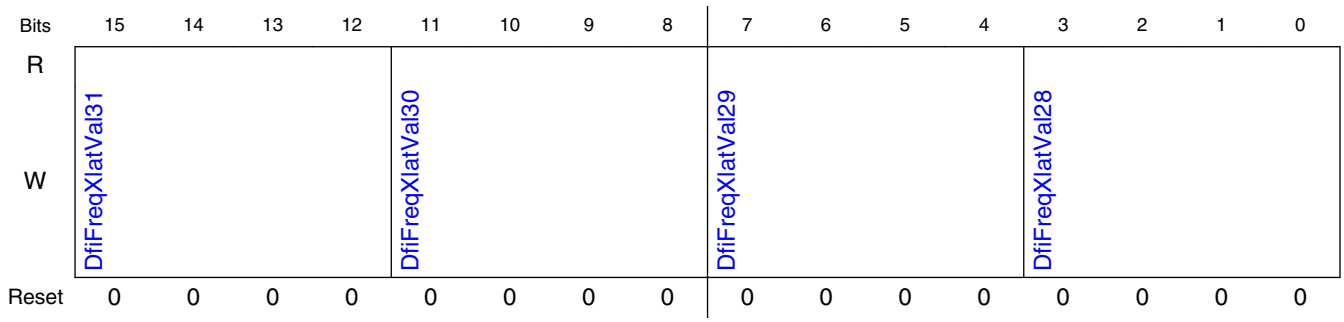
Field	Function
15-12 DfiFreqXlatVal27	The sequencer start vector used when dfi_freq value is 27.
11-8 DfiFreqXlatVal26	The sequencer start vector used when dfi_freq value is 26.
7-4 DfiFreqXlatVal25	The sequencer start vector used when dfi_freq value is 25.
3-0 DfiFreqXlatVal24	The sequencer start vector used when dfi_freq value is 24.

9.3.3.6.111 DFI Frequency Translation Register 7 (DfiFreqXlat7)

9.3.3.6.111.1 Offset

Register	Offset
DfiFreqXlat7	1EEh

9.3.3.6.111.2 Diagram



9.3.3.6.111.3 Fields

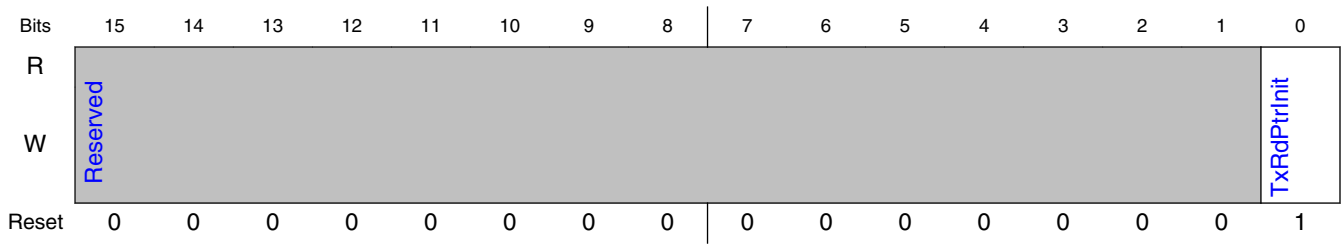
Field	Function
15-12 DfiFreqXlatVal31	The sequencer start vector used when dfi_freq value is 31.
11-8 DfiFreqXlatVal30	The sequencer start vector used when dfi_freq value is 30.
7-4 DfiFreqXlatVal29	The sequencer start vector used when dfi_freq value is 29.
3-0 DfiFreqXlatVal28	The sequencer start vector used when dfi_freq value is 28.

9.3.3.6.112 TxRdPtrInit control register (TxRdPtrInit)

9.3.3.6.112.1 Offset

Register	Offset
TxRdPtrInit	1F0h

9.3.3.6.112.2 Diagram



9.3.3.6.112.3 Fields

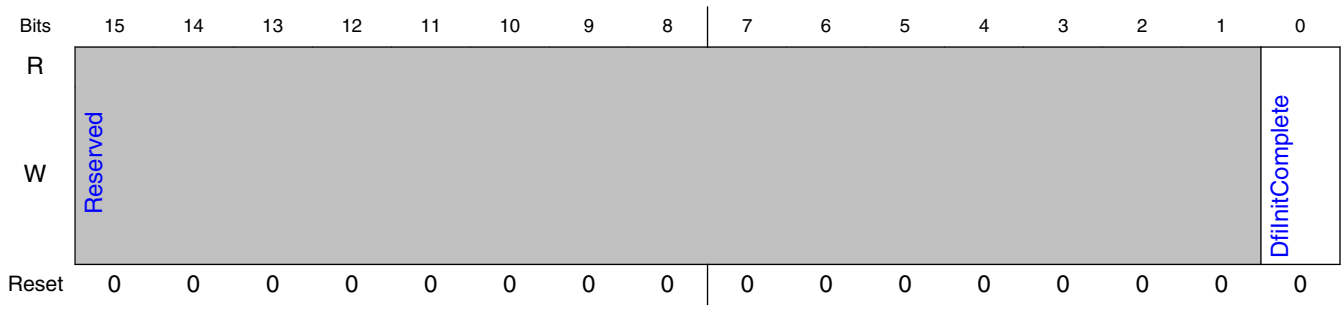
Field	Function
15-1 —	Reserved
0 TxRdPtrInit	This register directly controls TxRdPtrInit, and is meant to be written by the PState sequencer as part of the power state switching sequence. This register directly controls TxRdPtrInit, and is meant to be written by the PState sequencer as part of the power state switching sequence. It should be written to a 1 once the transition to LP2 is complete and 32 cycles before the de-asserting of init_complete. Likewise it should be written to 0 once the transition to an active PState is complete and 32 cycles before the assertion of init_complete. It should reset to a 1 on both warm and cold reset.

9.3.3.6.113 DFI Init Complete control (DfiInitComplete)

9.3.3.6.113.1 Offset

Register	Offset
DfiInitComplete	1F2h

9.3.3.6.113.2 Diagram



9.3.3.6.113.3 Fields

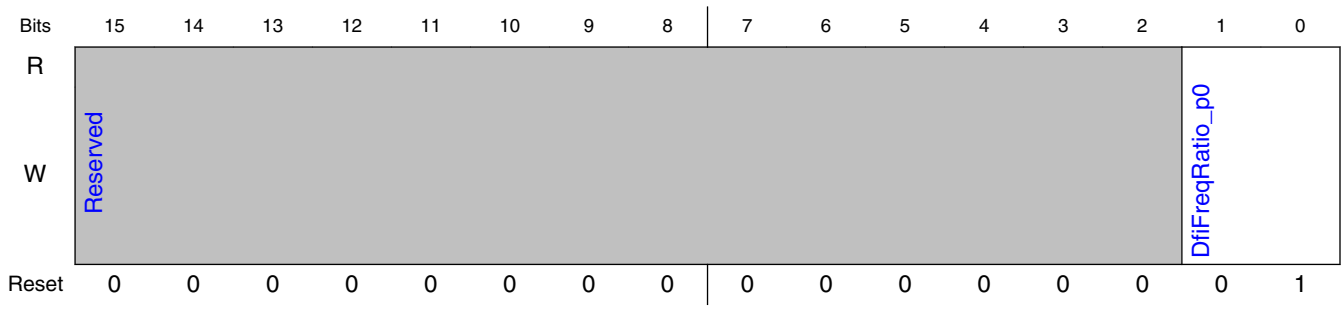
Field	Function
15-1 —	Reserved
0 DfiInitComplete	This register directly controls DfiInitComplete, and is meant to be written by the PState sequencer as part of the power state switching sequence. This register directly controls DfiInitComplete, and is meant to be written by the PState sequencer as part of the power state switching sequence. It should be written to a 1 once the transition to LP2 is complete. Likewise it should be written to 0 once the transition to an active and PState is complete. It should reset to a 0 on both warm and cold reset.

9.3.3.6.114 DFI Frequency Ratio (DfiFreqRatio_p0)

9.3.3.6.114.1 Offset

Register	Offset
DfiFreqRatio_p0	1F4h

9.3.3.6.114.2 Diagram



9.3.3.6.114.3 Fields

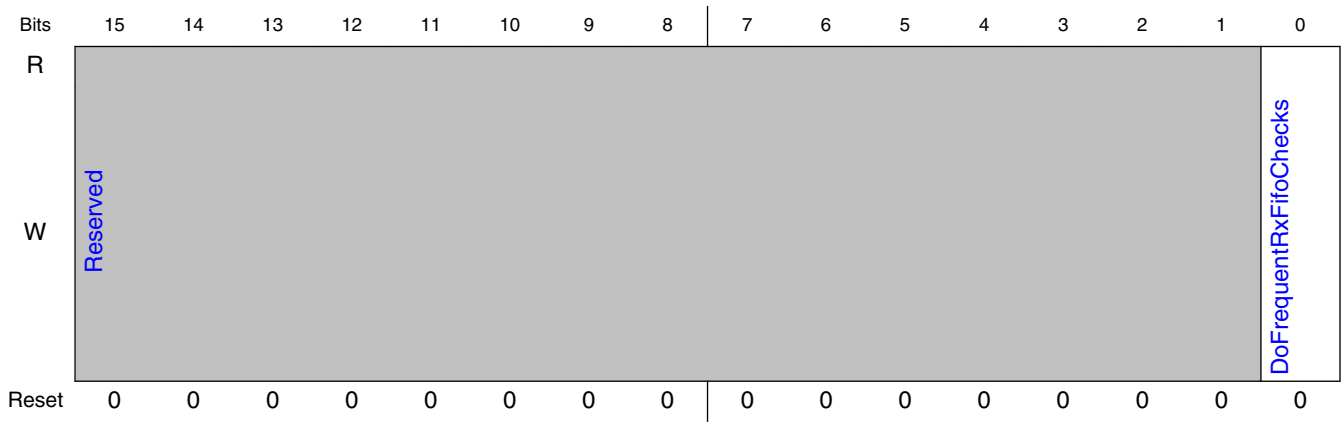
Field	Function
15-2 —	Reserved
1-0 DfiFreqRatio_p0	Used in dwc_ddrphy_pub_serdes to serialize or de-serialize DFI signals 00 = 1:1 mode 01 = 1:2 mode 1x = 1:4 mode* *Note: 1:4 is for future pub revision.

9.3.3.6.115 Enable more frequent consistency checks of the RX FIFOs (RxFifoChecks)

9.3.3.6.115.1 Offset

Register	Offset
RxFifoChecks	1F6h

9.3.3.6.115.2 Diagram



9.3.3.6.115.3 Fields

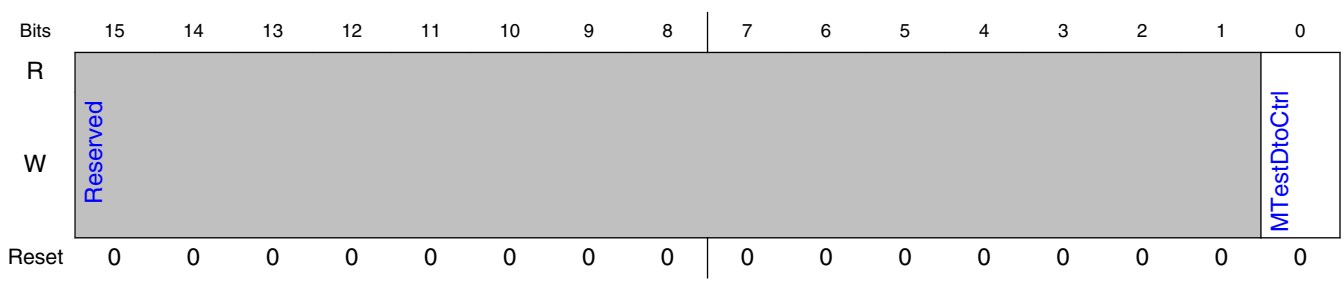
Field	Function
15-1 —	Reserved
0 DoFrequentRxFifoChecks	When 0, read data FIFO pointer consistency checks are performed only during sideband transactions (i. When 0, read data FIFO pointer consistency checks are performed only during sideband transactions (i.e., phyupd, ctrlupd, lp_data, phymstr) or when dfi_init_start is asserted. This option relies upon intentional quiescing of the DFI interface in order for the check to be performed. When 1, read data FIFO consistency checks are performed as when this bit is 0, and also at the beginning of each break in read data traffic from the DRAM. This options provides the lowest latency in reporting of a discrepancy in the read data FIFO pointer values.

9.3.3.6.116 (MTestDtoCtrl)

9.3.3.6.116.1 Offset

Register	Offset
MTestDtoCtrl	1FEh

9.3.3.6.116.2 Diagram



9.3.3.6.116.3 Fields

Field	Function
15-1 —	Reserved

Table continues on the next page...

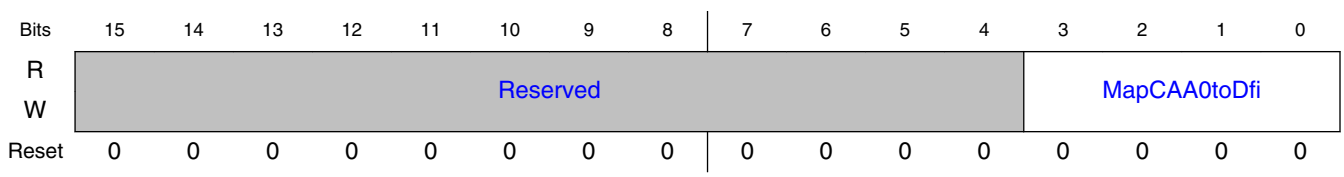
Field	Function
0 MTestDtoCtrl	MTESTdtoEn==[0], dwc_ddrphy_dto will be squelched (0) MTESTdtoEn==[1], dwc_ddrphy_dto will reflect the observability signal multiplexed on MTestCombo

9.3.3.6.117 Maps PHY CAA lane 0 from dfi0_address of the index of the register contents (MapCAA0toDfi)

9.3.3.6.117.1 Offset

Register	Offset
MapCAA0toDfi	200h

9.3.3.6.117.2 Diagram



9.3.3.6.117.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAA0toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 0. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 0. Unused for DDR3 and unused for DDR4 applications. Each register in the set of MapCAAtoDfi must have a unique value within the set. For example, if PHY CAA0 is mapped from dfi0_address[2], then Register MapCAA0toDfi should be 2. LP4 ACX3,ACX6,ACX10,ACX12 registers MapCAA0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAA6..9toDfi are reserved LP3 ACX3 not supported LP3 ACX6,ACX10,ACX12 registers MapCAA0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.118 Maps PHY CAA lane 1 from dfi0_address of the index of the register contents (MapCAA1toDfi)

9.3.3.6.118.1 Offset

Register	Offset
MapCAA1toDfi	202h

9.3.3.6.118.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												MapCAA1toDfi			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

9.3.3.6.118.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAA1toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 1. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 1. Unused for DDR3 and unused for DDR4 applications. Each register in the set of MapCAAtoDfi must have a unique value within the set. For example, if PHY CAA1 is mapped from dfi0_address[2], then Register MapCAA1toDfi should be 2. LP4 ACX3,ACX6,ACX10,ACX12 registers MapCAA0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAA6..9toDfi are reserved LP3 ACX3 not supported LP3 ACX6,ACX10,ACX12 registers MapCAA0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.119 Maps PHY CAA lane 2 from dfi0_address of the index of the register contents (MapCAA2toDfi)

9.3.3.6.119.1 Offset

Register	Offset
MapCAA2toDfi	204h

9.3.3.6.119.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												MapCAA2toDfi			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

9.3.3.6.119.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAA2toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 2. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 2. Unused for DDR3 and unused for DDR4 applications. Each register in the set of MapCAAtoDfi must have a unique value within the set. For example, if PHY CAA2 is mapped from dfi0_address[2], then Register MapCAA2toDfi should be 2. LP4 ACX3,ACX6,ACX10,ACX12 registers MapCAA0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAA6..9toDfi are reserved LP3 ACX3 not supported LP3 ACX6,ACX10,ACX12 registers MapCAA0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.120 Maps PHY CAA lane 3 from dfi0_address of the index of the register contents (MapCAA3toDfi)

9.3.3.6.120.1 Offset

Register	Offset
MapCAA3toDfi	206h

9.3.3.6.120.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												MapCAA3toDfi			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

9.3.3.6.120.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAA3toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 3. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 3. Unused for DDR3 and unused for DDR4 applications. Each register in the set of MapCAAtoDfi must have a unique value within the set. For example, if PHY CAA3 is mapped from dfi0_address[2], then Register MapCAA3toDfi should be 2. LP4 ACX3,ACX6,ACX10,ACX12 registers MapCAA0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAA6..9toDfi are reserved LP3 ACX3 not supported LP3 ACX6,ACX10,ACX12 registers MapCAA0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.121 Maps PHY CAA lane 4 from dfi0_address of the index of the register contents (MapCAA4toDfi)

9.3.3.6.121.1 Offset

Register	Offset
MapCAA4toDfi	208h

9.3.3.6.121.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												MapCAA4toDfi			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

9.3.3.6.121.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAA4toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 4. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 4. Unused for DDR3 and unused for DDR4 applications. Each register in the set of MapCAAtoDfi must have a unique value within the set. For example, if PHY CAA4 is mapped from dfi0_address[2], then Register MapCAA4toDfi should be 2. LP4 ACX3,ACX6,ACX10,ACX12 registers MapCAA0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAA6..9toDfi are reserved LP3 ACX3 not supported LP3 ACX6,ACX10,ACX12 registers MapCAA0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.122 Maps PHY CAA lane 5 from dfi0_address of the index of the register contents (MapCAA5toDfi)

9.3.3.6.122.1 Offset

Register	Offset
MapCAA5toDfi	20Ah

9.3.3.6.122.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												MapCAA5toDfi			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

9.3.3.6.122.3 Fields

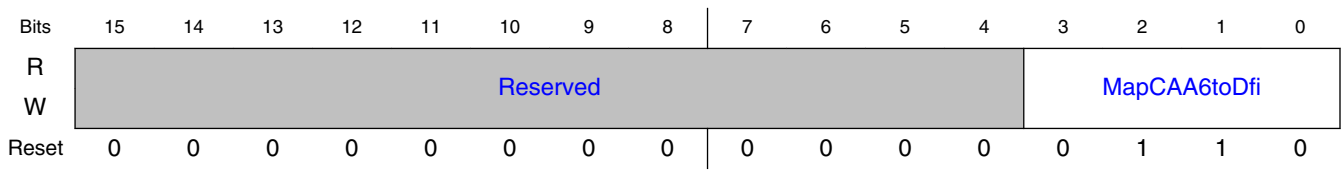
Field	Function
15-4 —	Reserved
3-0 MapCAA5toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 5. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 5. Unused for DDR3 and unused for DDR4 applications. Each register in the set of MapCAAtoDfi must have a unique value within the set. For example, if PHY CAA5 is mapped from dfi0_address[2], then Register MapCAA5toDfi should be 2. LP4 ACX3,ACX6,ACX10,ACX12 registers MapCAA0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAA6..9toDfi are reserved LP3 ACX3 not supported LP3 ACX6,ACX10,ACX12 registers MapCAA0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.123 Maps PHY CAA lane 6 from dfi0_address of the index of the register contents (MapCAA6toDfi)

9.3.3.6.123.1 Offset

Register	Offset
MapCAA6toDfi	20Ch

9.3.3.6.123.2 Diagram



9.3.3.6.123.3 Fields

Field	Function
15-4	Reserved

Table continues on the next page...

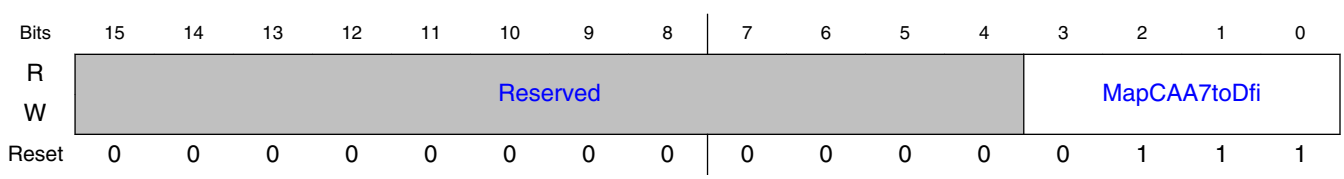
Field	Function
—	
3-0 MapCAA6toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 6. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 6. Unused for DDR3 and unused for DDR4 applications. Each register in the set of MapCAAtoDfi must have a unique value within the set. For example, if PHY CAA6 is mapped from dfi0_address[2], then Register MapCAA6toDfi should be 2. LP4 ACX3,ACX6,ACX10,ACX12 registers MapCAA0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAA6..9toDfi are reserved LP3 ACX3 not supported LP3 ACX6,ACX10,ACX12 registers MapCAA0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.124 Maps PHY CAA lane 7 from dfi0_address of the index of the register contents (MapCAA7toDfi)

9.3.3.6.124.1 Offset

Register	Offset
MapCAA7toDfi	20Eh

9.3.3.6.124.2 Diagram



9.3.3.6.124.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAA7toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 7. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 7.

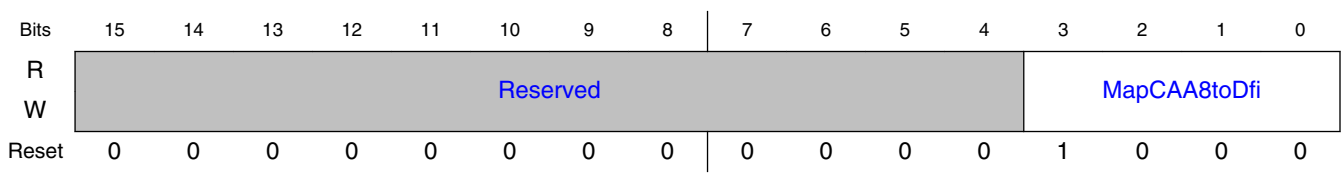
Field	Function
	Unused for DDR3 and unused for DDR4 applications. Each register in the set of MapCAAtoDfi must have a unique value within the set. For example, if PHY CAA7 is mapped from dfi0_address[2], then Register MapCAA7toDfi should be 2. LP4 ACX3,ACX6,ACX10,ACX12 registers MapCAA0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAA6..9toDfi are reserved LP3 ACX3 not supported LP3 ACX6,ACX10,ACX12 registers MapCAA0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.125 Maps PHY CAA lane 8 from dfi0_address of the index of the register contents (MapCAA8toDfi)

9.3.3.6.125.1 Offset

Register	Offset
MapCAA8toDfi	210h

9.3.3.6.125.2 Diagram



9.3.3.6.125.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAA8toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 8. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 8. Unused for DDR3 and unused for DDR4 applications. Each register in the set of MapCAAtoDfi must have a unique value within the set. For example, if PHY CAA8 is mapped from dfi0_address[2], then Register MapCAA8toDfi should be 2.

Field	Function
	LP4 ACX3,ACX6,ACX10,ACX12 registers MapCAA0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAA6..9toDfi are reserved LP3 ACX3 not supported LP3 ACX6,ACX10,ACX12 registers MapCAA0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.126 Maps PHY CAA lane 9 from dfi0_address of the index of the register contents (MapCAA9toDfi)

9.3.3.6.126.1 Offset

Register	Offset
MapCAA9toDfi	212h

9.3.3.6.126.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												MapCAA9toDfi			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1

9.3.3.6.126.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAA9toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 9. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi0_address to CAA 9. Unused for DDR3 and unused for DDR4 applications. Each register in the set of MapCAAtoDfi must have a unique value within the set. For example, if PHY CAA9 is mapped from dfi0_address[2], then Register MapCAA9toDfi should be 2. LP4 ACX3,ACX6,ACX10,ACX12 registers MapCAA0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAA6..9toDfi are reserved

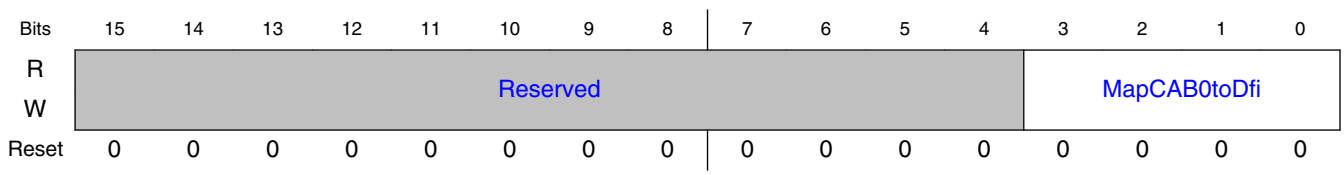
Field	Function
	LP3 ACX3 not supported LP3 ACX6,ACX10,ACX12 registers MapCAA0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.127 Maps PHY CAB lane 0 from dfi1_address of the index of the register contents (MapCAB0toDfi)

9.3.3.6.127.1 Offset

Register	Offset
MapCAB0toDfi	220h

9.3.3.6.127.2 Diagram



9.3.3.6.127.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAB0toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 0. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 0. Unused for DDR3 and unused for DDR4 applications. Unused for the 3-ANIB configuration. Each register in the set of MapCABtoDfi must have a unique value within the set. For example, if PHY CAB0 is mapped from dfi1_address[2], then Register MapCAB0toDfi should be 2. LP4 ACX3 registers MapCAB0..9toDfi are reserved, second channel not supportd LP4 ACX6,ACX10,ACX12 registers MapCAB0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAB6..9toDfi are reserved LP3 ACX3 not supported

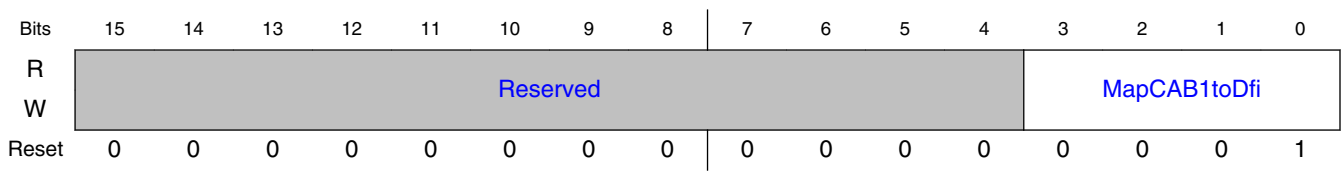
Field	Function
	LP3 ACX6 registers MapCAB0..9 are unused, second channel not supported LP3 ACX10,ACX12 registers MapCAB0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.128 Maps PHY CAB lane 1 from dfi1_address of the index of the register contents (MapCAB1toDfi)

9.3.3.6.128.1 Offset

Register	Offset
MapCAB1toDfi	222h

9.3.3.6.128.2 Diagram



9.3.3.6.128.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAB1toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 1. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 1. Unused for DDR3 and unused for DDR4 applications. Unused for the 3-ANIB configuration. Each register in the set of MapCABtoDfi must have a unique value within the set. For example, if PHY CAB1 is mapped from dfi1_address[2], then Register MapCAB1toDfi should be 2. LP4 ACX3 registers MapCAB0..9toDfi are reserved, second channel not supportd LP4 ACX6,ACX10,ACX12 registers MapCAB0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAB6..9toDfi are reserved LP3 ACX3 not supported

Field	Function
	LP3 ACX6 registers MapCAB0..9 are unused, second channel not supported LP3 ACX10,ACX12 registers MapCAB0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.129 Maps PHY CAB lane 2 from dfi1_address of the index of the register contents (MapCAB2toDfi)

9.3.3.6.129.1 Offset

Register	Offset
MapCAB2toDfi	224h

9.3.3.6.129.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												MapCAB2toDfi			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

9.3.3.6.129.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAB2toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 2. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 2. Unused for DDR3 and unused for DDR4 applications. Unused for the 3-ANIB configuration. Each register in the set of MapCABtoDfi must have a unique value within the set. For example, if PHY CAB2 is mapped from dfi1_address[2], then Register MapCAB2toDfi should be 2. LP4 ACX3 registers MapCAB0..9toDfi are reserved, second channel not supportd LP4 ACX6,ACX10,ACX12 registers MapCAB0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAB6..9toDfi are reserved LP3 ACX3 not supported

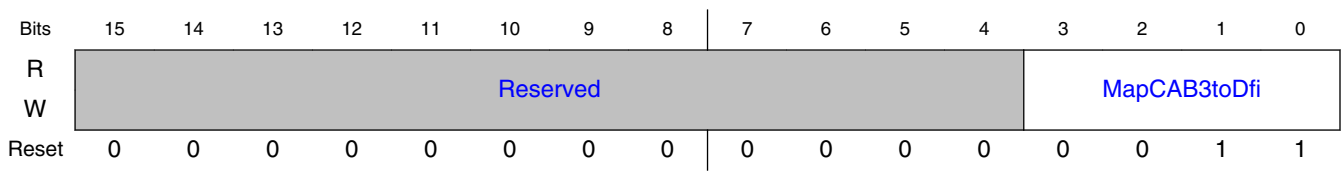
Field	Function
	LP3 ACX6 registers MapCAB0..9 are unused, second channel not supported LP3 ACX10,ACX12 registers MapCAB0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.130 Maps PHY CAB lane 3 from dfi1_address of the index of the register contents (MapCAB3toDfi)

9.3.3.6.130.1 Offset

Register	Offset
MapCAB3toDfi	226h

9.3.3.6.130.2 Diagram



9.3.3.6.130.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAB3toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 3. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 3. Unused for DDR3 and unused for DDR4 applications. Unused for the 3-ANIB configuration. Each register in the set of MapCABtoDfi must have a unique value within the set. For example, if PHY CAB3 is mapped from dfi1_address[2], then Register MapCAB3toDfi should be 2. LP4 ACX3 registers MapCAB0..9toDfi are reserved, second channel not supportd LP4 ACX6,ACX10,ACX12 registers MapCAB0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAB6..9toDfi are reserved LP3 ACX3 not supported

Field	Function
	LP3 ACX6 registers MapCAB0..9 are unused, second channel not supported LP3 ACX10,ACX12 registers MapCAB0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.131 Maps PHY CAB lane 4 from dfi1_address of the index of the register contents (MapCAB4toDfi)

9.3.3.6.131.1 Offset

Register	Offset
MapCAB4toDfi	228h

9.3.3.6.131.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												MapCAB4toDfi			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

9.3.3.6.131.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAB4toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 4. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 4. Unused for DDR3 and unused for DDR4 applications. Unused for the 3-ANIB configuration. Each register in the set of MapCABtoDfi must have a unique value within the set. For example, if PHY CAB4 is mapped from dfi1_address[2], then Register MapCAB4toDfi should be 2. LP4 ACX3 registers MapCAB0..9toDfi are reserved, second channel not supportd LP4 ACX6,ACX10,ACX12 registers MapCAB0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAB6..9toDfi are reserved LP3 ACX3 not supported

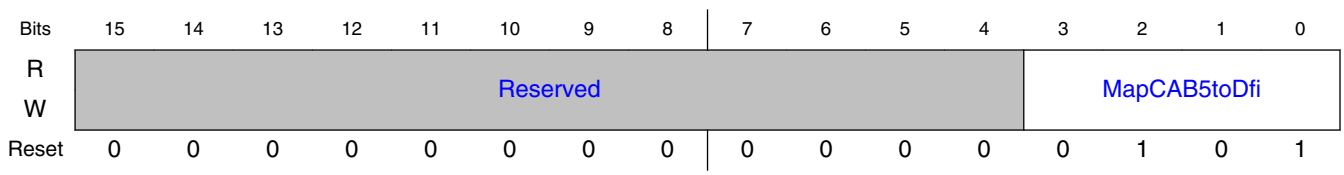
Field	Function
	LP3 ACX6 registers MapCAB0..9 are unused, second channel not supported LP3 ACX10,ACX12 registers MapCAB0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.132 Maps PHY CAB lane 5 from dfi1_address of the index of the register contents (MapCAB5toDfi)

9.3.3.6.132.1 Offset

Register	Offset
MapCAB5toDfi	22Ah

9.3.3.6.132.2 Diagram



9.3.3.6.132.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAB5toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 5. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 5. Unused for DDR3 and unused for DDR4 applications. Unused for the 3-ANIB configuration. Each register in the set of MapCABtoDfi must have a unique value within the set. For example, if PHY CAB5 is mapped from dfi1_address[2], then Register MapCAB5toDfi should be 2. LP4 ACX3 registers MapCAB0..9toDfi are reserved, second channel not supportd LP4 ACX6,ACX10,ACX12 registers MapCAB0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAB6..9toDfi are reserved LP3 ACX3 not supported

Field	Function
	LP3 ACX6 registers MapCAB0..9 are unused, second channel not supported LP3 ACX10,ACX12 registers MapCAB0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.133 Maps PHY CAB lane 6 from dfi1_address of the index of the register contents (MapCAB6toDfi)

9.3.3.6.133.1 Offset

Register	Offset
MapCAB6toDfi	22Ch

9.3.3.6.133.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												MapCAB6toDfi			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

9.3.3.6.133.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAB6toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 6. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 6. Unused for DDR3 and unused for DDR4 applications. Unused for the 3-ANIB configuration. Each register in the set of MapCABtoDfi must have a unique value within the set. For example, if PHY CAB6 is mapped from dfi1_address[2], then Register MapCAB6toDfi should be 2. LP4 ACX3 registers MapCAB0..9toDfi are reserved, second channel not supportd LP4 ACX6,ACX10,ACX12 registers MapCAB0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAB6..9toDfi are reserved LP3 ACX3 not supported

Field	Function
	LP3 ACX6 registers MapCAB0..9 are unused, second channel not supported LP3 ACX10,ACX12 registers MapCAB0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.134 Maps PHY CAB lane 7 from dfi1_address of the index of the register contents (MapCAB7toDfi)

9.3.3.6.134.1 Offset

Register	Offset
MapCAB7toDfi	22Eh

9.3.3.6.134.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												MapCAB7toDfi			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

9.3.3.6.134.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAB7toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 7. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 7. Unused for DDR3 and unused for DDR4 applications. Unused for the 3-ANIB configuration. Each register in the set of MapCABtoDfi must have a unique value within the set. For example, if PHY CAB7 is mapped from dfi1_address[2], then Register MapCAB7toDfi should be 2. LP4 ACX3 registers MapCAB0..9toDfi are reserved, second channel not supportd LP4 ACX6,ACX10,ACX12 registers MapCAB0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAB6..9toDfi are reserved LP3 ACX3 not supported

Field	Function
	LP3 ACX6 registers MapCAB0..9 are unused, second channel not supported LP3 ACX10,ACX12 registers MapCAB0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.135 Maps PHY CAB lane 8 from dfi1_address of the index of the register contents (MapCAB8toDfi)

9.3.3.6.135.1 Offset

Register	Offset
MapCAB8toDfi	230h

9.3.3.6.135.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												MapCAB8toDfi			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

9.3.3.6.135.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAB8toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 8. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 8. Unused for DDR3 and unused for DDR4 applications. Unused for the 3-ANIB configuration. Each register in the set of MapCABtoDfi must have a unique value within the set. For example, if PHY CAB8 is mapped from dfi1_address[2], then Register MapCAB8toDfi should be 2. LP4 ACX3 registers MapCAB0..9toDfi are reserved, second channel not supportd LP4 ACX6,ACX10,ACX12 registers MapCAB0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAB6..9toDfi are reserved LP3 ACX3 not supported

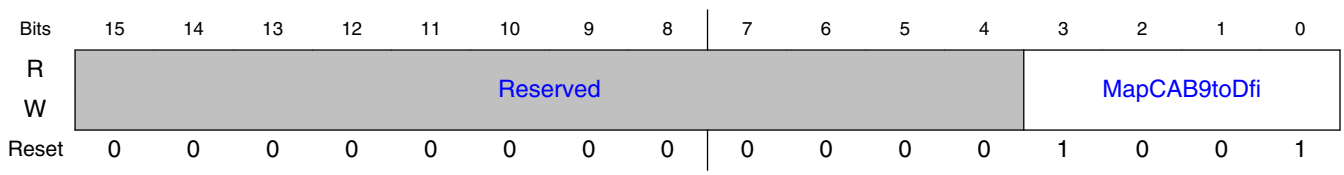
Field	Function
	LP3 ACX6 registers MapCAB0..9 are unused, second channel not supported LP3 ACX10,ACX12 registers MapCAB0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.136 Maps PHY CAB lane 9 from dfi1_address of the index of the register contents (MapCAB9toDfi)

9.3.3.6.136.1 Offset

Register	Offset
MapCAB9toDfi	232h

9.3.3.6.136.2 Diagram



9.3.3.6.136.3 Fields

Field	Function
15-4 —	Reserved
3-0 MapCAB9toDfi	For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 9. For LPDDR3 and LPDDR4 applications, these CSRs map a dfi1_address to CAB 9. Unused for DDR3 and unused for DDR4 applications. Unused for the 3-ANIB configuration. Each register in the set of MapCABtoDfi must have a unique value within the set. For example, if PHY CAB9 is mapped from dfi1_address[2], then Register MapCAB9toDfi should be 2. LP4 ACX3 registers MapCAB0..9toDfi are reserved, second channel not supportd LP4 ACX6,ACX10,ACX12 registers MapCAB0..5toDfi are used, values 0-5 are valid, values 6-15 are reserved registers MapCAB6..9toDfi are reserved LP3 ACX3 not supported

Field	Function
	LP3 ACX6 registers MapCAB0..9 are unused, second channel not supported LP3 ACX10,ACX12 registers MapCAB0..9toDfi are used, values 0-9 are valid, values 10-15 are reserved

9.3.3.6.137 Interrupt Enable Bits (PhyInterruptEnable)

9.3.3.6.137.1 Offset

Register	Offset
PhyInterruptEnable	236h

9.3.3.6.137.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PhyHWReservedEn					PhyRxFifoCheckEn	PhyVTDriftAlarmEn		PhyFWReservedEn					PhyTrngFailEn	PhyInitCmpltEn	PhyTrngCmpltEn
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.137.3 Fields

Field	Function
15-11 PhyHWReservedEn	Reserved
10 PhyRxFifoCheckEn	Enable for the RxFifo Pointers Check Interrupt 0 : Interrupt not enabled 1 : Interrupt enabled
9-8 PhyVTDriftAlarmEn	Enable for the PHY VT Drift Alarm interrupts. Enable for the PHY VT Drift Alarm interrupts. 0 : Interrupt not enabled 1 : Interrupt enabled
7-3	Reserved

Table continues on the next page...

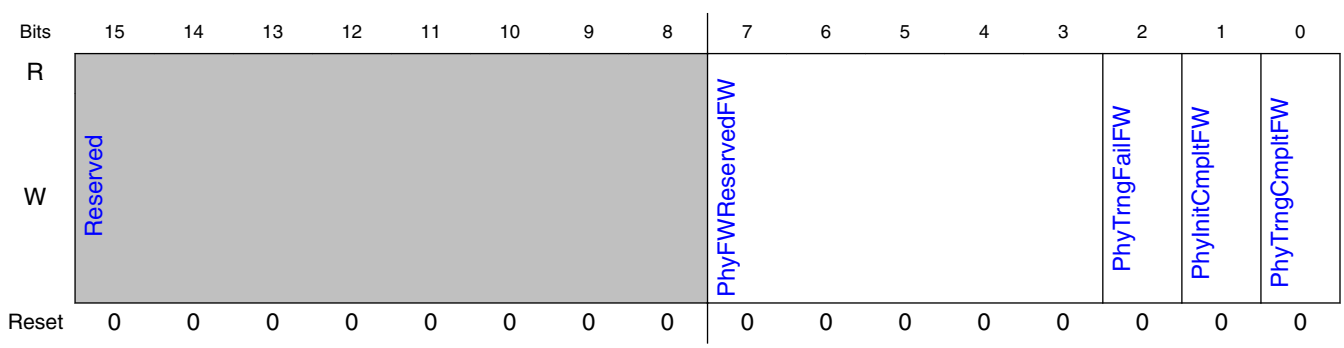
Field	Function
PhyFWReservedEn	
2 PhyTrngFailEn	Enable for the PHY Training Failure interrupt. Enable for the PHY Training Failure interrupt. 0 : Interrupt not enabled 1 : Interrupt enabled
1 PhyInitCmpltEn	Enable for the PHY Initialization Complete interrupt. Enable for the PHY Initialization Complete interrupt. 0 : Interrupt not enabled 1 : Interrupt enabled
0 PhyTrngCmpltEn	Enable for the PHY Training Complete interrupt. Enable for the PHY Training Complete interrupt. 0 : Interrupt not enabled 1 : Interrupt enabled

9.3.3.6.138 Interrupt Firmware Control Bits (PhyInterruptFWControl)

9.3.3.6.138.1 Offset

Register	Offset
PhyInterruptFWControl	238h

9.3.3.6.138.2 Diagram



9.3.3.6.138.3 Fields

Field	Function
15-8	Reserved

Table continues on the next page...

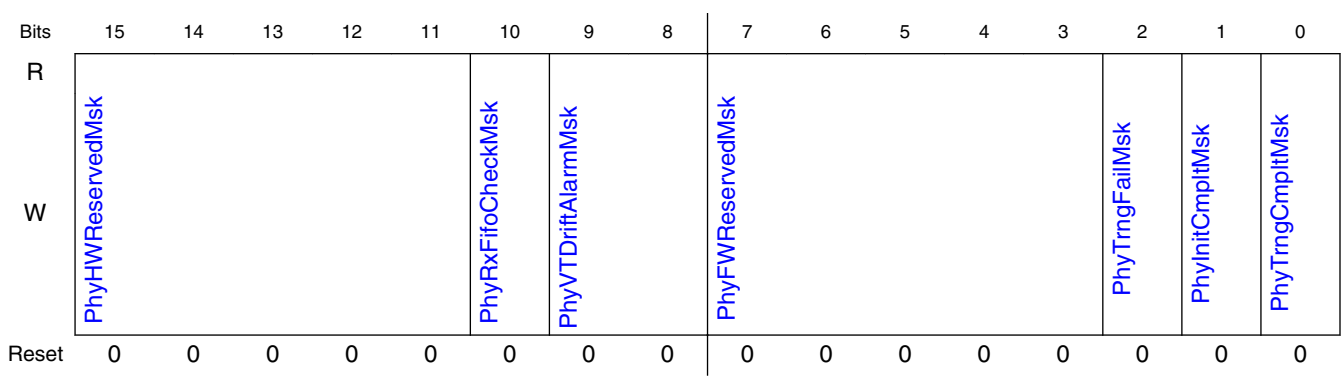
Field	Function
—	
7-3 PhyFWReservedFW	Reserved
2 PhyTrngFailFW	PHY Training Failure Firmware interrupt. PHY Training Failure Firmware interrupt. 0 : Interrupt not asserted 1 : Interrupt asserted
1 PhyInitCmpltFW	PHY Initialization Complete Firmware interrupt. PHY Initialization Complete Firmware interrupt. 0 : Interrupt not asserted 1 : Interrupt asserted
0 PhyTrngCmpltFW	PHY Training Complete Firmware interrupt. PHY Training Complete Firmware interrupt. 0 : Interrupt not asserted 1 : Interrupt asserted

9.3.3.6.139 Interrupt Mask Bits (PhyInterruptMask)

9.3.3.6.139.1 Offset

Register	Offset
PhyInterruptMask	23Ah

9.3.3.6.139.2 Diagram



9.3.3.6.139.3 Fields

Field	Function
15-11 PhyHWReservedMsk	Reserved
10 PhyRxFifoCheckMsk	Mask for the RxFifo Pointers Check Interrupt 0 : Interrupt not masked 1 : Interrupt masked
9-8 PhyVTDriftAlarmMsk	Mask for the PHY VT Drift Alarm interrupts. Mask for the PHY VT Drift Alarm interrupts. 0 : Interrupt not masked 1 : Interrupt masked
7-3 PhyFWReservedMsk	Reserved
2 PhyTrngFailMsk	Mask for the PHY Training Failure interrupt. Mask for the PHY Training Failure interrupt. 0 : Interrupt not masked 1 : Interrupt masked
1 PhyInitCmpltMsk	Mask for the PHY Initialization Complete interrupt. Mask for the PHY Initialization Complete interrupt. 0 : Interrupt not masked 1 : Interrupt masked
0 PhyTrngCmpltMsk	Mask for the PHY Training Complete interrupt. Mask for the PHY Training Complete interrupt. 0 : Interrupt not masked 1 : Interrupt masked

9.3.3.6.140 Interrupt Clear Bits (PhyInterruptClear)

9.3.3.6.140.1 Offset

Register	Offset
PhyInterruptClear	23Ch

9.3.3.6.140.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PhyHWReservedClr					PhyRxFifoCheckClr	PhyVTDriftAlarmClr		PhyFWReservedClr					PhyTrngFailClr	PhyInitCmpltClr	PhyTrngCmpltClr
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.140.3 Fields

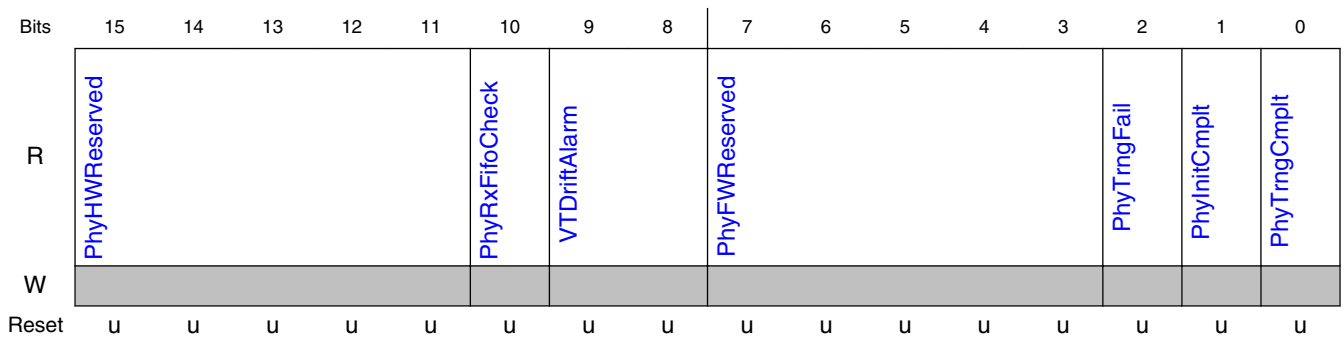
Field	Function
15-11 PhyHWReservedClr	Reserved
10 PhyRxFifoCheckClr	Clear for the RxFifo Pointers Check Interrupt 0 : Interrupt not affected 1 : Interrupt cleared
9-8 PhyVTDriftAlarmClr	Clear for the PHY VT Drift Alarm interrupt. Clear for the PHY VT Drift Alarm interrupt. 0 : Interrupt not affected 1 : Interrupt cleared
7-3 PhyFWReservedClr	Reserved
2 PhyTrngFailClr	Clear for the PHY Training Failure interrupt. Clear for the PHY Training Failure interrupt. 0 : Interrupt not affected 1 : Interrupt cleared
1 PhyInitCmpltClr	Clear for the PHY Initialization Complete interrupt. Clear for the PHY Initialization Complete interrupt. 0 : Interrupt not affected 1 : Interrupt cleared
0 PhyTrngCmpltClr	Clear for the PHY Training Complete interrupt. Clear for the PHY Training Complete interrupt. 0 : Interrupt not affected 1 : Interrupt cleared

9.3.3.6.141 Interrupt Status Bits (PhyInterruptStatus)

9.3.3.6.141.1 Offset

Register	Offset
PhyInterruptStatus	23Eh

9.3.3.6.141.2 Diagram



9.3.3.6.141.3 Fields

Field	Function
15-11 PhyHWReserved	Reserved
10 PhyRxFifoCheck	A mechanism in the PHY checks the Read Fifo pointers for consistency at times they are idle. A mechanism in the PHY checks the Read Fifo pointers for consistency at times they are idle. If a check fails, this interrupt is asserted, and detailed status is found in the RxFifoCheckStatus register. 0 : Interrupt not enabled and asserted 1 : Interrupt enabled and asserted
9-8 VTDriftAlarm	PHY VT Drift Alarm interrupt. PHY VT Drift Alarm interrupt. Assertion of this interrupt indicates the number of tap points in the master DLL has shifted by an amount greater than 2 * DfiPhyUpdIntThreshold. There is one bit for each master DLL and they are fully independent of one another. Please see the DfiPhyUpdIntThreshold CSR for additional information. 0 : Interrupt not enabled and asserted 1 : Interrupt enabled and asserted
7-3	Reserved

Table continues on the next page...

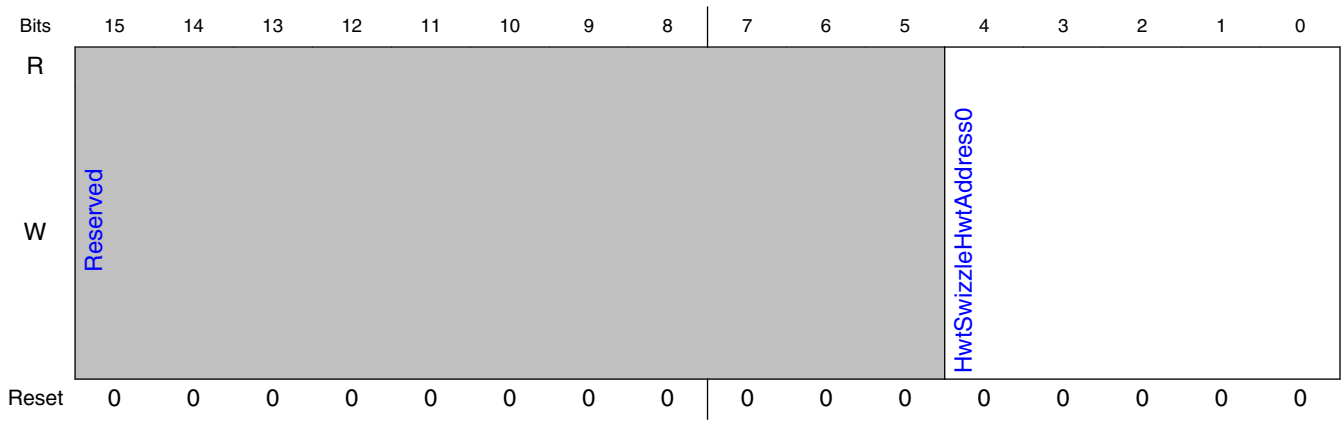
Field	Function
PhyFWReserved	
2 PhyTrngFail	PHY Training Failure interrupt. PHY Training Failure interrupt. Assertion of this interrupt indicates there was an issue during training which did not allow the PHY to successfully complete training. 0 : Interrupt not enabled and asserted 1 : Interrupt enabled and asserted
1 PhyInitCmplt	PHY Initialization Complete interrupt. PHY Initialization Complete interrupt. Assertion of this interrupt indicates that the PHY Initialization Engine has been fully loaded and the PHY is ready and able to assert dfi_init_complete following the assertion of dfi_init_start. It is possible this interrupt may not assert until after dfi_init_complete has asserted, depending on when dfi_init_start asserts. 0 : Interrupt not enabled and asserted 1 : Interrupt enabled and asserted
0 PhyTrngCmplt	PHY Training Complete interrupt. PHY Training Complete interrupt. Assertion of this interrupt indicates that the PHY has successfully trained the memory interface at all indicated frequencies and is ready for the PHY Initialization Engine to be loaded. 0 : Interrupt not enabled and asserted 1 : Interrupt enabled and asserted

9.3.3.6.142 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress0)

9.3.3.6.142.1 Offset

Register	Offset
HwtSwizzleHwtAddress0	240h

9.3.3.6.142.2 Diagram



9.3.3.6.142.3 Fields

Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress0	This set of registers is used in DDR3/DDR4 mode where a user has re-mapped the DFI inputs to the PHY. This set of registers is used in DDR3/DDR4 mode where a user has re-mapped the DFI inputs to the PHY. This register does not affect the mapping of those signals, or remap the tristate properties of the bumps. However, when the user remaps the DFI signals, the HWT inputs must be mapped in a similar fashion using this register. Each remappable hwt bit has a register associated with it. When the register value is zero, the bit is not re-mapped. When the register value is non-zero, the signal associated with the number in the table below is mapped to the signal associated with the register. So, for example, if you set the value for the register associated with hwt_act_n (HwtSwizzleHwtActN) to 27, then hwt_parity_in will be mapped to hwt_act_n. These registers should always be 0 for LPDDR3 and LPDDR4. Some registers are used only for DDR3 or DDR4. These are noted in the descriptions. Note: If the system uses DDR3-RDIMM or DDR4-RDIMM/LRDIMM, it is strongly recommended NOT to swizzle the PAR pin (parity). If the PAR pin is swizzled, the Dynamic Address Tristate feature must be disabled. (see the DisDynAdrTri field of the TristateMode CA Register). ----- Index: Signal

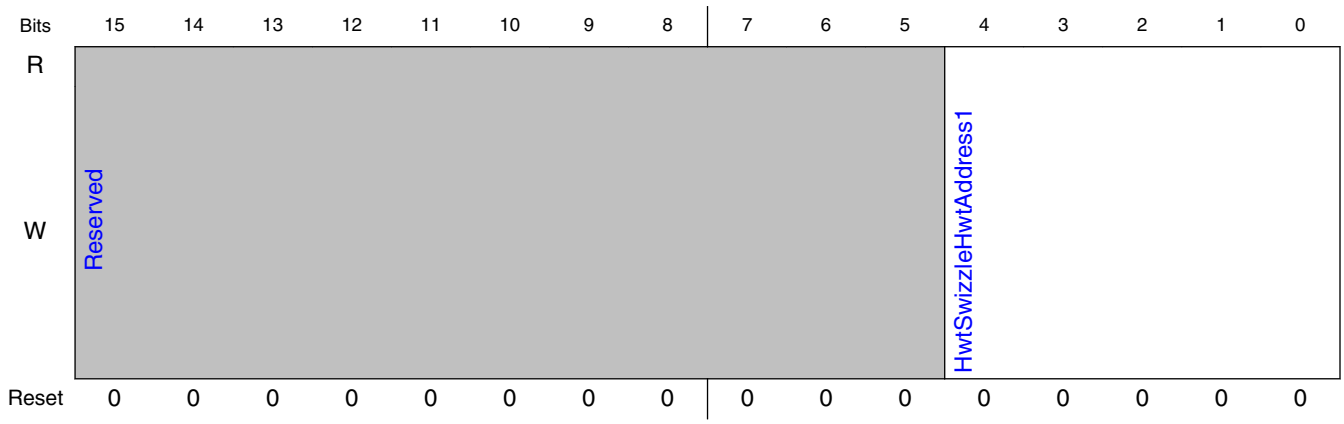
Field	Function
	0: No-remapping 1: hwt_address[0] 2: hwt_address[1] 3: hwt_address[2] 4: hwt_address[3] 5: hwt_address[4] 6: hwt_address[5] 7: hwt_address[6] 8: hwt_address[7] 9: hwt_address[8] 10: hwt_address[9] 11: hwt_address[10] 12: hwt_address[11] 13: hwt_address[12] 14: hwt_address[13] 15: hwt_address[14] (DDR3 Only) 16: hwt_address[15] (DDR3 Only) 17: hwt_address[17] (DDR4 Only) 18: hwt_act_n (DDR4 Only) 19: hwt_bank[0] 20: hwt_bank[1] 21: hwt_bank[2] (DDR3 Only) 22: hwt_bg[0] (DDR4 Only) 23: hwt_bg[1] (DDR4 Only) 24: hwt_cas_n 25: hwt_ras_n 26: hwt_we_n 27: hwt_parity_in

9.3.3.6.143 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress1)

9.3.3.6.143.1 Offset

Register	Offset
HwtSwizzleHwtAddress1	242h

9.3.3.6.143.2 Diagram



9.3.3.6.143.3 Fields

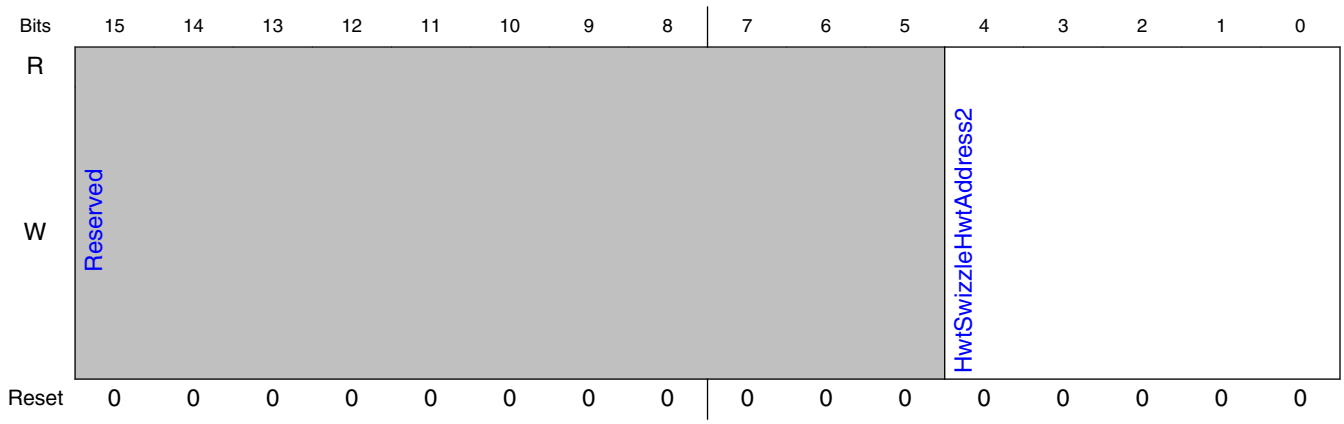
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress1	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.144 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress2)

9.3.3.6.144.1 Offset

Register	Offset
HwtSwizzleHwtAddress2	244h

9.3.3.6.144.2 Diagram



9.3.3.6.144.3 Fields

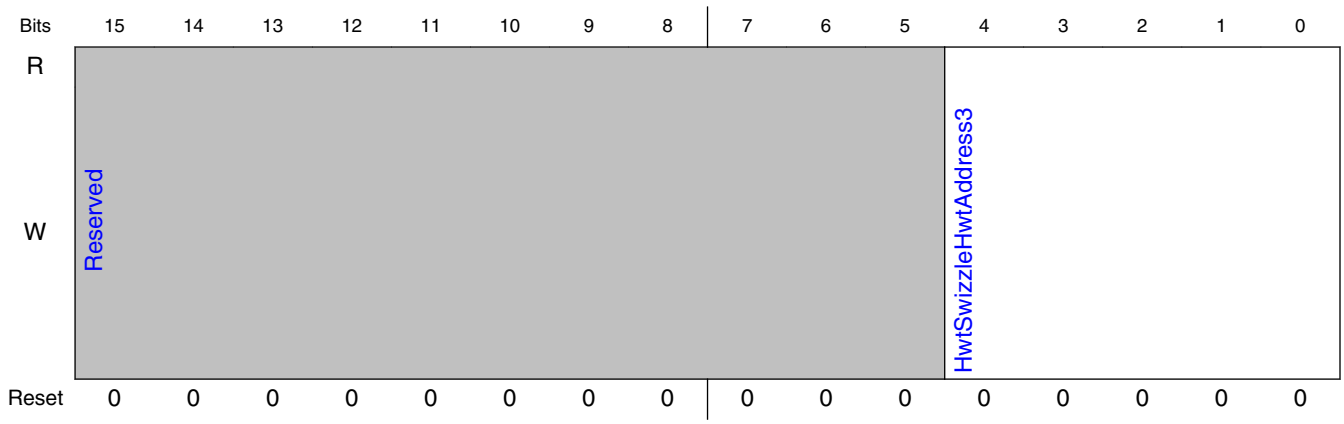
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress2	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.145 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress3)

9.3.3.6.145.1 Offset

Register	Offset
HwtSwizzleHwtAddress3	246h

9.3.3.6.145.2 Diagram



9.3.3.6.145.3 Fields

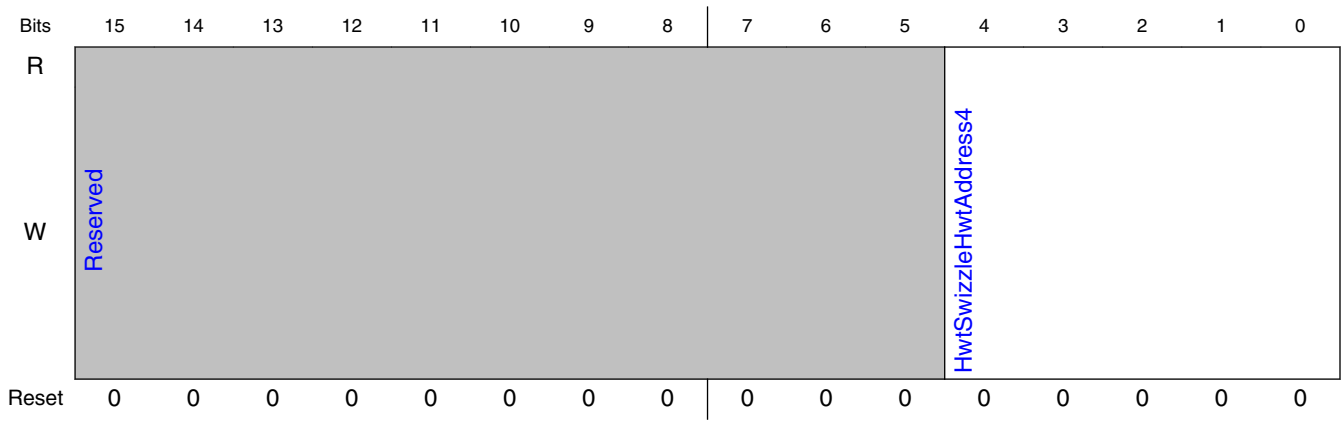
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress3	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.146 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress4)

9.3.3.6.146.1 Offset

Register	Offset
HwtSwizzleHwtAddress4	248h

9.3.3.6.146.2 Diagram



9.3.3.6.146.3 Fields

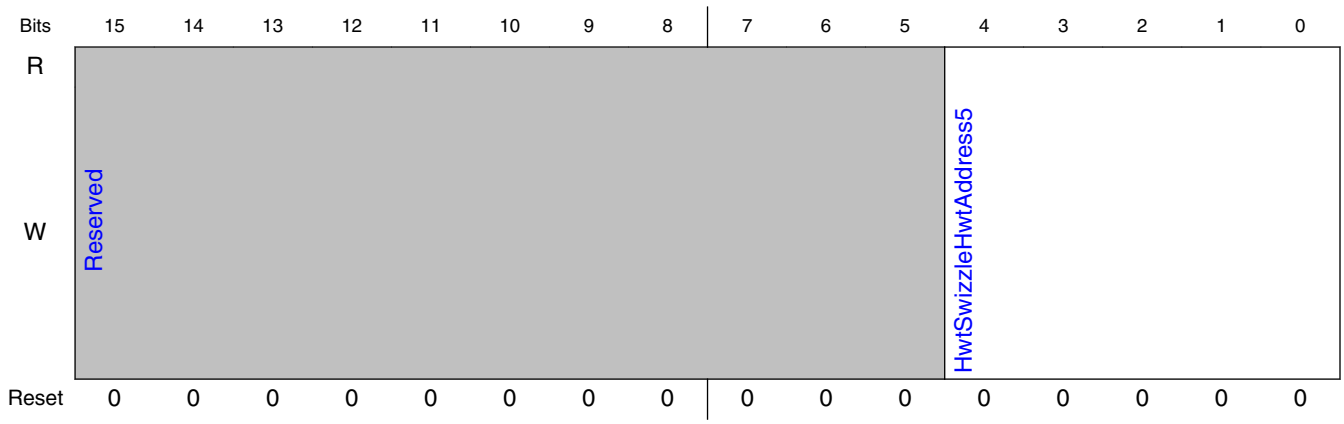
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress4	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.147 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress5)

9.3.3.6.147.1 Offset

Register	Offset
HwtSwizzleHwtAddress5	24Ah

9.3.3.6.147.2 Diagram



9.3.3.6.147.3 Fields

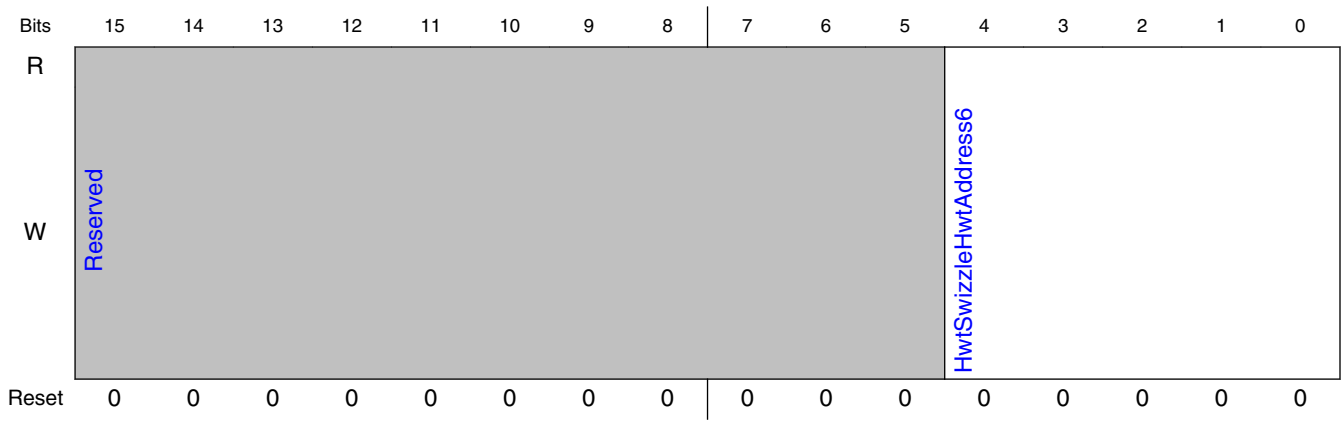
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress5	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.148 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress6)

9.3.3.6.148.1 Offset

Register	Offset
HwtSwizzleHwtAddress6	24Ch

9.3.3.6.148.2 Diagram



9.3.3.6.148.3 Fields

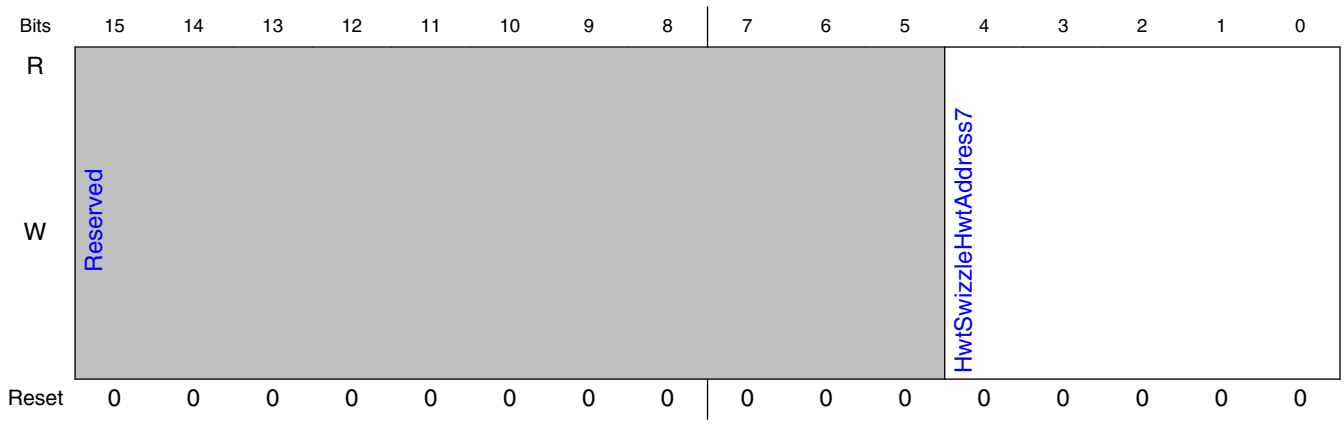
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress6	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.149 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress7)

9.3.3.6.149.1 Offset

Register	Offset
HwtSwizzleHwtAddress7	24Eh

9.3.3.6.149.2 Diagram



9.3.3.6.149.3 Fields

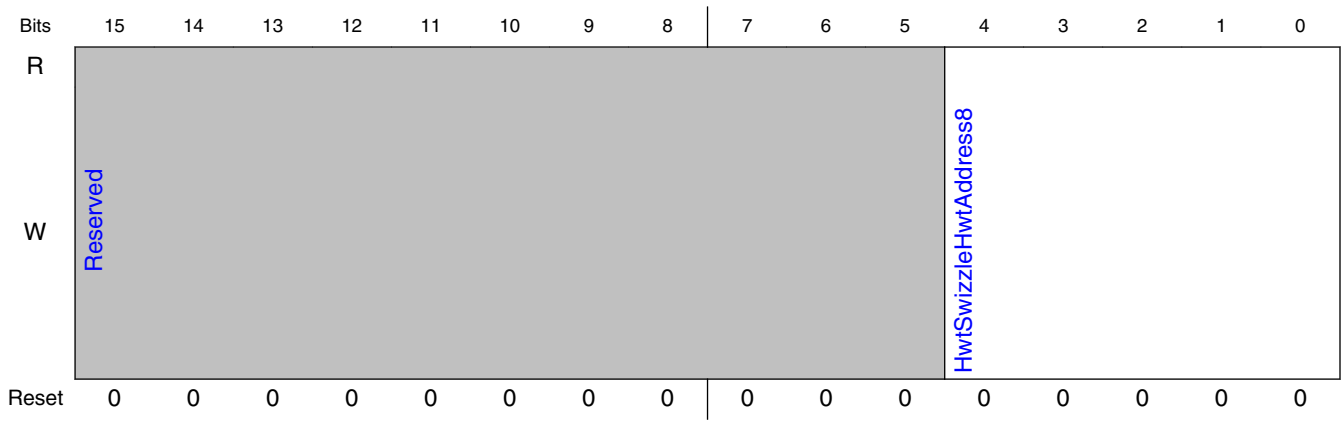
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress7	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.150 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress8)

9.3.3.6.150.1 Offset

Register	Offset
HwtSwizzleHwtAddress8	250h

9.3.3.6.150.2 Diagram



9.3.3.6.150.3 Fields

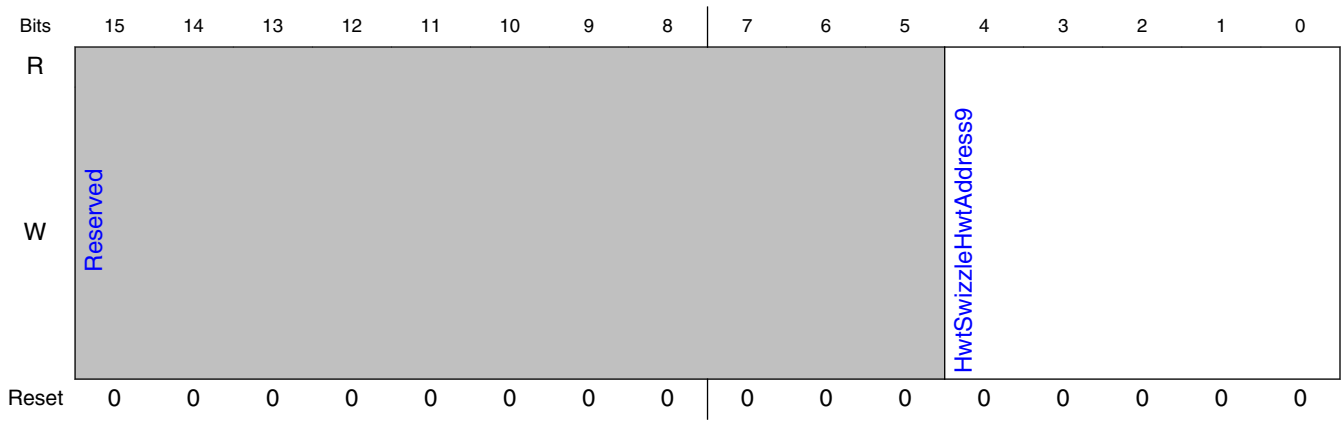
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress8	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.151 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress9)

9.3.3.6.151.1 Offset

Register	Offset
HwtSwizzleHwtAddress9	252h

9.3.3.6.151.2 Diagram



9.3.3.6.151.3 Fields

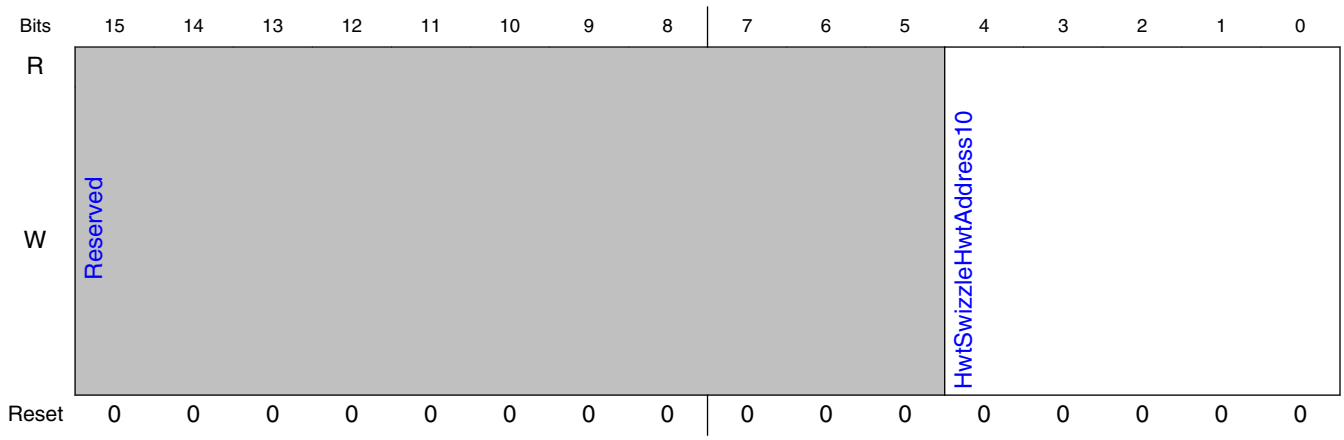
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress9	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.152 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress10)

9.3.3.6.152.1 Offset

Register	Offset
HwtSwizzleHwtAddress10	254h

9.3.3.6.152.2 Diagram



9.3.3.6.152.3 Fields

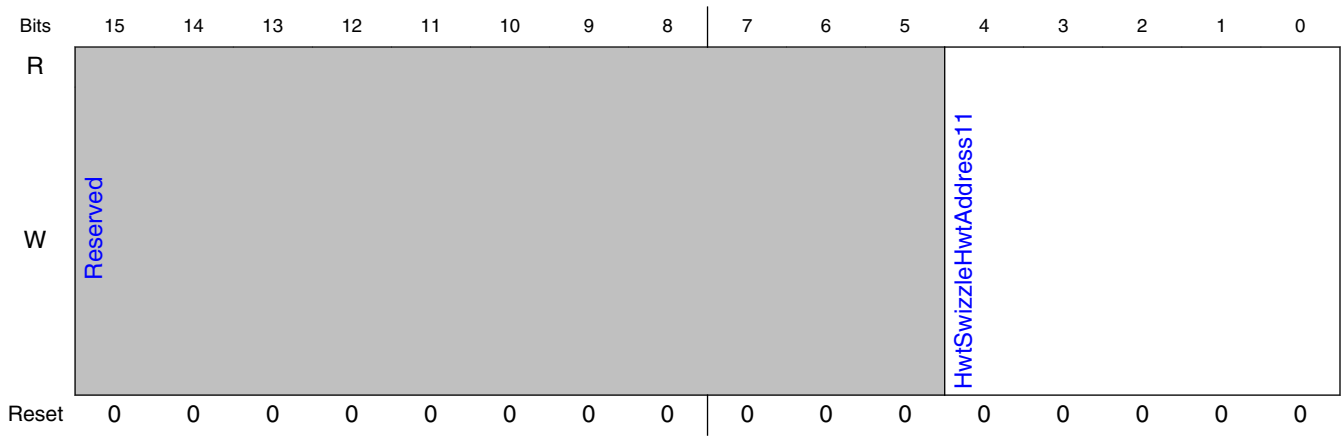
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress10	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.153 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress11)

9.3.3.6.153.1 Offset

Register	Offset
HwtSwizzleHwtAddress11	256h

9.3.3.6.153.2 Diagram



9.3.3.6.153.3 Fields

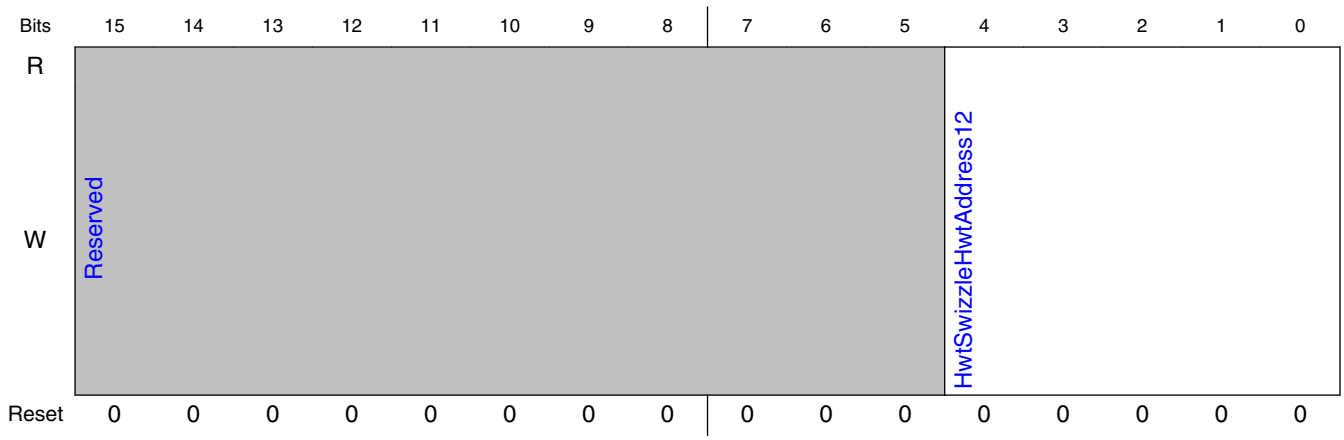
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress11	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.154 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress12)

9.3.3.6.154.1 Offset

Register	Offset
HwtSwizzleHwtAddress12	258h

9.3.3.6.154.2 Diagram



9.3.3.6.154.3 Fields

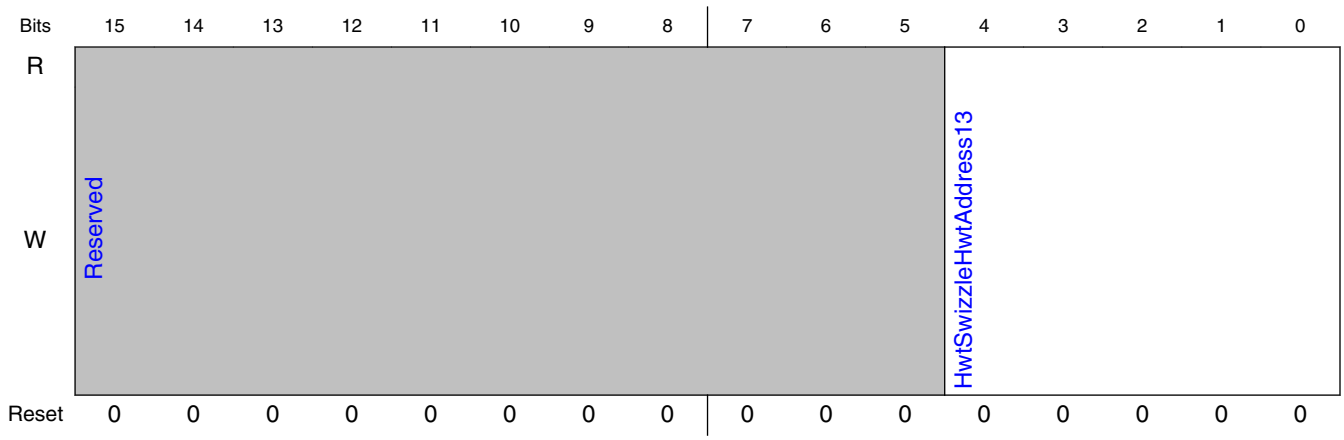
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress12	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.155 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress13)

9.3.3.6.155.1 Offset

Register	Offset
HwtSwizzleHwtAddress13	25Ah

9.3.3.6.155.2 Diagram



9.3.3.6.155.3 Fields

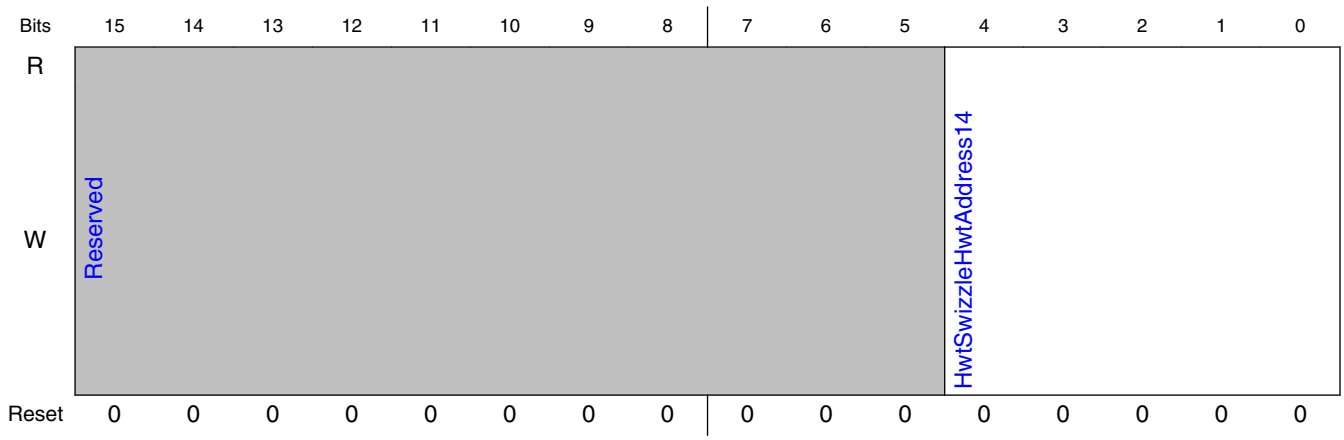
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress13	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.156 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress14)

9.3.3.6.156.1 Offset

Register	Offset
HwtSwizzleHwtAddress14	25Ch

9.3.3.6.156.2 Diagram



9.3.3.6.156.3 Fields

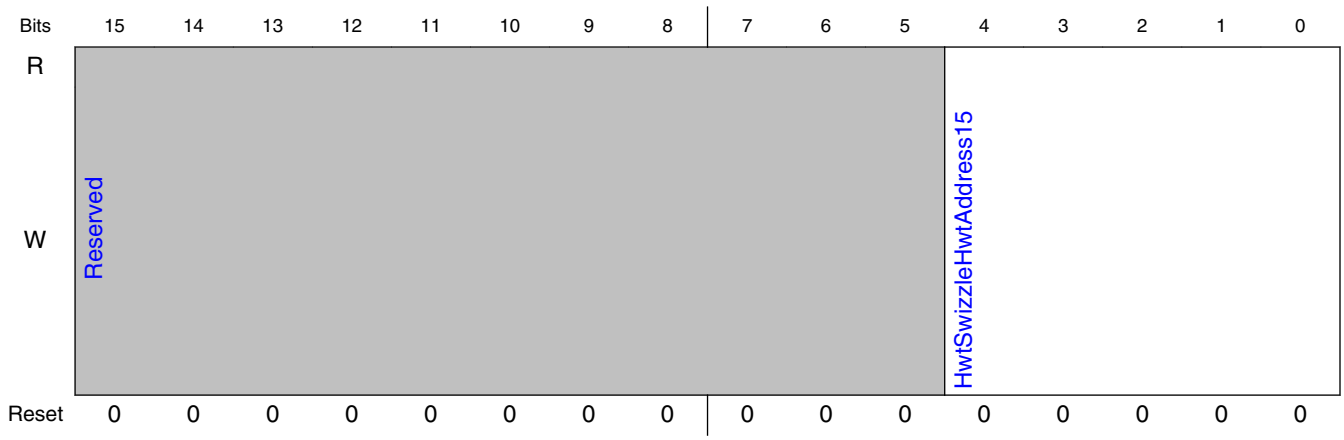
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress14	See Description of HwtSwizzleHwtAddress0 for details. See Description of HwtSwizzleHwtAddress0 for details. Used only for DDR3.

9.3.3.6.157 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress15)

9.3.3.6.157.1 Offset

Register	Offset
HwtSwizzleHwtAddress15	25Eh

9.3.3.6.157.2 Diagram



9.3.3.6.157.3 Fields

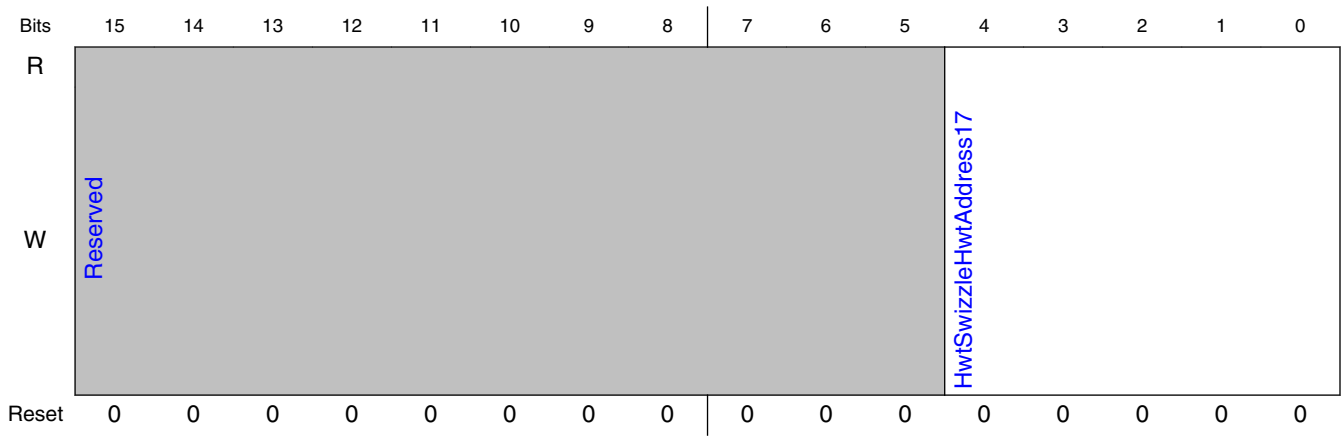
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress15	See Description of HwtSwizzleHwtAddress0 for details. See Description of HwtSwizzleHwtAddress0 for details. Used only for DDR3.

9.3.3.6.158 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtAddress17)

9.3.3.6.158.1 Offset

Register	Offset
HwtSwizzleHwtAddress17	260h

9.3.3.6.158.2 Diagram



9.3.3.6.158.3 Fields

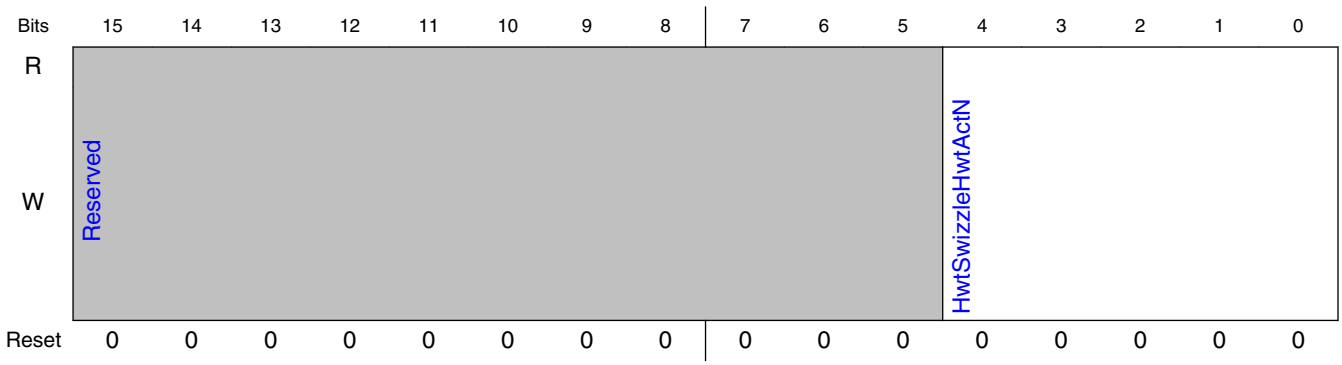
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtAddress17	See Description of HwtSwizzleHwtAddress0 for details. See Description of HwtSwizzleHwtAddress0 for details. Used only for DDR4.

9.3.3.6.159 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtActN)

9.3.3.6.159.1 Offset

Register	Offset
HwtSwizzleHwtActN	262h

9.3.3.6.159.2 Diagram



9.3.3.6.159.3 Fields

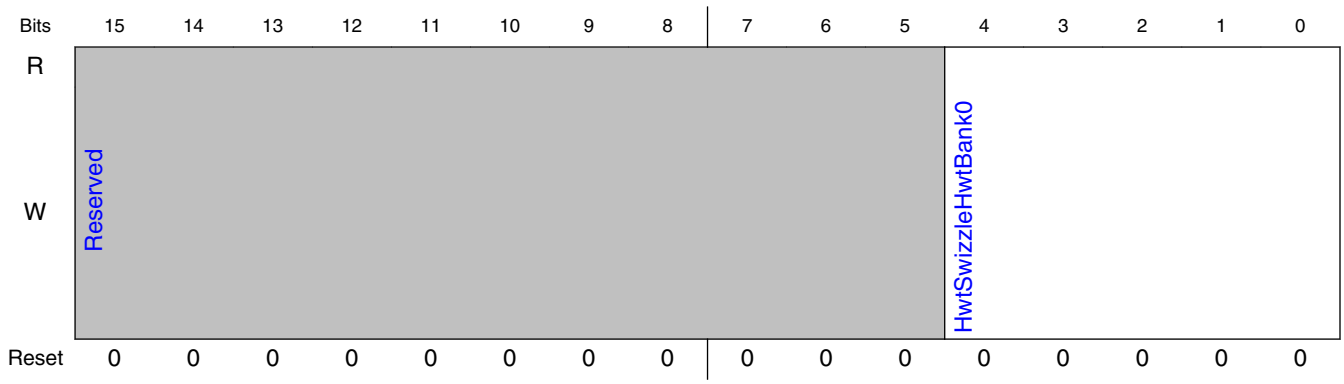
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtActN	See Description of HwtSwizzleHwtAddress0 for details. See Description of HwtSwizzleHwtAddress0 for details. Used only for DDR4.

9.3.3.6.160 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtBank0)

9.3.3.6.160.1 Offset

Register	Offset
HwtSwizzleHwtBank0	264h

9.3.3.6.160.2 Diagram



9.3.3.6.160.3 Fields

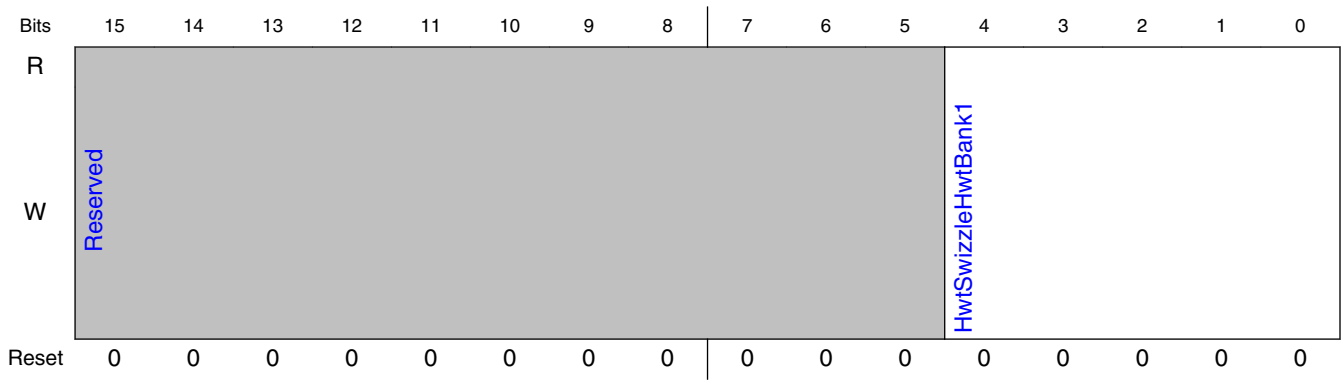
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtBank0	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.161 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtBank1)

9.3.3.6.161.1 Offset

Register	Offset
HwtSwizzleHwtBank1	266h

9.3.3.6.161.2 Diagram



9.3.3.6.161.3 Fields

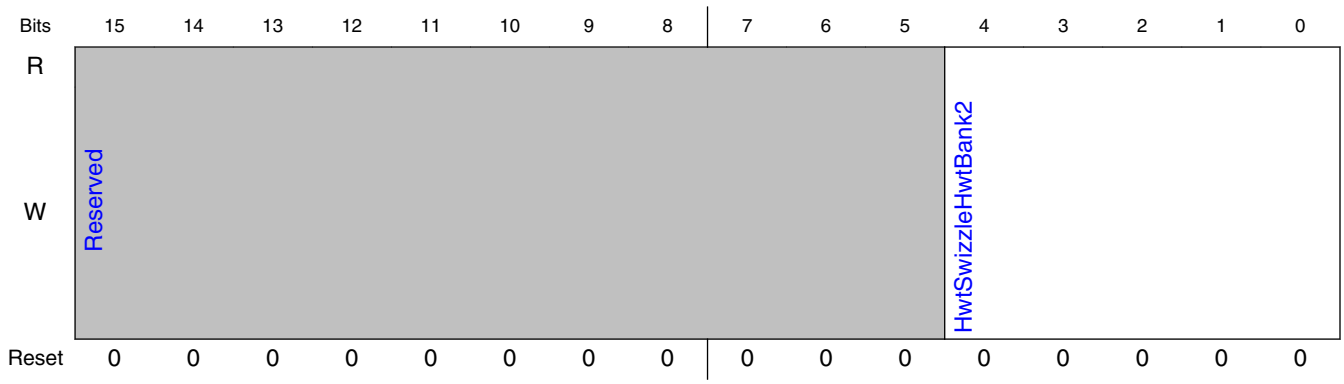
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtBank1	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.162 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtBank2)

9.3.3.6.162.1 Offset

Register	Offset
HwtSwizzleHwtBank2	268h

9.3.3.6.162.2 Diagram



9.3.3.6.162.3 Fields

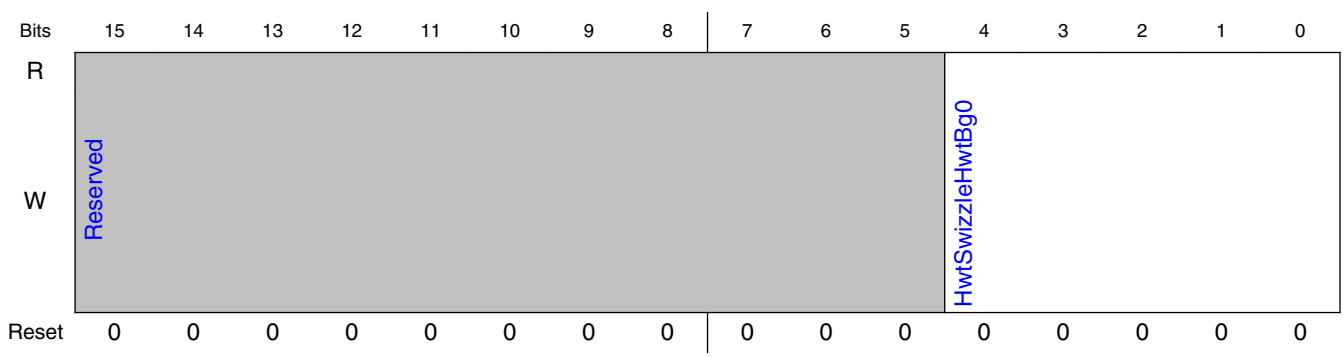
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtBank2	See Description of HwtSwizzleHwtAddress0 for details. See Description of HwtSwizzleHwtAddress0 for details. Used only for DDR3.

9.3.3.6.163 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtBg0)

9.3.3.6.163.1 Offset

Register	Offset
HwtSwizzleHwtBg0	26Ah

9.3.3.6.163.2 Diagram



9.3.3.6.163.3 Fields

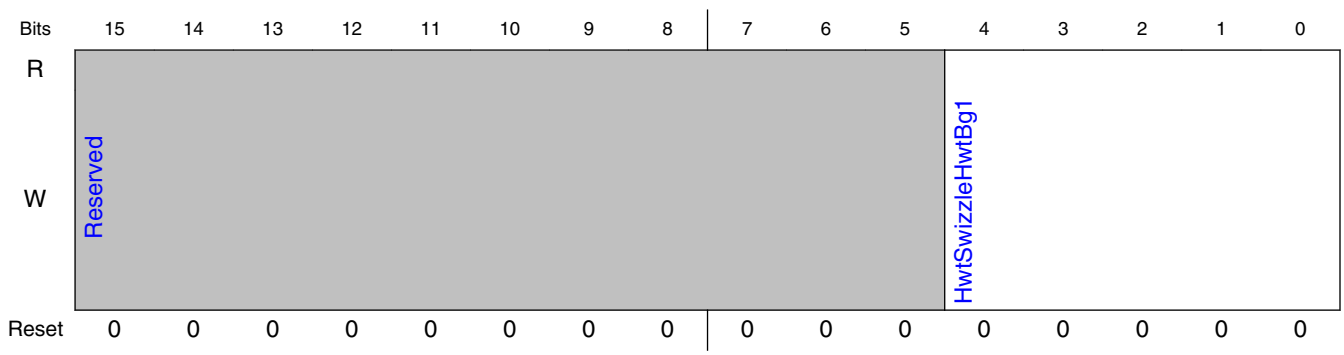
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtBg0	See Description of HwtSwizzleHwtAddress0 for details. See Description of HwtSwizzleHwtAddress0 for details. Used only for DDR4.

9.3.3.6.164 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtBg1)

9.3.3.6.164.1 Offset

Register	Offset
HwtSwizzleHwtBg1	26Ch

9.3.3.6.164.2 Diagram



9.3.3.6.164.3 Fields

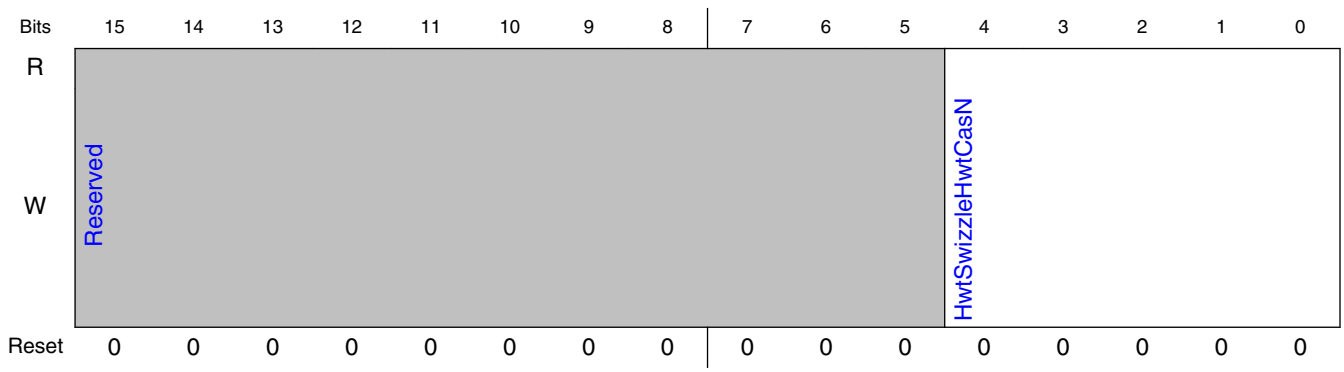
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtBg1	See Description of HwtSwizzleHwtAddress0 for details. See Description of HwtSwizzleHwtAddress0 for details. Used only for DDR4.

9.3.3.6.165 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtCasN)

9.3.3.6.165.1 Offset

Register	Offset
HwtSwizzleHwtCasN	26Eh

9.3.3.6.165.2 Diagram



9.3.3.6.165.3 Fields

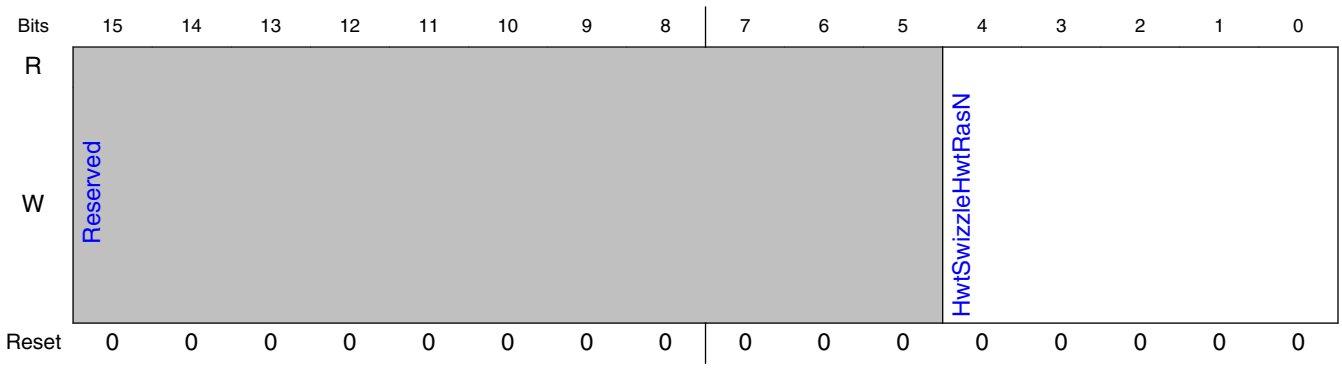
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtCasN	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.166 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtRasN)

9.3.3.6.166.1 Offset

Register	Offset
HwtSwizzleHwtRasN	270h

9.3.3.6.166.2 Diagram



9.3.3.6.166.3 Fields

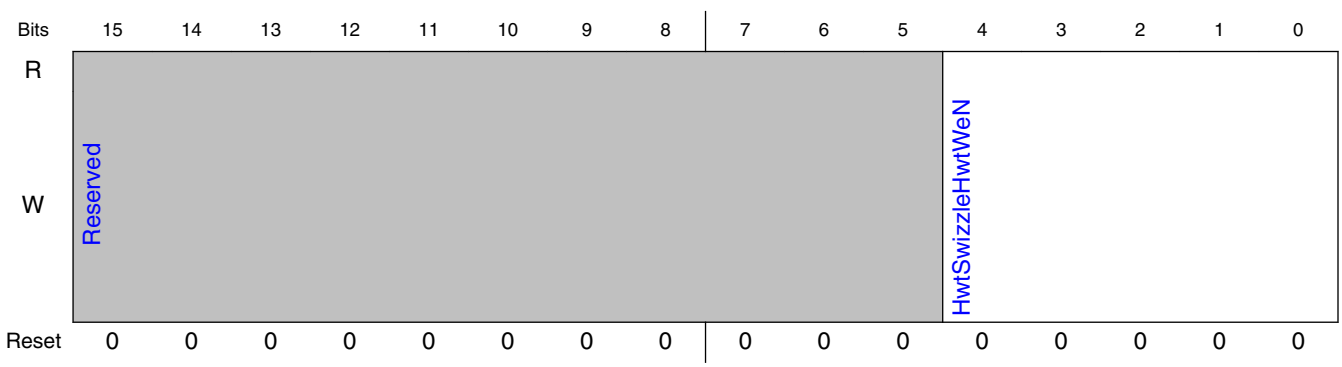
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtRasN	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.167 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtWeN)

9.3.3.6.167.1 Offset

Register	Offset
HwtSwizzleHwtWeN	272h

9.3.3.6.167.2 Diagram



9.3.3.6.167.3 Fields

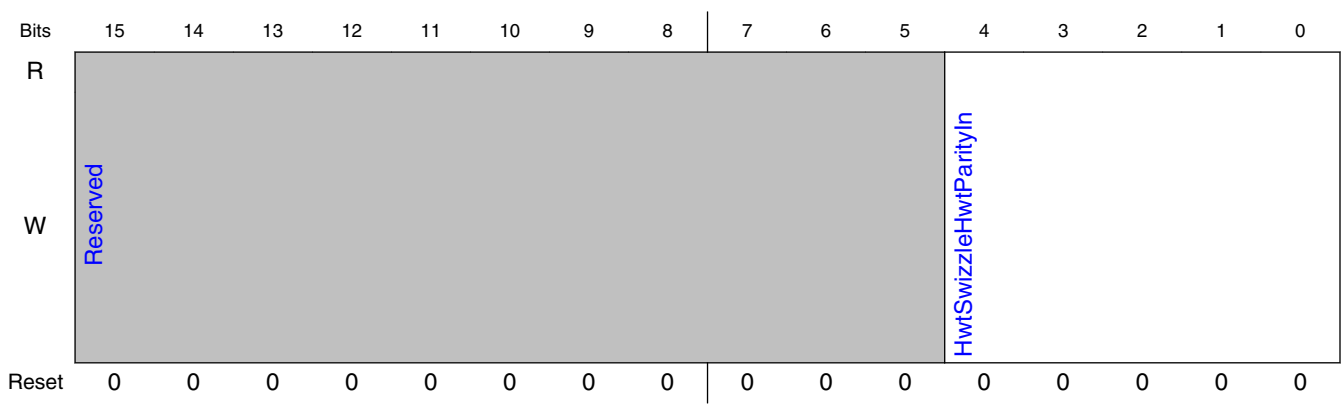
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtWeN	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.168 Signal swizzle selection for HWT swizzle (HwtSwizzleHwtParityIn)

9.3.3.6.168.1 Offset

Register	Offset
HwtSwizzleHwtParityIn	274h

9.3.3.6.168.2 Diagram



9.3.3.6.168.3 Fields

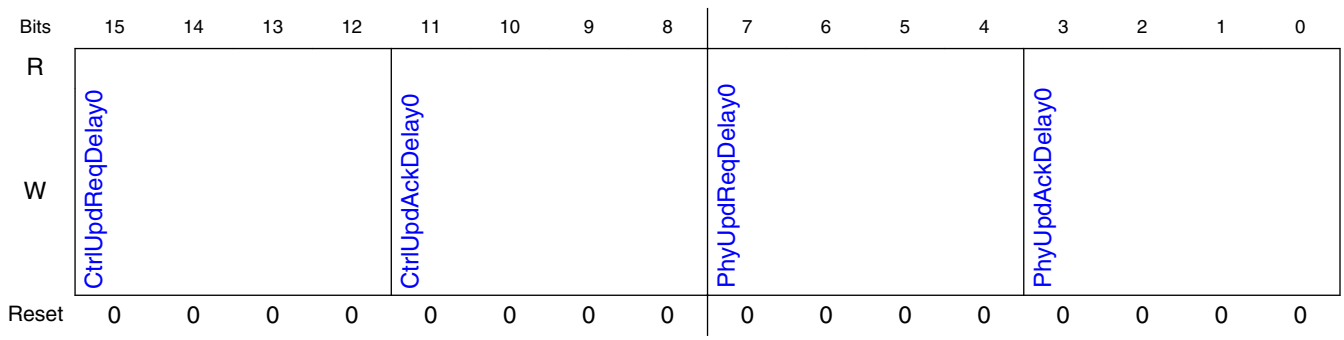
Field	Function
15-5 —	Reserved
4-0 HwtSwizzleHwtParityIn	See Description of HwtSwizzleHwtAddress0 for details.

9.3.3.6.169 Add assertion/deassertion delays on handshake signals Logic assumes that dfi signal assertions exceed the programmed delays (DfiHandshakeDelays0)

9.3.3.6.169.1 Offset

Register	Offset
DfiHandshakeDelays0	278h

9.3.3.6.169.2 Diagram



9.3.3.6.169.3 Fields

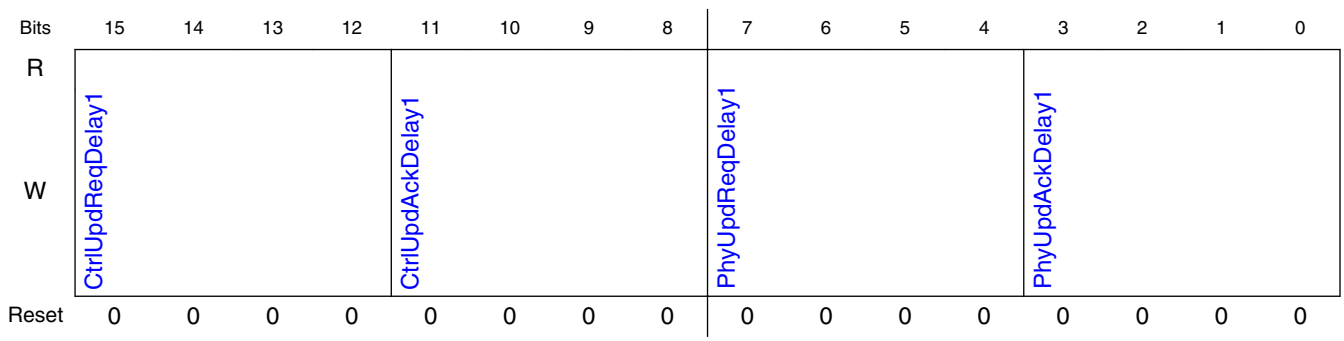
Field	Function
15-12 CtrlUpdReqDelay0	Adds 0-15 DfiClks of delay after the PHY completes all PHY update activities, before deasserting dfi0_ctrlupd_ack.
11-8 CtrlUpdAckDelay0	Adds 0-15 DfiClks of delay after dfi0_ctrlupd_req asserts, before the PHY takes any action.
7-4 PhyUpdReqDelay0	Adds 0-15 DfiClks of delay after the PHY completes all PHY update activities, before deasserting dfi0_phyupd_req.
3-0 PhyUpdAckDelay0	Adds 0-15 DfiClks of delay after dfi0_phyupd_ack asserts, before the PHY takes any action (such as starting DDL calibration).

9.3.3.6.170 Add assertion/deassertion delays on handshake signals Logic assumes that dfi signal assertions exceed the programmed delays (DfiHandshakeDelays1)

9.3.3.6.170.1 Offset

Register	Offset
DfiHandshakeDelays1	27Ah

9.3.3.6.170.2 Diagram



9.3.3.6.170.3 Fields

Field	Function
15-12 CtrlUpdReqDelay1	Adds 0-15 DfiClks of delay after the PHY completes all PHY update activities, before deasserting dfi1_ctrlupd_ack.
11-8 CtrlUpdAckDelay1	Adds 0-15 DfiClks of delay after dfi1_ctrlupd_req asserts, before the PHY takes any action.
7-4 PhyUpdReqDelay1	Adds 0-15 DfiClks of delay after the PHY completes all PHY update activities, before deasserting dfi1_phyupd_req.
3-0 PhyUpdAckDelay1	Adds 0-15 DfiClks of delay after dfi1_phyupd_ack asserts, before the PHY takes any action (such as starting DDL calibration).

9.3.3.6.171 Impedance Calibration Clock Ratio (CalUclkInfo_p1)

9.3.3.6.171.1 Offset

Register	Offset
CalUclkInfo_p1	20_0010h

9.3.3.6.171.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								CalUClkTicksPer1uS							
Reset	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0

9.3.3.6.171.3 Fields

Field	Function
15-10 —	Reserved
9-0 CalUClkTicksPer1uS	Must be programmed to the number of DfiClks in 1us (rounded up), with minimum value of 24. Must be programmed to the number of DfiClks in 1us (rounded up), with minimum value of 24. if (DfiClk < 24MHz) CalUclkInfo = 24 else CalUclkInfo = (number of DfiClks in 1us)

9.3.3.6.172 PHY Initialization Engine (PIE) Delay Register 0 (Seq0BDLY0_p1)

9.3.3.6.172.1 Offset

Register	Offset
Seq0BDLY0_p1	20_0016h

9.3.3.6.172.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Seq0BDLY0_p1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.172.3 Fields

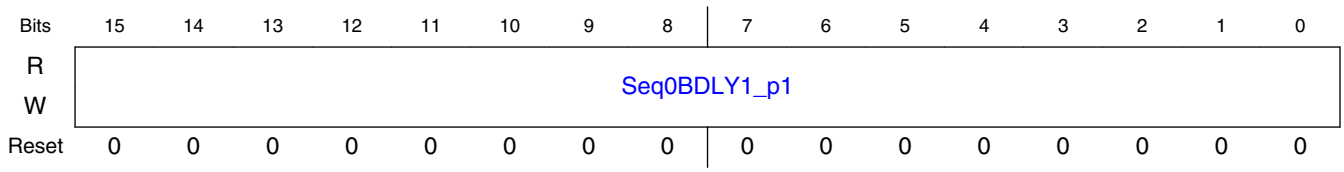
Field	Function
15-0 Seq0BDLY0_p1	PHY Initialization Engine (PIE) Delay Register 0 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 0 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.173 PHY Initialization Engine (PIE) Delay Register 1 (Seq0BDLY1_p1)

9.3.3.6.173.1 Offset

Register	Offset
Seq0BDLY1_p1	20_0018h

9.3.3.6.173.2 Diagram



9.3.3.6.173.3 Fields

Field	Function
15-0 Seq0BDLY1_p1	PHY Initialization Engine (PIE) Delay Register 1 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 1 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.174 PHY Initialization Engine (PIE) Delay Register 2 (Seq0BDLY2_p1)

9.3.3.6.174.1 Offset

Register	Offset
Seq0BDLY2_p1	20_001Ah

9.3.3.6.174.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Seq0BDLY2_p1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.174.3 Fields

Field	Function
15-0 Seq0BDLY2_p1	PHY Initialization Engine (PIE) Delay Register 2 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 2 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: -------- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --------- --------- ---------- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.175 PHY Initialization Engine (PIE) Delay Register 3 (Seq0BDLY3_p1)

9.3.3.6.175.1 Offset

Register	Offset
Seq0BDLY3_p1	20_001Ch

9.3.3.6.175.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Seq0BDLY3_p1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.175.3 Fields

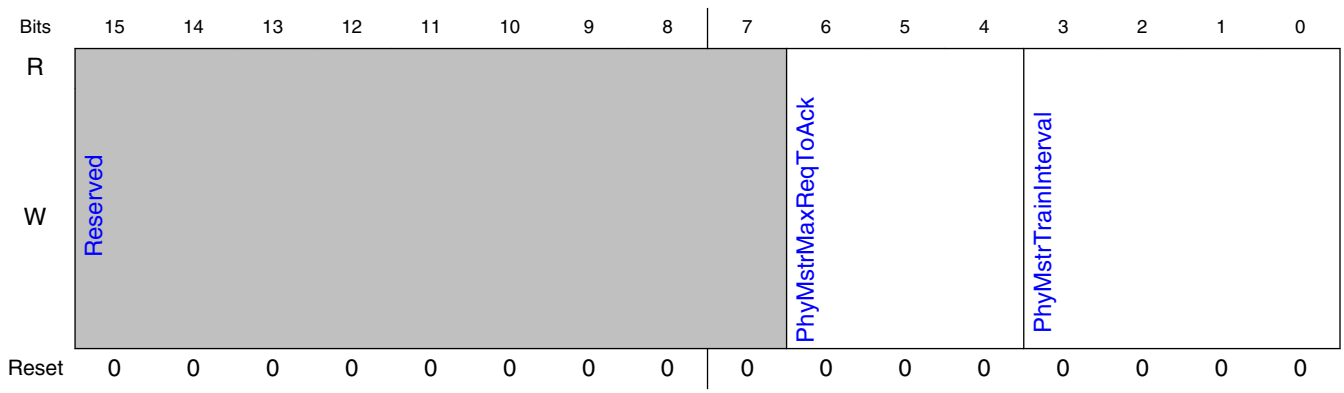
Field	Function
15-0 Seq0BDLY3_p1	PHY Initialization Engine (PIE) Delay Register 3 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 3 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.176 Setup Intervals for DFI PHY Master operations (PPTTrainSetup_p1)

9.3.3.6.176.1 Offset

Register	Offset
PPTTrainSetup_p1	20_0020h

9.3.3.6.176.2 Diagram



9.3.3.6.176.3 Fields

Field	Function
15-7 —	Reserved
6-4 PhyMstrMaxReqToAck	Bits 6:4 of this register specify the max time from tdfi_phymstr_req asserted to tdfi_phymstr_ack asserted. Bits 6:4 of this register specify the max time from tdfi_phymstr_req asserted to tdfi_phymstr_ack asserted. 3'b000 Disable PHY Master Interface 3'b001 set tPHYMSTR_resp. = 512 MEMCLKs 3'b010 set tPHYMSTR_resp. = 1024 MEMCLKs 3'b011 set tPHYMSTR_resp. = 2048 MEMCLKs 3'b100 set tPHYMSTR_resp. = 4096 MEMCLKs 3'b101 set tPHYMSTR_resp. = 8192 MEMCLKs 3'b110 set tPHYMSTR_resp. = 32768 MEMCLKs 3'b111 set tPHYMSTR_resp. = undefined
3-0	Bits 3:0 of this register specifies the time between the end of one training and the start of the next.

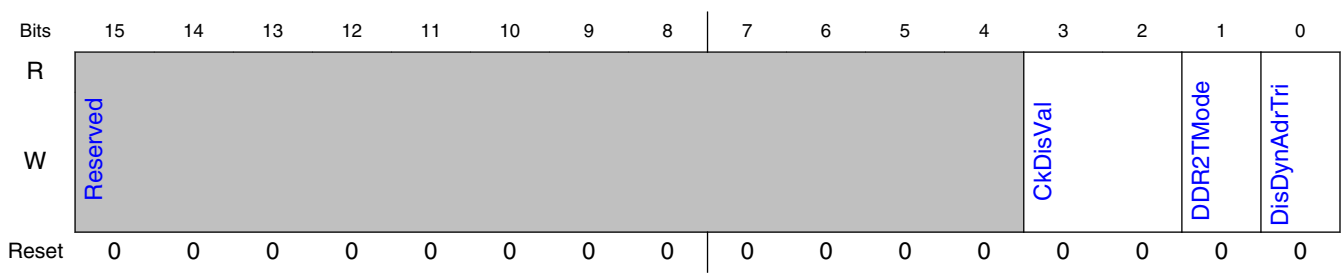
Field	Function
PhyMstrTrainInterval	Bits 3:0 of this register specifies the time between the end of one training and the start of the next. (As per section 11 of the Preliminary DFI 4.0 Specification, V2, it is the max expected time from dfi_init_complete asserted to tdfi_phymstr_ack asserted). 4'b0000 Disable PHY Master Interface 4'b0001 PHY MASTER Request Interval = 524288 MEMCLKs 4'b0010 PHY MASTER Request Interval = 1048576 MEMCLKs 4'b0011 PHY MASTER Request Interval = 2097152 MEMCLKs 4'b0100 PHY MASTER Request Interval = 4194304 MEMCLKs 4'b0101 PHY MASTER Request Interval = 8388608 MEMCLKs 4'b0110 PHY MASTER Request Interval = 16777216 MEMCLKs 4'b0111 PHY MASTER Request Interval = 33554432 MEMCLKs 4'b1000 PHY MASTER Request Interval = 67108864 MEMCLKs 4'b1001 PHY MASTER Request Interval = 134217728 MEMCLKs 4'b1010 PHY MASTER Request Interval = 268435456 MEMCLKs 4'b1011 - 4'b1111 PHY MASTER Request Interval = undefined

9.3.3.6.177 Mode select register for MEMCLK/Address/Command Tristates (TristateModeCA_p1)

9.3.3.6.177.1 Offset

Register	Offset
TristateModeCA_p1	20_0032h

9.3.3.6.177.2 Diagram



9.3.3.6.177.3 Fields

Field	Function
15-4 —	Reserved
3-2 CkDisVal	The PHY provides 4 memory clocks, n=0. The PHY provides 4 memory clocks, n=0..3. When the toggling of memory clock CK_t[n] is disabled with dfi_clk_disable[n]=1, the memory clock CK_t[n] is driven with Register CkDisVal[1]. When the toggling of memory clock CK_c[n] is disabled with dfi_clk_disable[n]=1, the memory clock CK_c[n] is driven with ~(Register CkDisVal[0]) (ie inverted). Note that the non-toggling differential memory clock CK_t[n],CK_c[n] may be driven with any combination, LL,LH,HL,HH; the default CK_t[n],CK_c[n]=LH.
1 DDR2TMode	Must be set to 1 for Dynamic Tristate to work when CA bus is 2T or Geardown mode. Must be set to 1 for Dynamic Tristate to work when CA bus is 2T or Geardown mode. This CSR bit has no effect when DisDynAdrTri==1
0 DisDynAdrTri	When DisDynAdrTri=1, Dynamic Tristating is disabled. When DisDynAdrTri=1, Dynamic Tristating is disabled. Dynamic Tristating is on by default. . Dynamic Tristating should be disabled (DisDynAdrTri=0) for these modes: 1. DDR3/2T if the controller cannot follow the 2T PHY tristate protocol. 2. DDR4/2T/2N if the controller cannot follow the 2T PHY tristate protocol. 3. LPDDR4 . When DisDynAdrTri=0 (default), In DDR3 mode, The following SDRAM pins can be dynamically tristated: A,BA,RAS_n,CAS_n,WE_n In DDR4 mode, The following SDRAM pins can be dynamically tristated: A,BA,BG,ACT_n In LPDDR3 mode, The following SDRAM pins can be dynamically tristated: CA[*] . In 1T mode, the PHY will tristate the relevant pins when all ranks are DESelected . In 2T mode (or geardown), the PHY will tristate the relevant pins when: D3 -- All ranks DESelected (all CS_L==1) AND RAS_n,CAS_n,WE_n,BA0 = 1,1,1,0 D4 -- All ranks DESelected (all CS_L==1) AND RAS_n,CAS_n,WE_n,BA0,ACT_n = 1,1,1,0,1 When in this mode, the controller should avoid sending the above patterns with any rank selected. [e.g. NOPs sent by the controller should have BA0 set to a non-zero value]

9.3.3.6.178 HWT MaxReadLatency. (HwtMRL_p1)

9.3.3.6.178.1 Offset

Register	Offset
HwtMRL_p1	20_0040h

9.3.3.6.178.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											HwtMRL_p1				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

9.3.3.6.178.3 Fields

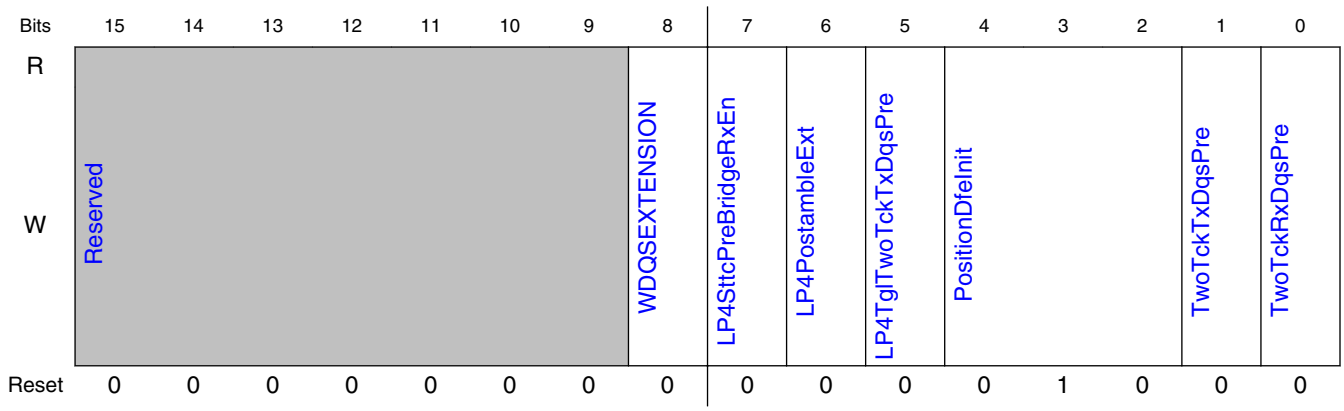
Field	Function
15-5 —	Reserved
4-0 HwtMRL_p1	This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. The CSR is in units of two mem clocks; that is, a unit change in the LSB is a change in MRL of two mem clocks. This MASTER copy of MRL is used by the PHY training hardware only.

9.3.3.6.179 Control the PHY logic related to the read and write DQS preamble (DqsPreambleControl_p1)

9.3.3.6.179.1 Offset

Register	Offset
DqsPreambleControl_p1	20_0048h

9.3.3.6.179.2 Diagram



9.3.3.6.179.3 Fields

Field	Function
15-9 —	Reserved
8 WDQSEXTENSION	When set, DQS_T and DQS_C will be driven differentially to 0 and 1, respectively, before and after a write burst, except during a memory read transaction. When set, DQS_T and DQS_C will be driven differentially to 0 and 1, respectively, before and after a write burst, except during a memory read transaction. See DesignWare Cores LPDDR4 MultiPHY: WDQS Extension Application Note. Use of WDQSEXTENSION requires POdtTailWidth=3 and POdtStartDelay=00
7 LP4SttcPreBridgeRxEn	Used in LPDDR4 static-preamble mode to bridge the RxEn between two reads to the same timing group when the bubble is 1 memclk. Used in LPDDR4 static-preamble mode to bridge the RxEn between two reads to the same timing group when the bubble is 1 memclk. LPDDR4 static-preamble mode is set in the DRAMs with MR1, OP[3]=0.
6 LP4PostambleExt	In LPDDR4 mode must be set to extend the write postamble. In LPDDR4 mode must be set to extend the write postamble. 0: half-memclk postamble, as in D4 1: one-and-one-half-memclk postamble; see LPDDR4 Spec MR3, OP[1] WR PST, vendor-specific function.
5 LP4TglTwoTckTxDqsPre	Used in LPDDR4 mode to modify the early preamble when Register TwoTckTxDqsPre=1 0: level first-memclk preamble 1: toggling first-memclk preamble
4-2 PositionDfeInit	For DDR4 phy only when receive DFE is enabled. For DDR4 phy only when receive DFE is enabled. PositionDfeInit[2]=1 causes DfeInit to be asserted late PositionDfeInit[1]=1 causes DfeInit to be asserted at the nominal time

Table continues on the next page...

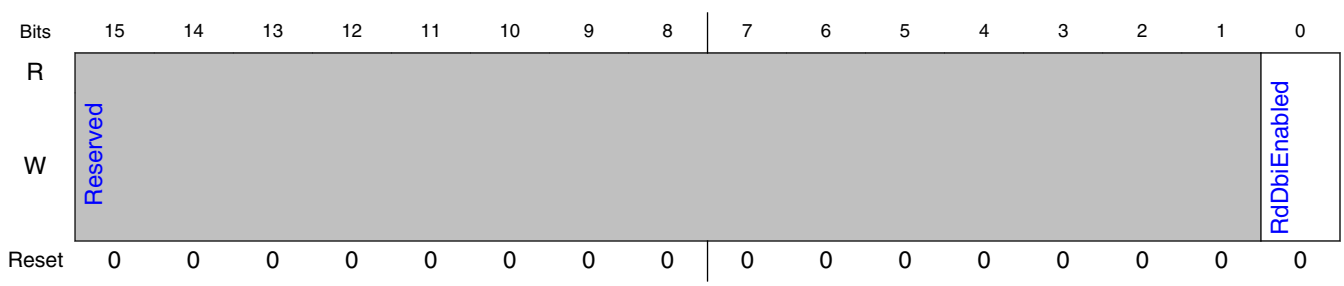
Field	Function
	PositionDfeInit[0]=1 causes DfeInit to be asserted early PositionDfeInit=3'b010 is the nominal, recommended value for leading edge used by receiver.
1 TwoTckTxDqsPre	0: Standard 1tck TxDqs Preamble 1: Enable Optional D4 2tck TxDqs Preamble The DDR4 MR4 A12 is Write Preamble, 1=2nCK, 0=1nCK.
0 TwoTckRxDqsPre	Widens the RxDqsEn window to allow larger drift in the incoming read DQS to take advantage of the larger/wider preamble generated by the DRAMSs when the D4 DRAMS are configured with DDR4 MR4 A11 Read Preamble=1 for causing a 2nCK read preamble. Widens the RxDqsEn window to allow larger drift in the incoming read DQS to take advantage of the larger/wider preamble generated by the DRAMSs when the D4 DRAMS are configured with DDR4 MR4 A11 Read Preamble=1 for causing a 2nCK read preamble. For LPDDR4, all read operations are 2nCK such that this control must be set to 1. Note the nominal trained-to-center point will be earlier relative to the first strobing edge of DQS than when not in this mode.

9.3.3.6.180 This Register is used to enable the Read-DBI function in each DBYTE (DMIPinPresent_p1)

9.3.3.6.180.1 Offset

Register	Offset
DMIPinPresent_p1	20_005Ah

9.3.3.6.180.2 Diagram



9.3.3.6.180.3 Fields

Field	Function
15-1	Reserved

Table continues on the next page...

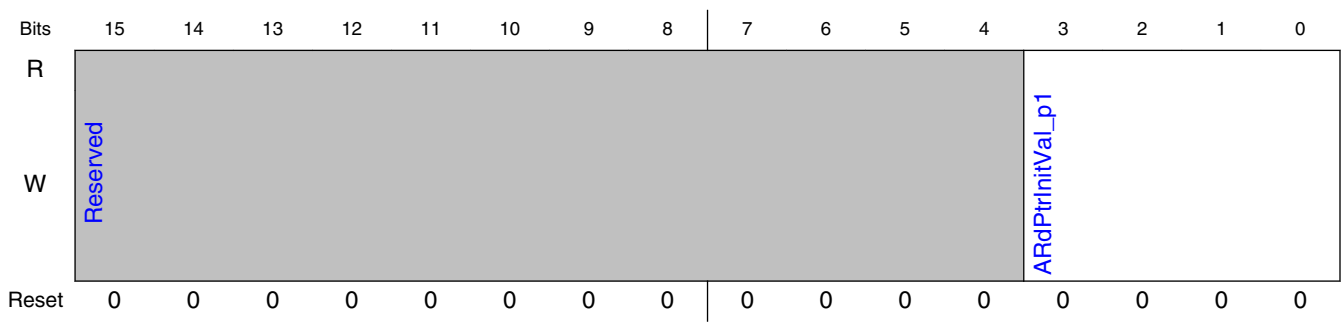
Field	Function
—	
0 RdDbiEnabled	This bit must be set to 1'b1 if Read-DBI is enabled in a connected DDR4 or LPDDR4 device. This bit must be set to 1'b1 if Read-DBI is enabled in a connected DDR4 or LPDDR4 device. If set, the following DRAM MR should also be set [DDR4.MR5.A12=1 or LPDDR4.MR3.OP[7]=1]

9.3.3.6.181 Address/Command FIFO ReadPointer Initial Value (ARdPtrInitVal_p1)

9.3.3.6.181.1 Offset

Register	Offset
ARdPtrInitVal_p1	20_005Ch

9.3.3.6.181.2 Diagram



9.3.3.6.181.3 Fields

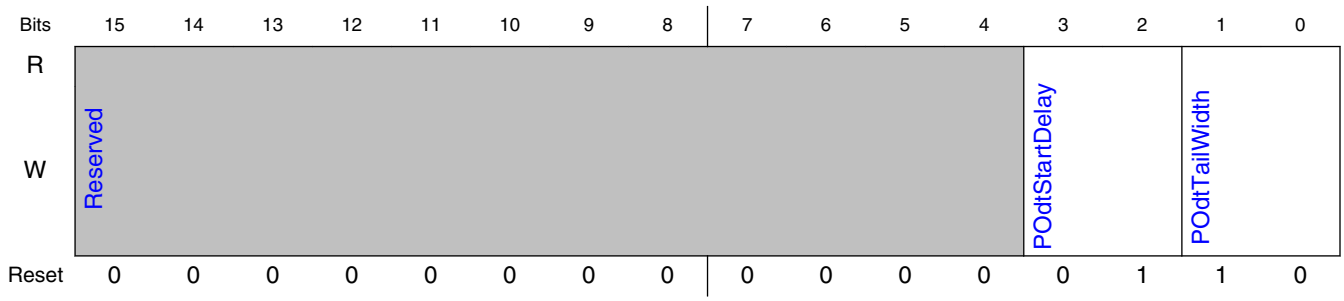
Field	Function
15-4 —	Reserved
3-0 ARdPtrInitVal_p1	This is the initial Pointer Offset for the free-running FIFOs in the DBYTE and ACX4 hardips. This is the initial Pointer Offset for the free-running FIFOs in the DBYTE and ACX4 hardips. The units of this register are in UI. The pointer separation should be chosen to compensate for all sources of skew and drift of the PHY DFICLK and PCLK networks. Please see PUB databook section 8.1.1 This CSR must be programmed in Step C of the PHY Initialization sequence

9.3.3.6.182 READ DATA On-Die Termination Timing Control (by PHY) (ProcOdtTimeCtl_p1)

9.3.3.6.182.1 Offset

Register	Offset
ProcOdtTimeCtl_p1	20_00ACh

9.3.3.6.182.2 Diagram



9.3.3.6.182.3 Fields

Field	Function
15-4 —	Reserved
3-2 POdtStartDelay	controls the start of ProcOdt, units of UI 3 delay start 2 UI, maximum delay of start of ProcOdt 2 delay start 1 UI, 1 delay start 0 UI, default 0 early by 1 UI, The time from ProcODT assertion to opening the window to receive DQS is (10 - POdtStartDelay) UI.
1-0 POdtTailWidth	controls the length of the tail of ProcOdt, units of UI 3 tail 3UI more than for Register POdtTailWidth=0, maximum 2 tail 2UI more than for Register POdtTailWidth=0, default 1 tail 1UI more than for Register POdtTailWidth=0 0 minimum length tail The time from ProcODT to closing the window to receive DQS to ProcODT POdtTailWidth is (2 + POdtTailWidth) UI

9.3.3.6.183 DLL gain control (DllGainCtl_p1)

9.3.3.6.183.1 Offset

Register	Offset
DllGainCtl_p1	20_00F8h

9.3.3.6.183.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				DllSeedSel				DllGainTV				DllGainIV			
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	1

9.3.3.6.183.3 Fields

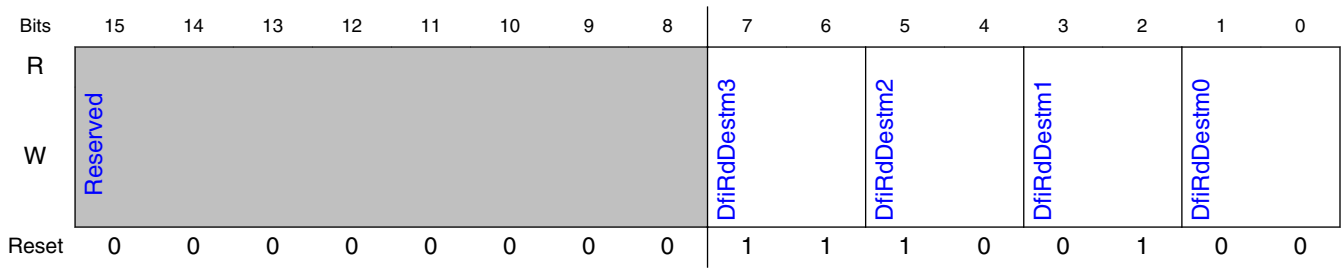
Field	Function
15-12 —	Reserved
11-8 DllSeedSel	Reserved, must be configured to be 0.
7-4 DllGainTV	Terminal value of DllGain, ie the value in effect when locking is done and the value used for maintaining lock, ie tracking pclk variation. Terminal value of DllGain, ie the value in effect when locking is done and the value used for maintaining lock, ie tracking pclk variation. DllGainTV must be greater than or equal to DllGainIV. The maximum value is 10. The minimum value is 6.
3-0 DllGainIV	Initial value of DllGain.

9.3.3.6.184 Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiRdDataCsDestMap_p1)

9.3.3.6.184.1 Offset

Register	Offset
DfiRdDataCsDestMap_p1	20_0160h

9.3.3.6.184.2 Diagram



9.3.3.6.184.3 Fields

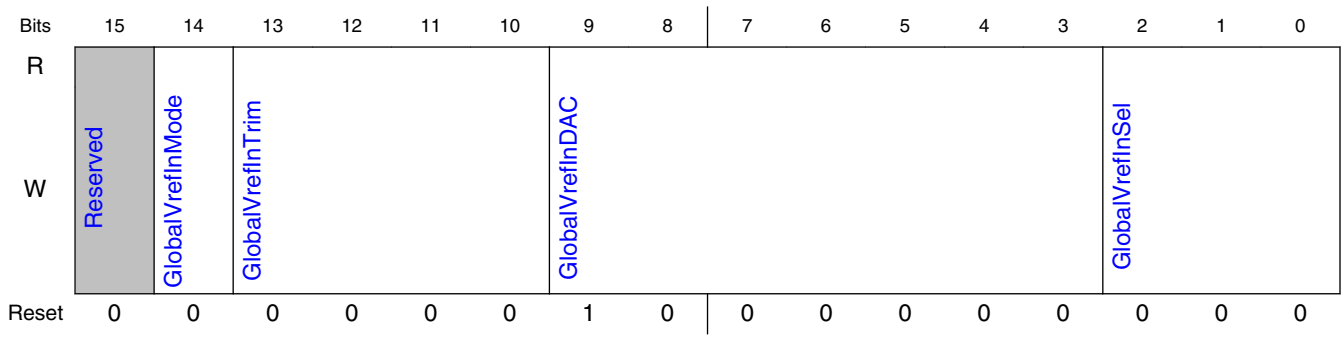
Field	Function
15-8 —	Reserved
7-6 DfiRdDestm3	Maps dfi_rddata_cs_n_p0[3] to dest DfiRdDestm3 timing For example, if 3 dfi_rddata_cs_n_p0[3] will use Register RxEn,ClkDlyTg3 timing.
5-4 DfiRdDestm2	Maps dfi_rddata_cs_n_p0[2] to dest DfiRdDestm2 timing For example, if 2 dfi_rddata_cs_n_p0[2] will use Register RxEn,ClkDlyTg2 timing.
3-2 DfiRdDestm1	Maps dfi_rddata_cs_n_p0[1] to dest DfiRdDestm1 timing For example, if 1 dfi_rddata_cs_n[_p01] will use Register RxEn,ClkDlyTg1 timing.
1-0 DfiRdDestm0	Maps dfi_rddata_cs_n_p0[0] to dest DfiRdDestm0 timing For example, if 0 dfi_rddata_cs_n_p0[0] will use Register RxEn,ClkDlyTg0 timing.

9.3.3.6.185 PHY Global Vref Controls (VrefInGlobal_p1)

9.3.3.6.185.1 Offset

Register	Offset
VrefInGlobal_p1	20_0164h

9.3.3.6.185.2 Diagram



9.3.3.6.185.3 Fields

Field	Function
15 —	Reserved
14 GlobalVrefInMode	RSVD
13-10 GlobalVrefInTrim	RSVD
9-3 GlobalVrefInDAC	DAC code for internal Vref generation The DAC has two ranges; the range is set by GlobalVrefInSel[2] ========== RANGE0 : DDR3,DDR4,LPDDR3 [GlobalVrefInSel[2] = 0] DAC Output Voltage = GlobalVrefInDAC == 6'h00 ? Hi-Z : 0. DAC code for internal Vref generation The DAC has two ranges; the range is set by GlobalVrefInSel[2] ========== RANGE0 : DDR3,DDR4,LPDDR3 [GlobalVrefInSel[2] = 0] DAC Output Voltage = GlobalVrefInDAC == 6'h00 ? Hi-Z : 0.345*VDDQ + (0.005*GlobalVrefInDAC)*VDDQ ========== RANGE1 : LPDDR4 [GlobalVrefInSel[2] = 1] DAC Output Voltage = GlobalVrefInDAC == 6'h00 ? Hi-Z : (0.005*(GlobalVrefInDAC-1))*VDDQ
2-0 GlobalVrefInSel	GlobalVrefInSel[1:0] controls the mode of the PHY VREF DAC and the BP_VREF pin ========== 2'b00 - PHY Vref DAC Range0 -- BP_VREF = Hi-Z 2'b01 - Reserved Encoding 2'b10 - PHY Vref DAC Range0 -- BP_VREF connected to PLL Analog Bus 2'b11 - PHY Vref DAC Range0 -- BP_VREF connected to PHY Vref DAC ========== GlobalVrefInSel[2] shall be set according to Dram Protocol: Protocol GlobalVrefInSel[2] ---------- DDR3 1'b0 DDR4 1'b0 LPDDR3 1'b0 LPDDR4 1'b1

9.3.3.6.186 Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiWrDataCsDestMap_p1)

9.3.3.6.186.1 Offset

Register	Offset
DfiWrDataCsDestMap_p1	20_0168h

9.3.3.6.186.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								DfiWrDestm3		DfiWrDestm2		DfiWrDestm1		DfiWrDestm0	
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	0

9.3.3.6.186.3 Fields

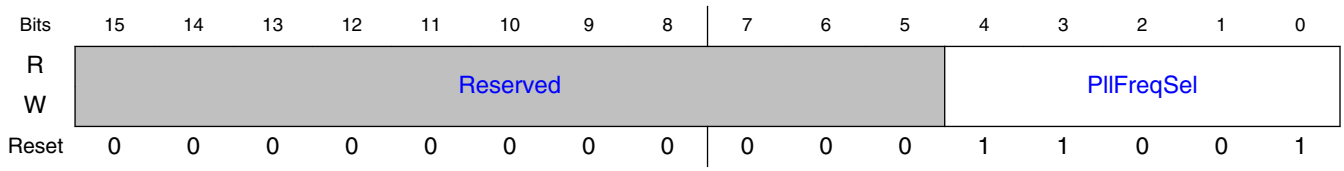
Field	Function
15-8 —	Reserved
7-6 DfiWrDestm3	Maps dfi_wrdata_cs_n_p0[3] to dest DfiWrDestm3 timing (use Register TxDq,DqsDlyTg3) For example, if 3 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg3 timing.
5-4 DfiWrDestm2	Maps dfi_wrdata_cs_n_p0[2] to dest DfiWrDestm2 timing For example, if 2 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg2 timing.
3-2 DfiWrDestm1	Maps dfi_wrdata_cs_n_p0[1] to dest DfiWrDestm1 timing For example, if 1 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg1 timing.
1-0 DfiWrDestm0	Maps dfi_wrdata_cs_n_p0[0] to dest DfiWrDestm0 timing For example, if 0 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg0 timing.

9.3.3.6.187 PState dependent PLL Control Register 2 (PllCtrl2_p1)

9.3.3.6.187.1 Offset

Register	Offset
PllCtrl2_p1	20_018Ah

9.3.3.6.187.2 Diagram



9.3.3.6.187.3 Fields

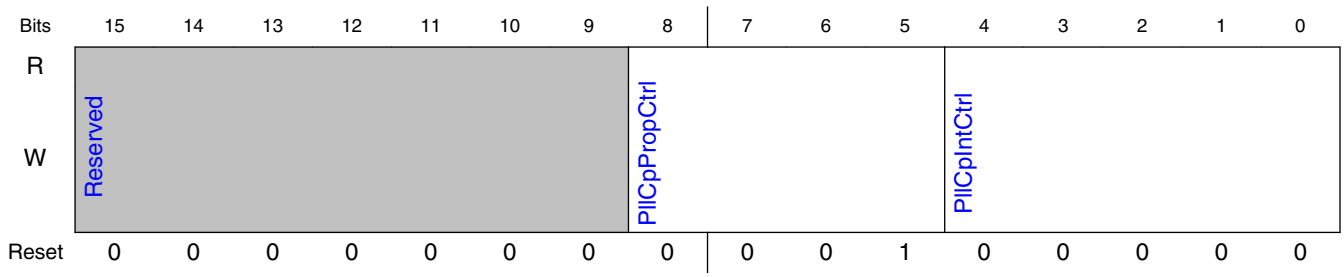
Field	Function
15-5 —	Reserved
4-0 PllFreqSel	Adjusts the loop parameters to compensate for different VCO bias points, and input/output clock division ratios. Adjusts the loop parameters to compensate for different VCO bias points, and input/output clock division ratios. Program based on reference clock freqency (DfiClk) with this table.

9.3.3.6.188 PState dependent PLL Control Register 1 (PllCtrl1_p1)

9.3.3.6.188.1 Offset

Register	Offset
PllCtrl1_p1	20_018Eh

9.3.3.6.188.2 Diagram



9.3.3.6.188.3 Fields

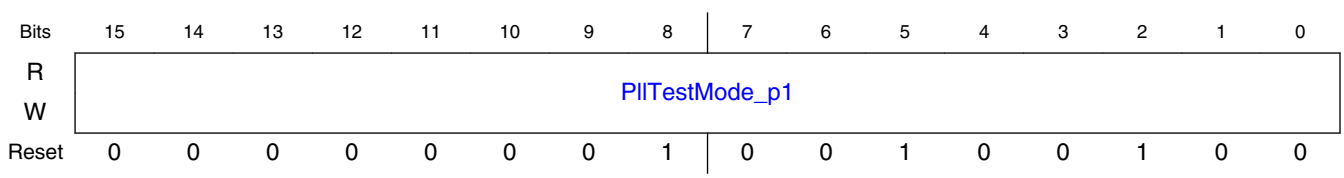
Field	Function
15-9 —	Reserved
8-5 PllCpPropCtrl	connects directly to cp_prop_cntrl<3:0> of PLL. connects directly to cp_prop_cntrl<3:0> of PLL. Charge pump proportional current control.
4-0 PllCpIntCtrl	connects directly to cp_int_cntrl<1:0> in PLL. connects directly to cp_int_cntrl<1:0> in PLL. Charge pump integrating current control.

9.3.3.6.189 Additional controls for PLL CP/VCO modes of operation (PllTestMode_p1)

9.3.3.6.189.1 Offset

Register	Offset
PllTestMode_p1	20_0194h

9.3.3.6.189.2 Diagram



9.3.3.6.189.3 Fields

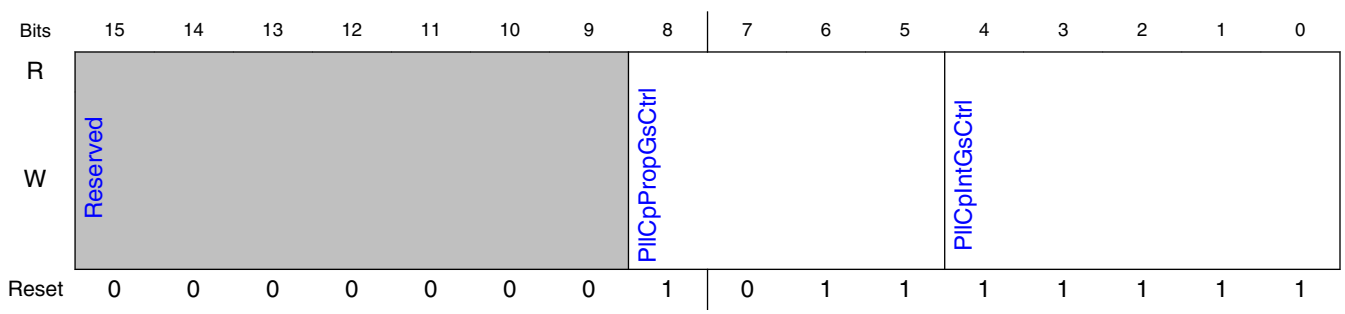
Field	Function
15-0 PllTestMode_p1	It is required to use default values for this CSR Directly connected to testmode[15:0] pin of PLL

9.3.3.6.190 PState dependent PLL Control Register 4 (PllCtrl4_p1)

9.3.3.6.190.1 Offset

Register	Offset
PllCtrl4_p1	20_0198h

9.3.3.6.190.2 Diagram



9.3.3.6.190.3 Fields

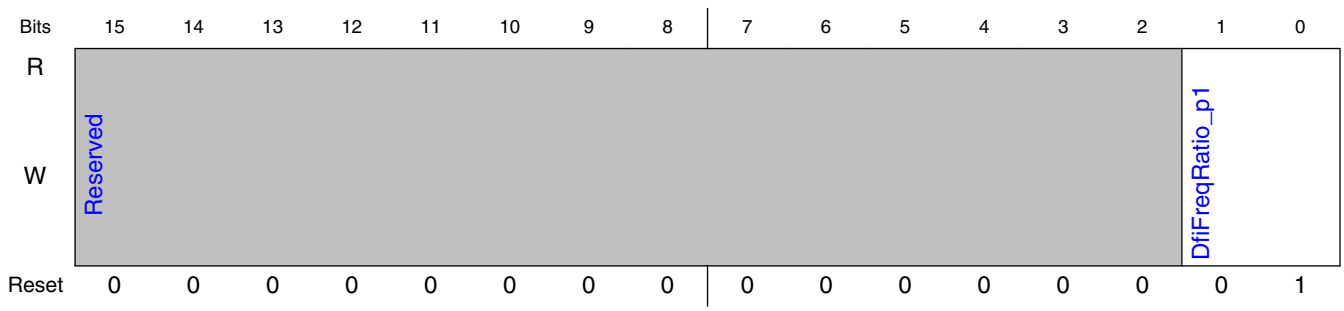
Field	Function
15-9 —	Reserved
8-5 PllCpPropGsCtrl	connects directly to cp_prop_gs_cntrl<3:0> of PLL. connects directly to cp_prop_gs_cntrl<3:0> of PLL. Charge pump proportional current control for fast relock and gearshift.
4-0 PllCpIntGsCtrl	connects directly to cp_int_gs_cntrl<4:0> in PLL. connects directly to cp_int_gs_cntrl<4:0> in PLL. Charge pump integrating current control for fast relock and gearshift.

9.3.3.6.191 DFI Frequency Ratio (DfiFreqRatio_p1)

9.3.3.6.191.1 Offset

Register	Offset
DfiFreqRatio_p1	20_01F4h

9.3.3.6.191.2 Diagram



9.3.3.6.191.3 Fields

Field	Function
15-2 —	Reserved
1-0 DfiFreqRatio_p1	Used in dwc_ddrphy_pub_serdes to serialize or de-serialize DFI signals 00 = 1:1 mode 01 = 1:2 mode 1x = 1:4 mode* *Note: 1:4 is for future pub revision.

9.3.3.6.192 Impedance Calibration Clock Ratio (CalUclkInfo_p2)

9.3.3.6.192.1 Offset

Register	Offset
CalUclkInfo_p2	40_0010h

9.3.3.6.192.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						CalUClkTicksPer1uS									
Reset	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0

9.3.3.6.192.3 Fields

Field	Function
15-10 —	Reserved
9-0 CalUClkTicksPer1uS	Must be programmed to the number of DfiClks in 1us (rounded up), with minimum value of 24. Must be programmed to the number of DfiClks in 1us (rounded up), with minimum value of 24. if (DfiClk < 24MHz) CalUclkInfo = 24 else CalUclkInfo = (number of DfiClks in 1us)

9.3.3.6.193 PHY Initialization Engine (PIE) Delay Register 0 (Seq0BDLY0_p2)

9.3.3.6.193.1 Offset

Register	Offset
Seq0BDLY0_p2	40_0016h

9.3.3.6.193.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Seq0BDLY0_p2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.193.3 Fields

Field	Function
15-0 Seq0BDLY0_p2	PHY Initialization Engine (PIE) Delay Register 0 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value.

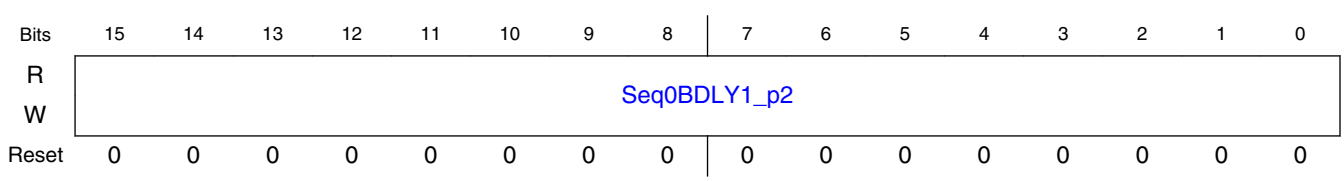
Field	Function
	PHY Initialization Engine (PIE) Delay Register 0 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.194 PHY Initialization Engine (PIE) Delay Register 1 (Seq0BDLY1_p2)

9.3.3.6.194.1 Offset

Register	Offset
Seq0BDLY1_p2	40_0018h

9.3.3.6.194.2 Diagram



9.3.3.6.194.3 Fields

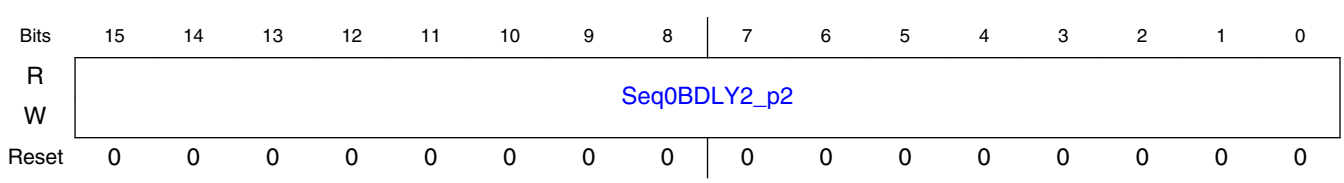
Field	Function
15-0 Seq0BDLY1_p2	PHY Initialization Engine (PIE) Delay Register 1 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 1 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.195 PHY Initialization Engine (PIE) Delay Register 2 (Seq0BDLY2_p2)

9.3.3.6.195.1 Offset

Register	Offset
Seq0BDLY2_p2	40_001Ah

9.3.3.6.195.2 Diagram



9.3.3.6.195.3 Fields

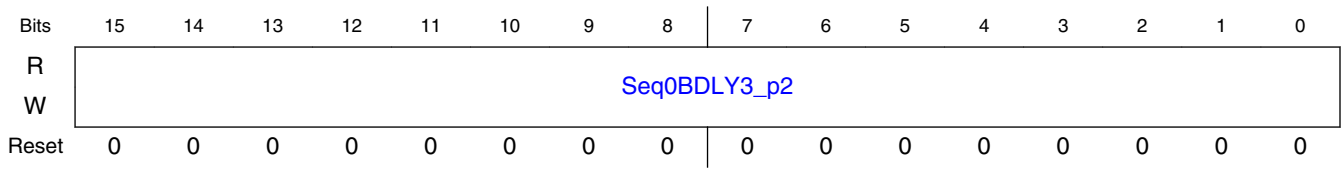
Field	Function
15-0 Seq0BDLY2_p2	PHY Initialization Engine (PIE) Delay Register 2 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 2 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.196 PHY Initialization Engine (PIE) Delay Register 3 (Seq0BDLY3_p2)

9.3.3.6.196.1 Offset

Register	Offset
Seq0BDLY3_p2	40_001Ch

9.3.3.6.196.2 Diagram



9.3.3.6.196.3 Fields

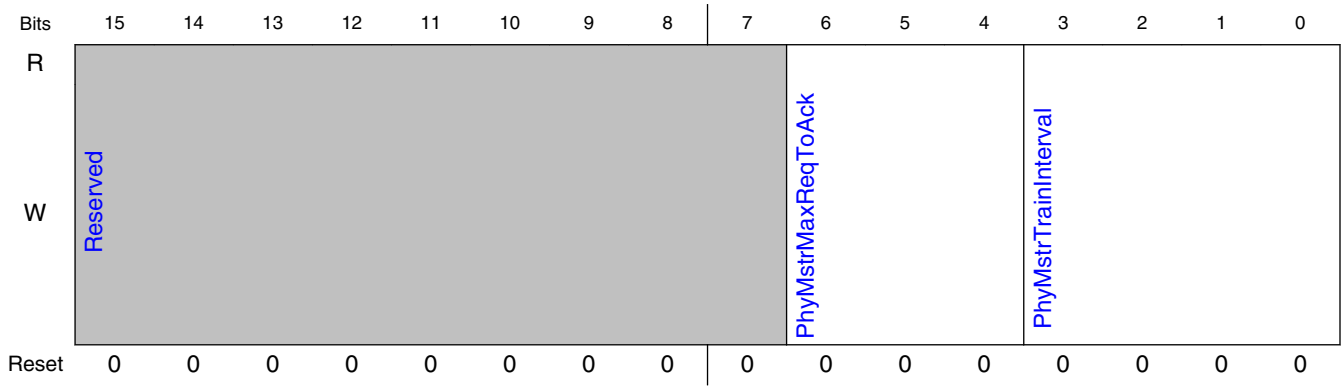
Field	Function
15-0 Seq0BDLY3_p2	PHY Initialization Engine (PIE) Delay Register 3 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 3 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.197 Setup Intervals for DFI PHY Master operations (PPTTrainSetup_p2)

9.3.3.6.197.1 Offset

Register	Offset
PPTTrainSetup_p2	40_0020h

9.3.3.6.197.2 Diagram



9.3.3.6.197.3 Fields

Field	Function
15-7 —	Reserved
6-4 PhyMstrMaxReqToAck	Bits 6:4 of this register specify the max time from tdfi_phymstr_req asserted to tdfi_phymstr_ack asserted. Bits 6:4 of this register specify the max time from tdfi_phymstr_req asserted to tdfi_phymstr_ack asserted. 3'b000 Disable PHY Master Interface 3'b001 set tPHYMSTR_resp. = 512 MEMCLKs 3'b010 set tPHYMSTR_resp. = 1024 MEMCLKs 3'b011 set tPHYMSTR_resp. = 2048 MEMCLKs 3'b100 set tPHYMSTR_resp. = 4096 MEMCLKs 3'b101 set tPHYMSTR_resp. = 8192 MEMCLKs 3'b110 set tPHYMSTR_resp. = 32768 MEMCLKs 3'b111 set tPHYMSTR_resp. = undefined
3-0 PhyMstrTrainInterval	Bits 3:0 of this register specifies the time between the end of one training and the start of the next. Bits 3:0 of this register specifies the time between the end of one training and the start of the next. (As per section 11 of the Preliminary DFI 4.0 Specification, V2, it is the max expected time from dfi_init_complete asserted to tdfi_phymstr_ack asserted). 4'b0000 Disable PHY Master Interface 4'b0001 PHY MASTER Request Interval = 524288 MEMCLKs 4'b0010 PHY MASTER Request Interval = 1048576 MEMCLKs

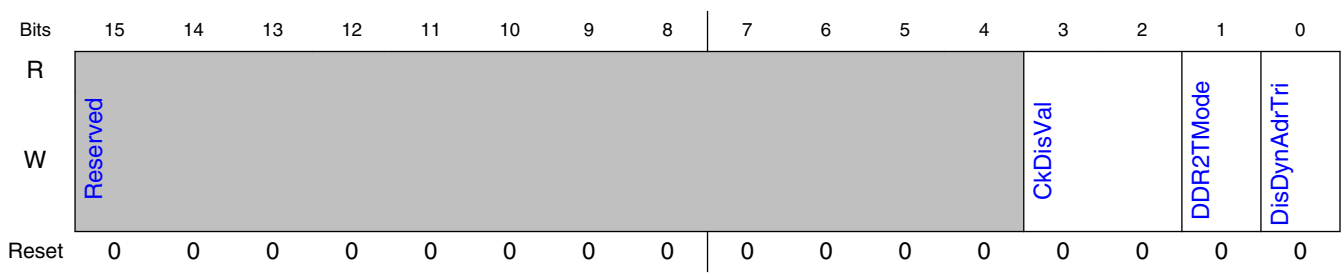
Field	Function
	4'b0011 PHY MASTER Request Interval = 2097152 MEMCLKs
	4'b0100 PHY MASTER Request Interval = 4194304 MEMCLKs
	4'b0101 PHY MASTER Request Interval = 8388608 MEMCLKs
	4'b0110 PHY MASTER Request Interval = 16777216 MEMCLKs
	4'b0111 PHY MASTER Request Interval = 33554432 MEMCLKs
	4'b1000 PHY MASTER Request Interval = 67108864 MEMCLKs
	4'b1001 PHY MASTER Request Interval = 134217728 MEMCLKs
	4'b1010 PHY MASTER Request Interval = 268435456 MEMCLKs
	4'b1011 - 4'b1111 PHY MASTER Request Interval = undefined

9.3.3.6.198 Mode select register for MEMCLK/Address/Command Tristates (TristateModeCA_p2)

9.3.3.6.198.1 Offset

Register	Offset
TristateModeCA_p2	40_0032h

9.3.3.6.198.2 Diagram



9.3.3.6.198.3 Fields

Field	Function
15-4 —	Reserved
3-2 CkDisVal	The PHY provides 4 memory clocks, n=0. The PHY provides 4 memory clocks, n=0..3. When the toggling of memory clock CK_t[n] is disabled with dfi_clk_disable[n]=1,

Table continues on the next page...

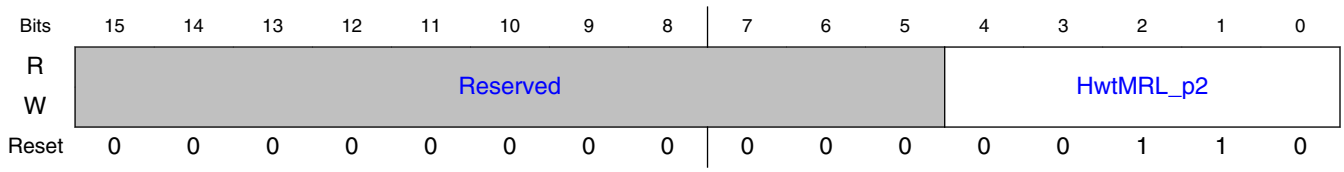
Field	Function
	the memory clock CK_t[n] is driven with Register CkDisVal[1]. When the toggling of memory clock CK_c[n] is disabled with dfi_clk_disable[n]=1, the memory clock CK_c[n] is driven with ~(Register CkDisVal[0]) (ie inverted). Note that the non-toggling differential memory clock CK_t[n],CK_c[n] may be driven with any combination, LL,LH,HL,HH; the default CK_t[n],CK_c[n]=LH.
1 DDR2TMode	Must be set to 1 for Dynamic Tristate to work when CA bus is 2T or Geardown mode. Must be set to 1 for Dynamic Tristate to work when CA bus is 2T or Geardown mode. This CSR bit has no effect when DisDynAdrTri==1
0 DisDynAdrTri	When DisDynAdrTri=1, Dynamic Tristating is disabled. When DisDynAdrTri=1, Dynamic Tristating is disabled. Dynamic Tristating is on by default. . Dynamic Tristating should be disabled (DisDynAdrTri=0) for these modes: 1. DDR3/2T if the controller cannot follow the 2T PHY tristate protocol. 2. DDR4/2T/2N if the controller cannot follow the 2T PHY tristate protocol. 3. LPDDR4 . When DisDynAdrTri=0 (default), In DDR3 mode, The following SDRAM pins can be dynamically tristated: A,BA,RAS_n,CAS_n,WE_n In DDR4 mode, The following SDRAM pins can be dynamically tristated: A,BA,BG,ACT_n In LPDDR3 mode, The following SDRAM pins can be dynamically tristated: CA[*] . In 1T mode, the PHY will tristate the relevant pins when all ranks are DESelected . In 2T mode (or geardown), the PHY will tristate the relevant pins when: D3 -- All ranks DESelected (all CS_L==1) AND RAS_n,CAS_n,WE_n,BA0 = 1,1,1,0 D4 -- All ranks DESelected (all CS_L==1) AND RAS_n,CAS_n,WE_n,BA0,ACT_n = 1,1,1,0,1 When in this mode, the controller should avoid sending the above patterns with any rank selected. [e.g. NOPs sent by the controller should have BA0 set to a non-zero value]

9.3.3.6.199 HWT MaxReadLatency. (HwtMRL_p2)

9.3.3.6.199.1 Offset

Register	Offset
HwtMRL_p2	40_0040h

9.3.3.6.199.2 Diagram



9.3.3.6.199.3 Fields

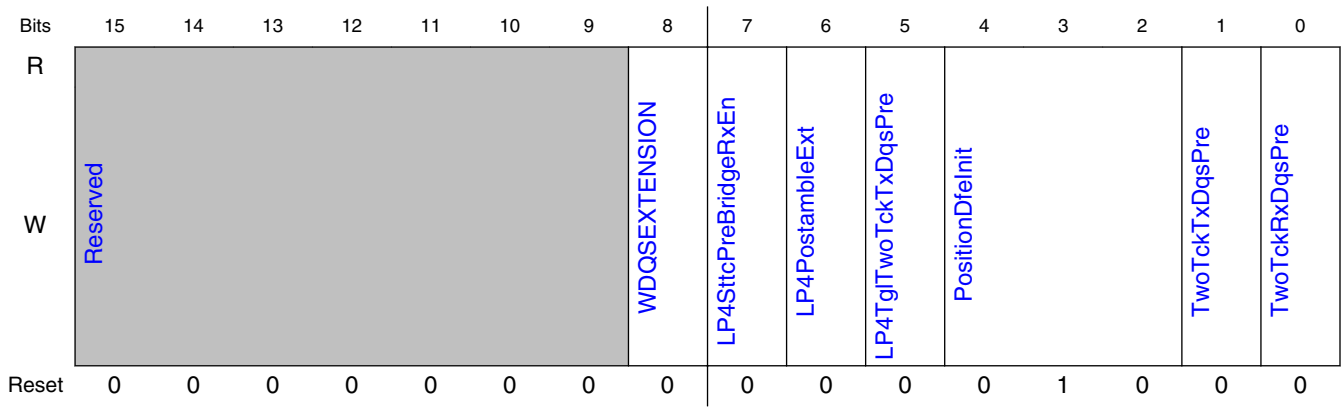
Field	Function
15-5 —	Reserved
4-0 HwtMRL_p2	This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. The CSR is in units of two mem clocks; that is, a unit change in the LSB is a change in MRL of two mem clocks. This MASTER copy of MRL is used by the PHY training hardware only.

9.3.3.6.200 Control the PHY logic related to the read and write DQS preamble (DqsPreambleControl_p2)

9.3.3.6.200.1 Offset

Register	Offset
DqsPreambleControl_p2	40_0048h

9.3.3.6.200.2 Diagram



9.3.3.6.200.3 Fields

Field	Function
15-9 —	Reserved
8 WDQSEXTENSION	When set, DQS_T and DQS_C will be driven differentially to 0 and 1, respectively, before and after a write burst, except during a memory read transaction. When set, DQS_T and DQS_C will be driven differentially to 0 and 1, respectively, before and after a write burst, except during a memory read transaction. See DesignWare Cores LPDDR4 MultiPHY: WDQS Extension Application Note. Use of WDQSEXTENSION requires POdtTailWidth=3 and POdtStartDelay=00
7 LP4SttcPreBridgeRxEn	Used in LPDDR4 static-preamble mode to bridge the RxEn between two reads to the same timing group when the bubble is 1 memclk. Used in LPDDR4 static-preamble mode to bridge the RxEn between two reads to the same timing group when the bubble is 1 memclk. LPDDR4 static-preamble mode is set in the DRAMs with MR1, OP[3]=0.
6 LP4PostambleExt	In LPDDR4 mode must be set to extend the write postamble. In LPDDR4 mode must be set to extend the write postamble. 0: half-memclk postamble, as in D4 1: one-and-one-half-memclk postamble; see LPDDR4 Spec MR3, OP[1] WR PST, vendor-specific function.
5 LP4TglTwoTckTxDqsPre	Used in LPDDR4 mode to modify the early preamble when Register TwoTckTxDqsPre=1 0: level first-memclk preamble 1: toggling first-memclk preamble
4-2 PositionDfeInit	For DDR4 phy only when receive DFE is enabled. For DDR4 phy only when receive DFE is enabled. PositionDfeInit[2]=1 causes DfeInit to be asserted late PositionDfeInit[1]=1 causes DfeInit to be asserted at the nominal time

Table continues on the next page...

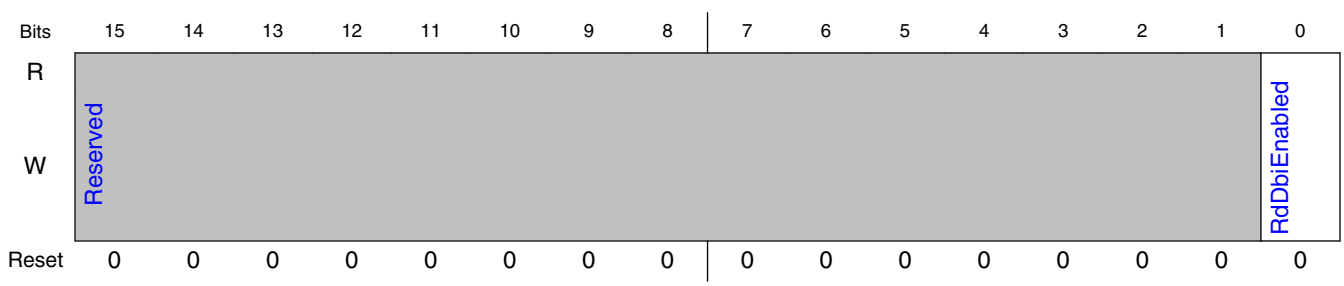
Field	Function
	PositionDfeInit[0]=1 causes DfeInit to be asserted early PositionDfeInit=3'b010 is the nominal, recommended value for leading edge used by receiver.
1 TwoTckTxDqsPre	0: Standard 1tck TxDqs Preamble 1: Enable Optional D4 2tck TxDqs Preamble The DDR4 MR4 A12 is Write Preamble, 1=2nCK, 0=1nCK.
0 TwoTckRxDqsPre	Widens the RxDqsEn window to allow larger drift in the incoming read DQS to take advantage of the larger/wider preamble generated by the DRAMSs when the D4 DRAMS are configured with DDR4 MR4 A11 Read Preamble=1 for causing a 2nCK read preamble. Widens the RxDqsEn window to allow larger drift in the incoming read DQS to take advantage of the larger/wider preamble generated by the DRAMSs when the D4 DRAMS are configured with DDR4 MR4 A11 Read Preamble=1 for causing a 2nCK read preamble. For LPDDR4, all read operations are 2nCK such that this control must be set to 1. Note the nominal trained-to-center point will be earlier relative to the first strobing edge of DQS than when not in this mode.

9.3.3.6.201 This Register is used to enable the Read-DBI function in each DBYTE (DMIPinPresent_p2)

9.3.3.6.201.1 Offset

Register	Offset
DMIPinPresent_p2	40_005Ah

9.3.3.6.201.2 Diagram



9.3.3.6.201.3 Fields

Field	Function
15-1	Reserved

Table continues on the next page...

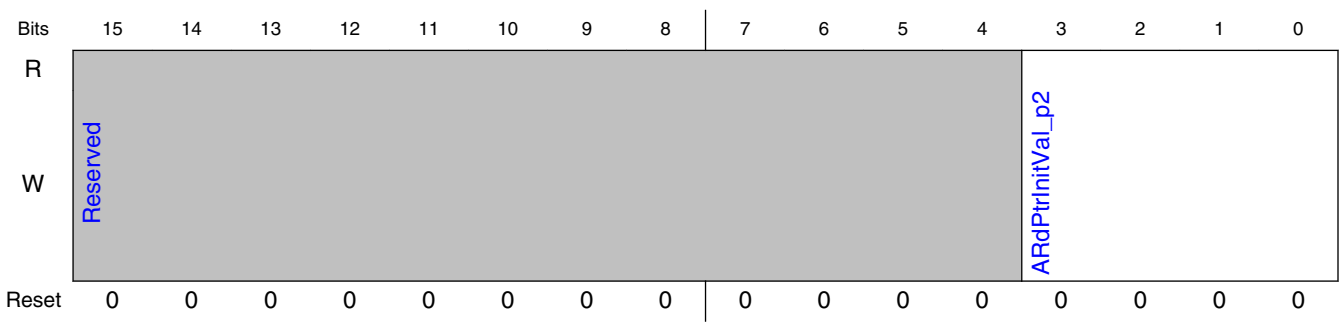
Field	Function
—	
0 RdDbiEnabled	This bit must be set to 1'b1 if Read-DBI is enabled in a connected DDR4 or LPDDR4 device. This bit must be set to 1'b1 if Read-DBI is enabled in a connected DDR4 or LPDDR4 device. If set, the following DRAM MR should also be set [DDR4.MR5.A12=1 or LPDDR4.MR3.OP[7]=1]

9.3.3.6.202 Address/Command FIFO ReadPointer Initial Value (ARdPtrInitVal_p2)

9.3.3.6.202.1 Offset

Register	Offset
ARdPtrInitVal_p2	40_005Ch

9.3.3.6.202.2 Diagram



9.3.3.6.202.3 Fields

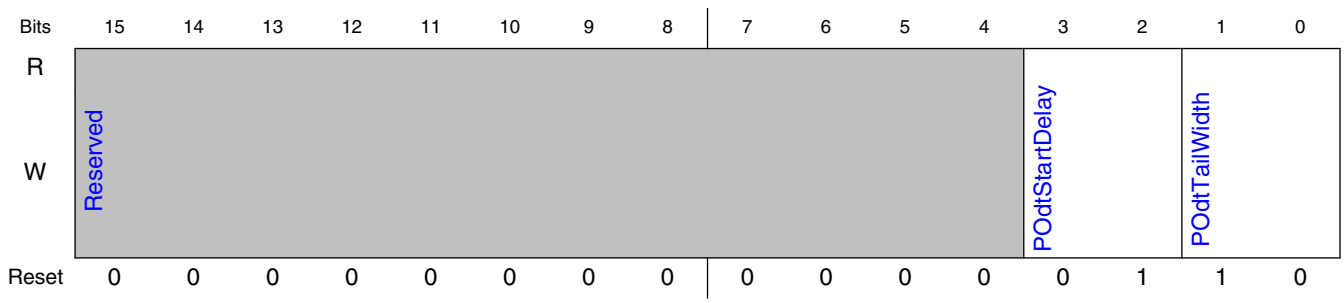
Field	Function
15-4 —	Reserved
3-0 ARdPtrInitVal_p2	This is the initial Pointer Offset for the free-running FIFOs in the DBYTE and ACX4 hardips. This is the initial Pointer Offset for the free-running FIFOs in the DBYTE and ACX4 hardips. The units of this register are in UI. The pointer separation should be chosen to compensate for all sources of skew and drift of the PHY DFICLK and PCLK networks. Please see PUB databook section 8.1.1 This CSR must be programmed in Step C of the PHY Initialization sequence

9.3.3.6.203 READ DATA On-Die Termination Timing Control (by PHY) (ProcOdtTimeCtl_p2)

9.3.3.6.203.1 Offset

Register	Offset
ProcOdtTimeCtl_p2	40_00ACh

9.3.3.6.203.2 Diagram



9.3.3.6.203.3 Fields

Field	Function
15-4 —	Reserved
3-2 POdtStartDelay	controls the start of ProcOdt, units of UI 3 delay start 2 UI, maximum delay of start of ProcOdt 2 delay start 1 UI, 1 delay start 0 UI, default 0 early by 1 UI, The time from ProcODT assertion to opening the window to receive DQS is (10 - POdtStartDelay) UI.
1-0 POdtTailWidth	controls the length of the tail of ProcOdt, units of UI 3 tail 3UI more than for Register POdtTailWidth=0, maximum 2 tail 2UI more than for Register POdtTailWidth=0, default 1 tail 1UI more than for Register POdtTailWidth=0 0 minimum length tail The time from ProcODT to closing the window to receive DQS to ProcODT POdtTailWidth is (2 + POdtTailWidth) UI

9.3.3.6.204 DLL gain control (DllGainCtl_p2)

9.3.3.6.204.1 Offset

Register	Offset
DllGainCtl_p2	40_00F8h

9.3.3.6.204.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				DllSeedSel				DllGainTV				DllGainIV			
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	1

9.3.3.6.204.3 Fields

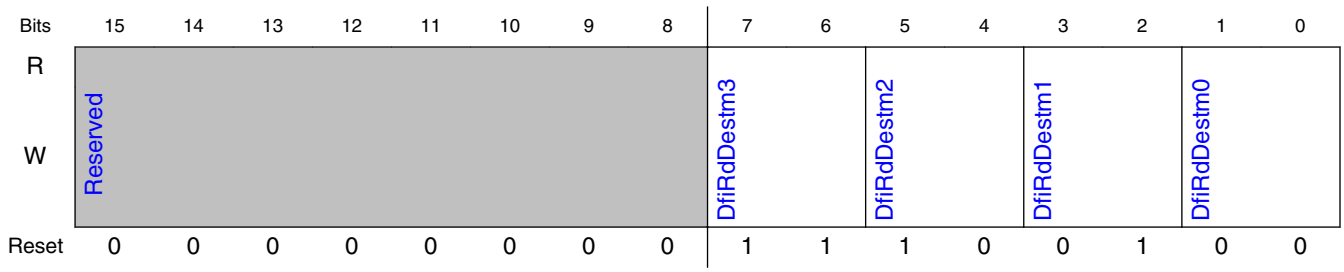
Field	Function
15-12 —	Reserved
11-8 DllSeedSel	Reserved, must be configured to be 0.
7-4 DllGainTV	Terminal value of DllGain, ie the value in effect when locking is done and the value used for maintaining lock, ie tracking pclk variation. Terminal value of DllGain, ie the value in effect when locking is done and the value used for maintaining lock, ie tracking pclk variation. DllGainTV must be greater than or equal to DllGainIV. The maximum value is 10. The minimum value is 6.
3-0 DllGainIV	Initial value of DllGain.

9.3.3.6.205 Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiRdDataCsDestMap_p2)

9.3.3.6.205.1 Offset

Register	Offset
DfiRdDataCsDestMap_p2	40_0160h

9.3.3.6.205.2 Diagram



9.3.3.6.205.3 Fields

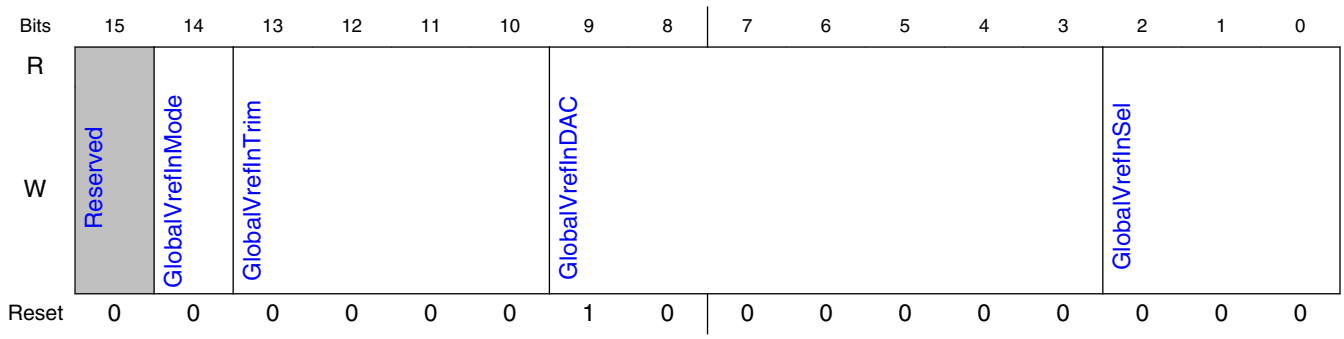
Field	Function
15-8 —	Reserved
7-6 DfiRdDestm3	Maps dfi_rddata_cs_n_p0[3] to dest DfiRdDestm3 timing For example, if 3 dfi_rddata_cs_n_p0[3] will use Register RxEn,ClkDlyTg3 timing.
5-4 DfiRdDestm2	Maps dfi_rddata_cs_n_p0[2] to dest DfiRdDestm2 timing For example, if 2 dfi_rddata_cs_n_p0[2] will use Register RxEn,ClkDlyTg2 timing.
3-2 DfiRdDestm1	Maps dfi_rddata_cs_n_p0[1] to dest DfiRdDestm1 timing For example, if 1 dfi_rddata_cs_n[_p01] will use Register RxEn,ClkDlyTg1 timing.
1-0 DfiRdDestm0	Maps dfi_rddata_cs_n_p0[0] to dest DfiRdDestm0 timing For example, if 0 dfi_rddata_cs_n_p0[0] will use Register RxEn,ClkDlyTg0 timing.

9.3.3.6.206 PHY Global Vref Controls (VrefInGlobal_p2)

9.3.3.6.206.1 Offset

Register	Offset
VrefInGlobal_p2	40_0164h

9.3.3.6.206.2 Diagram



9.3.3.6.206.3 Fields

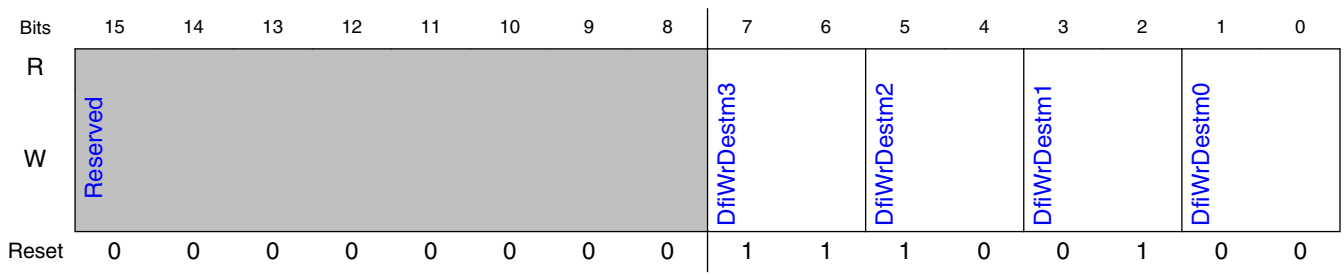
Field	Function
15 —	Reserved
14 GlobalVrefInMode	RSVD
13-10 GlobalVrefInTrim	RSVD
9-3 GlobalVrefInDAC	DAC code for internal Vref generation The DAC has two ranges; the range is set by GlobalVrefInSel[2] ========== RANGE0 : DDR3,DDR4,LPDDR3 [GlobalVrefInSel[2] = 0] DAC Output Voltage = GlobalVrefInDAC == 6'h00 ? Hi-Z : 0. DAC code for internal Vref generation The DAC has two ranges; the range is set by GlobalVrefInSel[2] ========== RANGE0 : DDR3,DDR4,LPDDR3 [GlobalVrefInSel[2] = 0] DAC Output Voltage = GlobalVrefInDAC == 6'h00 ? Hi-Z : 0.345*VDDQ + (0.005*GlobalVrefInDAC)*VDDQ ========== RANGE1 : LPDDR4 [GlobalVrefInSel[2] = 1] DAC Output Voltage = GlobalVrefInDAC == 6'h00 ? Hi-Z : (0.005*(GlobalVrefInDAC-1))*VDDQ
2-0 GlobalVrefInSel	GlobalVrefInSel[1:0] controls the mode of the PHY VREF DAC and the BP_VREF pin ========== 2'b00 - PHY Vref DAC Range0 -- BP_VREF = Hi-Z 2'b01 - Reserved Encoding 2'b10 - PHY Vref DAC Range0 -- BP_VREF connected to PLL Analog Bus 2'b11 - PHY Vref DAC Range0 -- BP_VREF connected to PHY Vref DAC ========== GlobalVrefInSel[2] shall be set according to Dram Protocol: Protocol GlobalVrefInSel[2] ---------- DDR3 1'b0 DDR4 1'b0 LPDDR3 1'b0 LPDDR4 1'b1

9.3.3.6.207 Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiWrDataCsDestMap_p2)

9.3.3.6.207.1 Offset

Register	Offset
DfiWrDataCsDestMap_p2	40_0168h

9.3.3.6.207.2 Diagram



9.3.3.6.207.3 Fields

Field	Function
15-8 —	Reserved
7-6 DfiWrDestm3	Maps dfi_wrdata_cs_n_p0[3] to dest DfiWrDestm3 timing (use Register TxDq,DqsDlyTg3) For example, if 3 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg3 timing.
5-4 DfiWrDestm2	Maps dfi_wrdata_cs_n_p0[2] to dest DfiWrDestm2 timing For example, if 2 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg2 timing.
3-2 DfiWrDestm1	Maps dfi_wrdata_cs_n_p0[1] to dest DfiWrDestm1 timing For example, if 1 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg1 timing.
1-0 DfiWrDestm0	Maps dfi_wrdata_cs_n_p0[0] to dest DfiWrDestm0 timing For example, if 0 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg0 timing.

9.3.3.6.208 PState dependent PLL Control Register 2 (PllCtrl2_p2)

9.3.3.6.208.1 Offset

Register	Offset
PllCtrl2_p2	40_018Ah

9.3.3.6.208.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											PllFreqSel				
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1

9.3.3.6.208.3 Fields

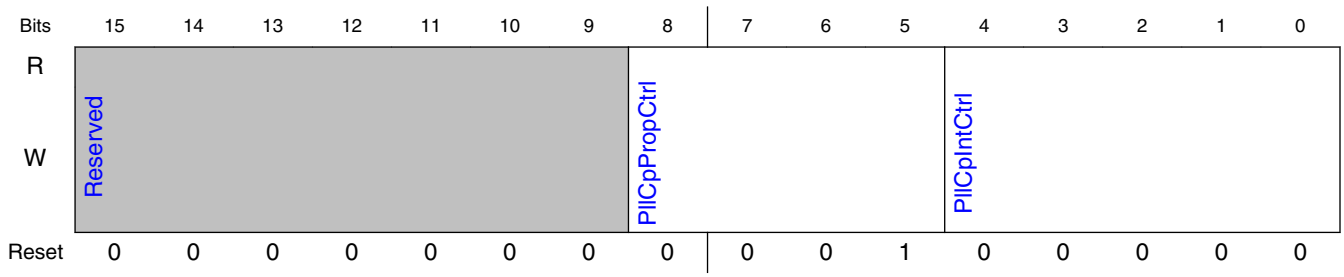
Field	Function
15-5 —	Reserved
4-0 PllFreqSel	Adjusts the loop parameters to compensate for different VCO bias points, and input/output clock division ratios. Adjusts the loop parameters to compensate for different VCO bias points, and input/output clock division ratios. Program based on reference clock freqency (DfiClk) with this table.

9.3.3.6.209 PState dependent PLL Control Register 1 (PllCtrl1_p2)

9.3.3.6.209.1 Offset

Register	Offset
PllCtrl1_p2	40_018Eh

9.3.3.6.209.2 Diagram



9.3.3.6.209.3 Fields

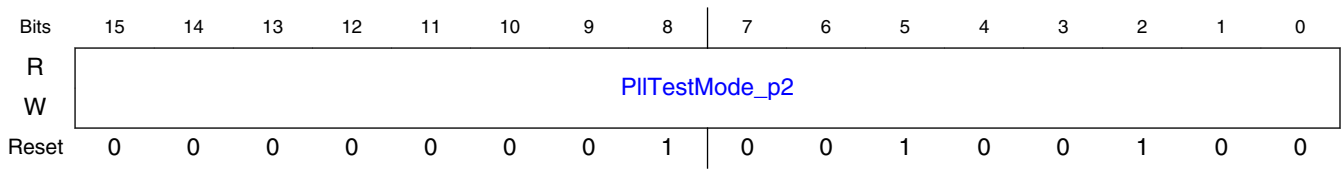
Field	Function
15-9 —	Reserved
8-5 PllCpPropCtrl	connects directly to cp_prop_cntrl<3:0> of PLL. connects directly to cp_prop_cntrl<3:0> of PLL. Charge pump proportional current control.
4-0 PllCpIntCtrl	connects directly to cp_int_cntrl<1:0> in PLL. connects directly to cp_int_cntrl<1:0> in PLL. Charge pump integrating current control.

9.3.3.6.210 Additional controls for PLL CP/VCO modes of operation (PllTestMode_p2)

9.3.3.6.210.1 Offset

Register	Offset
PllTestMode_p2	40_0194h

9.3.3.6.210.2 Diagram



9.3.3.6.210.3 Fields

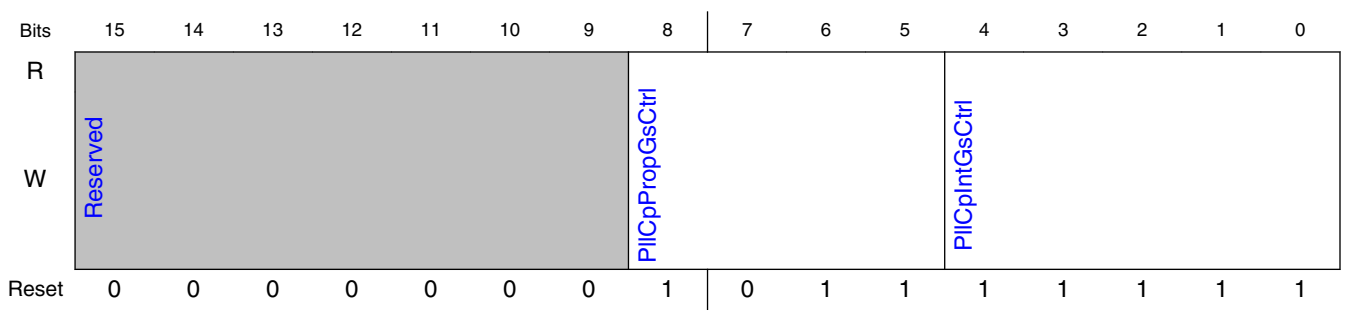
Field	Function
15-0 PllTestMode_p2	It is required to use default values for this CSR Directly connected to testmode[15:0] pin of PLL

9.3.3.6.211 PState dependent PLL Control Register 4 (PllCtrl4_p2)

9.3.3.6.211.1 Offset

Register	Offset
PllCtrl4_p2	40_0198h

9.3.3.6.211.2 Diagram



9.3.3.6.211.3 Fields

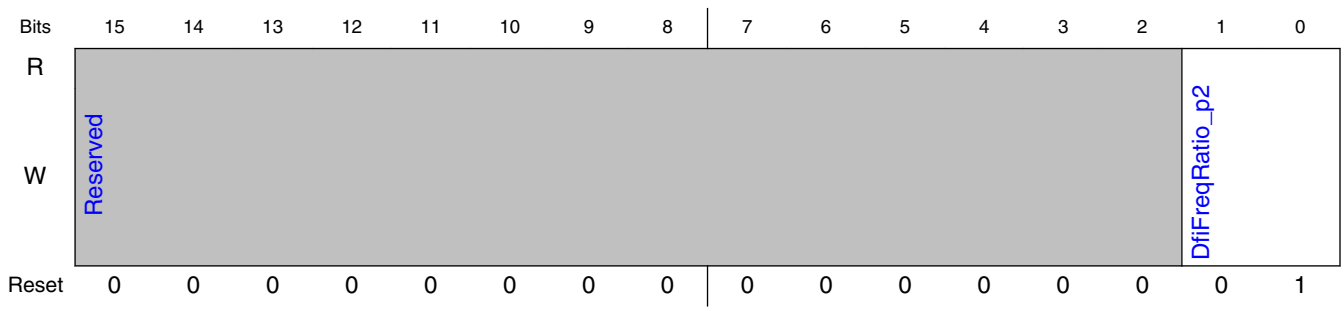
Field	Function
15-9 —	Reserved
8-5 PllCpPropGsCtrl	connects directly to cp_prop_gs_cntrl<3:0> of PLL. connects directly to cp_prop_gs_cntrl<3:0> of PLL. Charge pump proportional current control for fast relock and gearshift.
4-0 PllCpIntGsCtrl	connects directly to cp_int_gs_cntrl<4:0> in PLL. connects directly to cp_int_gs_cntrl<4:0> in PLL. Charge pump integrating current control for fast relock and gearshift.

9.3.3.6.212 DFI Frequency Ratio (DfiFreqRatio_p2)

9.3.3.6.212.1 Offset

Register	Offset
DfiFreqRatio_p2	40_01F4h

9.3.3.6.212.2 Diagram



9.3.3.6.212.3 Fields

Field	Function
15-2 —	Reserved
1-0 DfiFreqRatio_p2	Used in dwc_ddrphy_pub_serdes to serialize or de-serialize DFI signals 00 = 1:1 mode 01 = 1:2 mode 1x = 1:4 mode* *Note: 1:4 is for future pub revision.

9.3.3.6.213 Impedance Calibration Clock Ratio (CalUclkInfo_p3)

9.3.3.6.213.1 Offset

Register	Offset
CalUclkInfo_p3	60_0010h

9.3.3.6.213.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						CalUClkTicksPer1uS									
Reset	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0

9.3.3.6.213.3 Fields

Field	Function
15-10 —	Reserved
9-0 CalUClkTicksPer1uS	Must be programmed to the number of DfiClks in 1us (rounded up), with minimum value of 24. Must be programmed to the number of DfiClks in 1us (rounded up), with minimum value of 24. if (DfiClk < 24MHz) CalUclkInfo = 24 else CalUclkInfo = (number of DfiClks in 1us)

9.3.3.6.214 PHY Initialization Engine (PIE) Delay Register 0 (Seq0BDLY0_p3)

9.3.3.6.214.1 Offset

Register	Offset
Seq0BDLY0_p3	60_0016h

9.3.3.6.214.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Seq0BDLY0_p3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.214.3 Fields

Field	Function
15-0 Seq0BDLY0_p3	PHY Initialization Engine (PIE) Delay Register 0 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value.

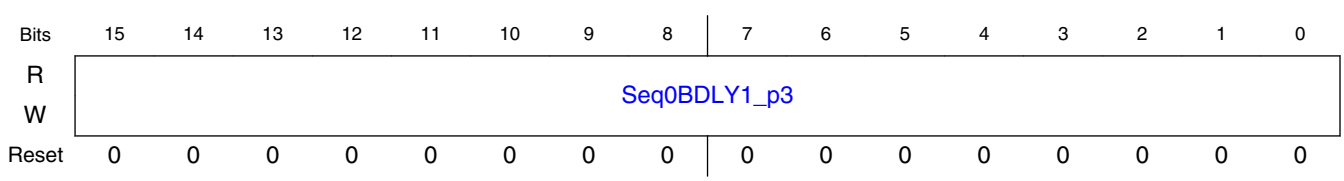
Field	Function
	PHY Initialization Engine (PIE) Delay Register 0 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.215 PHY Initialization Engine (PIE) Delay Register 1 (Seq0BDLY1_p3)

9.3.3.6.215.1 Offset

Register	Offset
Seq0BDLY1_p3	60_0018h

9.3.3.6.215.2 Diagram



9.3.3.6.215.3 Fields

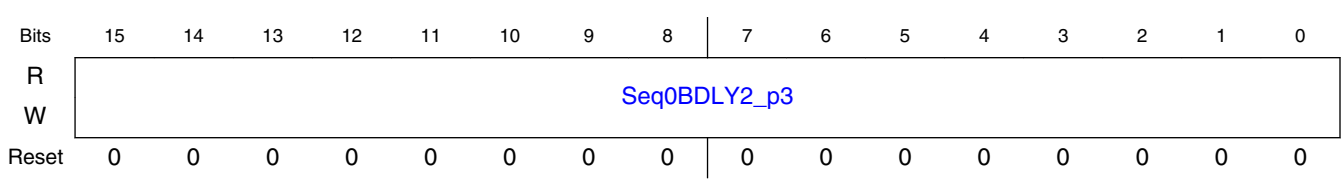
Field	Function
15-0 Seq0BDLY1_p3	PHY Initialization Engine (PIE) Delay Register 1 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 1 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.216 PHY Initialization Engine (PIE) Delay Register 2 (Seq0BDLY2_p3)

9.3.3.6.216.1 Offset

Register	Offset
Seq0BDLY2_p3	60_001Ah

9.3.3.6.216.2 Diagram



9.3.3.6.216.3 Fields

Field	Function
15-0 Seq0BDLY2_p3	PHY Initialization Engine (PIE) Delay Register 2 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 2 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.217 PHY Initialization Engine (PIE) Delay Register 3 (Seq0BDLY3_p3)

9.3.3.6.217.1 Offset

Register	Offset
Seq0BDLY3_p3	60_001Ch

9.3.3.6.217.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Seq0BDLY3_p3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.3.6.217.3 Fields

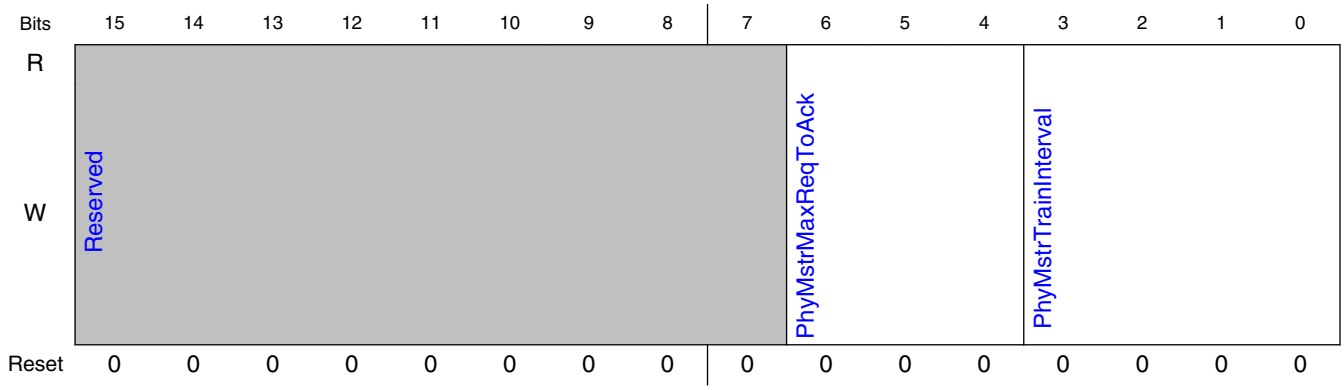
Field	Function
15-0 Seq0BDLY3_p3	PHY Initialization Engine (PIE) Delay Register 3 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. PHY Initialization Engine (PIE) Delay Register 3 This register is available for selection by the NOP and WAIT instructions in the PIE for the delay value. Programmed by dwc_ddrphy_phyinit_I_loadPIEImage() to support frequency changes. Programming Seq0BDLY[3,2,1,0] Registers: --- These registers are used to control various delays during PLL, DLL initialization. These regiseters should be programmed with the corresponding number of MEMCLK cycles representing a fixed amount of delay irrespective of DfiClk Frequency. When calculating the number of cycles to be programmed for each PState, it should be noted that the sequencer multiplies programmed values by a factor of 8. Register Required Delay Programmed Value --- --- --- Seq0BDLY0 0.5us (0.5us / MEMCLK period) / 8 Seq0BDLY1 1.0us (1.0us / MEMCLK period) / 8 Seq0BDLY2 10.0us (10.0us / MEMCLK period) / 8 64 if (MEMCLK Freq <= 400MHz) Seq0BDLY3 - 132 if (400MHz Freq < MEMCLK Freq <= 533MHz) 176 if (MEMCLK Freq > 533MHz)

9.3.3.6.218 Setup Intervals for DFI PHY Master operations (PPTTrainSetup_p3)

9.3.3.6.218.1 Offset

Register	Offset
PPTTrainSetup_p3	60_0020h

9.3.3.6.218.2 Diagram



9.3.3.6.218.3 Fields

Field	Function
15-7 —	Reserved
6-4 PhyMstrMaxReqToAck	Bits 6:4 of this register specify the max time from tdfi_phymstr_req asserted to tdfi_phymstr_ack asserted. Bits 6:4 of this register specify the max time from tdfi_phymstr_req asserted to tdfi_phymstr_ack asserted. 3'b000 Disable PHY Master Interface 3'b001 set tPHYMSTR_resp. = 512 MEMCLKs 3'b010 set tPHYMSTR_resp. = 1024 MEMCLKs 3'b011 set tPHYMSTR_resp. = 2048 MEMCLKs 3'b100 set tPHYMSTR_resp. = 4096 MEMCLKs 3'b101 set tPHYMSTR_resp. = 8192 MEMCLKs 3'b110 set tPHYMSTR_resp. = 32768 MEMCLKs 3'b111 set tPHYMSTR_resp. = undefined
3-0 PhyMstrTrainInterval	Bits 3:0 of this register specifies the time between the end of one training and the start of the next. Bits 3:0 of this register specifies the time between the end of one training and the start of the next. (As per section 11 of the Preliminary DFI 4.0 Specification, V2, it is the max expected time from dfi_init_complete asserted to tdfi_phymstr_ack asserted). 4'b0000 Disable PHY Master Interface 4'b0001 PHY MASTER Request Interval = 524288 MEMCLKs 4'b0010 PHY MASTER Request Interval = 1048576 MEMCLKs

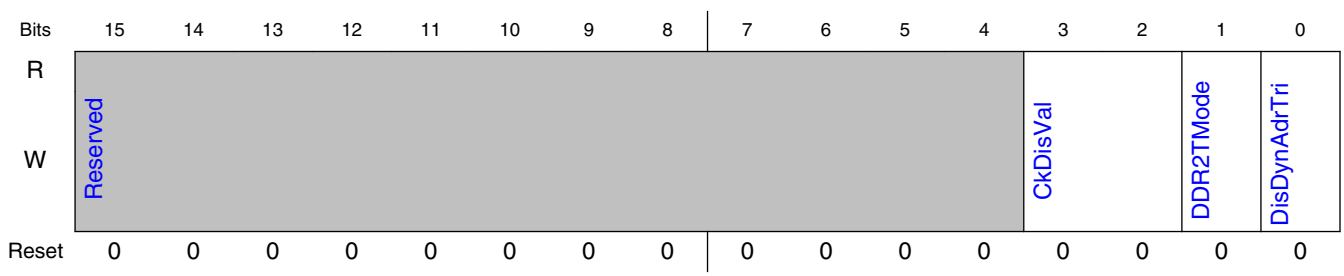
Field	Function
	4'b0011 PHY MASTER Request Interval = 2097152 MEMCLKs
	4'b0100 PHY MASTER Request Interval = 4194304 MEMCLKs
	4'b0101 PHY MASTER Request Interval = 8388608 MEMCLKs
	4'b0110 PHY MASTER Request Interval = 16777216 MEMCLKs
	4'b0111 PHY MASTER Request Interval = 33554432 MEMCLKs
	4'b1000 PHY MASTER Request Interval = 67108864 MEMCLKs
	4'b1001 PHY MASTER Request Interval = 134217728 MEMCLKs
	4'b1010 PHY MASTER Request Interval = 268435456 MEMCLKs
	4'b1011 - 4'b1111 PHY MASTER Request Interval = undefined

9.3.3.6.219 Mode select register for MEMCLK/Address/Command Tristates (TristateModeCA_p3)

9.3.3.6.219.1 Offset

Register	Offset
TristateModeCA_p3	60_0032h

9.3.3.6.219.2 Diagram



9.3.3.6.219.3 Fields

Field	Function
15-4 —	Reserved
3-2 CkDisVal	The PHY provides 4 memory clocks, n=0. The PHY provides 4 memory clocks, n=0..3. When the toggling of memory clock CK_t[n] is disabled with dfi_clk_disable[n]=1,

Table continues on the next page...

Field	Function
	the memory clock CK_t[n] is driven with Register CkDisVal[1]. When the toggling of memory clock CK_c[n] is disabled with dfi_clk_disable[n]=1, the memory clock CK_c[n] is driven with ~(Register CkDisVal[0]) (ie inverted). Note that the non-toggling differential memory clock CK_t[n],CK_c[n] may be driven with any combination, LL,LH,HL,HH; the default CK_t[n],CK_c[n]=LH.
1 DDR2TMode	Must be set to 1 for Dynamic Tristate to work when CA bus is 2T or Geardown mode. Must be set to 1 for Dynamic Tristate to work when CA bus is 2T or Geardown mode. This CSR bit has no effect when DisDynAdrTri==1
0 DisDynAdrTri	When DisDynAdrTri=1, Dynamic Tristating is disabled. When DisDynAdrTri=1, Dynamic Tristating is disabled. Dynamic Tristating is on by default. . Dynamic Tristating should be disabled (DisDynAdrTri=0) for these modes: 1. DDR3/2T if the controller cannot follow the 2T PHY tristate protocol. 2. DDR4/2T/2N if the controller cannot follow the 2T PHY tristate protocol. 3. LPDDR4 . When DisDynAdrTri=0 (default), In DDR3 mode, The following SDRAM pins can be dynamically tristated: A,BA,RAS_n,CAS_n,WE_n In DDR4 mode, The following SDRAM pins can be dynamically tristated: A,BA,BG,ACT_n In LPDDR3 mode, The following SDRAM pins can be dynamically tristated: CA[*] . In 1T mode, the PHY will tristate the relevant pins when all ranks are DESelected . In 2T mode (or geardown), the PHY will tristate the relevant pins when: D3 -- All ranks DESelected (all CS_L==1) AND RAS_n,CAS_n,WE_n,BA0 = 1,1,1,0 D4 -- All ranks DESelected (all CS_L==1) AND RAS_n,CAS_n,WE_n,BA0,ACT_n = 1,1,1,0,1 When in this mode, the controller should avoid sending the above patterns with any rank selected. [e.g. NOPs sent by the controller should have BA0 set to a non-zero value]

9.3.3.6.220 HWT MaxReadLatency. (HwtMRL_p3)

9.3.3.6.220.1 Offset

Register	Offset
HwtMRL_p3	60_0040h

9.3.3.6.220.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											HwtMRL_p3				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

9.3.3.6.220.3 Fields

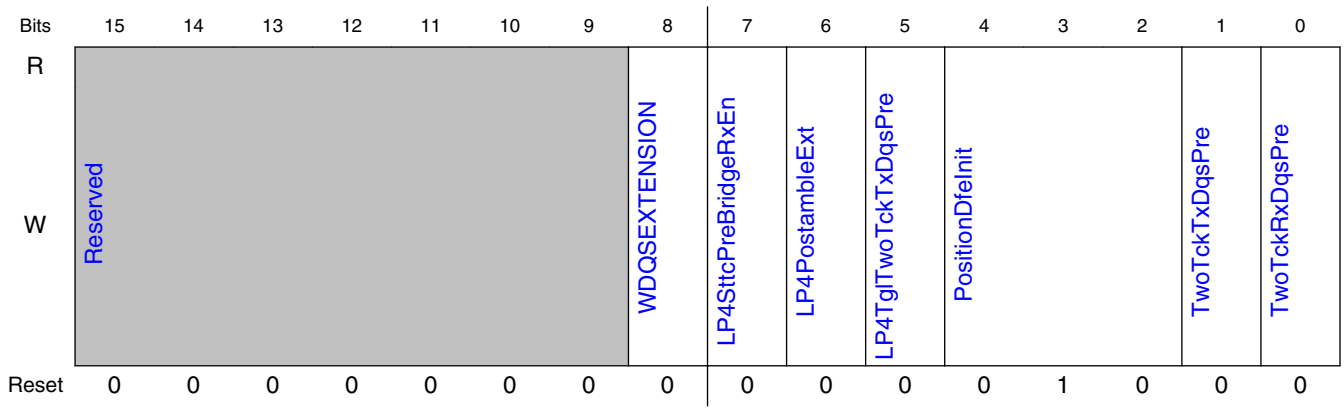
Field	Function
15-5 —	Reserved
4-0 HwtMRL_p3	This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. This Max Read Latency CSR is to be trained to ensure the rx-data fifo is not read until after all dbytes have their read data valid. The CSR is in units of two mem clocks; that is, a unit change in the LSB is a change in MRL of two mem clocks. This MASTER copy of MRL is used by the PHY training hardware only.

9.3.3.6.221 Control the PHY logic related to the read and write DQS preamble (DqsPreambleControl_p3)

9.3.3.6.221.1 Offset

Register	Offset
DqsPreambleControl_p3	60_0048h

9.3.3.6.221.2 Diagram



9.3.3.6.221.3 Fields

Field	Function
15-9 —	Reserved
8 WDQSEXTENSION	When set, DQS_T and DQS_C will be driven differentially to 0 and 1, respectively, before and after a write burst, except during a memory read transaction. When set, DQS_T and DQS_C will be driven differentially to 0 and 1, respectively, before and after a write burst, except during a memory read transaction. See DesignWare Cores LPDDR4 MultiPHY: WDQS Extension Application Note. Use of WDQSEXTENSION requires POdtTailWidth=3 and POdtStartDelay=00
7 LP4SttcPreBridgeRxEn	Used in LPDDR4 static-preamble mode to bridge the RxEn between two reads to the same timing group when the bubble is 1 memclk. Used in LPDDR4 static-preamble mode to bridge the RxEn between two reads to the same timing group when the bubble is 1 memclk. LPDDR4 static-preamble mode is set in the DRAMs with MR1, OP[3]=0.
6 LP4PostambleExt	In LPDDR4 mode must be set to extend the write postamble. In LPDDR4 mode must be set to extend the write postamble. 0: half-memclk postamble, as in D4 1: one-and-one-half-memclk postamble; see LPDDR4 Spec MR3, OP[1] WR PST, vendor-specific function.
5 LP4TglTwoTckTxDqsPre	Used in LPDDR4 mode to modify the early preamble when Register TwoTckTxDqsPre=1 0: level first-memclk preamble 1: toggling first-memclk preamble
4-2 PositionDfeInit	For DDR4 phy only when receive DFE is enabled. For DDR4 phy only when receive DFE is enabled. PositionDfeInit[2]=1 causes DfeInit to be asserted late PositionDfeInit[1]=1 causes DfeInit to be asserted at the nominal time

Table continues on the next page...

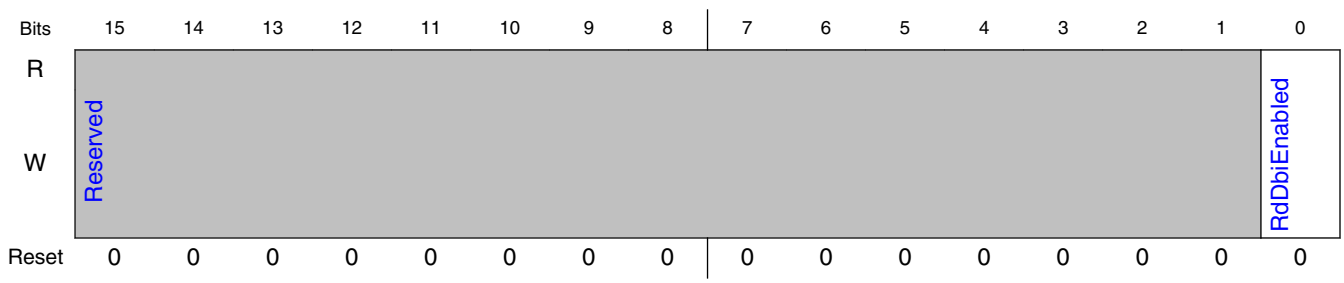
Field	Function
	PositionDfeInit[0]=1 causes DfeInit to be asserted early PositionDfeInit=3'b010 is the nominal, recommended value for leading edge used by receiver.
1 TwoTckTxDqsPre	0: Standard 1tck TxDqs Preamble 1: Enable Optional D4 2tck TxDqs Preamble The DDR4 MR4 A12 is Write Preamble, 1=2nCK, 0=1nCK.
0 TwoTckRxDqsPre	Widens the RxDqsEn window to allow larger drift in the incoming read DQS to take advantage of the larger/wider preamble generated by the DRAMSs when the D4 DRAMS are configured with DDR4 MR4 A11 Read Preamble=1 for causing a 2nCK read preamble. Widens the RxDqsEn window to allow larger drift in the incoming read DQS to take advantage of the larger/wider preamble generated by the DRAMSs when the D4 DRAMS are configured with DDR4 MR4 A11 Read Preamble=1 for causing a 2nCK read preamble. For LPDDR4, all read operations are 2nCK such that this control must be set to 1. Note the nominal trained-to-center point will be earlier relative to the first strobing edge of DQS than when not in this mode.

9.3.3.6.222 This Register is used to enable the Read-DBI function in each DBYTE (DMIPinPresent_p3)

9.3.3.6.222.1 Offset

Register	Offset
DMIPinPresent_p3	60_005Ah

9.3.3.6.222.2 Diagram



9.3.3.6.222.3 Fields

Field	Function
15-1	Reserved

Table continues on the next page...

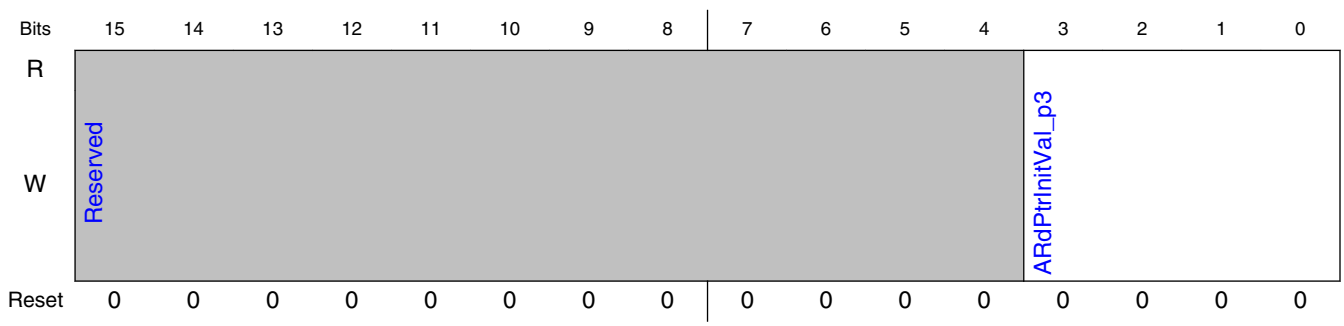
Field	Function
—	
0 RdDbiEnabled	This bit must be set to 1'b1 if Read-DBI is enabled in a connected DDR4 or LPDDR4 device. This bit must be set to 1'b1 if Read-DBI is enabled in a connected DDR4 or LPDDR4 device. If set, the following DRAM MR should also be set [DDR4.MR5.A12=1 or LPDDR4.MR3.OP[7]=1]

9.3.3.6.223 Address/Command FIFO ReadPointer Initial Value (ARdPtrInitVal_p3)

9.3.3.6.223.1 Offset

Register	Offset
ARdPtrInitVal_p3	60_005Ch

9.3.3.6.223.2 Diagram



9.3.3.6.223.3 Fields

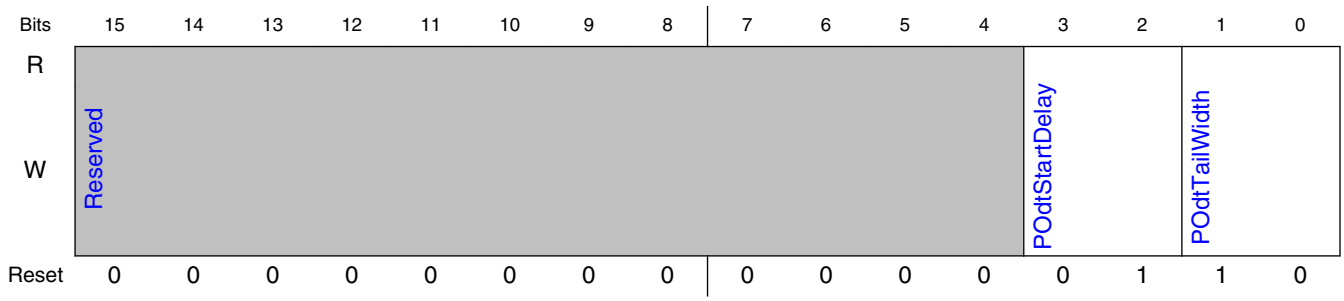
Field	Function
15-4 —	Reserved
3-0 ARdPtrInitVal_p3	This is the initial Pointer Offset for the free-running FIFOs in the DBYTE and ACX4 hardips. This is the initial Pointer Offset for the free-running FIFOs in the DBYTE and ACX4 hardips. The units of this register are in UI. The pointer separation should be chosen to compensate for all sources of skew and drift of the PHY DFICLK and PCLK networks. Please see PUB databook section 8.1.1 This CSR must be programmed in Step C of the PHY Initialization sequence

9.3.3.6.224 READ DATA On-Die Termination Timing Control (by PHY) (ProcOdtTimeCtl_p3)

9.3.3.6.224.1 Offset

Register	Offset
ProcOdtTimeCtl_p3	60_00ACh

9.3.3.6.224.2 Diagram



9.3.3.6.224.3 Fields

Field	Function
15-4 —	Reserved
3-2 POdtStartDelay	controls the start of ProcOdt, units of UI 3 delay start 2 UI, maximum delay of start of ProcOdt 2 delay start 1 UI, 1 delay start 0 UI, default 0 early by 1 UI, The time from ProcODT assertion to opening the window to receive DQS is (10 - POdtStartDelay) UI.
1-0 POdtTailWidth	controls the length of the tail of ProcOdt, units of UI 3 tail 3UI more than for Register POdtTailWidth=0, maximum 2 tail 2UI more than for Register POdtTailWidth=0, default 1 tail 1UI more than for Register POdtTailWidth=0 0 minimum length tail The time from ProcODT to closing the window to receive DQS to ProcODT POdtTailWidth is (2 + POdtTailWidth) UI

9.3.3.6.225 DLL gain control (DllGainCtl_p3)

9.3.3.6.225.1 Offset

Register	Offset
DllGainCtl_p3	60_00F8h

9.3.3.6.225.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				DllSeedSel				DllGainTV				DllGainIV			
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	1

9.3.3.6.225.3 Fields

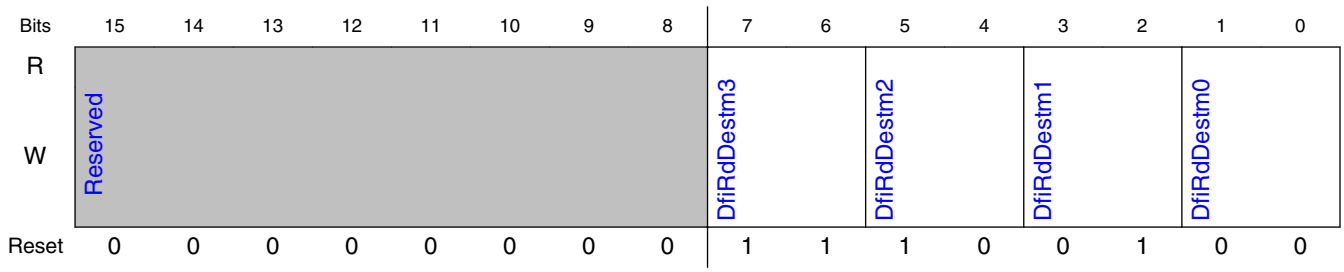
Field	Function
15-12 —	Reserved
11-8 DllSeedSel	Reserved, must be configured to be 0.
7-4 DllGainTV	Terminal value of DllGain, ie the value in effect when locking is done and the value used for maintaining lock, ie tracking pclk variation. Terminal value of DllGain, ie the value in effect when locking is done and the value used for maintaining lock, ie tracking pclk variation. DllGainTV must be greater than or equal to DllGainIV. The maximum value is 10. The minimum value is 6.
3-0 DllGainIV	Initial value of DllGain.

9.3.3.6.226 Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiRdDataCsDestMap_p3)

9.3.3.6.226.1 Offset

Register	Offset
DfiRdDataCsDestMap_p3	60_0160h

9.3.3.6.226.2 Diagram



9.3.3.6.226.3 Fields

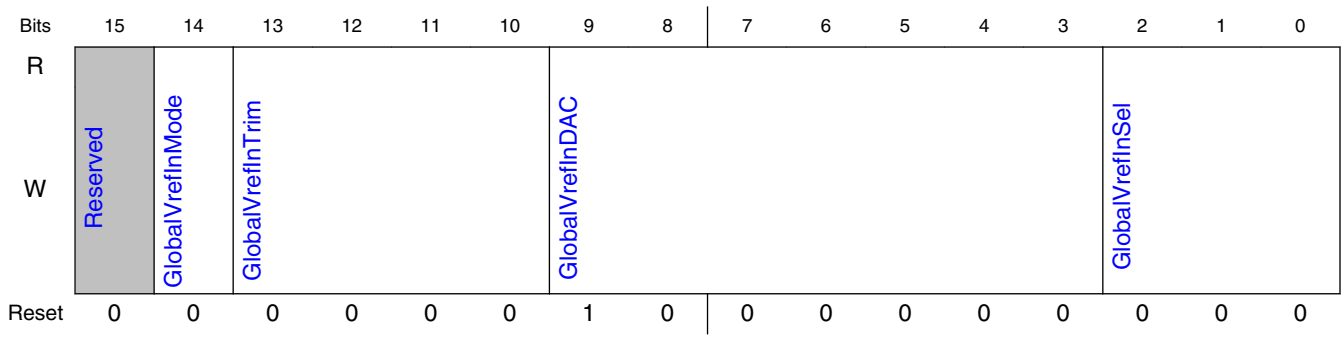
Field	Function
15-8 —	Reserved
7-6 DfiRdDestm3	Maps dfi_rddata_cs_n_p0[3] to dest DfiRdDestm3 timing For example, if 3 dfi_rddata_cs_n_p0[3] will use Register RxEn,ClkDlyTg3 timing.
5-4 DfiRdDestm2	Maps dfi_rddata_cs_n_p0[2] to dest DfiRdDestm2 timing For example, if 2 dfi_rddata_cs_n_p0[2] will use Register RxEn,ClkDlyTg2 timing.
3-2 DfiRdDestm1	Maps dfi_rddata_cs_n_p0[1] to dest DfiRdDestm1 timing For example, if 1 dfi_rddata_cs_n[_p01] will use Register RxEn,ClkDlyTg1 timing.
1-0 DfiRdDestm0	Maps dfi_rddata_cs_n_p0[0] to dest DfiRdDestm0 timing For example, if 0 dfi_rddata_cs_n_p0[0] will use Register RxEn,ClkDlyTg0 timing.

9.3.3.6.227 PHY Global Vref Controls (VrefInGlobal_p3)

9.3.3.6.227.1 Offset

Register	Offset
VrefInGlobal_p3	60_0164h

9.3.3.6.227.2 Diagram



9.3.3.6.227.3 Fields

Field	Function
15 —	Reserved
14 GlobalVrefInMode	RSVD
13-10 GlobalVrefInTrim	RSVD
9-3 GlobalVrefInDAC	DAC code for internal Vref generation The DAC has two ranges; the range is set by GlobalVrefInSel[2] ========== RANGE0 : DDR3,DDR4,LPDDR3 [GlobalVrefInSel[2] = 0] DAC Output Voltage = GlobalVrefInDAC == 6'h00 ? Hi-Z : 0. DAC code for internal Vref generation The DAC has two ranges; the range is set by GlobalVrefInSel[2] ========== RANGE0 : DDR3,DDR4,LPDDR3 [GlobalVrefInSel[2] = 0] DAC Output Voltage = GlobalVrefInDAC == 6'h00 ? Hi-Z : 0.345*VDDQ + (0.005*GlobalVrefInDAC)*VDDQ ========== RANGE1 : LPDDR4 [GlobalVrefInSel[2] = 1] DAC Output Voltage = GlobalVrefInDAC == 6'h00 ? Hi-Z : (0.005*(GlobalVrefInDAC-1))*VDDQ
2-0 GlobalVrefInSel	GlobalVrefInSel[1:0] controls the mode of the PHY VREF DAC and the BP_VREF pin ========== 2'b00 - PHY Vref DAC Range0 -- BP_VREF = Hi-Z 2'b01 - Reserved Encoding 2'b10 - PHY Vref DAC Range0 -- BP_VREF connected to PLL Analog Bus 2'b11 - PHY Vref DAC Range0 -- BP_VREF connected to PHY Vref DAC ========== GlobalVrefInSel[2] shall be set according to Dram Protocol: Protocol GlobalVrefInSel[2] ---------- DDR3 1'b0 DDR4 1'b0 LPDDR3 1'b0 LPDDR4 1'b1

9.3.3.6.228 Maps dfi_rddata_cs_n to destination dimm timing group for use with D4 LRDIMM (DfiWrDataCsDestMap_p3)

9.3.3.6.228.1 Offset

Register	Offset
DfiWrDataCsDestMap_p3	60_0168h

9.3.3.6.228.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								DfiWrDestm3		DfiWrDestm2		DfiWrDestm1		DfiWrDestm0	
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	0

9.3.3.6.228.3 Fields

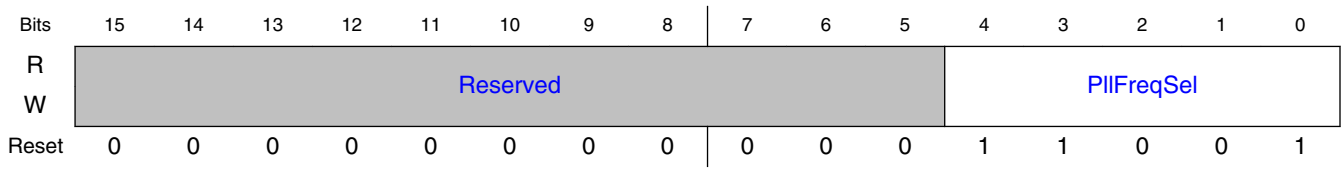
Field	Function
15-8 —	Reserved
7-6 DfiWrDestm3	Maps dfi_wrdata_cs_n_p0[3] to dest DfiWrDestm3 timing (use Register TxDq,DqsDlyTg3) For example, if 3 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg3 timing.
5-4 DfiWrDestm2	Maps dfi_wrdata_cs_n_p0[2] to dest DfiWrDestm2 timing For example, if 2 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg2 timing.
3-2 DfiWrDestm1	Maps dfi_wrdata_cs_n_p0[1] to dest DfiWrDestm1 timing For example, if 1 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg1 timing.
1-0 DfiWrDestm0	Maps dfi_wrdata_cs_n_p0[0] to dest DfiWrDestm0 timing For example, if 0 dfi_wrdata_cs_n_p0[0] will use Register TxDq,DqsDlyTg0 timing.

9.3.3.6.229 PState dependent PLL Control Register 2 (PllCtrl2_p3)

9.3.3.6.229.1 Offset

Register	Offset
PllCtrl2_p3	60_018Ah

9.3.3.6.229.2 Diagram



9.3.3.6.229.3 Fields

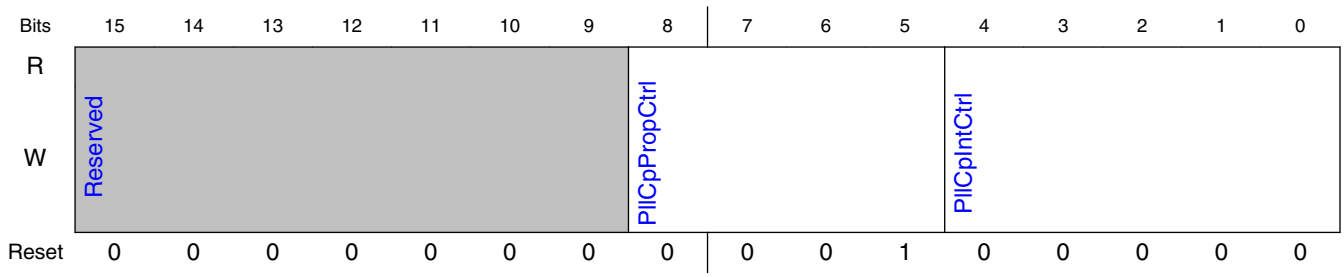
Field	Function
15-5 —	Reserved
4-0 PllFreqSel	Adjusts the loop parameters to compensate for different VCO bias points, and input/output clock division ratios. Adjusts the loop parameters to compensate for different VCO bias points, and input/output clock division ratios. Program based on reference clock freqency (DfiClk) with this table.

9.3.3.6.230 PState dependent PLL Control Register 1 (PllCtrl1_p3)

9.3.3.6.230.1 Offset

Register	Offset
PllCtrl1_p3	60_018Eh

9.3.3.6.230.2 Diagram



9.3.3.6.230.3 Fields

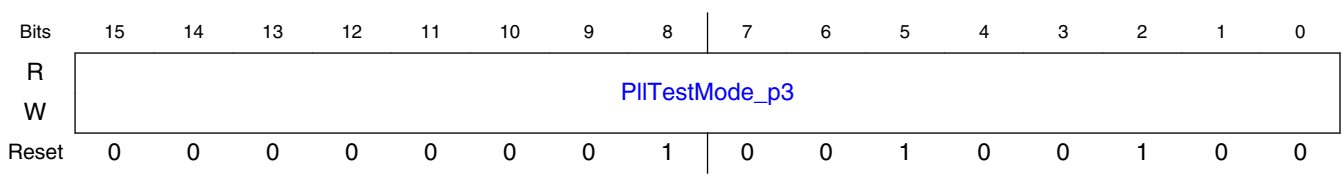
Field	Function
15-9 —	Reserved
8-5 PllCpPropCtrl	connects directly to cp_prop_cntrl<3:0> of PLL. connects directly to cp_prop_cntrl<3:0> of PLL. Charge pump proportional current control.
4-0 PllCpIntCtrl	connects directly to cp_int_cntrl<1:0> in PLL. connects directly to cp_int_cntrl<1:0> in PLL. Charge pump integrating current control.

9.3.3.6.231 Additional controls for PLL CP/VCO modes of operation (PllTestMode_p3)

9.3.3.6.231.1 Offset

Register	Offset
PllTestMode_p3	60_0194h

9.3.3.6.231.2 Diagram



9.3.3.6.231.3 Fields

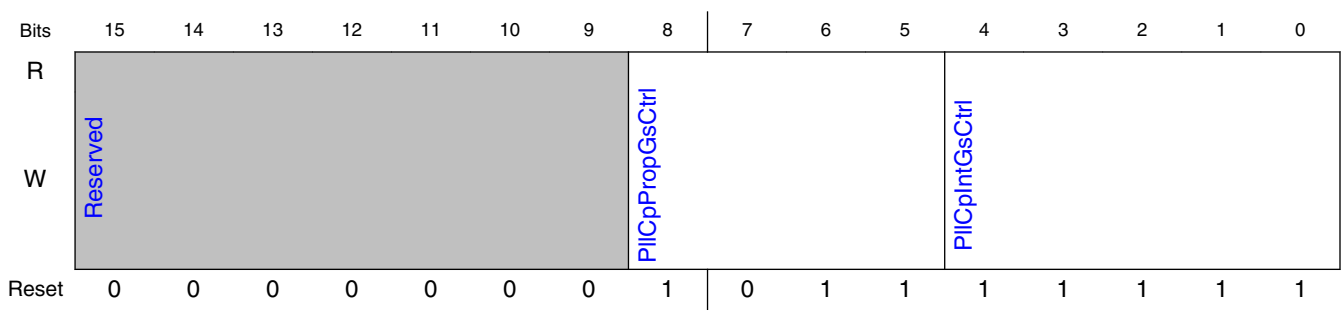
Field	Function
15-0 PllTestMode_p3	It is required to use default values for this CSR Directly connected to testmode[15:0] pin of PLL

9.3.3.6.232 PState dependent PLL Control Register 4 (PllCtrl4_p3)

9.3.3.6.232.1 Offset

Register	Offset
PllCtrl4_p3	60_0198h

9.3.3.6.232.2 Diagram



9.3.3.6.232.3 Fields

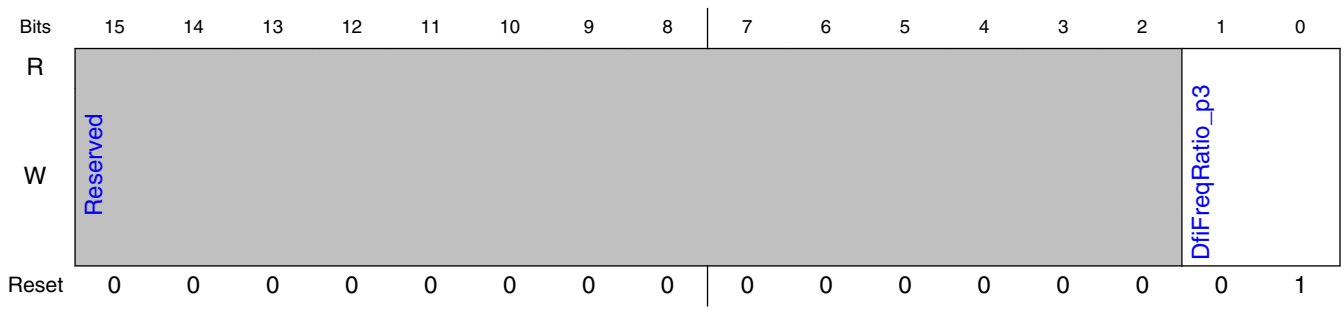
Field	Function
15-9 —	Reserved
8-5 PllCpPropGsCtrl	connects directly to cp_prop_gs_cntrl<3:0> of PLL. connects directly to cp_prop_gs_cntrl<3:0> of PLL. Charge pump proportional current control for fast relock and gearshift.
4-0 PllCpIntGsCtrl	connects directly to cp_int_gs_cntrl<4:0> in PLL. connects directly to cp_int_gs_cntrl<4:0> in PLL. Charge pump integrating current control for fast relock and gearshift.

9.3.3.6.233 DFI Frequency Ratio (DfiFreqRatio_p3)

9.3.3.6.233.1 Offset

Register	Offset
DfiFreqRatio_p3	60_01F4h

9.3.3.6.233.2 Diagram



9.3.3.6.233.3 Fields

Field	Function
15-2 —	Reserved
1-0 DfiFreqRatio_p3	Used in dwc_ddrphy_pub_serdes to serialize or de-serialize DFI signals 00 = 1:1 mode 01 = 1:2 mode 1x = 1:4 mode* *Note: 1:4 is for future pub revision.

9.4 AHB-to-APBH Bridge with DMA (APBH-Bridge-DMA)

9.4.1 Overview

The AHB-to-APBH bridge provides the chip with an inexpensive peripheral attachment bus running on the AHB's HCLK. (The H in APBH denotes that the APB*H* is synchronous to HCLK.)

As shown in the figure below, the AHB-to-APBH bridge includes the AHB-to-APB PIO bridge for a memory-mapped I/O to the APB devices, as well as a central DMA facility for devices on this bus and a vectored interrupt controller for the Arm core. Each one of the APB peripherals, including the vectored interrupt controller, is documented in their respective chapters.

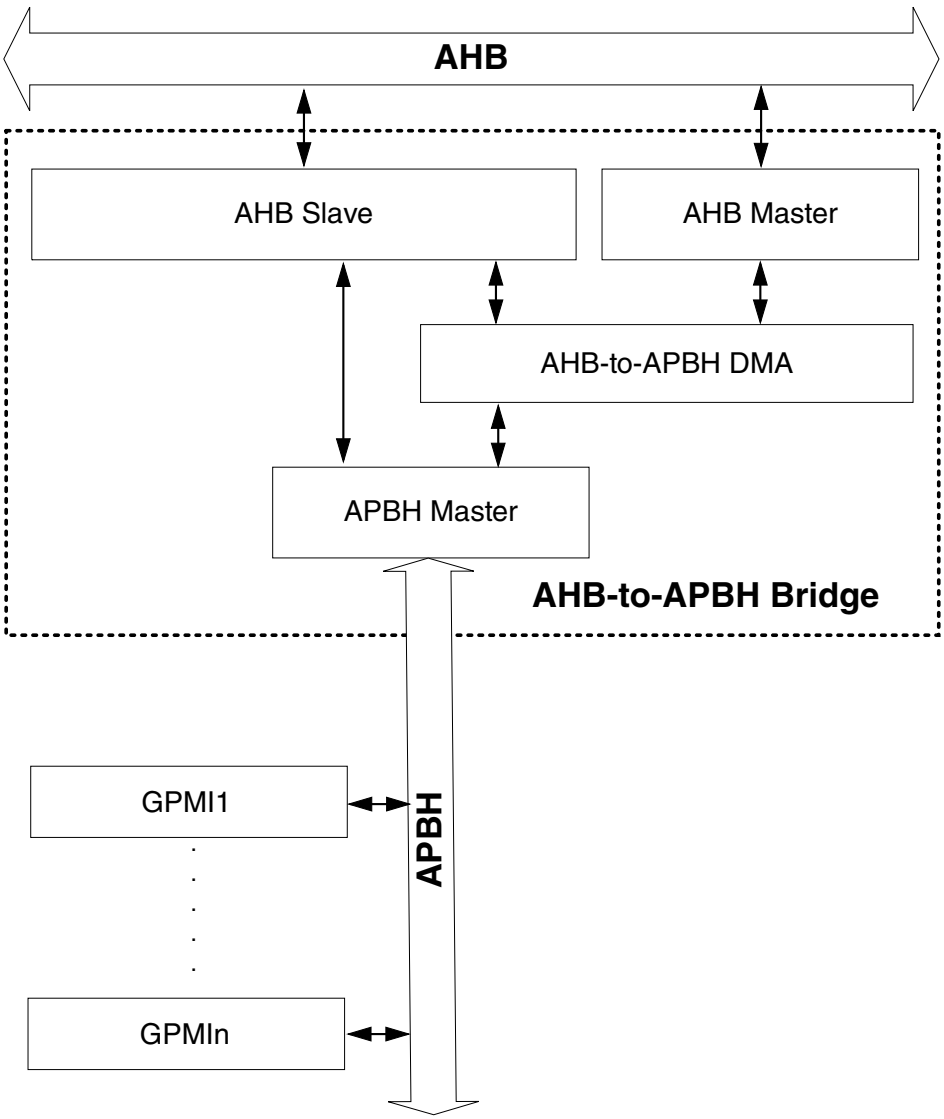


Figure 9-7. AHB-to-APBH Bridge DMA Block Diagram

The DMA controller uses the APBH bus to transfer read and write data to and from each peripheral. There is no separate DMA bus for these devices. Contention between the DMA's use of the APBH bus and the AHB-to-APB bridge functions' use of the APBH is mediated by an internal arbitration logic. For contention between these two units, the DMA is favored and the AHB slave will report "not ready" through its HREADY output until the bridge transfer can complete. The arbiter tracks repeated lockouts and inverts the priority, guaranteeing the Arm platform every fourth transfer on the APB.

9.4.2 APBH DMA

The DMA supports four channels of DMA services, as shown in the following table. The shared DMA resource allows each independent channel to follow a simple chained command list. Command chains are built up using the general structure, as shown in Figure 9-8.

Table 9-5. APBH DMA channel assignments

APBH DMA Channel #	Usage
0	GPMI0
1	GPMI1
2	GPMI2
3	GPMI3

A single command structure or channel command word specifies a number of operations to be performed by the DMA in support of a given device. Thus, the Arm platform can set up large units of work, chaining together many DMA channel command words, pass them off to the DMA, and have no further concern for the device until the DMA completion interrupt occurs. The goal is to have enough intelligence in the DMA and the devices to keep the interrupt frequency from any device below 1 KHz (arrival intervals longer than 1 ms).

A single command structure can issue 32-bit PIO write operations to key registers in the associated device using the same APB bus and controls that it uses to write DMA data bytes to the device. For example, this allows a chain of operations to be issued to the GPMI controller to send NAND command bytes, address bytes, and data transfers where the command and the address structure is completely under software control, but the administration of that transfer is handled autonomously by the DMA. Each DMA structure can have 0–15 PIO words appended to it. The CMDPIOWORDS field, if non-zero, instructs the DMA engine to copy these words to the APB, beginning at the first register address offset for the peripheral and incrementing the register offset each cycle.

The DMA master generates only normal read/write transfers to the APBH. It does *not* generate set, clear, or toggle (SCT) transfers.

After any requested PIO words have been transferred to the peripheral, the DMA examines the two-bit command field in the channel command structure. Table 9-6 shows the four commands implemented by the DMA.

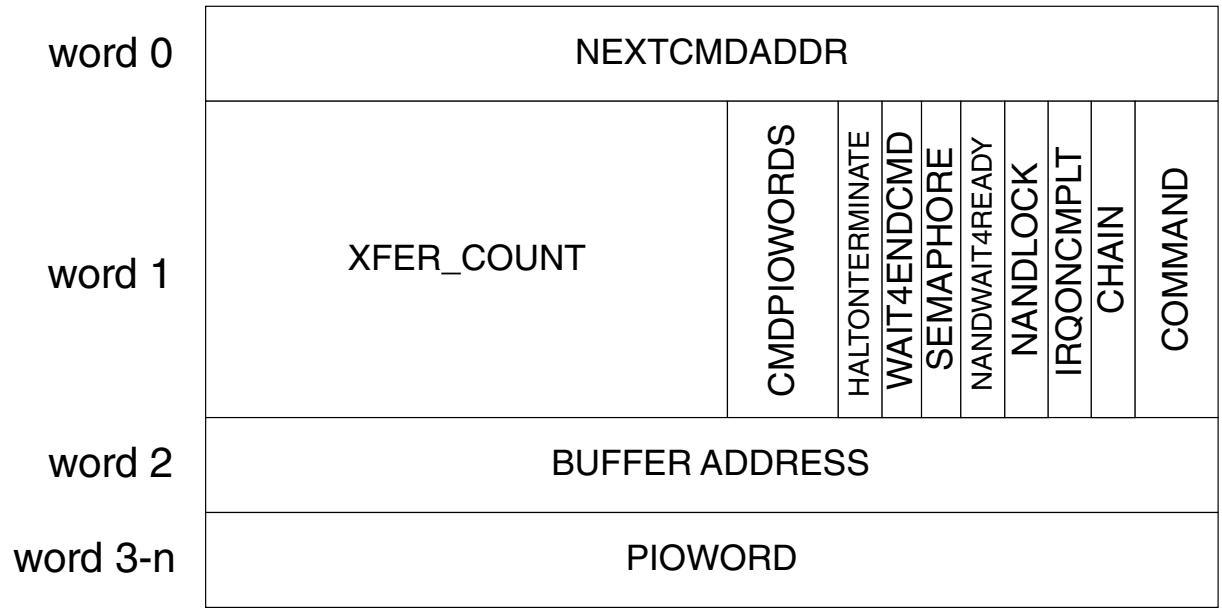


Figure 9-8. AHB-to-APBH Bridge DMA channel command structure

Table 9-6. APBH DMA commands

DMA Command	Usage
00	NO_DMA_XFER. Perform any requested PIO word transfers, but terminate the command before any DMA transfer.
01	DMA_WRITE. Perform any requested PIO word transfers, then perform a DMA transfer from the peripheral for the specified number of bytes.
10	DMA_READ. Perform any requested PIO word transfers and then perform a DMA transfer to the peripheral for the specified number of bytes.
11	DMA_SENSE. Perform any requested PIO word transfers, then perform a conditional branch to the next chained device. Follow the NEXTCMD_ADDR pointer if the peripheral sense is false. Follow the BUFFER_ADDRESS as a chain pointer if the peripheral sense line is true. This command becomes a no-operation for any channel other than a GPMI channel.

DMA_WRITE operations copy data bytes to the system memory (on-chip RAM or SDRAM) from the associated peripheral.

DMA_READ operations copy data bytes to the APB peripheral from the system memory. The DMA engine contains a shared byte aligner that aligns bytes from system memory to or from the peripherals. Peripherals always assume little-endian-aligned data arrives or departs on their 32-bit APB. The DMA_READ transfer uses the BUFFER_ADDRESS word in the command structure to point to the DMA data buffer to be read by the DMA_READ command.

The NO_DMA_XFER command is used to write PIO words to a device without performing any DMA data byte transfers. This command is useful in such applications as activating the NAND devices CHECKSTATUS operation. The check status command reads a status byte from the NAND device, performs an XOR and MASK against an

expected value supplied as part of the PIO transfer. Once the read check completes (see NAND Read Status Polling Example), the NO_DMA_XFER command completes. The result in the peripheral is that its sense line is driven by the results of the comparison. The sense flip-flop is only updated by CHECKSTATUS for the device that is executed. At some future point, the chain contains a DMA command structure with the fourth and final command value, that is, the DMA_SENSE command.

As each DMA command completes, it triggers the DMA to load the next DMA command structure in the chain. The normal flow list of DMA commands is found by following the NEXTCMD_ADDR pointer in the DMA command structure. The DMA_SENSE command uses the DMA buffer pointer word of the command structure to point to an alternate DMA command structure chain or list. The DMA_SENSE command examines the sense line of the associated peripheral. If the sense line is false, then the DMA follows the standard list found whose next command is found from the pointer in the NEXTCMD_ADDR word of the command structure. If the sense line is true, then the DMA follows the alternate list whose next command is found from the pointer in the DMA Buffer Pointer word of the DMA_SENSE command structure (see Figure 9-8). The sense command ignores the CHAIN bit, so that both pointers must be valid when the DMA comes to a sense command.

If the wait-for-end-command bit (WAIT4ENDCMD) is set in a command structure, the DMA channel waits for the device to signal completion of a command by toggling the endcmd signal before proceeding to load and execute the next command structure. Then, if DECREMENT_SEMAPHORE is set, the semaphore is decremented after the end command is seen.

A detailed bit-field view of the DMA command structure is shown in the following table, which shows a field that specifies the number of bytes to be transferred by this DMA command. The transfer-count mechanism is duplicated in the associated peripheral, either as an implied or as a specified count in the peripheral.

Table 9-7. DMA channel command word in system memory

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
NEXT_COMMAND_ADDRESS																															
Number DMA Bytes to Transfer																Number PIO Words to Write							HALTONTERMINATE	WAIT4ENDCMD	DECREMENT SEMAPHORE	NANDwAIT4READY	NANDLOCK	IRQ_COMPLETE	CHAIN	COMMAND	

Table continues on the next page...

Table 9-7. DMA channel command word in system memory (continued)

DMA Buffer or Alternate CCW
Zero or More PIO Words to Write to the Associated Peripheral Starting at its Base Address on the APBH Bus

Figure 9-8 also shows the CHAIN bit in bit 2 of the second word of the command structure. This bit is set to 1, if the NEXT_COMMAND_ADDRESS contains a pointer to another DMA command structure. If a null pointer (0) is loaded into the NEXT_COMMAND_ADDRESS, it is not detected by the DMA hardware. Only the CHAIN bit indicates whether a valid list exists beyond the current structure.

If the IRQ_COMPLETE bit is set in the command structure, then the last act of the DMA before loading the next command is to set the interrupt-status bit corresponding to the current channel. The sticky interrupt request bit in the DMA CSR remains set until cleared by the software. It can be used to interrupt the Arm platform.

The NAND_LOCK bit is monitored by the DMA channel arbiter. Once a NAND channel (from channel 0 to channel 3) succeeds in the arbiter with its NAND_LOCK bit set, then the arbiter ignores the other NAND channels until a command is completed in which the NAND_LOCK is not set. Notice that the semantic here is that the NAND_LOCK state is to limit scheduling of a non-locked DMA. A DMA channel can go from unlocked to locked in the arbiter at the beginning of a command when the NAND_LOCK bit is set. When the last DMA command of an atomic sequence is completed, the lock should be removed. To accomplish this, the last command does not have the NAND_LOCK bit. It is still locked in the atomic state within the arbiter when the command starts, so that it is the only NAND command that can be executed. At the end, it drops from the atomic state within the arbiter.

The NAND_WAIT4READY bit also has a special use for GPMI channels (from channel 0 to channel 3), i.e., the NAND device channels. The GPMI peripheral supplies a sample of the ready line from the NAND device. This ready value is used to hold off of a command with this bit set until the ready line is asserted to 1. Once the arbiter sees a command with a wait-for-ready set, it holds off that channel until ready is asserted.

Receiving an IRQ for HALTONTERMINATE (HOT) is a feature in the APBH DMA descriptor that allows GPMI to signal to the DMA engine that an error has occurred. If a command is stalled due to an error, a HOT signal is sent from the peripheral to the DMA engine and causes an IRQ after terminating the DMA descriptor being executed.

Therefore, it is recommended that software use this signal as follows:

- Always set HALTONTERMINATE to 1 in a DMA descriptor. That way, if a peripheral signals HOT, the transfer will end, leaving the peripheral block and the DMA engine synchronized (but at the end of a command).
- When an IRQ from an APBH channel is received, and the IRQ is determined to be due to an error (as opposed to an IRQONCOMPLETE interrupt) the software should:
  - Reset the channel.
  - Determine the error from error reporting in the peripheral block, then manage the error in the peripheral that is attached to that channel in whatever appropriate way exists for that device (software recovery, device reset, block reset, etc).

Each channel has an eight-bit counting semaphore that controls whether it is in the idle state. When the semaphore is non-zero, the channel is ready to run, process commands and perform DMA transfers. Whenever a command finishes its DMA transfer, it checks the DECREMENT_SEMAPHORE bit. If set, it decrements the counting semaphore. If the semaphore goes to 0 as a result, then the channel enters the idle state and remains there until the semaphore is incremented by the software. When the semaphore goes to non-zero and the channel is in its idle state, then it uses the value in the APBH_CHn_NXTCMDAR register (next command address register) to fetch a pointer to the next command to process.

NOTE

This is a double indirect case. This method allows the software to append to a running command list under the protection of the counting semaphore.

To start processing the first time, software creates the command list to be processed. It writes the address of the first command into the APBH_CHn_NXTCMDAR register, and then writes 1 to the counting semaphore in APBH_CHn_SEMA. The DMA channel loads APBH_CHn_CURCMDAR register and then enters the normal state machine processing for the next command. When the software writes a value to the counting semaphore, it is added to the semaphore count by hardware, protecting the case where both hardware and software are trying to change the semaphore on the same clock edge.

Software can examine the value of APBH_CHn_CURCMDAR at any time to determine the location of the command structure currently being processed.

9.4.3 NAND Read Status Polling Example

The following figure shows a more complicated scenario.

This subset of a NAND device workload shows that the first two command structures are used during the data-write phase of an NAND device write operation (CLE and ALE transfers omitted for clarity).

- After writing the data, one must wait until the NAND device status register indicates that the write charge has been transferred. This is built into the workload using a check status command in the NAND in a loop created from the next two DMA command structures.
- The NO_DMA_TRANSFER command is shown here performing the read check, followed by a DMA_SENSE command to branch the DMA command structure list, based on the status of a bit in the external NAND device.

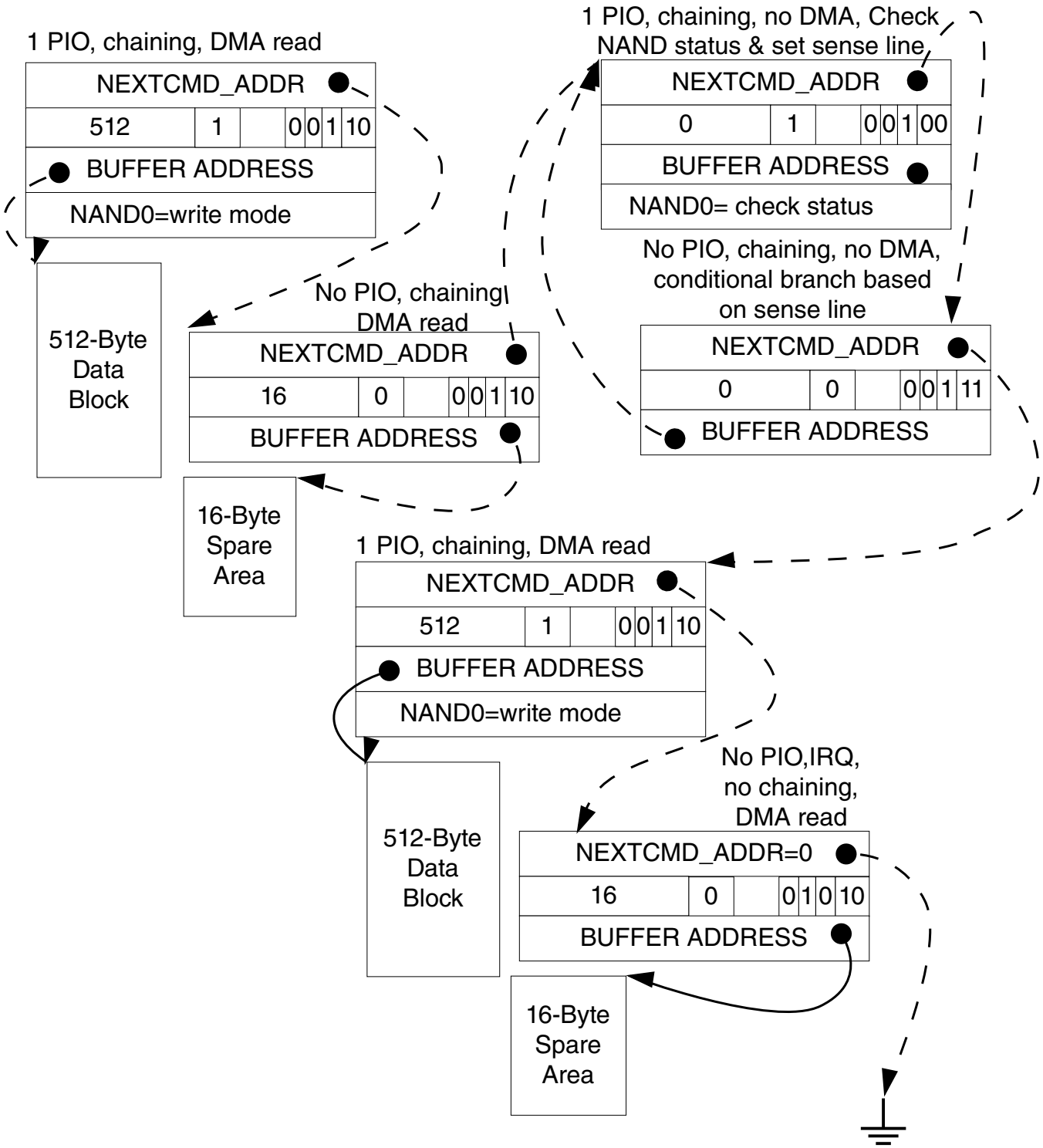


Figure 9-9. AHB-to-APBH Bridge DMA NAND Read Status Polling with DMA Sense Command

The example in the above figure shows the workload continuing immediately to the next NAND page transfer. However, one could perform a second sense operation to see if an error has occurred after the write. One could then point the sense command alternate branch at a NO_DMA_XFER command with the interrupt bit set. If the CHAIN bit is not

set on this failure branch, then the Arm platform is interrupted immediately, and the channel process is also immediately terminated in the presence of a workload-detected NAND error bit.

Note that each word of the three-word DMA command structure corresponds to a PIO register of the DMA that is accessible on the APBH bus. Normally, the DMA copies the next command structure onto these registers for processing at the start of each command by following the value of the pointer previously loaded into the NEXTCMD_ADDR register.

To start DMA processing for the first command, initialize the PIO registers of the desired channel, as follows:

- First, load the next command address register with a pointer to the first command to be loaded.
- Then, write 1 to the counting semaphore register. This causes the DMA to schedule the targeted channel for the DMA command structure load, just as if it had finished its previous command.

9.4.4 APBH Memory Map/Register Definition

APBH Hardware Register Format Summary

APBH memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3300_0000	AHB to APBH Bridge Control and Status Register 0 (APBH_CTRL0)	32	R/W	E000_0000h	9.4.4.1/ 2205
3300_0004	AHB to APBH Bridge Control and Status Register 0 (APBH_CTRL0_SET)	32	R/W	E000_0000h	9.4.4.1/ 2205
3300_0008	AHB to APBH Bridge Control and Status Register 0 (APBH_CTRL0_CLR)	32	R/W	E000_0000h	9.4.4.1/ 2205
3300_000C	AHB to APBH Bridge Control and Status Register 0 (APBH_CTRL0_TOG)	32	R/W	E000_0000h	9.4.4.1/ 2205
3300_0010	AHB to APBH Bridge Control and Status Register 1 (APBH_CTRL1)	32	R/W	0000_0000h	9.4.4.2/ 2207
3300_0014	AHB to APBH Bridge Control and Status Register 1 (APBH_CTRL1_SET)	32	R/W	0000_0000h	9.4.4.2/ 2207
3300_0018	AHB to APBH Bridge Control and Status Register 1 (APBH_CTRL1_CLR)	32	R/W	0000_0000h	9.4.4.2/ 2207

Table continues on the next page...

APBH memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3300_001C	AHB to APBH Bridge Control and Status Register 1 (APBH_CTRL1_TOG)	32	R/W	0000_0000h	9.4.4.2/ 2207
3300_0020	AHB to APBH Bridge Control and Status Register 2 (APBH_CTRL2)	32	R/W	0000_0000h	9.4.4.3/ 2210
3300_0024	AHB to APBH Bridge Control and Status Register 2 (APBH_CTRL2_SET)	32	R/W	0000_0000h	9.4.4.3/ 2210
3300_0028	AHB to APBH Bridge Control and Status Register 2 (APBH_CTRL2_CLR)	32	R/W	0000_0000h	9.4.4.3/ 2210
3300_002C	AHB to APBH Bridge Control and Status Register 2 (APBH_CTRL2_TOG)	32	R/W	0000_0000h	9.4.4.3/ 2210
3300_0030	AHB to APBH Bridge Channel Register (APBH_CHANNEL_CTRL)	32	R/W	0000_0000h	9.4.4.4/ 2215
3300_0034	AHB to APBH Bridge Channel Register (APBH_CHANNEL_CTRL_SET)	32	R/W	0000_0000h	9.4.4.4/ 2215
3300_0038	AHB to APBH Bridge Channel Register (APBH_CHANNEL_CTRL_CLR)	32	R/W	0000_0000h	9.4.4.4/ 2215
3300_003C	AHB to APBH Bridge Channel Register (APBH_CHANNEL_CTRL_TOG)	32	R/W	0000_0000h	9.4.4.4/ 2215
3300_0040	AHB to APBH DMA Device Assignment Register (APBH_DEVSEL)	32	R	0000_0000h	9.4.4.5/ 2216
3300_0050	AHB to APBH DMA burst size (APBH_DMA_BURST_SIZE)	32	R/W	0055_5555h	9.4.4.6/ 2217
3300_0060	AHB to APBH DMA Debug Register (APBH_DEBUG)	32	R/W	0000_0000h	9.4.4.7/ 2218
3300_0100	APBH DMA Channel n Current Command Address Register (APBH_CH0_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_0110	APBH DMA Channel n Next Command Address Register (APBH_CH0_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_0120	APBH DMA Channel n Command Register (APBH_CH0_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_0130	APBH DMA Channel n Buffer Address Register (APBH_CH0_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_0140	APBH DMA Channel n Semaphore Register (APBH_CH0_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_0150	AHB to APBH DMA Channel n Debug Information (APBH_CH0_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_0160	AHB to APBH DMA Channel n Debug Information (APBH_CH0_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_0170	APBH DMA Channel n Current Command Address Register (APBH_CH1_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_0180	APBH DMA Channel n Next Command Address Register (APBH_CH1_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_0190	APBH DMA Channel n Command Register (APBH_CH1_CMD)	32	R	0000_0000h	9.4.4.10/ 2220

Table continues on the next page...

APBH memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3300_01A0	APBH DMA Channel n Buffer Address Register (APBH_CH1_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_01B0	APBH DMA Channel n Semaphore Register (APBH_CH1_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_01C0	AHB to APBH DMA Channel n Debug Information (APBH_CH1_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_01D0	AHB to APBH DMA Channel n Debug Information (APBH_CH1_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_01E0	APBH DMA Channel n Current Command Address Register (APBH_CH2_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_01F0	APBH DMA Channel n Next Command Address Register (APBH_CH2_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_0200	APBH DMA Channel n Command Register (APBH_CH2_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_0210	APBH DMA Channel n Buffer Address Register (APBH_CH2_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_0220	APBH DMA Channel n Semaphore Register (APBH_CH2_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_0230	AHB to APBH DMA Channel n Debug Information (APBH_CH2_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_0240	AHB to APBH DMA Channel n Debug Information (APBH_CH2_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_0250	APBH DMA Channel n Current Command Address Register (APBH_CH3_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_0260	APBH DMA Channel n Next Command Address Register (APBH_CH3_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_0270	APBH DMA Channel n Command Register (APBH_CH3_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_0280	APBH DMA Channel n Buffer Address Register (APBH_CH3_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_0290	APBH DMA Channel n Semaphore Register (APBH_CH3_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_02A0	AHB to APBH DMA Channel n Debug Information (APBH_CH3_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_02B0	AHB to APBH DMA Channel n Debug Information (APBH_CH3_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_02C0	APBH DMA Channel n Current Command Address Register (APBH_CH4_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_02D0	APBH DMA Channel n Next Command Address Register (APBH_CH4_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_02E0	APBH DMA Channel n Command Register (APBH_CH4_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_02F0	APBH DMA Channel n Buffer Address Register (APBH_CH4_BAR)	32	R	0000_0000h	9.4.4.11/ 2222

Table continues on the next page...

APBH memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3300_0300	APBH DMA Channel n Semaphore Register (APBH_CH4_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_0310	AHB to APBH DMA Channel n Debug Information (APBH_CH4_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_0320	AHB to APBH DMA Channel n Debug Information (APBH_CH4_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_0330	APBH DMA Channel n Current Command Address Register (APBH_CH5_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_0340	APBH DMA Channel n Next Command Address Register (APBH_CH5_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_0350	APBH DMA Channel n Command Register (APBH_CH5_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_0360	APBH DMA Channel n Buffer Address Register (APBH_CH5_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_0370	APBH DMA Channel n Semaphore Register (APBH_CH5_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_0380	AHB to APBH DMA Channel n Debug Information (APBH_CH5_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_0390	AHB to APBH DMA Channel n Debug Information (APBH_CH5_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_03A0	APBH DMA Channel n Current Command Address Register (APBH_CH6_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_03B0	APBH DMA Channel n Next Command Address Register (APBH_CH6_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_03C0	APBH DMA Channel n Command Register (APBH_CH6_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_03D0	APBH DMA Channel n Buffer Address Register (APBH_CH6_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_03E0	APBH DMA Channel n Semaphore Register (APBH_CH6_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_03F0	AHB to APBH DMA Channel n Debug Information (APBH_CH6_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_0400	AHB to APBH DMA Channel n Debug Information (APBH_CH6_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_0410	APBH DMA Channel n Current Command Address Register (APBH_CH7_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_0420	APBH DMA Channel n Next Command Address Register (APBH_CH7_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_0430	APBH DMA Channel n Command Register (APBH_CH7_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_0440	APBH DMA Channel n Buffer Address Register (APBH_CH7_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_0450	APBH DMA Channel n Semaphore Register (APBH_CH7_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223

Table continues on the next page...

APBH memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3300_0460	AHB to APBH DMA Channel n Debug Information (APBH_CH7_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_0470	AHB to APBH DMA Channel n Debug Information (APBH_CH7_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_0480	APBH DMA Channel n Current Command Address Register (APBH_CH8_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_0490	APBH DMA Channel n Next Command Address Register (APBH_CH8_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_04A0	APBH DMA Channel n Command Register (APBH_CH8_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_04B0	APBH DMA Channel n Buffer Address Register (APBH_CH8_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_04C0	APBH DMA Channel n Semaphore Register (APBH_CH8_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_04D0	AHB to APBH DMA Channel n Debug Information (APBH_CH8_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_04E0	AHB to APBH DMA Channel n Debug Information (APBH_CH8_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_04F0	APBH DMA Channel n Current Command Address Register (APBH_CH9_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_0500	APBH DMA Channel n Next Command Address Register (APBH_CH9_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_0510	APBH DMA Channel n Command Register (APBH_CH9_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_0520	APBH DMA Channel n Buffer Address Register (APBH_CH9_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_0530	APBH DMA Channel n Semaphore Register (APBH_CH9_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_0540	AHB to APBH DMA Channel n Debug Information (APBH_CH9_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_0550	AHB to APBH DMA Channel n Debug Information (APBH_CH9_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_0560	APBH DMA Channel n Current Command Address Register (APBH_CH10_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_0570	APBH DMA Channel n Next Command Address Register (APBH_CH10_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_0580	APBH DMA Channel n Command Register (APBH_CH10_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_0590	APBH DMA Channel n Buffer Address Register (APBH_CH10_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_05A0	APBH DMA Channel n Semaphore Register (APBH_CH10_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_05B0	AHB to APBH DMA Channel n Debug Information (APBH_CH10_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224

Table continues on the next page...

APBH memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3300_05C0	AHB to APBH DMA Channel n Debug Information (APBH_CH10_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_05D0	APBH DMA Channel n Current Command Address Register (APBH_CH11_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_05E0	APBH DMA Channel n Next Command Address Register (APBH_CH11_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_05F0	APBH DMA Channel n Command Register (APBH_CH11_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_0600	APBH DMA Channel n Buffer Address Register (APBH_CH11_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_0610	APBH DMA Channel n Semaphore Register (APBH_CH11_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_0620	AHB to APBH DMA Channel n Debug Information (APBH_CH11_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_0630	AHB to APBH DMA Channel n Debug Information (APBH_CH11_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_0640	APBH DMA Channel n Current Command Address Register (APBH_CH12_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_0650	APBH DMA Channel n Next Command Address Register (APBH_CH12_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_0660	APBH DMA Channel n Command Register (APBH_CH12_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_0670	APBH DMA Channel n Buffer Address Register (APBH_CH12_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_0680	APBH DMA Channel n Semaphore Register (APBH_CH12_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_0690	AHB to APBH DMA Channel n Debug Information (APBH_CH12_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_06A0	AHB to APBH DMA Channel n Debug Information (APBH_CH12_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_06B0	APBH DMA Channel n Current Command Address Register (APBH_CH13_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_06C0	APBH DMA Channel n Next Command Address Register (APBH_CH13_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_06D0	APBH DMA Channel n Command Register (APBH_CH13_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_06E0	APBH DMA Channel n Buffer Address Register (APBH_CH13_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_06F0	APBH DMA Channel n Semaphore Register (APBH_CH13_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_0700	AHB to APBH DMA Channel n Debug Information (APBH_CH13_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_0710	AHB to APBH DMA Channel n Debug Information (APBH_CH13_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227

Table continues on the next page...

APBH memory map (continued)

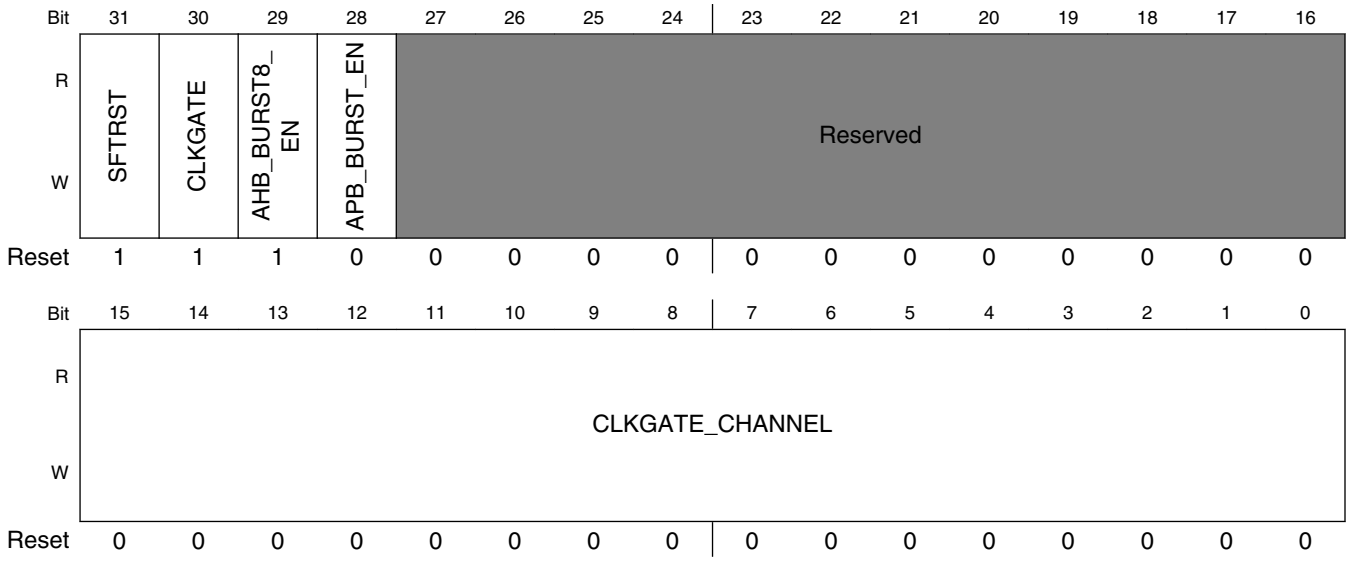
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3300_0720	APBH DMA Channel n Current Command Address Register (APBH_CH14_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_0730	APBH DMA Channel n Next Command Address Register (APBH_CH14_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_0740	APBH DMA Channel n Command Register (APBH_CH14_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_0750	APBH DMA Channel n Buffer Address Register (APBH_CH14_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_0760	APBH DMA Channel n Semaphore Register (APBH_CH14_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_0770	AHB to APBH DMA Channel n Debug Information (APBH_CH14_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_0780	AHB to APBH DMA Channel n Debug Information (APBH_CH14_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_0790	APBH DMA Channel n Current Command Address Register (APBH_CH15_CURCMDAR)	32	R	0000_0000h	9.4.4.8/ 2219
3300_07A0	APBH DMA Channel n Next Command Address Register (APBH_CH15_NXTCMDAR)	32	R/W	0000_0000h	9.4.4.9/ 2220
3300_07B0	APBH DMA Channel n Command Register (APBH_CH15_CMD)	32	R	0000_0000h	9.4.4.10/ 2220
3300_07C0	APBH DMA Channel n Buffer Address Register (APBH_CH15_BAR)	32	R	0000_0000h	9.4.4.11/ 2222
3300_07D0	APBH DMA Channel n Semaphore Register (APBH_CH15_SEMA)	32	R/W	0000_0000h	9.4.4.12/ 2223
3300_07E0	AHB to APBH DMA Channel n Debug Information (APBH_CH15_DEBUG1)	32	R	00A0_0000h	9.4.4.13/ 2224
3300_07F0	AHB to APBH DMA Channel n Debug Information (APBH_CH15_DEBUG2)	32	R	0000_0000h	9.4.4.14/ 2227
3300_0800	APBH Bridge Version Register (APBH_VERSION)	32	R/W	0301_0000h	9.4.4.15/ 2227

9.4.4.1 AHB to APBH Bridge Control and Status Register 0 (APBH_CTRL0*n*)

The APBH CTRL 0 provides overall control of the AHB to APBH bridge and DMA.

This register contains module softreset, clock gating, channel clock gating/freeze bits.

Address: 3300_0000h base + 0h offset + (4d × i), where i=0d to 3d



APBH_CTRL0*n* field descriptions

Field	Description
31 SFTRST	Set this bit to zero to enable normal APBH DMA operation. Set this bit to one (default) to disable clocking with the APBH DMA and hold it in its reset (lowest power) state. This bit can be turned on and then off to reset the APBH DMA block to its default state.
30 CLKGATE	This bit must be set to zero for normal operation. When set to one it gates off the clocks to the block.
29 AHB_BURST8_EN	Set this bit to one (default) to enable AHB 8-beat burst. Set to zero to disable 8-beat burst on AHB interface.
28 APB_BURST_EN	Set this bit to one to enable apb master do a continous transfers when a device request a burst dma. Set to zero will treat a burst dma request as 4/8 individual requests.
27–16 RSVD0	This field is reserved. Reserved, always set to zero.
CLKGATE_CHANNEL	These bits must be set to zero for normal operation of each channel. When set to one they gate off the individual clocks to the channels. 0x0001 **NAND0** — 0x0002 **NAND1** — 0x0004 **NAND2** — 0x0008 **NAND3** — 0x0010 **NAND4** — 0x0020 **NAND5** — 0x0040 **NAND6** — 0x0080 **NAND7** — 0x0100 **SSP** —

9.4.4.2 AHB to APBH Bridge Control and Status Register 1 (APBH_CTRL1*n*)

The APBH CTRL one provides overall control of the interrupts generated by the AHB to APBH DMA. This register contains the per channel interrupt status bits and the per channel interrupt enable bits. Each channel has a dedicated interrupt vector in the vectored interrupt controller.

EXAMPLE

```
                    BF_WR(APBH_CTRL1, CH5_CMDCMPLT_IRQ, 0);  // use bitfield write macro
                    BF_APBH_CTRL1.CH5_CMDCMPLT_IRQ = 0;      // or, assign to register
struct's bitfield
```

Address: 3300_0000h base + 10h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CH15_CMDCMPLT_IRQ_EN	CH14_CMDCMPLT_IRQ_EN	CH13_CMDCMPLT_IRQ_EN	CH12_CMDCMPLT_IRQ_EN	CH11_CMDCMPLT_IRQ_EN	CH10_CMDCMPLT_IRQ_EN	CH9_CMDCMPLT_IRQ_EN	CH8_CMDCMPLT_IRQ_EN	CH7_CMDCMPLT_IRQ_EN	CH6_CMDCMPLT_IRQ_EN	CH5_CMDCMPLT_IRQ_EN	CH4_CMDCMPLT_IRQ_EN	CH3_CMDCMPLT_IRQ_EN	CH2_CMDCMPLT_IRQ_EN	CH1_CMDCMPLT_IRQ_EN	CH0_CMDCMPLT_IRQ_EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CH15_CMDCMPLT_IRQ	CH14_CMDCMPLT_IRQ	CH13_CMDCMPLT_IRQ	CH12_CMDCMPLT_IRQ	CH11_CMDCMPLT_IRQ	CH10_CMDCMPLT_IRQ	CH9_CMDCMPLT_IRQ	CH8_CMDCMPLT_IRQ	CH7_CMDCMPLT_IRQ	CH6_CMDCMPLT_IRQ	CH5_CMDCMPLT_IRQ	CH4_CMDCMPLT_IRQ	CH3_CMDCMPLT_IRQ	CH2_CMDCMPLT_IRQ	CH1_CMDCMPLT_IRQ	CH0_CMDCMPLT_IRQ
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

APBH_CTRL1*n* field descriptions

Field	Description
31 CH15_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 15.
30 CH14_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 14.

Table continues on the next page...

APBH_CTRL1*n* field descriptions (continued)

Field	Description
29 CH13_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 13.
28 CH12_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 12.
27 CH11_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 11.
26 CH10_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 10.
25 CH9_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 9.
24 CH8_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 8.
23 CH7_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 7.
22 CH6_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 6.
21 CH5_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 5.
20 CH4_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 4.
19 CH3_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 3.
18 CH2_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 2.

Table continues on the next page...

APBH_CTRL1*n* field descriptions (continued)

Field	Description
17 CH1_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 1.
16 CH0_ CMDCMPLT_ IRQ_EN	Setting this bit enables the generation of an interrupt request for APBH DMA channel 0.
15 CH15_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA Channel 15. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
14 CH14_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA Channel 14. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
13 CH13_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA Channel 13. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
12 CH12_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA Channel 12. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
11 CH11_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA Channel 11. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
10 CH10_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA Channel 10. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
9 CH9_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA Channel 9. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
8 CH8_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA Channel 8. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
7 CH7_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA channel 7. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
6 CH6_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA channel 6. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.

Table continues on the next page...

APBH_CTRL1*n* field descriptions (continued)

Field	Description
5 CH5_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA channel 5. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
4 CH4_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA channel 4. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
3 CH3_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA channel 3. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
2 CH2_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA channel 2. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
1 CH1_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA channel 1. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.
0 CH0_ CMDCMPLT_ IRQ	Interrupt request status bit for APBH DMA channel 0. This sticky bit is set by DMA hardware and reset by software. It is ANDed with its corresponding enable bit to generate an interrupt.

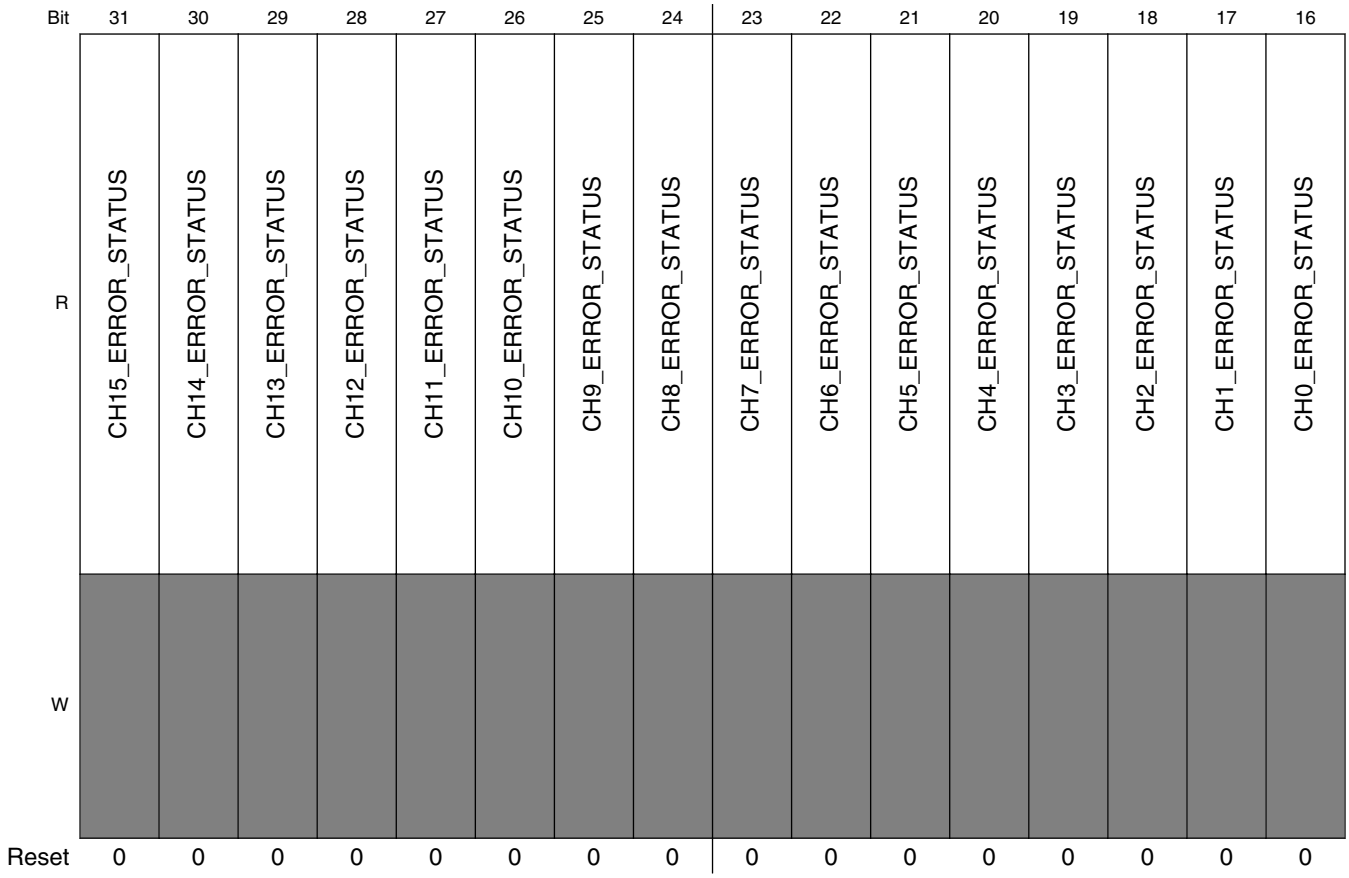
9.4.4.3 AHB to APBH Bridge Control and Status Register 2 (APBH_CTRL2*n*)

The APBH CTRL 2 provides channel error interrupts generated by the AHB to APBH DMA. This register contains the per channel interrupt status bits and the per channel interrupt enable bits. Each channel has a dedicated interrupt vector in the vectored interrupt controller.

EXAMPLE

```
                    BF_WR(APBH_CTRL1, CH5_CMDCMPLT_IRQ, 0);  // use bitfield write macro
                    BF_APBH_CTRL1.CH5_CMDCMPLT_IRQ = 0;      // or, assign to register
struct's bitfield
```

Address: 3300_0000h base + 20h offset + (4d × i), where i=0d to 3d



APBH_CTRL2*n* field descriptions

Field	Description
31 CH15_ERROR_STATUS	Error status bit for APBH DMA Channel 15. Valid when corresponding Error IRQ is set. 1 - AHB bus error

Table continues on the next page...

APBH_CTRL2*n* field descriptions (continued)

Field	Description
	0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
30 CH14_ERROR_ STATUS	Error status bit for APBH DMA Channel 14. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
29 CH13_ERROR_ STATUS	Error status bit for APBH DMA Channel 13. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
28 CH12_ERROR_ STATUS	Error status bit for APBH DMA Channel 12. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
27 CH11_ERROR_ STATUS	Error status bit for APBH DMA Channel 11. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
26 CH10_ERROR_ STATUS	Error status bit for APBH DMA Channel 10. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
25 CH9_ERROR_ STATUS	Error status bit for APBH DMA Channel 9. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
24 CH8_ERROR_ STATUS	Error status bit for APBH DMA Channel 8. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination.

Table continues on the next page...

APBH_CTRL2*n* field descriptions (continued)

Field	Description
	0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
23 CH7_ERROR_STATUS	Error status bit for APBX DMA Channel 7. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
22 CH6_ERROR_STATUS	Error status bit for APBX DMA Channel 6. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
21 CH5_ERROR_STATUS	Error status bit for APBX DMA Channel 5. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
20 CH4_ERROR_STATUS	Error status bit for APBX DMA Channel 4. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
19 CH3_ERROR_STATUS	Error status bit for APBX DMA Channel 3. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
18 CH2_ERROR_STATUS	Error status bit for APBX DMA Channel 2. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
17 CH1_ERROR_STATUS	Error status bit for APBX DMA Channel 1. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.

Table continues on the next page...

APBH_CTRL2*n* field descriptions (continued)

Field	Description
16 CH0_ERROR_STATUS	Error status bit for APBX DMA Channel 0. Valid when corresponding Error IRQ is set. 1 - AHB bus error 0 - channel early termination. 0x0 **TERMINATION** — An early termination from the device causes error IRQ. 0x1 **BUS_ERROR** — An AHB bus error causes error IRQ.
15 CH15_ERROR_IRQ	Error interrupt status bit for APBH DMA Channel 15. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
14 CH14_ERROR_IRQ	Error interrupt status bit for APBH DMA Channel 14. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
13 CH13_ERROR_IRQ	Error interrupt status bit for APBH DMA Channel 13. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
12 CH12_ERROR_IRQ	Error interrupt status bit for APBH DMA Channel 12. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
11 CH11_ERROR_IRQ	Error interrupt status bit for APBH DMA Channel 11. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
10 CH10_ERROR_IRQ	Error interrupt status bit for APBH DMA Channel 10. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
9 CH9_ERROR_IRQ	Error interrupt status bit for APBH DMA Channel 9. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
8 CH8_ERROR_IRQ	Error interrupt status bit for APBH DMA Channel 8. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
7 CH7_ERROR_IRQ	Error interrupt status bit for APBX DMA Channel 7. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
6 CH6_ERROR_IRQ	Error interrupt status bit for APBX DMA Channel 6. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
5 CH5_ERROR_IRQ	Error interrupt status bit for APBX DMA Channel 5. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
4 CH4_ERROR_IRQ	Error interrupt status bit for APBX DMA Channel 4. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
3 CH3_ERROR_IRQ	Error interrupt status bit for APBX DMA Channel 3. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
2 CH2_ERROR_IRQ	Error interrupt status bit for APBX DMA Channel 2. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.

Table continues on the next page...

APBH_CTRL2*n* field descriptions (continued)

Field	Description
1 CH1_ERROR_IRQ	Error interrupt status bit for APBX DMA Channel 1. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.
0 CH0_ERROR_IRQ	Error interrupt status bit for APBX DMA Channel 0. This sticky bit is set by DMA hardware and reset by software. It is ORed with the corresponding cmdcmplt irq to generate an irq to Arm.

9.4.4.4 AHB to APBH Bridge Channel Register (APBH_CHANNEL_CTRL*n*)

The APBH CHANNEL CTRL provides reset/freeze control of each DMA channel. This register contains individual channel reset/freeze bits.

Address: 3300_0000h base + 30h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RESET_CHANNEL																FREEZE_CHANNEL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

APBH_CHANNEL_CTRL*n* field descriptions

Field	Description
31–16 RESET_CHANNEL	Setting a bit in this field causes the DMA controller to take the corresponding channel through its reset state. The bit is reset after the channel resources are cleared. 0x0001 **NAND0** — 0x0002 **NAND1** — 0x0004 **NAND2** — 0x0008 **NAND3** — 0x0010 **NAND4** — 0x0020 **NAND5** — 0x0040 **NAND6** — 0x0080 **NAND7** — 0x0100 **SSP** —
FREEZE_CHANNEL	Setting a bit in this field will freeze the DMA channel associated with it. This field is a direct input to the DMA channel arbiter. When frozen, the channel is deined access to the central DMA resources. 0x0001 **NAND0** — 0x0002 **NAND1** — 0x0004 **NAND2** — 0x0008 **NAND3** — 0x0010 **NAND4** — 0x0020 **NAND5** — 0x0040 **NAND6** —

Table continues on the next page...

APBH_CHANNEL_CTRL*n* field descriptions (continued)

Field	Description
	0x0080 **NAND7** — 0x0100 **SSP** —

9.4.4.5 AHB to APBH DMA Device Assignment Register (APBH_DEVSEL)

This register allows reassignment of the APBH device connected to the DMA Channels.

In this chip, APBH DMA channel resource is enough for high speed peripherals, so this register is of no use and reserved.

Address: 3300_0000h base + 40h offset = 3300_0040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

APBH_DEVSEL field descriptions

Field	Description
31–30 CH15	This field is reserved. Reserved.
29–28 CH14	This field is reserved. Reserved.
27–26 CH13	This field is reserved. Reserved.
25–24 CH12	This field is reserved. Reserved.
23–22 CH11	This field is reserved. Reserved.
21–20 CH10	This field is reserved. Reserved.
19–18 CH9	This field is reserved. Reserved.
17–16 CH8	This field is reserved. Reserved.
15–14 CH7	This field is reserved. Reserved.
13–12 CH6	This field is reserved. Reserved.

Table continues on the next page...

APBH_DEVSEL field descriptions (continued)

Field	Description
11–10 CH5	This field is reserved. Reserved.
9–8 CH4	This field is reserved. Reserved.
7–6 CH3	This field is reserved. Reserved.
5–4 CH2	This field is reserved. Reserved.
3–2 CH1	This field is reserved. Reserved.
CH0	This field is reserved. Reserved.

9.4.4.6 AHB to APBH DMA burst size (APBH_DMA_BURST_SIZE)

This register programs the apbh burst size of the APBH DMA devices when a DMA burst request is issued.

This register provides a mechanism for assigning the device.

Address: 3300_0000h base + 50h offset = 3300_0050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		CH8	
Reset	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CH7		CH6		CH5		CH4		CH3		CH2		CH1		CH0	
Reset	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

APBH_DMA_BURST_SIZE field descriptions

Field	Description
31–30 CH15	This field is reserved. Reserved.
29–28 CH14	This field is reserved. Reserved.
27–26 CH13	This field is reserved. Reserved.
25–24 CH12	This field is reserved. Reserved.
23–22 CH11	This field is reserved. Reserved.

Table continues on the next page...

APBH_DMA_BURST_SIZE field descriptions (continued)

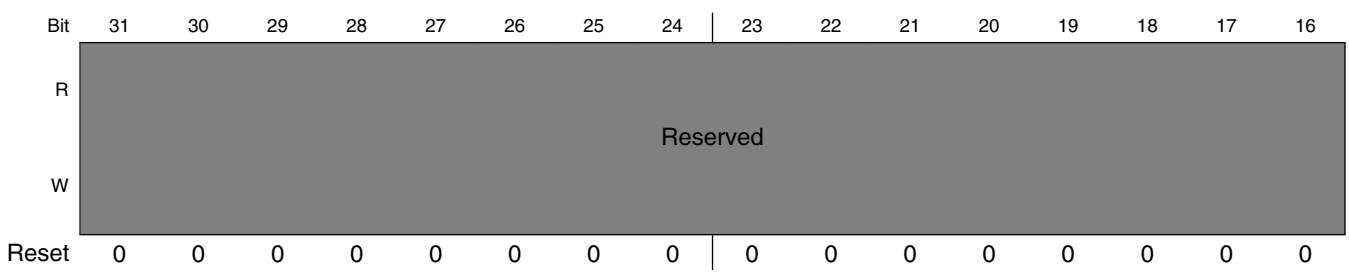
Field	Description
21–20 CH10	This field is reserved. Reserved.
19–18 CH9	This field is reserved. Reserved.
17–16 CH8	DMA burst size for SSP. 0x0 **BURST0** — 0x1 **BURST4** — 0x2 **BURST8** —
15–14 CH7	DMA burst size for GPMI channel 7. Do not change. GPMI only support burst size 4.
13–12 CH6	DMA burst size for GPMI channel 6. Do not change. GPMI only support burst size 4.
11–10 CH5	DMA burst size for GPMI channel 5. Do not change. GPMI only support burst size 4.
9–8 CH4	DMA burst size for GPMI channel 4. Do not change. GPMI only support burst size 4.
7–6 CH3	DMA burst size for GPMI channel 3. Do not change. GPMI only support burst size 4.
5–4 CH2	DMA burst size for GPMI channel 2. Do not change. GPMI only support burst size 4.
3–2 CH1	DMA burst size for GPMI channel 1. Do not change. GPMI only support burst size 4.
CH0	DMA burst size for GPMI channel 0. Do not change. GPMI only support burst size 4.

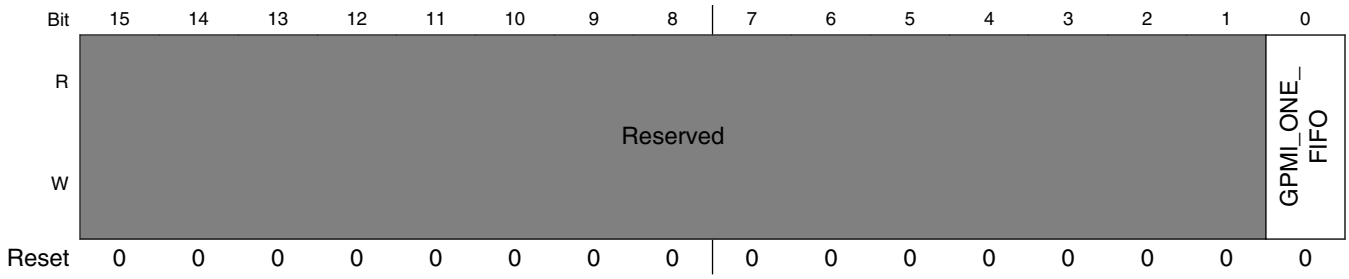
9.4.4.7 AHB to APBH DMA Debug Register (APBH_DEBUG)

This register is for debug purpose.

The debug register is for internal use only. Not recommend for customer useage.

Address: 3300_0000h base + 60h offset = 3300_0060h





APBH_DEBUG field descriptions

Field	Description
31–1 -	This field is reserved. Reserved, always set to zero.
0 GPMI_ONE_FIFO	Set to 0ne and the 8 GPMI channels will share the DMA FIFO, and when set to zero, the 8 GPMI channels will use its own DMA FIFO.

9.4.4.8 APBH DMA Channel n Current Command Address Register (APBH_CH*n*_CURCMDAR)

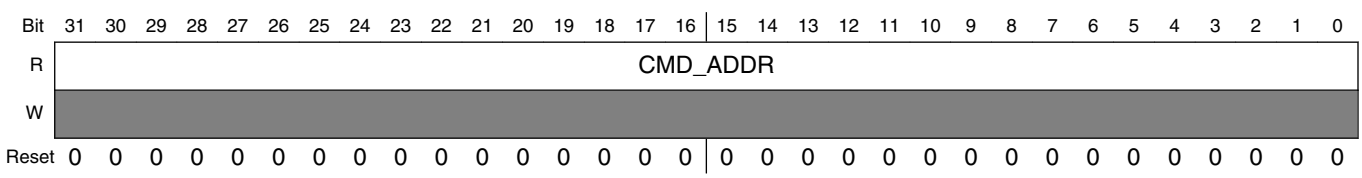
The APBH DMA channel n current command address register points to the multiword command that is currently being executed. Commands are threaded on the command address.

APBH DMA Channel n is controlled by a variable sized command structure. This register points to the command structure currently being executed.

EXAMPLE

```
pCurCmd = (apbh_chn_cmd_t *) APBH_CHn_CURCMDAR_RD(0);              // read the whole
register, since there is only one field
pCurCmd = (apbh_chn_cmd_t *) BF_RDn(APBH_CHn_CURCMDAR, 0, CMD_ADDR);  // or, use multi-
register bitfield read macro
pCurCmd = (apbh_chn_cmd_t *) APBH_CHn_CURCMDAR(0).CMD_ADDR;        // or, assign from
bitfield of indexed register's struct
```

Address: 3300_0000h base + 100h offset + (112d × i), where i=0d to 15d



APBH_CH*n*_CURCMDAR field descriptions

Field	Description
CMD_ADDR	Pointer to command structure currently being processed for channel n.

9.4.4.9 APBH DMA Channel n Next Command Address Register (APBH_CH*n*_NXTCMDAR)

The APBH DMA Channel n Next Command Address register contains the address of the next multiword command to be executed. Commands are threaded on the command address. Set CHAIN to 1 in the DMA command word to process command lists.

APBH DMA Channel n is controlled by a variable sized command structure. Software loads this register with the address of the first command structure to process and increments the Channel n semaphore to start processing. This register points to the next command structure to be executed when the current command is completed.

EXAMPLE

```
APBH_CHn_NXTCMDAR_WR(0, (reg32_t) pCommandTwoStructure);         // write the entire
register, since there is only one field
BF_WRn(APBH_CHn_NXTCMDAR, 0, (reg32_t) pCommandTwoStructure);       // or, use multi-
register bitfield write macro
APBH_CHn_NXTCMDAR(0).CMD_ADDR = (reg32_t) pCommandTwoStructure;  // or, assign to bitfield
of indexed register's struct
```

Address: 3300_0000h base + 110h offset + (112d × i), where i=0d to 15d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CMD_ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

APBH_CH*n*_NXTCMDAR field descriptions

Field	Description
CMD_ADDR	Pointer to next command structure for channel n.

9.4.4.10 APBH DMA Channel n Command Register (APBH_CH*n*_CMD)

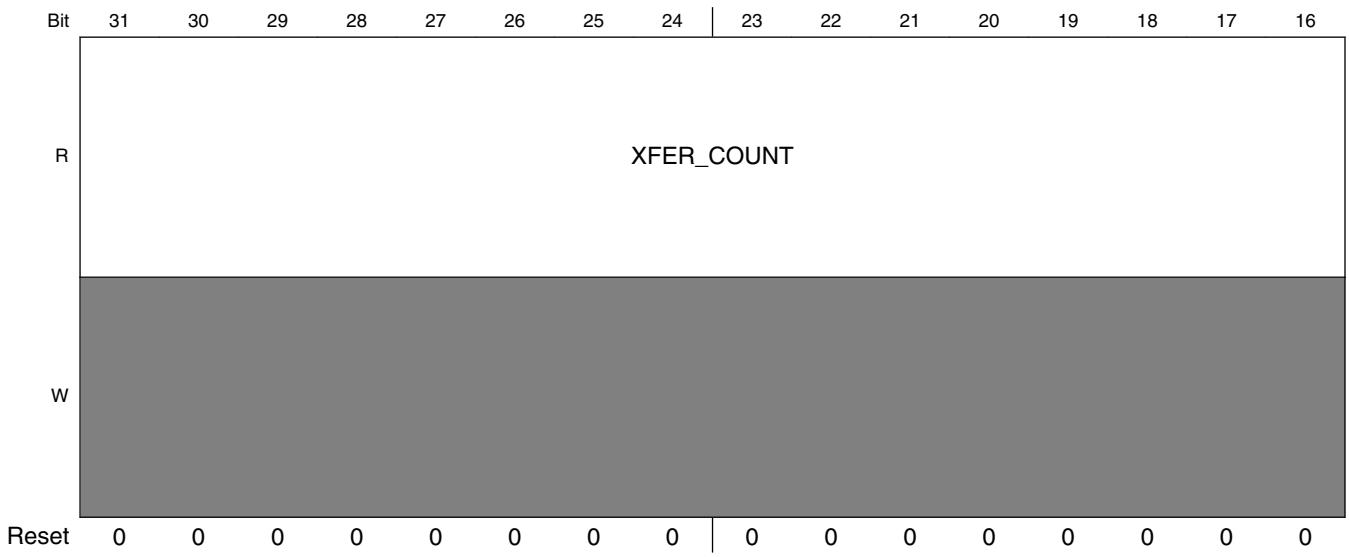
The APBH DMA Channel n command register specifies the DMA transaction to perform for the current command chain item.

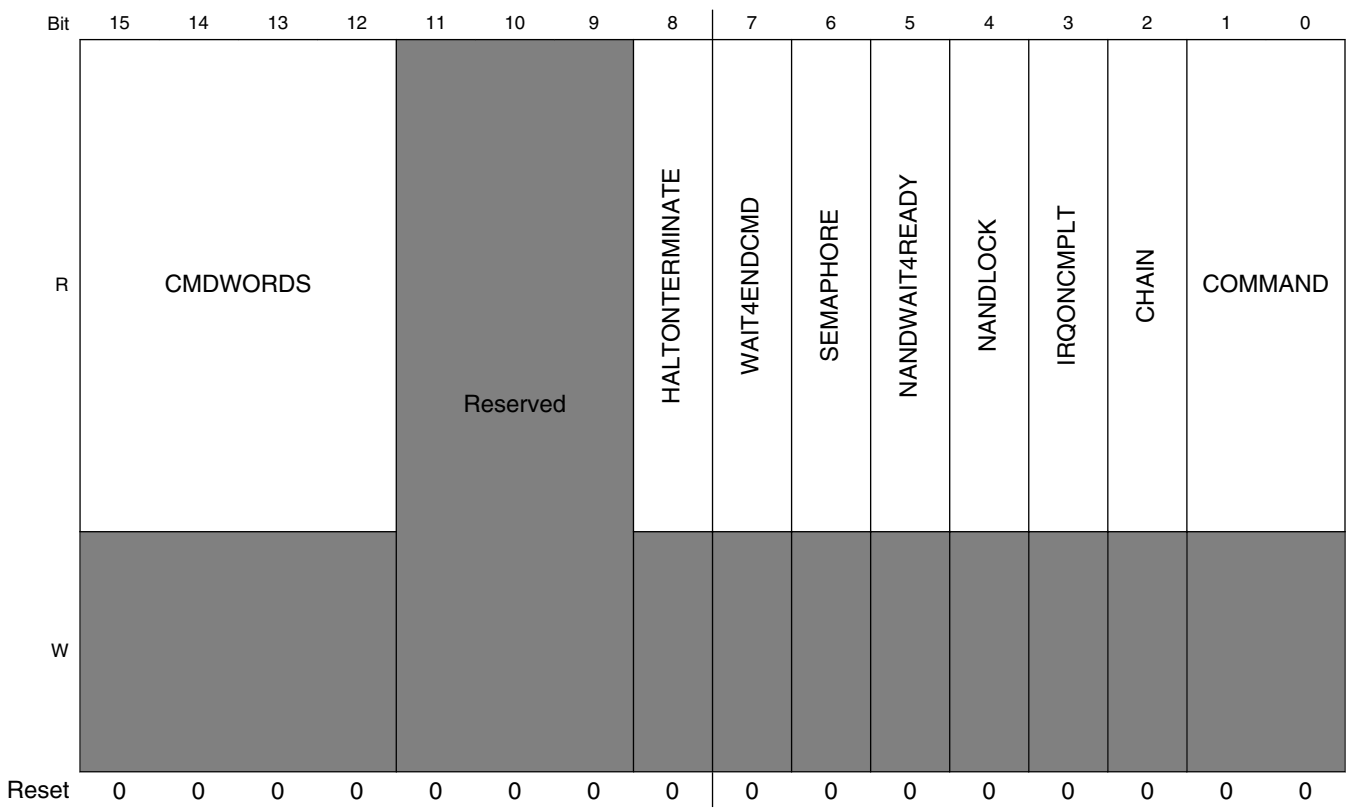
The command register controls the overall operation of each DMA command for this channel. It includes the number of bytes to transfer to or from the device, the number of APB PIO command words included with this command structure, whether to interrupt at command completion, whether to chain an additional command to the end of this one and whether this transfer is a read or write DMA transfer.

EXAMPLE

```
apbh_chn_cmd_t  dma_cmd;
dma_cmd.XFER_COUNT = 512;                                   // transfer 512 bytes
dma_cmd.COMMAND = BV_APBH_CHn_CMD_COMMAND__DMA_WRITE;  // transfer to system memory from
peripheral device
dma_cmd.CHAIN = 1;                                          // chain an additional command
structure on to the list
dma_cmd.IRQONCMPLT = 1;                                     // generate an interrupt on
completion of this command structure
```

Address: 3300_0000h base + 120h offset + (112d × i), where i=0d to 15d





APBH_CH*n*_CMD field descriptions

Field	Description
31–16 XFER_COUNT	This field indicates the number of bytes to transfer to or from the appropriate PIO register in the GPMI0 device. A value of 0 indicates a 64 KBytes transfer.
15–12 CMDWORDS	This field indicates the number of command words to send to the GPMI0, starting with the base PIO address of the GPMI0 control register and incrementing from there. Zero means transfer NO command words
11–9 -	This field is reserved. Reserved, always set to zero.
8 HALTONTERMINATE	A value of one indicates that the channel will immeditately terminate the current descriptor and halt the DMA channel if a terminate signal is set. A value of 0 will still cause an immediate terminate of the channel if the terminate signal is set, but the channel will continue as if the count had been exhausted, meaning it will honor IRQONCMPLT, CHAIN, SEMAPHORE, and WAIT4ENDCMD.
7 WAIT4ENDCMD	A value of one indicates that the channel will wait for the end of command signal to be sent from the APBH device to the DMA before starting the next DMA command.
6 SEMAPHORE	A value of one indicates that the channel will decrement its semaphore at the completion of the current command structure. If the semaphore decrements to zero, then this channel stalls until software increments it again.
5 NANDWAIT4READY	A value of one indicates that the NAND DMA channel will will wait until the NAND device reports "ready" before executing the command. It is ignored for non-NAND DMA channels.
4 NANDLOCK	A value of one indicates that the NAND DMA channel will remain "locked" in the arbiter at the expense of other NAND DMA channels. It is ignored for non-NAND DMA channels.
3 IRQONCMPLT	A value of one indicates that the channel will cause the interrupt status bit to be set upon completion of the current command, i.e. after the DMA transfer is complete.
2 CHAIN	A value of one indicates that another command is chained onto the end of the current command structure. At the completion of the current command, this channel will follow the pointer in APBH_CHn_CMDAR to find the next command.
COMMAND	This bitfield indicates the type of current command: 0x0 **NO_DMA_XFER** — Perform any requested PIO word transfers but terminate command before any DMA transfer. 0x1 **DMA_WRITE** — Perform any requested PIO word transfers and then perform a DMA transfer from the peripheral for the specified number of bytes. 0x2 **DMA_READ** — Perform any requested PIO word transfers and then perform a DMA transfer to the peripheral for the specified number of bytes. 0x3 **DMA_SENSE** — Perform any requested PIO word transfers and then perform a conditional branch to the next chained device. Follow the NEXCMD_ADDR pointer if the perpheral sense is true. Follow the BUFFER_ADDRESS as a chain pointer if the peripheral sense line is false.

9.4.4.11 APBH DMA Channel n Buffer Address Register (APBH_CH*n*_BAR)

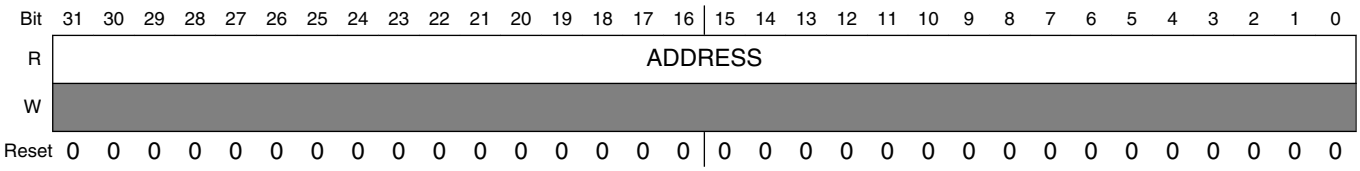
The APBH DMA Channel n buffer address register contains a pointer to the data buffer for the transfer. For immediate forms, the data is taken from this register. This is a byte address which means transfers can start on any byte boundary.

This register holds a pointer to the data buffer in system memory. After the command values have been read into the DMA controller and the device controlled by this channel, then the DMA transfer will begin, to or from the buffer pointed to by this register.

EXAMPLE

```
apbh_chn_bar_t  dma_data;
dma_data.ADDRESS = (reg32_t) pDataBuffer;
```

Address: 3300_0000h base + 130h offset + (112d × i), where i=0d to 15d



APBH_CH*n*_BAR field descriptions

Field	Description
ADDRESS	Address of system memory buffer to be read or written over the AHB bus.

9.4.4.12 APBH DMA Channel n Semaphore Register (APBH_CH*n*_SEMA)

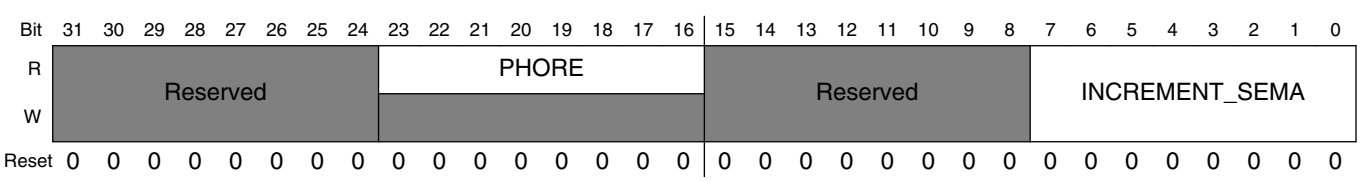
The APBH DMA Channel n semaphore register is used to synchronize the Arm platform instruction stream and the DMA chain processing state.

Each DMA channel has an 8 bit counting semaphore that is used to synchronize between the program stream and and the DMA chain processing. DMA processing continues until the DMA attempts to decrement a semaphore that has already reached a value of zero. When the attempt is made, the DMA channel is stalled until software increments the semaphore count.

EXAMPLE

```
BF_WR(APBH_CHn_SEMA, 0, INCREMENT_SEMA, 2);     // increment semaphore by two
current_sema = BF_RD(APBH_CHn_SEMA, 0, PHORE);  // get instantaneous value
```

Address: 3300_0000h base + 140h offset + (112d × i), where i=0d to 15d



APBH_CH*n*_SEMA field descriptions

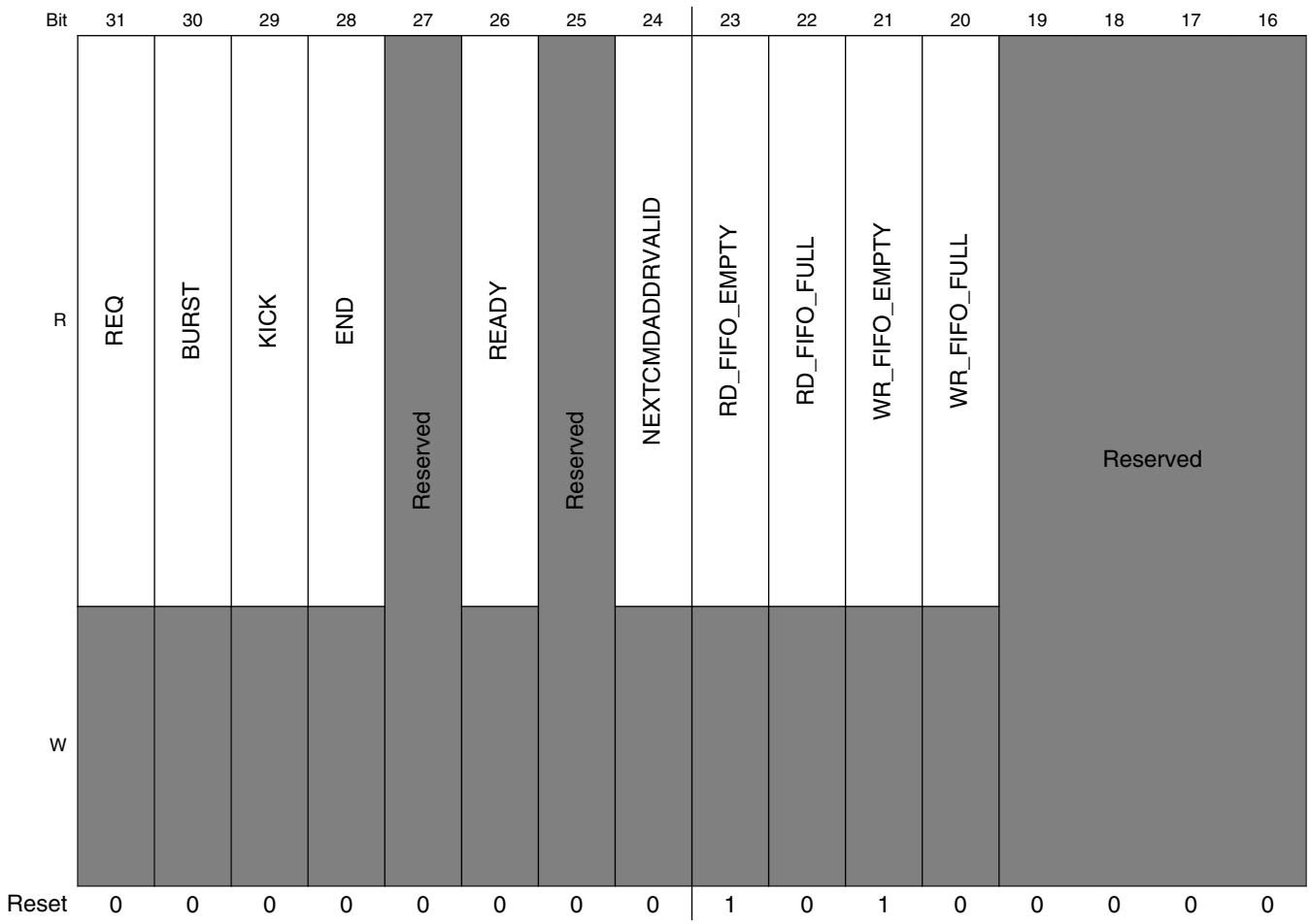
Field	Description
31–24 -	This field is reserved. Reserved, always set to zero.
23–16 PHORE	This read-only field shows the current (instantaneous) value of the semaphore counter.
15–8 -	This field is reserved. Reserved, always set to zero.
INCREMENT_ SEMA	The value written to this field is added to the semaphore count in an atomic way such that simultaneous software adds and DMA hardware substracts happening on the same clock are protected. This bit field reads back a value of 0x00. Writing a value of 0x02 increments the semaphore count by two, unless the DMA channel decrements the count on the same clock, then the count is incremented by a net one.

9.4.4.13 AHB to APBH DMA Channel n Debug Information (APBH_CH*n*_DEBUG1)

This register gives debug visibility into the APBH DMA Channel n state machine and controls.

This register allows debug visibility of the APBH DMA Channel n.

Address: 3300_0000h base + 150h offset + (112d × i), where i=0d to 15d



APBH_CH*n*_DEBUG1 field descriptions

Field	Description
31 REQ	This bit reflects the current state of the DMA Request Signal from the APB device
30 BURST	This bit reflects the current state of the DMA Burst Signal from the APB device
29 KICK	This bit reflects the current state of the DMA Kick Signal sent to the APB Device
28 END	This bit reflects the current state of the DMA End Command Signal sent from the APB Device
27 SENSE	This field is reserved. This bit is reserved for this DMA Channel and always reads 0.
26 READY	This bit is reserved for this DMA Channel and always reads 0.
25 LOCK	This field is reserved. This bit is reserved for this Channel and always reads 0.
24 NEXTCMDADDRVALID	This bit reflects the internal bit which indicates whether the channel's next command address is valid.
23 RD_FIFO_EMPTY	This bit reflects the current state of the DMA Channel's Read FIFO Empty signal.
22 RD_FIFO_FULL	This bit reflects the current state of the DMA Channel's Read FIFO Full signal.
21 WR_FIFO_EMPTY	This bit reflects the current state of the DMA Channel's Write FIFO Empty signal.
20 WR_FIFO_FULL	This bit reflects the current state of the DMA Channel's Write FIFO Full signal.
19–5 RSVD1	This field is reserved. Reserved
STATEMACHINE	PIO Display of the DMA Channel n state machine state. 0x00 **IDLE** — This is the idle state of the DMA state machine. 0x01 **REQ_CMD1** — State in which the DMA is waiting to receive the first word of a command. 0x02 **REQ_CMD3** — State in which the DMA is waiting to receive the third word of a command. 0x03 **REQ_CMD2** — State in which the DMA is waiting to receive the second word of a command. 0x04 **XFER_DECODE** — The state machine processes the descriptor command field in this state and branches accordingly. 0x05 **REQ_WAIT** — The state machine waits in this state for the PIO APB cycles to complete. 0x06 **REQ_CMD4** — State in which the DMA is waiting to receive the fourth word of a command, or waiting to receive the PIO words when PIO count is greater than 1. 0x07 **PIO_REQ** — This state determines whether another PIO cycle needs to occur before starting DMA transfers. 0x08 **READ_FLUSH** — During a read transfers, the state machine enters this state waiting for the last bytes to be pushed out on the APB. 0x09 **READ_WAIT** — When an AHB read request occurs, the state machine waits in this state for the AHB transfer to complete. 0x0C **WRITE** — During DMA Write transfers, the state machine waits in this state until the AHB master arbiter accepts the request from this channel. 0x0D **READ_REQ** — During DMA Read transfers, the state machine waits in this state until the AHB master arbiter accepts the request from this channel.

Table continues on the next page...

APBH_CH*n*_DEBUG1 field descriptions (continued)

Field	Description
	0x0E **CHECK_CHAIN** — Upon completion of the DMA transfers, this state checks the value of the Chain bit and branches accordingly. 0x0F **XFER_COMPLETE** — The state machine goes to this state after the DMA transfers are complete, and determines what step to take next. 0x14 **TERMINATE** — When a terminate signal is set, the state machine enters this state until the current AHB transfer is completed. 0x15 **WAIT_END** — When the Wait for Command End bit is set, the state machine enters this state until the DMA device indicates that the command is complete. 0x1C **WRITE_WAIT** — During DMA Write transfers, the state machine waits in this state until the AHB master completes the write to the AHB memory space. 0x1D **HALT_AFTER_TERM** — If HALTONTERMINATE is set and a terminate signal is set, the state machine enters this state and effectively halts. A channel reset is required to exit this state 0x1E **CHECK_WAIT** — If the Chain bit is a 0, the state machine enters this state and effectively halts. 0x1F **WAIT_READY** — When the NAND Wait for Ready bit is set, the state machine enters this state until the GPMI device indicates that the external device is ready.

9.4.4.14 AHB to APBH DMA Channel n Debug Information (APBH_CH*n*_DEBUG2)

This register gives debug visibility for the APB and AHB byte counts for DMA Channel n.

This register allows debug visibility of the APBH DMA Channel n.

Address: 3300_0000h base + 160h offset + (112d × i), where i=0d to 15d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	APB_BYTES																AHB_BYTES															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

APBH_CH*n*_DEBUG2 field descriptions

Field	Description
31–16 APB_BYTES	This value reflects the current number of APB bytes remaining to be transfered in the current transfer.
AHB_BYTES	This value reflects the current number of AHB bytes remaining to be transfered in the current transfer.

9.4.4.15 APBH Bridge Version Register (APBH_VERSION)

This register always returns a known read value for debug purposes it indicates the version of the block.

This register indicates the RTL version in use.

EXAMPLE

```
if (APBH_VERSION.B.MAJOR != 3)
   Error();
```

Address: 3300_0000h base + 800h offset = 3300_0800h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								STEP															
W																																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

APBH_VERSION field descriptions

Field	Description
31–24 MAJOR	Fixed read-only value reflecting the MAJOR field of the RTL version.
23–16 MINOR	Fixed read-only value reflecting the MINOR field of the RTL version.
STEP	Fixed read-only value reflecting the stepping of the RTL version.

9.5 62BIT Correcting ECC Accelerator (BCH)

9.5.1 Overview

The hardware ECC accelerator provides a forward error-correction function for improving the reliability of various storage media that may be attached to the device.

For example, NAND flash devices use a spare area to store ecc codes to correct some hard bit errors in data stored within the device, allowing higher device yields and, therefore, lower NAND device costs.

The Bose, Ray-Chaudhuri, Hocquenghem (BCH) Encoder and Decoder module is capable of correcting from 2 to 62 single bit errors within a block of data no larger than about 1900 bytes (512 bytes or 1024 bytes are typical) in applications such as protecting data and resources stored on modern NAND flash devices. The correction level in the BCH block is programmable to provide flexibility for varying applications and configurations of flash page size. The design can be programmed to encode protection of 2 to 62 bit errors when writing flash and to correct the corresponding number of errors on decode. The correction level when decoding MUST be programmed to the same correction level as was used during the encode phase.

BCH-codes are a type of block-code, which implies that all error-correction is performed over a block of N-symbols. The BCH operation will be performed over GF(2^{13} = 8192) or GF(2^{14} = 16384), which is the Galois Field consisting of 8191 or 16383 one-bit symbols. BCH-encoding (or encode for any block-code) can be performed by two algorithms: systematic encoding or multiplicative encoding. Systematic encoding is the process of reading all the symbols which constitute a block, dividing continuously these symbols by the generator polynomial for the GF(8192) or GF(16384) and appending the resulting ***t*** parity symbols to the block to create a BCH codeword (where ***t*** is the number of correctable bits).

The BCH encode process creates *t**13 (or *t**14)-bit parity symbols for each data block when the data is written to the flash device. The parity symbols are written to the flash device after the corresponding data block, and together these are collectively called the codeword. The codeword can be used during the decode process to correct errors that occur in either the data or parity blocks.

The BCH decoder processes code words in a 4-step fashion:

1. Syndrome Calculation (SC): This is the process of reading in all of the symbols of the codeword and continuously dividing by the generator polynomial for the field. 2**t* syndromes must be calculated for each codeword and inspection of the syndromes determines if there are errors: a non-zero set of syndromes indicates one or more errors. This process is implemented parallel hardware to minimize processing time since it must be done every time the decode is performed.
2. Key Equation Solver (KES): The syndromes represent 2t-linear equations with 2t-unknown variables. The process of solving these equations and selecting from the numerous solutions constitutes the KES module. When the KES block completes its operations, it generates an error locator polynomial (sigma) that is used in the proceeding block to determine the locations and values of the errors.
3. Chien Search (CS): This block takes input from the KES block and uses the Chien Algorithm for finding the locations of the errors based on the error locator polynomial. The method basically involves substituting all 8191 symbols from the GF(8192) or 16383 symbols from the GF(16383) into the locator polynomial. All evaluations that produce a zero solution indicate locations of the various errors. Since each located error corresponds to a single bit, the bit in the original data may be corrected by simply flipping the polarity of the incorrect location.
4. Correction: this block has to convert the symbol index and mask information to memory byte indexes and masks.

The BCH block, shown in the figure, was designed to operate in a pipelined fashion to maximize throughput. Aside from the initial latency to fill the pipeline stages, the BCH throughput is about 7/4 cycles/byte. Thus, the bottleneck in performing NAND reads and error corrections is the BCH rate. Current GPMI read rates are approximately 1/2 cycles/byte maximally for the current generation of NAND flash. Fortunately, BCH has a different master clock from GPMI, this gives some flexibility to match the throughput rate. The CPU is not directly involved in generating parity symbols, checking for errors, or correcting them.

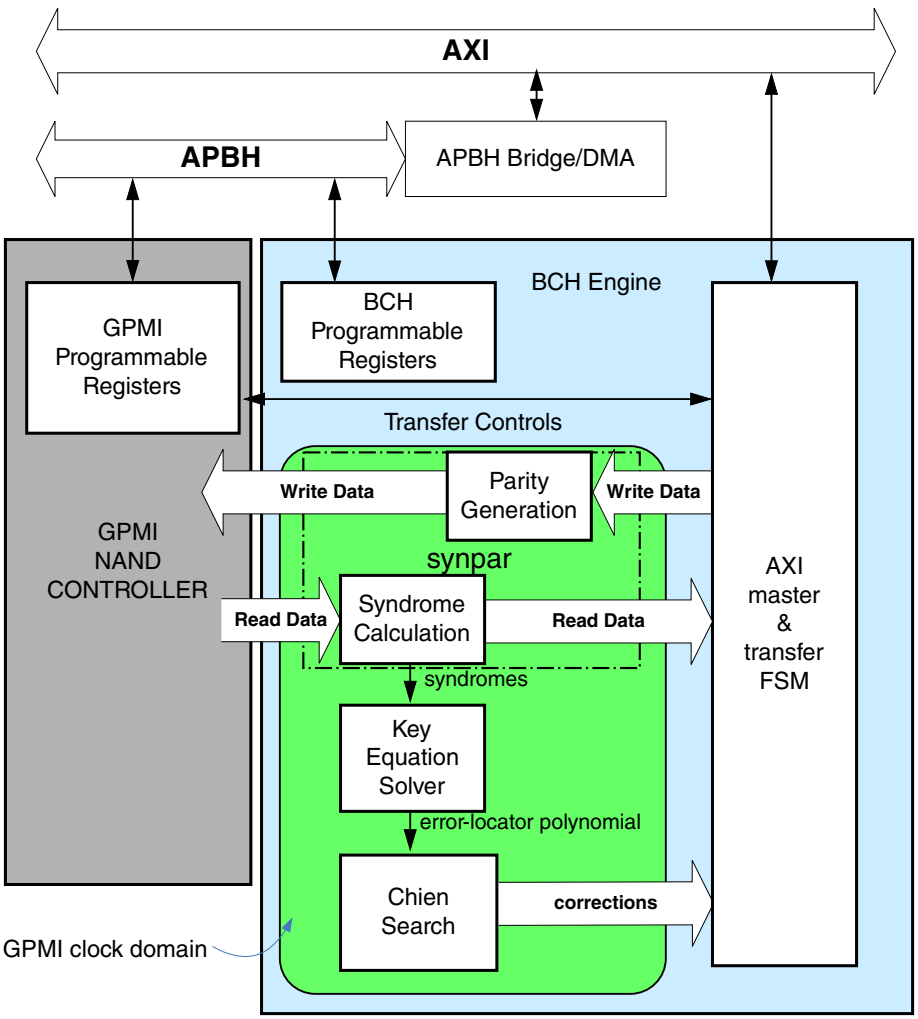


Figure 9-10. Hardware BCH Accelerator

9.5.2 Operation

Before performing any NAND flash read or write operations, software should first program the BCH's flash layout registers (see Flash Page Layout) to specify how data is to be formatted on the flash device. The BCH hardware allows full programmability over the flash page layout to enable users flexibility in balancing ECC correction levels and ever-changing flash page sizes.

To initiate a NAND Flash write, software will program a GPMI DMA operation. The DMA need only program the GPMI control registers (and handle the requisite flash addressing handshakes) since the BCH will handle all data operations using its AXI bus interface. The BCH will then send the data to the GPMI controller to be written to flash as it computes the parity symbols. At the end of each data block the BCH will insert the parity symbols into the data stream so that the GPMI sees only a continuous stream of data to be written.

NAND Flash read operations operate in a similar manner. As the GPMI controller reads the device, all data is sent to the BCH hardware for error detection/correction. The BCH controller writes all incoming read data to system memory and in parallel computes the syndromes used to detect bit errors. If errors are detected within a block, the BCH hardware activates the error correction logic to determine where bit errors have occurred and ultimately correct them in the data buffer in system memory. After an entire flash page has been read and corrected, the BCH will signal an interrupt to the CPU.

The figure below indicates how data read from the GPMI is operated on within the BCH hardware. As the BCH receives data from the GPMI (top row), it is written to memory by the BCH's Bus Interface Unit (BIU) (second row). For blocks requiring correction, the KES logic will be activated after the entire block has been received. Once the error locator polynomial has been computed, the corrections are determined by the Chien Search and fed back to the BIU, which performs a read, modify, write operation on the buffer in memory to correct the data.

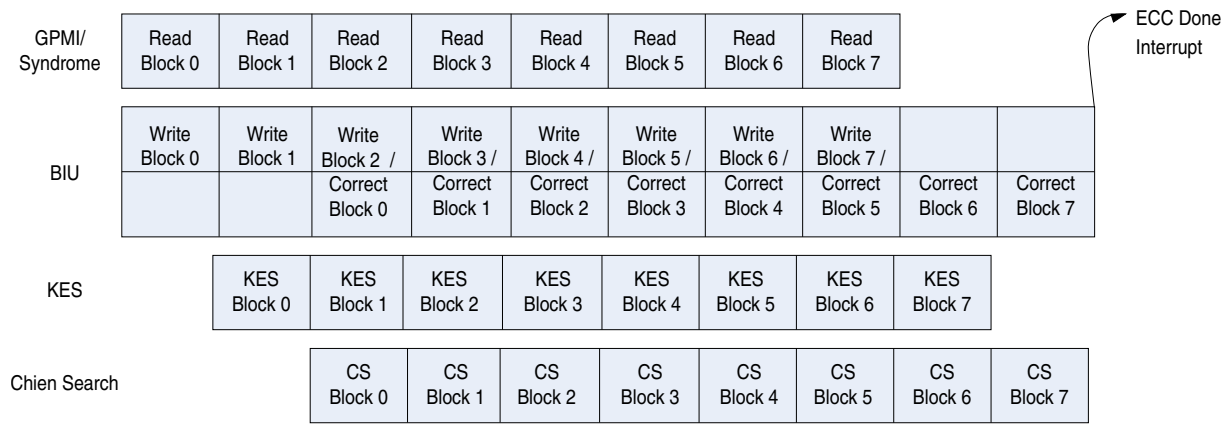


Figure 9-11. Block Pipeline while Reading Flash

9.5.2.1 BCH Limitations and Assumptions

- The BCH is programmable to support 2 to 62 bit error correction. ECC0 is supported as a pass-through, non-correcting mode.
- Data block sizes must be a multiple of 4 bytes and be aligned in system memory.
- The BCH supports a programmable number of metadata/auxiliary data bytes, from 0 to 255.
- Metadata will be written at the beginning of the flash page to facilitate fast access for filesystem operations.
- Metadata may be treated as an independent block for ECC purposes or combined with the first data block to conserve bits in the flash.
- The BCH does not support a partial page write (this can be accomplished by programming the BCH layout registers such that the BCH only sees a portion of the page).
- Flash read operations can read the entire page or the first block on the page.
- The BCH also supports a memory-to-memory mode of operation that does not require the use of DMA or the GPMI.

9.5.2.2 Flash Page Layout

The BCH supports a fully programmable flash page layout. The BCH maintains four independent layout registers that can describe four completely different NAND devices or layouts.

When the BCH initiates an operation, it selects one of the layouts by using the chip select as an index into the BCH_LAYOUTSELECT register that determines which layout should be used for the operation.

Three possible (generic) flash layout schemes are supported, as indicated in the figure below. (In each case, the metadata size may also be programmed to 0 bytes). Metadata may either be combined with the first block of data or the size of the first data block can be programmed to 0 to allow the metadata to be protected by its own ECC parity bits.

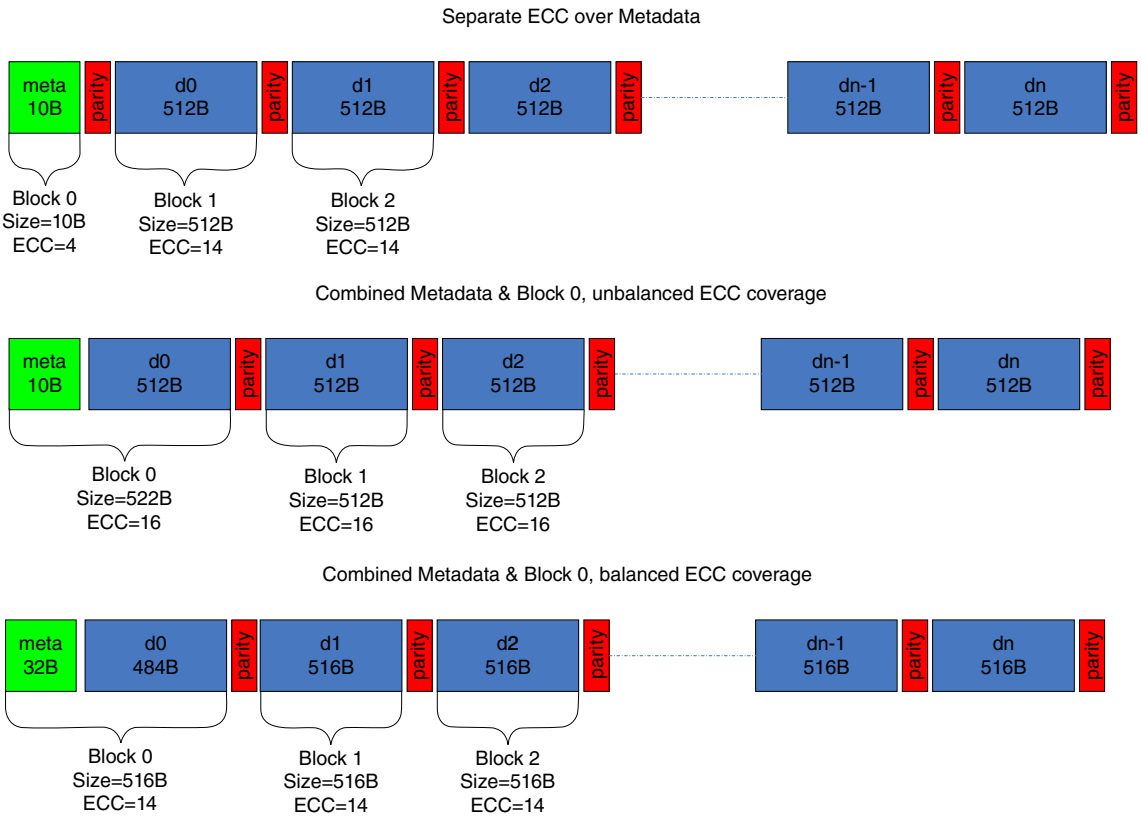


Figure 9-12. FLASH Page Layout Options

Each layout is determined by a pair of registers that define the following parameters:

- DATA0_SIZE: Indicates the number of data bytes in the first block on the page (this should not include parity or metadata bytes). This should be set to 0 when the metadata is to be covered separately with its own ECC. This must be a multiple of 4 bytes.
- ECC0: Indicates the ECC level to be used for the first block on the flash (data0+metadata).
- META_SIZE: Indicates the number of bytes (from 0-255) that are stored as metadata.

- NBLOCKS: Indicates the number of subsequent DATAN blocks on the flash, or the number of blocks following the DATA0 block.
- DATAN_SIZE: Indicates the number of data bytes in all subsequent data blocks. This MUST be a multiple of 4 bytes.
- ECCN: Indicates the ECC level to be used for the subsequent data blocks.
- GF0 or GFN: Indicates the Galois field the meta / data blocks are using
- PAGE_SIZE: Indicates the total number of bytes available per page on the physical flash device. This includes the spare area and is typically 4096+128, 4096+218, or 2048+64 bytes.

9.5.2.3 Determining the ECC layout for a device

Since the BCH is programmable, a system can trade off ECC levels for flash size and layout configurations.

The following examples indicate how to determine a valid layout based on the required storage space and flash size. For all cases, the size of the parity will be 13 (or 14 for GF(2^{14}))*ECC level ***bits***-- so for ECC8, 13 (or 14) bytes are required (per block).

9.5.2.3.1 4K+218 flash, 10 bytes metadata, 512 byte data blocks, separate metadata, Assuming GF(2^{13})

In this case, we have 8 data blocks each consisting of 512 bytes. Since the flash has 218 spare bytes (1744 bits), first estimate an ECC level for the data blocks by first subtracting the number of metadata bytes from the spare bytes (218 – 10 = 208 bytes = 1664 bits) then dividing the number of bits by 8 (number of blocks) and then by 13 (bits per ECC level).

(218 - 10) x 8 = 1664/13(8) = 16

Therefore all the data blocks could be covered by ECC16 if the metadata had no parity. This isn't acceptable, so assume ECC14 for all the data blocks. Now calculate the number of free bits for the metadata parity as

1664 - (14) x 13 x 8 = 208

Therefore, 208 bits remain for metadata parity. Dividing by 13 (bits/ECC) gives 16, so the metadata can be covered with ECC16. The settings for this device would then be

Table 9-8. Settings for 4K+218 FLASH

Setting	Value
PAGE_SIZE	4096+218=4314=0x10DA
META_SIZE	10=0x0A
DATA0_SIZE	0
ECC0	16=0x10
GF0	GF(2^{13})
DATAN_SIZE	512=0x200 (in register interface, assigned as 0x80)
ECCN	14=0x0E
GFN	GF(2^{13})
NBLOCKS	8

9.5.2.3.2 4K+128 flash, 10 bytes metadata, 1024 byte data blocks, separate metadata, assuming GF(2^{13}) for data and GF(2^{14}) for metadata

This flash will have 118 bytes available for ECC (after subtracting the metadata size), therefore, 994 bits.

Dividing by 4*14 (number of blocks * ECC level) we get 17.75, therefore we can support ECC16 on the data blocks. The number of free spare bits becomes 944 - 16 * 4 * 14 = 944 - 896 = 48, divided by 13 = 3.69, therefore the metadata can be also covered by ECC2.

Table 9-9. Settings for 4K+128 FLASH

Setting	Value
PAGE_SIZE	4096+128=4224=0x1080
META_SIZE	10=0x0A
DATA0_SIZE	0
ECC0	2
GF0	GF(2^{13})
DATAN_SIZE	1024=0x400 (in register interface, assigned as 0x100)
ECCN	16
GFN	GF(2^{14})
NBLOCKS	4

In this case, there will be additional unused spare bits, with the BCH will pad out with zeros.

9.5.2.4 Data Buffers in System Memory

While the data on the flash is interleaved with parity symbols, the BCH assumes that the data buffers in memory are contiguous.

Metadata read from the flash will be stored to the location pointed to by the GPMI_AUXILIARY register and data will be written to the address specified in the GPMI_PAYLOAD register as is shown in the following figure where the block length is 512 bytes for example. Since the number of blocks on a flash page is programmable, the BCH also writes individual block correction status to the auxiliary pointer at the word-aligned address following the end of the metadata. Optionally, the computed syndromes may also be written to the auxiliary area if the DEBUGSYNDROME bit is set in the control register.

As blocks complete processing, the bus master will accumulate the status for each block and write it to the auxiliary data buffer following the metadata. The metadata area will be padded with 0's until the next word boundary and the status for blocks 0-3 will be written to the next word. The status for subsequent blocks will then be written to the buffer. The status for the first block (metadata block) is also stored in the STATUS_BLK0 register in the BCH_STATUS register. The completion codes for the blocks are indicated in the Table 9-10. Note that the definition of the bytes and their ordering in the auxiliary and payload storage areas are user defined. When this data is read back from the flash and put into memory, it will resemble the original buffer that was written out to the flash.

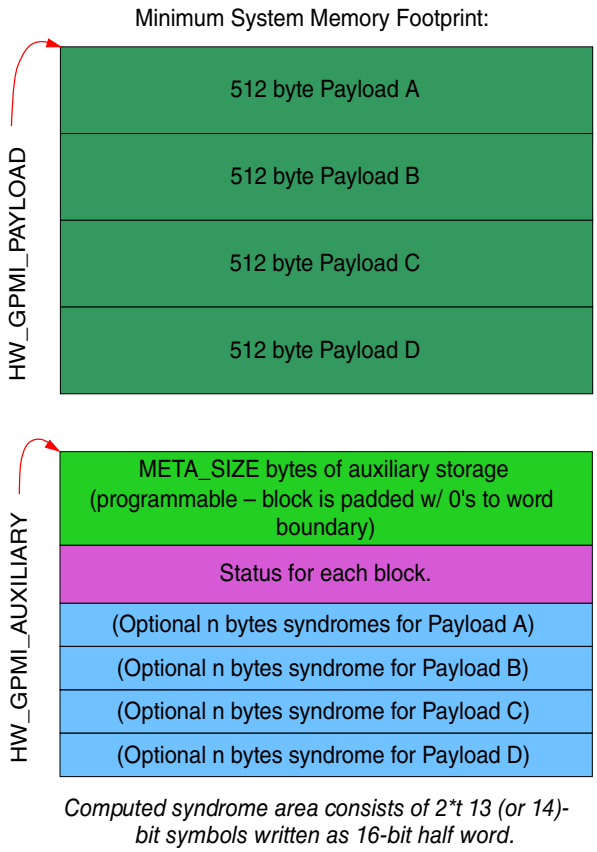
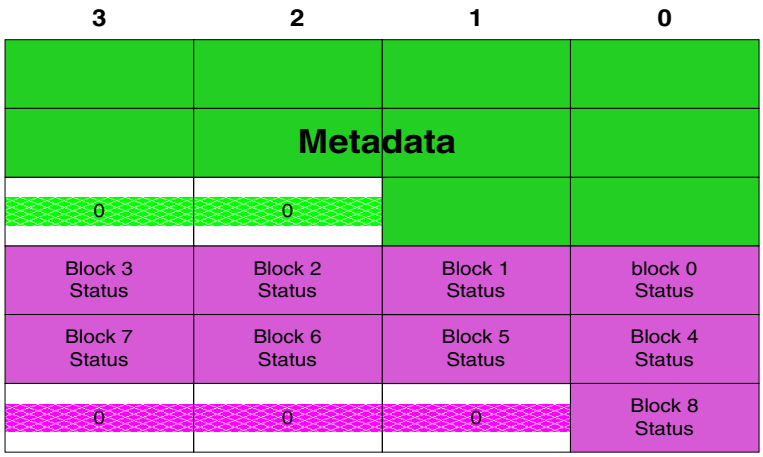


Figure 9-13. BCH Data Buffers in Memory

Table 9-10. Status Block Completion Codes

Code	Description
0xFF	Block is erased
0xFE	Block is uncorrectable
0x00	No errors found
0x01-0x3E	Number of errors corrected

The following figure shows the layout of the bytes within the status field.



Status bytes are allocated based on the NBLOCKS programmed into the flash format register. The number of status bytes are computed by NBLOCK+1. The status area will be padded with zeros to the next word boundary.

Syndrome data written for debug purposes follows the end of the status block.

Figure 9-14. Memory-to-Memory Operations

9.5.3 Memory to Memory (Loopback) Operation

The BCH supports a memory-to-memory mode of operation where both the encoded and decoded buffers reside in system memory.

This can be useful for applications where data must be protected by ECC, but the storage device does not reside on the GPMI bus.

The BCH operation in memory to memory mode is much simpler than in GPMI mode since DMAs are not required to manage the operation. Instead, software simply writes the BCH_DATAPTR and BCH_METAPTR with the addresses of the data and metadata (auxiliary) buffers and the BCH_ENCODEPTR with the address of the buffer for encoded data. To initiate the operation, software simply sets the M2M_ENCODE and M2M_ENABLE bits in the control register. The BCH can be programmed to either issue an interrupt at the end of the operation or software may poll the status bits for completion.

Memory to memory decode operations work in a similar manner. The encoded data address is written to the BCH_ENCODEPTR and the data and meta pointers are written to buffers that correspond to the desired decoded data addresses. To initiate a decode, software must set the M2M_ENCODE bit to 0 while writing the M2M_ENABLE bit. Note that the addresses written to the BCH_DATAPTR, BCH_METAPTR and BCH_ENCODEPTR registers should always be aligned on a 4 byte boundary. In other words, the 2 lower bits of the address should always be written with zeros.

9.5.4 Programming the BCH/GPMI Interfaces

Programming the BCH for NAND operations consists largely of disabling the soft reset and clock bits (SFTRST and CLKGATE) from the BCH_CTRL register and then programming the flash layout registers to correspond to the format of the attached NAND device(s).

The BCH_LAYOUTSELECT register should also be programmed to map the chip select of each attached device into one of the four layout registers.

The bulk of the programming is actually applied to the GPMI through PIO operations embedded in DMA command structures. The DMA will perform all the requisite handshaking with the GPMI interface to negotiate the address portion of the transfer, then the BCH will handle all the movement of data from memory to the GPMI (writes) or the GPMI to memory (reads). The BCH will direct all data blocks to the buffer pointed to by the PAYLOAD_BUFFER and the metadata will be written to the AUXILIARY_BUFFER. Both of these registers are located in the GPMI PIO data space and are communicated to the BCH hardware at the beginning of the transfer. Thus, the normal multi-NAND DMA based device interleaving is preserved, that is, four NANDs on four separate chip selects can be scheduled for read or write operations using the BCH. Whichever channel finishes its ready wait first and enters the DMA arbiter with its lock bit set owns the GPMI command interface and through it owns the BCH resources for the duration of its processing.

9.5.4.1 BCH Encoding for NAND Writes

The BCH encoder flowchart in Figure 9-15 shows the detailed steps involved in programming and using the BCH encoder. This flowchart shows how to use the BCH block with the GPMI.

To use the BCH encoder with the GPMI's DMA, create a DMA command chain containing ten descriptor structures, as shown in Figure 9-17 and detailed in the DMA structure code example that follows it in DMA Structure Code Example. The ten descriptors perform the following tasks:

1. Disable the BCH block (in case it was enabled) and issue NAND write setup command byte (under CLE) and address bytes (under ALE).
2. Configure and enable the BCH and GPMI blocks to perform the NAND write.
3. Disable the BCH block and issue NAND write execute command byte (under CLE).

4. Wait for the NAND device to finish writing the data by watching the ready signal.
5. Check for NAND timeout through PSENSE.
6. Issue NAND status command byte (under CLE).
7. Read the status and compare against expected.
8. If status is incorrect or incomplete, branch to error handling descriptor chain.
9. Otherwise, write is complete and emit GPMI interrupt.

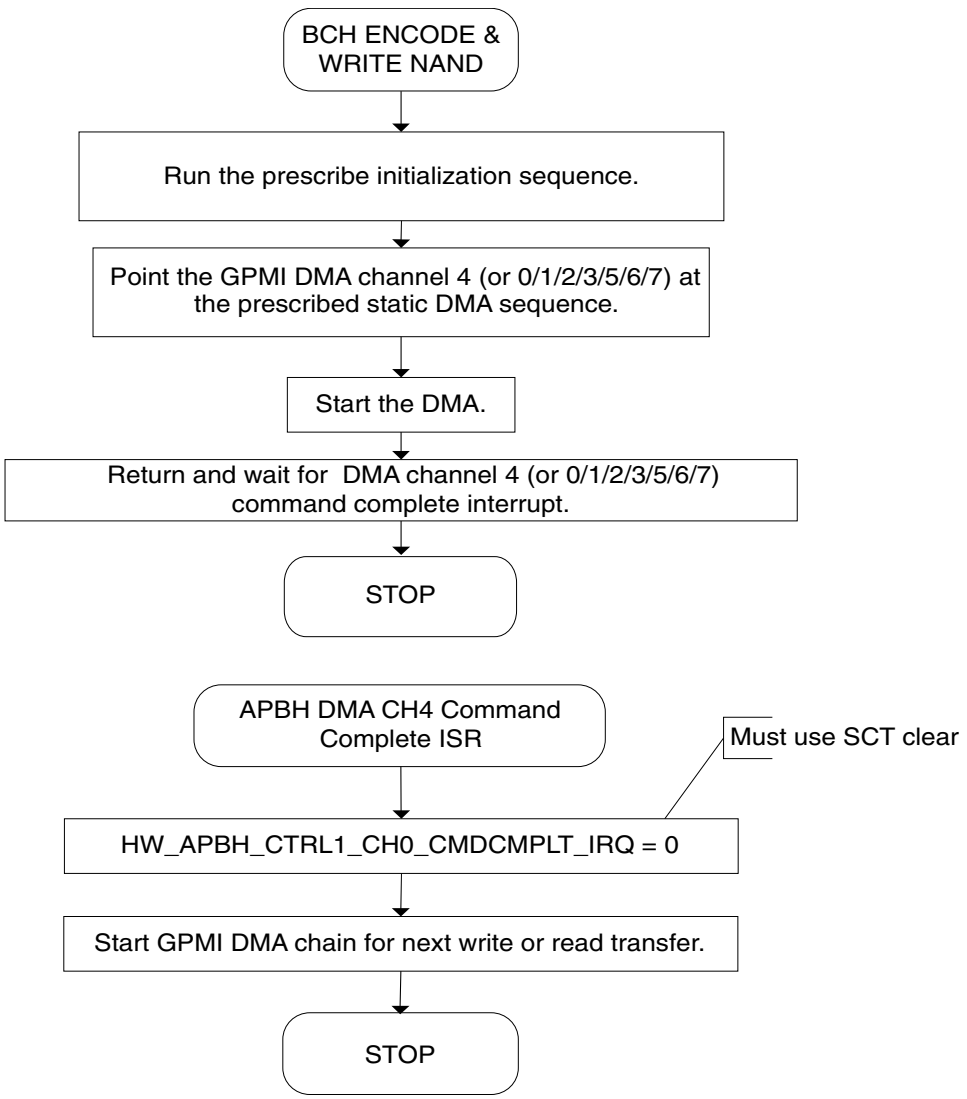


Figure 9-15. BCH Encode Flowchart

Descriptor Legend

NEXT CMD ADDR										
CMD	<=	xfer_count	cmdwords	wait4endcmd	semaphore	nandwait4ready	nandlock	irqoncmplt	chain	command
BUFFER ADDR										
HW_GPMI_CTRL0	<=	command_mode	word_length	lock_cs	CS	address	address_increment	xfer_count		
HW_GPMI_COMPARE	<=	mask				reference				
HW_GPMI_ECCCTRL	<=	ecc_cmd			enable_ecc				buffer_mask	
HW_GPMI_ECCCOUNT										
HW_GPMI_PAYLOAD										
HW_GPMI_AUXILIARY										

Figure 9-16. BCH DMA Descriptor Legend

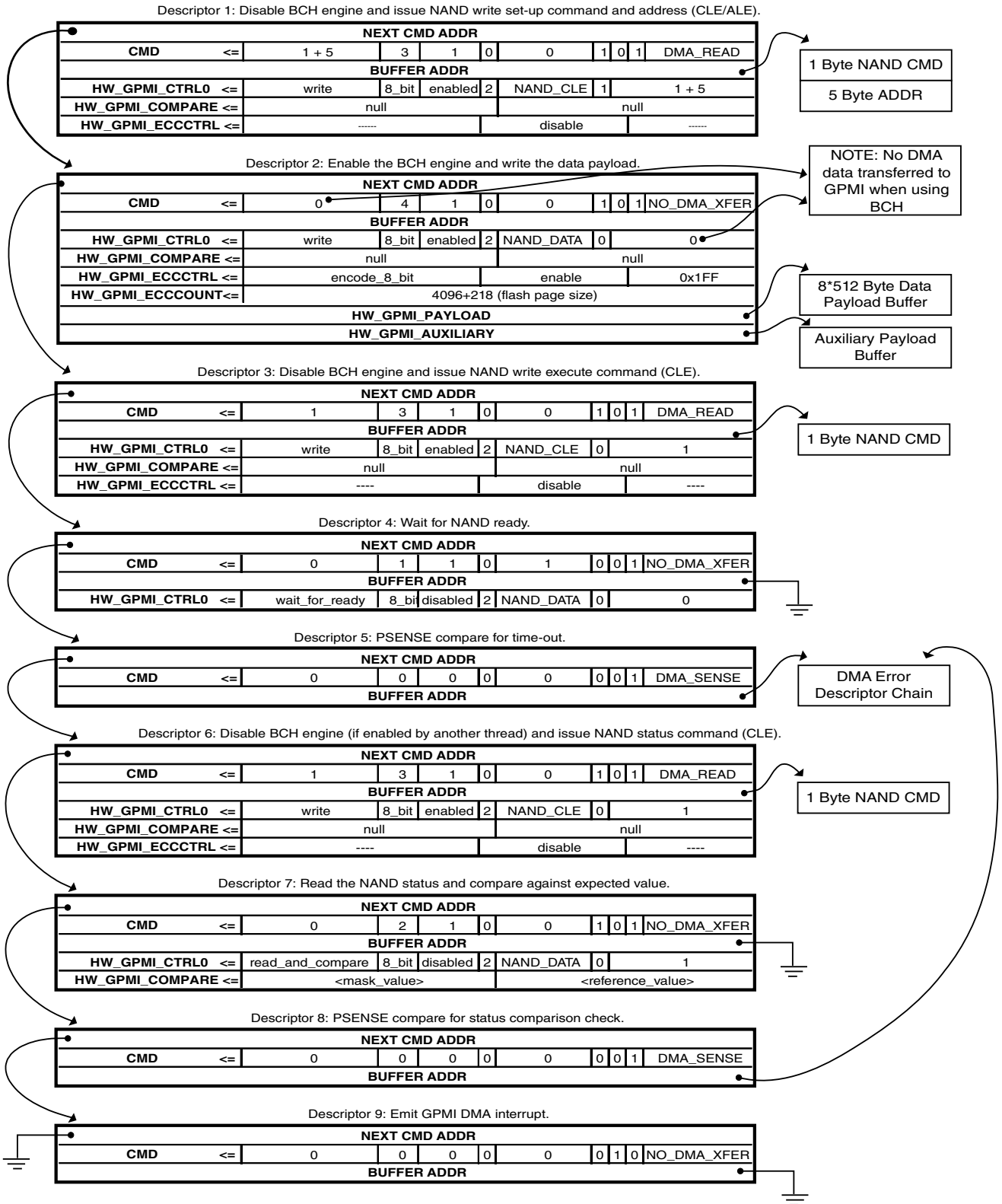


Figure 9-17. BCH Encode DMA Descriptor Chain

9.5.4.1.1 DMA Structure Code Example

The following code sample illustrates the coding for one write transaction involving 4096 bytes of data payload (eight 512-byte blocks) and 10 bytes of auxiliary payload (also referred to as metadata) to a 4K NAND page sitting on GPMI CS2.

```
//-----------------------------------------------------------------------------
// generic DMA/GPMI/ECC descriptor struct, order sensitive!
//-----------------------------------------------------------------------------
typedef struct {
  // DMA related fields
  unsigned int dma_nxtcmdar;
  unsigned int dma_cmd;
  unsigned int dma_bar;
  // GPMI related fields
  unsigned int gpmi_ctrl0;
  unsigned int gpmi_compare;
  unsigned int gpmi_eccctrl;
  unsigned int gpmi_ecccount;
  unsigned int gpmi_data_ptr;
  unsigned int gpmi_aux_ptr;
} GENERIC_DESCRIPTOR;
//-----------------------------------------------------------------------------
// allocate 10 descriptors for doing a NAND ECC Write
//-----------------------------------------------------------------------------
GENERIC_DESCRIPTOR write[10];
//-----------------------------------------------------------------------------
// DMA descriptor pointer to handle error conditions from psense checks
//-----------------------------------------------------------------------------
unsigned int * dma_error_handler;
//-----------------------------------------------------------------------------
// 8 byte NAND command and address buffer
// any alignment is ok, it is read by the GPMI DMA
//   byte 0 is write setup command
//   bytes 1-5 is the NAND address
//   byte 6 is write execute command
//   byte 7 is status command
//-----------------------------------------------------------------------------
unsigned char nand_cmd_addr_buffer[8];
//-----------------------------------------------------------------------------
// 4096 byte payload buffer used for reads or writes
// needs to be word aligned
//-----------------------------------------------------------------------------
unsigned int write_payload_buffer[(4096/4)];
//-----------------------------------------------------------------------------
// 65 byte meta-data to be written to NAND
// needs to be word aligned
//-----------------------------------------------------------------------------
unsigned int write_aux_buffer[65];
//-----------------------------------------------------------------------------
// Descriptor 1: issue NAND write setup command (CLE/ALE)
//-----------------------------------------------------------------------------
write[0].dma_nxtcmdar = &write[1];                          // point to the next descriptor
write[0].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (1 + 5)|   // 1 byte command, 5 byte address
                   BF_APBH_CHn_CMD_CMDWORDS      (3)    |   // send 3 words to the GPMI
                 BF_APBH_CHn_CMD_WAIT4ENDCMD   (1)    |  // wait for command to finish
before
                                                            //    continuing
                   BF_APBH_CHn_CMD_SEMAPHORE     (0)    |
                   BF_APBH_CHn_CMD_NANDWAIT4READY(0)    |
                   BF_APBH_CHn_CMD_NANDLOCK      (1)    |   // prevent other DMA channels
from
                                                            //    taking over
                   BF_APBH_CHn_CMD_IRQONCMPLT    (0)    |
                   BF_APBH_CHn_CMD_CHAIN         (1)    |   // follow chain to next command
```

```
                BV_FLD(APBH_CHn_CMD, COMMAND, DMA_READ); // read data from DMA, write to
NAND
write[0].dma_bar = &nand_cmd_addr_buffer;           // byte 0 write setup, bytes 1 - 5 NAND
address
// 3 words sent to the GPMI
write[0].gpmi_ctrl0 = BV_FLD(GPMI_CTRL0, COMMAND_MODE, WRITE)    | // write to the NAND
                      BV_FLD(GPMI_CTRL0, WORD_LENGTH,  8_BIT)    |
                      BV_FLD(GPMI_CTRL0, LOCK_CS,      ENABLED)  |
                    BF_GPMI_CTRL0_CS              (2)      | // must correspond to NAND CS
used
                      BV_FLD(GPMI_CTRL0, ADDRESS,      NAND_CLE) |
                      BF_GPMI_CTRL0_ADDRESS_INCREMENT  (1)       | // send command and
address
                    BF_GPMI_CTRL0_XFER_COUNT       (1 + 5);    // 1 byte command, 5 byte
address
write[0].gpmi_compare = NULL;                            // field not used but necessary to
set eccctrl
write[0].gpmi_eccctrl = BV_FLD(GPMI_ECCCTRL, ENABLE_ECC, DISABLE); // disable the ECC block
//--------------------------------------------------------------------------
// Descriptor 2: write the data payload (DATA)
//--------------------------------------------------------------------------
write[1].dma_nxtcmdar = &write[2];                       // point to the next descriptor
write[1].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT      (0)|  // NOTE: No DMA data transfer
                   BF_APBH_CHn_CMD_CMDWORDS        (4)|  // send 4 words to the GPMI
                   BF_APBH_CHn_CMD_WAIT4ENDCMD     (1)|  // Wait to end
                   BF_APBH_CHn_CMD_SEMAPHORE       (0)|
                   BF_APBH_CHn_CMD_NANDWAIT4READY  (0)|
                   BF_APBH_CHn_CMD_NANDLOCK        (1)|  // maintain resource lock
                   BF_APBH_CHn_CMD_IRQONCMPLT      (0)|
                   BF_APBH_CHn_CMD_CHAIN           (1)|  // follow chain to next command
                   BV_FLD(APBH_CHn_CMD, COMMAND, DMA_NO_XFER);  // No data transferred
write[1].dma_bar = &write_payload_buffer;                  // pointer for the 4K byte
data area
// 4 words sent to the GPMI
write[1].gpmi_ctrl0 = BV_FLD(GPMI_CTRL0, COMMAND_MODE, WRITE)    | // write to the NAND
                      BV_FLD(GPMI_CTRL0, WORD_LENGTH,  8_BIT)    |
                      BV_FLD(GPMI_CTRL0, LOCK_CS,      ENABLED)  |
                    BF_GPMI_CTRL0_CS                (2)       | // must correspond to NAND
CS used
                      BV_FLD(GPMI_CTRL0, ADDRESS,      NAND_DATA)|
                      BF_GPMI_CTRL0_ADDRESS_INCREMENT  (0)       |
                      BF_GPMI_CTRL0_XFER_COUNT         (0);        // NOTE: this field
contains
                                                                 //  the total amount
                              // DMA transferred to GPMI via DMA (0)!
write[1].gpmi_compare = NULL;                              // field not used but necessary
to
set eccctrl
write[1].gpmi_eccctrl = BV_FLD(GPMI_ECCCTRL, ECC_CMD,    ENCODE_8_BIT) | // specify t = 8
mode
                        BV_FLD(GPMI_ECCCTRL, ENABLE_ECC, ENABLE)       | // enable ECC module
                     BF_GPMI_ECCCTRL_BUFFER_MASK   (0x1FF);          // write all 8 data
blocks
                                                                   //   and 1 aux block
write[1].gpmi_ecccount = BF_GPMI_ECCCOUNT_COUNT(4096+218);      // specify number of bytes
                                                                //   written to NAND
write[1].gpmi_data_pointer = &write_payload_pointer;            // data buffer address
write[1].gpmi_aux_pointer  = &write_aux_pointer;                // metadata pointer
//--------------------------------------------------------------------------
// Descriptor 3: issue NAND write execute command (CLE)
//--------------------------------------------------------------------------
write[2].dma_nxtcmdar = &write[3];                          // point to the next descriptor
write[2].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (1) |     // 1 byte command
                   BF_APBH_CHn_CMD_CMDWORDS      (3) |     // send 3 words to the GPMI
                  BF_APBH_CHn_CMD_WAIT4ENDCMD   (1) |     // wait for command to finish
before
                                                          //    continuing
                   BF_APBH_CHn_CMD_SEMAPHORE     (0) |
                   BF_APBH_CHn_CMD_NANDWAIT4READY(0) |
                   BF_APBH_CHn_CMD_NANDLOCK      (1) |     // maintain resource lock
```

```
                  BF_APBH_CHn_CMD_IRQONCMPLT    (0) |
                  BF_APBH_CHn_CMD_CHAIN         (1) |     // follow chain to next command
                BV_FLD(APBH_CHn_CMD, COMMAND, DMA_READ);   // read data from DMA, write to
NAND
write[2].dma_bar = &nand_cmd_addr_buffer[6];             // point to byte 6, write execute
command
// 3 words sent to the GPMI
write[2].gpmi_ctrl0 = BV_FLD(GPMI_CTRL0, COMMAND_MODE, WRITE)    | // write to the NAND
                      BV_FLD(GPMI_CTRL0, WORD_LENGTH,  8_BIT)    |
                      BV_FLD(GPMI_CTRL0, LOCK_CS,      ENABLED)  |
                    BF_GPMI_CTRL0_CS               (2)      | // must correspond to NAND CS
used
                      BV_FLD(GPMI_CTRL0, ADDRESS,      NAND_CLE) |
                      BF_GPMI_CTRL0_ADDRESS_INCREMENT  (0)       |
                      BF_GPMI_CTRL0_XFER_COUNT         (1);       // 1 byte command
write[2].gpmi_compare = NULL;                           // field not used but necessary to set
eccctrl
write[2].gpmi_eccctrl = BV_FLD(GPMI_ECCCTRL, ENABLE_ECC, DISABLE); // disable the ECC block
//--------------------------------------------------------------------------
// Descriptor 4: wait for ready (CLE)
//--------------------------------------------------------------------------
write[3].dma_nxtcmdar = &write[4];                       // point to the next descriptor

write[3].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (0) |   // no dma transfer
                   BF_APBH_CHn_CMD_CMDWORDS      (1) |   // send 1 word to the GPMI
                   BF_APBH_CHn_CMD_WAIT4ENDCMD   (1) |   // wait for command to finish before
                                                         //      continuing
                   BF_APBH_CHn_CMD_SEMAPHORE     (0) |
                   BF_APBH_CHn_CMD_NANDWAIT4READY(1) |   // wait for nand to be ready
                   BF_APBH_CHn_CMD_NANDLOCK      (0) |   // relinquish nand lock
                   BF_APBH_CHn_CMD_IRQONCMPLT    (0) |
                   BF_APBH_CHn_CMD_CHAIN         (1) |   // follow chain to next command
                   BV_FLD(APBH_CHn_CMD, COMMAND, NO_DMA_XFER);          // no dma transfer
write[3].dma_bar = NULL;                                                // field not used
// 1 word sent to the GPMI
write[3].gpmi_ctrl0 = BV_FLD(GPMI_CTRL0, COMMAND_MODE, WAIT_FOR_READY) | // wait for NAND
ready
                      BV_FLD(GPMI_CTRL0, WORD_LENGTH,  8_BIT)         |
                      BV_FLD(GPMI_CTRL0, LOCK_CS,      DISABLED)      |
                  BF_GPMI_CTRL0_CS              (2)         | // must correspond to NAND CS
used
                      BV_FLD(GPMI_CTRL0, ADDRESS,      NAND_DATA)     |
                      BF_GPMI_CTRL0_ADDRESS_INCREMENT  (0)            |
                      BF_GPMI_CTRL0_XFER_COUNT         (0);
//--------------------------------------------------------------------------
// Descriptor 5: psense compare (time out check)
//--------------------------------------------------------------------------
write[4].dma_nxtcmdar = &write[5];                         // point to the next descriptor
write[4].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (0)       | // no dma transfer
                   BF_APBH_CHn_CMD_CMDWORDS      (0)       | // no words sent to GPMI
                   BF_APBH_CHn_CMD_WAIT4ENDCMD   (0)       | // do not wait to continue
                   BF_APBH_CHn_CMD_SEMAPHORE     (0)       |
                   BF_APBH_CHn_CMD_NANDWAIT4READY(0)       |
                   BF_APBH_CHn_CMD_NANDLOCK      (0)       |
                   BF_APBH_CHn_CMD_IRQONCMPLT    (0)       |
                   BF_APBH_CHn_CMD_CHAIN         (1)       | // follow chain to next command
                   BV_FLD(APBH_CHn_CMD, COMMAND, DMA_SENSE);  // perform a sense check
write[4].dma_bar = dma_error_handler;                // if sense check fails, branch to error
handler
//--------------------------------------------------------------------------
// Descriptor 6: issue NAND status command (CLE)
//--------------------------------------------------------------------------
write[5].dma_nxtcmdar = &write[6];                         // point to the next descriptor
write[5].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (1)     |  // 1 byte command
                   BF_APBH_CHn_CMD_CMDWORDS      (3)     |  // send 3 words to the GPMI
                 BF_APBH_CHn_CMD_WAIT4ENDCMD   (1)     |  // wait for command to finish
before
continuing
                   BF_APBH_CHn_CMD_SEMAPHORE     (0)     |
                   BF_APBH_CHn_CMD_NANDWAIT4READY(0)     |
```

```
                    BF_APBH_CHn_CMD_NANDLOCK      (1)    |  // prevent other DMA channels from
taking over
                     BF_APBH_CHn_CMD_IRQONCMPLT    (0)    |
                     BF_APBH_CHn_CMD_CHAIN         (1)    |  // follow chain to next command
                   BV_FLD(APBH_CHn_CMD, COMMAND, DMA_READ);  // read data from DMA, write to
NAND
write[5].dma_bar = &nand_cmd_addr_buffer[7];                 // point to byte 7, status
command
write[5].gpmi_compare = NULL;                              // field not used but necessary to set
eccctrl
write[5].gpmi_eccctrl = BV_FLD(GPMI_ECCCTRL, ENABLE_ECC, DISABLE); // disable the ECC block
// 3 words sent to the GPMI
write[5].gpmi_ctrl0 = BV_FLD(GPMI_CTRL0, COMMAND_MODE, WRITE)    | // write to the NAND
                      BV_FLD(GPMI_CTRL0, WORD_LENGTH,  8_BIT)    |
                      BV_FLD(GPMI_CTRL0, LOCK_CS,      ENABLED)  |
                    BF_GPMI_CTRL0_CS               (2)     | // must correspond to NAND CS
used
                      BV_FLD(GPMI_CTRL0, ADDRESS,      NAND_CLE) |
                      BF_GPMI_CTRL0_ADDRESS_INCREMENT  (0)       |
                      BF_GPMI_CTRL0_XFER_COUNT         (1);        // 1 byte command
//--------------------------------------------------------------------------
// Descriptor 7: read status and compare (DATA)
//--------------------------------------------------------------------------
write[6].dma_nxtcmdar = &write[7];                           // point to the next descriptor
write[6].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (0)  |    // no dma transfer
                   BF_APBH_CHn_CMD_CMDWORDS      (2)  |    // send 2 words to the GPMI
                  BF_APBH_CHn_CMD_WAIT4ENDCMD   (1)  |    // wait for command to finish
before
                                                           //      continuing
                   BF_APBH_CHn_CMD_SEMAPHORE     (0)  |
                   BF_APBH_CHn_CMD_NANDWAIT4READY(0)  |
                   BF_APBH_CHn_CMD_NANDLOCK      (1)  |    // maintain resource lock
                   BF_APBH_CHn_CMD_IRQONCMPLT    (0)  |
                   BF_APBH_CHn_CMD_CHAIN         (1)  |    // follow chain to next command
                   BV_FLD(APBH_CHn_CMD, COMMAND, NO_DMA_XFER);  // no dma transfer
write[6].dma_bar = NULL;                                     // field not used
// 2 word sent to the GPMI
write[6].gpmi_ctrl0 = BV_FLD(GPMI_CTRL0, COMMAND_MODE, READ_AND_COMPARE) | // read from the
                                                                  //       NAND and
//                                                               // compare to expect
                      BV_FLD(GPMI_CTRL0, WORD_LENGTH,  8_BIT)     |
                      BV_FLD(GPMI_CTRL0, LOCK_CS,      DISABLED)  |
                    BF_GPMI_CTRL0_CS             (2)       |   // must correspond to NAND CS
used
                      BV_FLD(GPMI_CTRL0, ADDRESS,      NAND_DATA) |
                      BF_GPMI_CTRL0_ADDRESS_INCREMENT  (0)        |
                      BF_GPMI_CTRL0_XFER_COUNT         (1);
write[6].gpmi_compare = <MASK_AND_REFERENCE_VALUE>;  // NOTE: mask and reference values are
NAND
                                                          // SPECIFIC to evaluate the NAND
status
//--------------------------------------------------------------------------
// Descriptor 8: psense compare (time out check)
//--------------------------------------------------------------------------
write[7].dma_nxtcmdar = &write[8];                             // point to the next descriptor
write[7].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (0)         | // no dma transfer
                   BF_APBH_CHn_CMD_CMDWORDS      (0)         | // no words sent to GPMI
                   BF_APBH_CHn_CMD_WAIT4ENDCMD   (0)         | // do not wait to continue
                   BF_APBH_CHn_CMD_SEMAPHORE     (0)         |
                   BF_APBH_CHn_CMD_NANDWAIT4READY(0)         |
                   BF_APBH_CHn_CMD_NANDLOCK      (0)         | // relinquish nand lock
                   BF_APBH_CHn_CMD_IRQONCMPLT    (0)         |
                   BF_APBH_CHn_CMD_CHAIN         (1)         | // follow chain to next command
                   BV_FLD(APBH_CHn_CMD, COMMAND, DMA_SENSE);  // perform a sense check
write[7].dma_bar = dma_error_handler;              // if sense check fails, branch to error
handler
//--------------------------------------------------------------------------
// Descriptor 9: emit GPMI interrupt
//--------------------------------------------------------------------------
write[8].dma_nxtcmdar = NULL;                                // not used since this is
```

```
last
descriptor
write[8].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT     (0)          | // no dma transfer
                   BF_APBH_CHn_CMD_CMDWORDS       (0)          | // no words sent to GPMI
                   BF_APBH_CHn_CMD_WAIT4ENDCMD    (0)          | // do not wait to continue
                   BF_APBH_CHn_CMD_SEMAPHORE      (0)          |
                   BF_APBH_CHn_CMD_NANDWAIT4READY(0)           |
                   BF_APBH_CHn_CMD_NANDLOCK       (0)          |
                   BF_APBH_CHn_CMD_IRQONCMPLT     (1)          | // emit GPMI interrupt
                   BF_APBH_CHn_CMD_CHAIN          (0)          | // terminate DMA chain
processing
                   BV_FLD(APBH_CHn_CMD, COMMAND, NO_DMA_XFER);  // no dma transfer
```

9.5.4.1.2 Using the BCH Encoder

To use the BCH encoder, first turn off the module-wide soft reset bit in both the GPMI and BCH blocks before starting any DMA activity.

Turning off the soft reset must take place by itself, prior to programming the rest of the control registers. Turn off the BCH bus master soft reset bit. Turn off the clock gate bits.

Program the remainder of the GPMI, BCH and APBH DMA as follows:

```
// bring APBH out of reset
APBH_CTRL0_CLR(BM_APBH_CTRL0_SFRST);
APBH_CTRL0_CLR(BM_APBH_CTRL0_CLKGATE);

// bring BCH out of reset
BCH_CTRL_CLR(BM_BCH_CTRL_SFTRST);
BCH_CTRL_CLR(BM_BCH_CTRL_CLKGATE);

// bring gpmi out of reset
GPMI_CTRL0_CLR(BM_GPMI_CTRL0_SFTRST);
GPMI_CTRL0_CLR(BM_GPMI_CTRL0_CLKGATE);
GPMI_CTRL1_SET(BM_GPMI_CTRL1_DEV_RESET | // deassert reset
                   BM_GPMI_CTRL1_BCH_MODE  );    // enable BCH mode

// enable pinctrl
PINCTRL_CTRL_WR(0x00000000);

// enable gpmi pins
PINCTRL_MUXSEL0_CLR(0x0000ffff);    // data bits
PINCTRL_MUXSEL1_CLR(0x03ffffff);    // control bits
PINCTRL_MUXSEL8_CLR(0x0003f3ff);    // control bits
PINCTRL_MUXSEL8_SET(0x00015155);    // control bits
```

Note that for writing NANDs (ECC encoding), only GPMI DMA command complete interrupts are used. The BCH engine is used for writing to the NAND but may optionally produces an interrupt. From the sample code in DMA Structure Code Example :

- DMA descriptor 1 prepares the NAND for data write by using the GPMI to issue a write setup command byte under CLE, then sends a 5-byte address under ALE. The BCH engine is disabled and not used for these commands.
- DMA descriptor 2 enables the BCH engine for encoding to begin the initial writing of the NAND data by specifying where the data and auxiliary payload are coming from in system memory.

- DMA descriptor 3 issues the write commit command byte under CLE to the NAND.
- DMA descriptor 4 waits for the NAND to complete the write commit/transfer by watching the NAND's ready line status. This descriptor relinquishes the NANDLOCK on the GPMI to enable the other DMA channels to initiate NAND transactions on different NAND CS lines.
- DMA descriptor 6 issues a NAND status command byte under "CLE" to check the status of the NAND device following the page write.
- DMA descriptor 7 reads back the NAND status and compares the status with an expected value. If there are differences, then the DMA processing engine follows an error-handling DMA descriptor path.
- DMA descriptor 8 disables the BCH engine and emits a GPMI interrupt to indicate that the NAND write has been completed.

9.5.4.2 BCH Decoding for NAND Reads

When a page is read from NAND flash, BCH syndromes will be computed and, if correctable errors are found, they will be corrected on a per block basis within the NAND page. This decoding process is fully overlapped with other NAND data reads and with CPU execution. The BCH decoder flowchart in the figure below shows the steps involved in programming the decoder. The hardware flow of reading and decoding a 4096-byte page is shown in Figure 9-19.

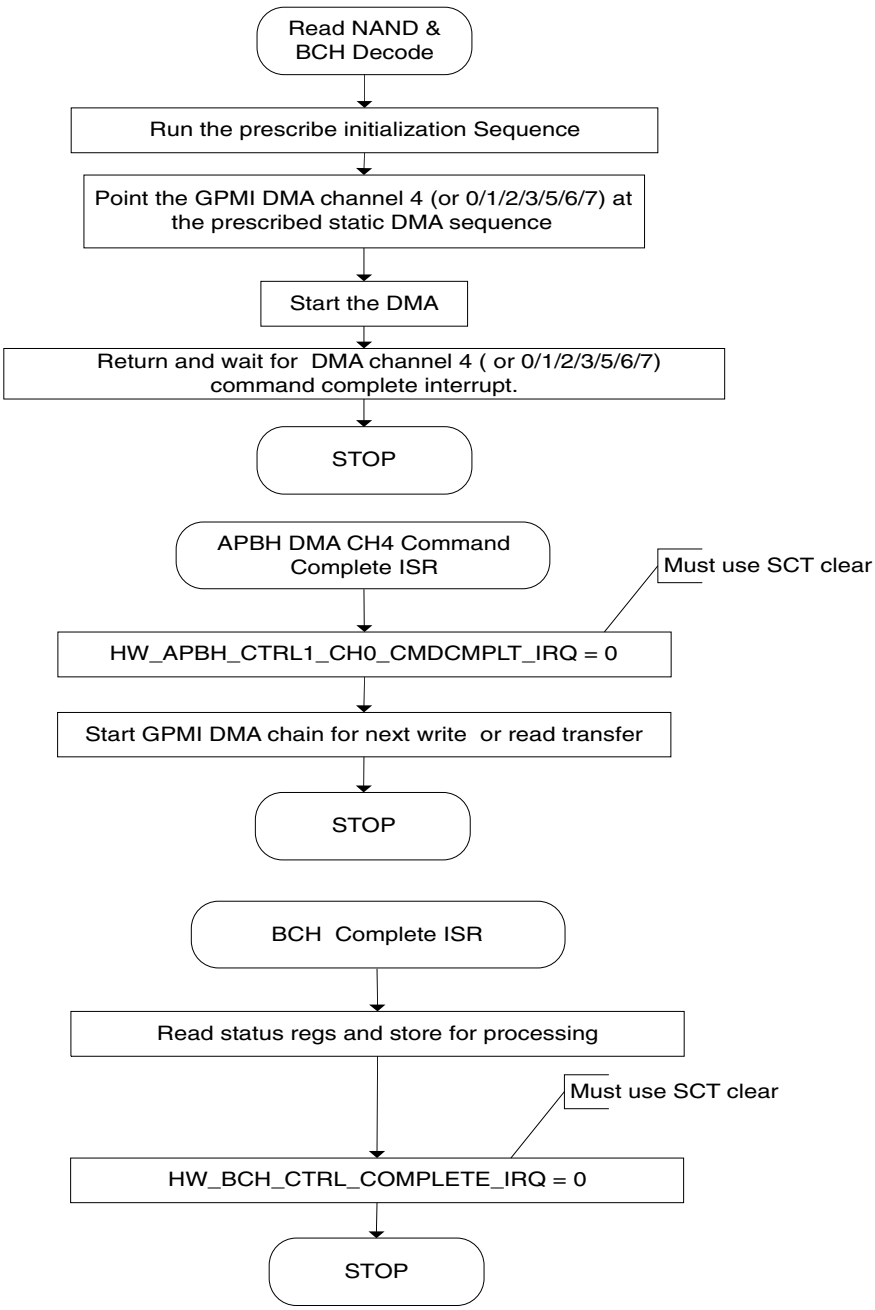


Figure 9-18. BCH Decode Flowchart

Conceptually, an APHB DMA Channel (0,1,2 or 3) command chain with seven command structures linked together is used to perform the BCH decode operation (as shown in Figure 9-19).

Note

The GPMI's DMA command structures controls the BCH decode operation.

To use the BCH decoder with the GPMI's DMA, create a DMA command chain containing seven descriptor structures, as shown in the figure below and detailed in the DMA structure code example that follows it in DMA Structure Code Example. The seven DMA descriptors perform the following tasks:

1. Issue NAND read setup command byte (under "CLE") and address bytes (under "ALE").
2. Issue NAND read execute command byte (under "CLE").
3. Wait for the NAND device to complete accessing the block data by watching the ready signal.
4. Check for NAND timeout through "PSENSE".
5. Configure and enable the BCH block and read the NAND block data.
6. Disable the BCH block.
7. Descriptor NOP to allow NANDLOCK in the previous descriptor to the thread-safe.

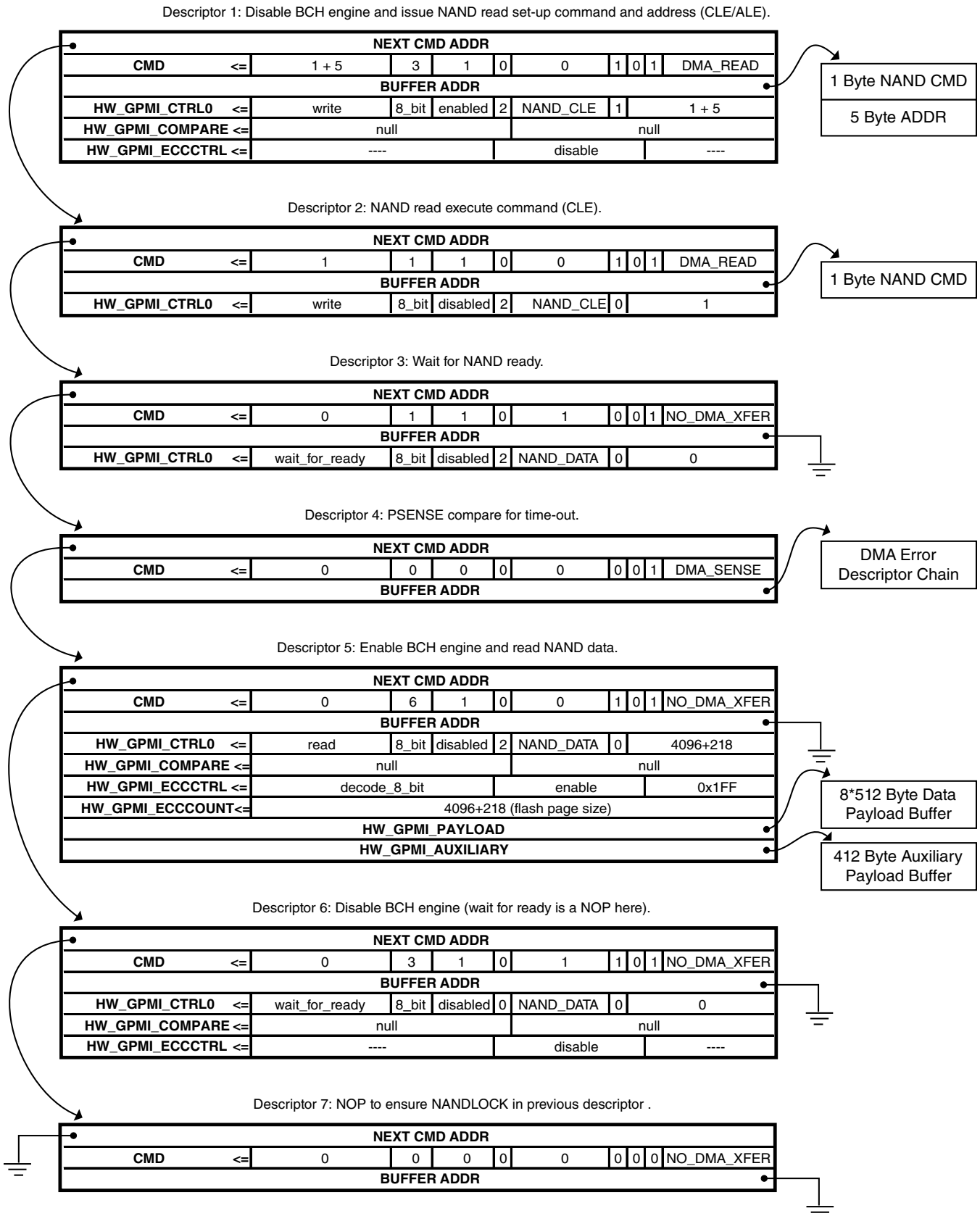


Figure 9-19. BCH Decode DMA Descriptor Chain

9.5.4.2.1 DMA Structure Code Example

The following sample code illustrates the coding for one read transaction, consisting of a seven DMA command structure chain for reading all 4096 bytes of payload data (eight 512-byte blocks) and 65 bytes of metadata with the associative parity bytes (8 * (18) + 9) from a 4K NAND page sitting on GPMI CS2.

```
//-------------------------------------------------------------------------
// generic DMA/GPMI/ECC descriptor struct, order sensitive!
//-------------------------------------------------------------------------
typedef struct {
  // DMA related fields
  unsigned int dma_nxtcmdar;
  unsigned int dma_cmd;
  unsigned int dma_bar;
  // GPMI related fields
  unsigned int gpmi_ctrl0;
  unsigned int gpmi_compare;
  unsigned int gpmi_eccctrl;
  unsigned int gpmi_ecccount;
  unsigned int gpmi_data_ptr;
  unsigned int gpmi_aux_ptr;
} GENERIC_DESCRIPTOR;
//-------------------------------------------------------------------------
// allocate 7 descriptors for doing a NAND ECC Read
//-------------------------------------------------------------------------
GENERIC_DESCRIPTOR read[7];
//-------------------------------------------------------------------------
// DMA descriptor pointer to handle error conditions from psense checks
//-------------------------------------------------------------------------
unsigned int * dma_error_handler;
//-------------------------------------------------------------------------
// 7 byte NAND command and address buffer
// any alignment is ok, it is read by the GPMI DMA
//   byte 0 is read setup command
//   bytes 1-5 is the NAND address
//   byte 6 is read execute command
//-------------------------------------------------------------------------
unsigned char nand_cmd_addr_buffer[7];
//-------------------------------------------------------------------------
// 4096 byte payload buffer used for reads or writes
// needs to be word aligned
//-------------------------------------------------------------------------
unsigned int read_payload_buffer[(4096/4)];
//-------------------------------------------------------------------------
// 412 byte auxiliary buffer used for reads
// needs to be word aligned
//-------------------------------------------------------------------------
unsigned int read_aux_buffer[(412/4)];
//-------------------------------------------------------------------------
// Descriptor 1: issue NAND read setup command (CLE/ALE)
//-------------------------------------------------------------------------
read[0].dma_nxtcmdar = &read[1];                              // point to the next descriptor
read[0].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (1 + 5) |  // 1 byte command, 5 byte address
                  BF_APBH_CHn_CMD_CMDWORDS      (3)     |  // send 3 words to the GPMI
                  BF_APBH_CHn_CMD_WAIT4ENDCMD   (1)     |  // wait for command to finish
                                                           //     before continuing
                  BF_APBH_CHn_CMD_SEMAPHORE     (0)     |
                  BF_APBH_CHn_CMD_NANDWAIT4READY(0)     |
                  BF_APBH_CHn_CMD_NANDLOCK      (1)     |  // prevent other DMA channels from
                                                           //    taking over
                  BF_APBH_CHn_CMD_IRQONCMPLT    (0)     |
                  BF_APBH_CHn_CMD_CHAIN         (1)     |   // follow chain to next command
                  BV_FLD(APBH_CHn_CMD, COMMAND, DMA_READ);  // read data from DMA, write to
```

```
NAND
read[0].dma_bar = &nand_cmd_addr_buffer;            // byte 0 read setup, bytes 1 - 5 NAND
address
// 3 words sent to the GPMI
read[0].gpmi_ctrl0 = BV_FLD(GPMI_CTRL0, COMMAND_MODE, WRITE)     | // write to the NAND
                     BV_FLD(GPMI_CTRL0, WORD_LENGTH,  8_BIT)     |
                     BV_FLD(GPMI_CTRL0, LOCK_CS,      ENABLED)   |
                     BF_GPMI_CTRL0_CS                 (2)        | // must correspond to NAND
CS used
                     BV_FLD(GPMI_CTRL0, ADDRESS,      NAND_CLE)  |
                     BF_GPMI_CTRL0_ADDRESS_INCREMENT  (1)        | // send command and address
                     BF_GPMI_CTRL0_XFER_COUNT         (1 + 5);    // 1 byte command, 5 byte
address
read[0].gpmi_compare = NULL;                                     // field not used but necessary to set
eccctrl
read[0].gpmi_eccctrl = BV_FLD(GPMI_ECCCTRL, ENABLE_ECC, DISABLE); // disable the ECC block
//-------------------------------------------------------------------------
// Descriptor 2: issue NAND read execute command (CLE)
//-------------------------------------------------------------------------
read[1].dma_nxtcmdar = &read[2];                                 // point to the next descriptor
read[1].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (1)    |   // 1 byte read command
                  BF_APBH_CHn_CMD_CMDWORDS      (1)    |   // send 1 word to GPMI
                  BF_APBH_CHn_CMD_WAIT4ENDCMD   (1)    |   // wait for command to finish
before
                                                         //    continuing
                  BF_APBH_CHn_CMD_SEMAPHORE     (0)    |
                  BF_APBH_CHn_CMD_NANDWAIT4READY(0)    |
                  BF_APBH_CHn_CMD_NANDLOCK      (1)    |   // prevent other DMA channels from
                                                         //     taking over
                  BF_APBH_CHn_CMD_IRQONCMPLT    (0)    |
                  BF_APBH_CHn_CMD_CHAIN         (1)    |   // follow chain to next command
                  BV_FLD(APBH_CHn_CMD, COMMAND, DMA_READ);     // read data from DMA, write
to NAND
read[1].dma_bar = &nand_cmd_addr_buffer[6];              // point to byte 6, read execute
command
// 1 word sent to the GPMI
read[1].gpmi_ctrl0 = BV_FLD(GPMI_CTRL0, COMMAND_MODE, WRITE)     | // write to the NAND
                     BV_FLD(GPMI_CTRL0, WORD_LENGTH,  8_BIT)     |
                     BV_FLD(GPMI_CTRL0, LOCK_CS,      DISABLED)  |
                     BF_GPMI_CTRL0_CS                 (2)        | // must correspond to NAND
CS used
                     BV_FLD(GPMI_CTRL0, ADDRESS,      NAND_CLE)  |
                     BF_GPMI_CTRL0_ADDRESS_INCREMENT  (0)        |
                     BF_GPMI_CTRL0_XFER_COUNT         (1);        // 1 byte command
//-------------------------------------------------------------------------
// Descriptor 3: wait for ready (DATA)
//-------------------------------------------------------------------------
read[2].dma_nxtcmdar = &read[3];                                 // point to the next descriptor
read[2].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (0)      | // no dma transfer
                  BF_APBH_CHn_CMD_CMDWORDS      (1)      | // send 1 word to GPMI
                  BF_APBH_CHn_CMD_WAIT4ENDCMD   (1)      | // wait for command to finish
before
                                                         //     continuing
                  BF_APBH_CHn_CMD_SEMAPHORE     (0)      |
                  BF_APBH_CHn_CMD_NANDWAIT4READY(1)      | // wait for nand to be ready
                  BF_APBH_CHn_CMD_NANDLOCK      (0)      | // relinquish nand lock
                  BF_APBH_CHn_CMD_IRQONCMPLT    (0)      |
                  BF_APBH_CHn_CMD_CHAIN         (1)      | // follow chain to next command
                  BV_FLD(APBH_CHn_CMD, COMMAND, NO_DMA_XFER);   // no dma transfer
read[2].dma_bar = NULL;                                          // field not used
// 1 word sent to the GPMI
read[2].gpmi_ctrl0 = BV_FLD(GPMI_CTRL0, COMMAND_MODE, WAIT_FOR_READY) | // wait for NAND
ready
                     BV_FLD(GPMI_CTRL0, WORD_LENGTH,  8_BIT)          |
                     BV_FLD(GPMI_CTRL0, LOCK_CS,      DISABLED)       |
                     BF_GPMI_CTRL0_CS                 (2)             | // must correspond
to NAND CS used
                     BV_FLD(GPMI_CTRL0, ADDRESS,      NAND_DATA)      |
                     BF_GPMI_CTRL0_ADDRESS_INCREMENT  (0)             |
                     BF_GPMI_CTRL0_XFER_COUNT         (0);
```

```
//------------------------------------------------------------------------
// Descriptor 4: psense compare (time out check)
//------------------------------------------------------------------------
read[3].dma_nxtcmdar = &read[4];                                    // point to the next
descriptor
read[3].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (0)         |       // no dma transfer
                  BF_APBH_CHn_CMD_CMDWORDS      (0)         |       // no words sent to GPMI
                  BF_APBH_CHn_CMD_WAIT4ENDCMD   (0)         |       // do not wait to continue
                  BF_APBH_CHn_CMD_SEMAPHORE     (0)         |
                  BF_APBH_CHn_CMD_NANDWAIT4READY(0)         |
                  BF_APBH_CHn_CMD_NANDLOCK      (0)         |
                  BF_APBH_CHn_CMD_IRQONCMPLT    (0)         |
                  BF_APBH_CHn_CMD_CHAIN         (1)         |       // follow chain to next
command
                  BV_FLD(APBH_CHn_CMD, COMMAND, DMA_SENSE);         // perform a sense check
read[3].dma_bar = dma_error_handler;                        // if sense check fails, branch to
error handler
//------------------------------------------------------------------------
// Descriptor 5: read 4K page plus 65 byte meta-data Nand data
//               and send it to ECC block (DATA)
//------------------------------------------------------------------------
read[4].dma_nxtcmdar = &read[5];                              // point to the next descriptor
read[4].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (0)     | // no dma transfer
                  BF_APBH_CHn_CMD_CMDWORDS      (6)     | // send 6 words to GPMI
                  BF_APBH_CHn_CMD_WAIT4ENDCMD   (1)     | // wait for command to finish before
                                                          //       continuing
                  BF_APBH_CHn_CMD_SEMAPHORE     (0)     |
                  BF_APBH_CHn_CMD_NANDWAIT4READY(0)     |
                  BF_APBH_CHn_CMD_NANDLOCK      (1)     | // prevent other DMA channels from
taking over
                  BF_APBH_CHn_CMD_IRQONCMPLT    (0)     | // ECC block generates BCH interrupt
                                                          //      on completion
                  BF_APBH_CHn_CMD_CHAIN         (1)     | // follow chain to next command
                  BV_FLD(APBH_CHn_CMD, COMMAND, NO_DMA_XFER);     // no DMA transfer,
                                                                  // ECC block handles
transfer
read[4].dma_bar = NULL;                                           // field not used
// 6 words sent to the GPMI
read[4].gpmi_ctrl0 = BV_FLD(GPMI_CTRL0, COMMAND_MODE, READ)       |  // read from the NAND
                     BV_FLD(GPMI_CTRL0, WORD_LENGTH,  8_BIT)      |
                     BV_FLD(GPMI_CTRL0, LOCK_CS,      DISABLED)   |
                     BF_GPMI_CTRL0_CS                 (2)         |  // must correspond to
NAND CS used
                     BV_FLD(GPMI_CTRL0, ADDRESS,      NAND_DATA)  |
                     BF_GPMI_CTRL0_ADDRESS_INCREMENT  (0)         |
                     BF_GPMI_CTRL0_XFER_COUNT         (4096+218);    // eight 512 byte data
blocks
                                                                     // metadata, and parity

read[4].gpmi_compare = NULL;                              // field not used but necessary to set
eccctrl
// GPMI ECCCTRL PIO This launches the 4K byte transfer through BCH's
// bus master. Setting the ECC_ENABLE bit redirects the data flow
// within the GPMI so that read data flows to the BCH engine instead
// of flowing to the GPMI's DMA channel.
read[4].gpmi_eccctrl = BV_FLD(GPMI_ECCCTRL, ECC_CMD,    DECODE_8_BIT) |    // specify t = 8
mode
                       BV_FLD(GPMI_ECCCTRL, ENABLE_ECC, ENABLE)       |    // enable ECC
module
                       BF_GPMI_ECCCTRL_BUFFER_MASK     (0X1FF);  // read all 8 data blocks
and 1 aux block
read[4].gpmi_ecccount = BF_GPMI_ECCCOUNT_COUNT(4096+218);         // specify number of bytes
                                                                  //        read from NAND
read[4].gpmi_data_ptr = &read_payload_buffer;                     // pointer for the 4K byte
                                                                  //    data area
read[4].gpmi_aux_ptr = &read_aux_buffer;                          // pointer for the 65 byte
aux area +
                                                                  // parity and syndrome
bytes for both
                                                                 // data and aux blocks.
```

```
//------------------------------------------------------------------------
// Descriptor 6: disable ECC block
//------------------------------------------------------------------------
read[5].dma_nxtcmdar = &read[6];                                  // point to the next descriptor
read[5].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (0)       |     // no dma transfer
                  BF_APBH_CHn_CMD_CMDWORDS      (3)       |     // send 3 words to GPMI
                  BF_APBH_CHn_CMD_WAIT4ENDCMD   (1)       |     // wait for command to finish
before
                                                                  //     continuing
                  BF_APBH_CHn_CMD_SEMAPHORE     (0)       |
                  BF_APBH_CHn_CMD_NANDWAIT4READY(1)       |     // wait for nand to be ready
                  BF_APBH_CHn_CMD_NANDLOCK      (1)       |     // need nand lock to be
                                                                  // thread safe while turn-off BCH
                  BF_APBH_CHn_CMD_IRQONCMPLT    (0)       |
                  BF_APBH_CHn_CMD_CHAIN         (1)       |     // follow chain to next command
                  BV_FLD(APBH_CHn_CMD, COMMAND, NO_DMA_XFER);     // no dma transfer
read[5].dma_bar = NULL;                                           // field not used
// 3 words sent to the GPMI
read[5].gpmi_ctrl0 = BV_FLD(GPMI_CTRL0, COMMAND_MODE, READ)       |
                     BV_FLD(GPMI_CTRL0, WORD_LENGTH,  8_BIT)      |
                     BV_FLD(GPMI_CTRL0, LOCK_CS,      DISABLED)   |
                     BF_GPMI_CTRL0_CS                 (2)        | // must correspond to
NAND CS used
                     BV_FLD(GPMI_CTRL0, ADDRESS,      NAND_DATA) |
                     BF_GPMI_CTRL0_ADDRESS_INCREMENT  (0)        |
                     BF_GPMI_CTRL0_XFER_COUNT         (0);
read[5].gpmi_compare = NULL;                                      // field not used but necessary to set
eccctrl
read[5].gpmi_eccctrl = BV_FLD(GPMI_ECCCTRL, ENABLE_ECC, DISABLE);  // disable the ECC block
//------------------------------------------------------------------------
// Descriptor 7: deassert nand lock
//------------------------------------------------------------------------
read[6].dma_nxtcmdar = NULL;                                      // not used since this is last
descriptor
read[6].dma_cmd = BF_APBH_CHn_CMD_XFER_COUNT    (0)   |     // no dma transfer
                  BF_APBH_CHn_CMD_CMDWORDS      (0)   |     // no words sent to GPMI
                  BF_APBH_CHn_CMD_WAIT4ENDCMD   (0)   |     // wait for command to finish
before
                                                            //        continuing
                  BF_APBH_CHn_CMD_SEMAPHORE     (0)   |
                  BF_APBH_CHn_CMD_NANDWAIT4READY(0)   |
                  BF_APBH_CHn_CMD_NANDLOCK      (0)   |     // relinquish nand lock
                  BF_APBH_CHn_CMD_IRQONCMPLT    (0)   |     // BCH engine generates interrupt
                  BF_APBH_CHn_CMD_CHAIN         (0)   |     // terminate DMA chain processing
                  BV_FLD(APBH_CHn_CMD, COMMAND, NO_DMA_XFER);     // no dma transfer
read[6].dma_bar = NULL;                                           // field not used
```

9.5.4.2.2 Using the Decoder

As illustrated in Figure 9-19 and the sample code in DMA Structure Code Example :

- DMA descriptor 1 prepares the NAND for data read by using the GPMI to issue a NAND read setup command byte under CLE, then sends a 5-byte address under ALE. The BCH engine is not used for these commands.
- DMA descriptor 2 issues a one-byte read execute command to the NAND device that triggers its read access. The NAND then goes not ready.
- DMA descriptor 3 performs a wait for ready operation allowing the DMA chain to remain dormant until the NAND device completes its read access time.

- DMA descriptor 5 handles the reading and error correction of the NAND data. This command's PIOs activate the BCH engine to write the read NAND data to system memory and to process it for any errors that need to be corrected. This DMA descriptor contains two PIO values that are system memory addresses pointing to the PAYLOAD data area and to the AUXILIARY data area. These addresses are used by the BCH engine's AHB master to move data into system memory and to correct it. While this example is reading an entire 4K page—payload plus metadata—it is equally possible to read just one 512-byte payload block or just the uniquely protected metadata block in a single 7 DMA structure transfer.
- DMA descriptor 6 disables the BCH engine with the NANDLOCK asserted. This is necessary to ensure that the GPMI resource is not arbitrated to another DMA channel when multiple DMA channels are active concurrently.
- DMA descriptor 7 de-asserts the NANDLOCK to free up the GPMI resource to another channel.

As the BCH block receives data from the GPMI:

- The decoder transforms the read NAND data block into a BCH code word and computes the codeword syndrome.
- If no errors are present, then the BCH block can immediately report back to firmware. This report is passed as the BCH_CTRL_COMPLETE_IRQ interrupt status bit and the associated status registers in BCH_STATUS0/1 registers.
- If an error is present, then the BCH block corrects the necessary data block or parity block bytes, if possible (not all errors are correctable).

As the BCH decoder reads the data and parity blocks, it records a special condition, i.e., that all of the bits of a payload data block or metadata block are one, including any associated parity bytes. The all-ones case for both parity and data indicates an erased block in the NAND device.

The BCH_STATUS0 register contains a 4-bit field that indicates the final status of the auxiliary block. A value of 0x0 indicates no errors found for a block.

- A value of 1 to 20 inclusive indicates that many correctable errors were found and fixed.
- A value of 0xFE indicates uncorrectable errors detected on the block.

- A value of 0xFF indicates that the block was in the special ALL ONES state and is therefore considered to be an ERASED block.
- All other values are disallowed by the hardware design.

Recall that up to eight NAND devices can have DMA chains in-flight at once, i.e. they can all be contending for access to the GPMI data bus. It is impossible to predict which NAND device will enter the BCH engine with a transfer first, because each chain includes a wait4ready command structure. As a result, firmware should look at the BCH_STATUS0_COMPLETED_CE bit field to determine which block is being reported in the status register. There is also a 16-bit HANDLE field in the GPMI_ECCCTRL register that is passed down the pipeline with each transaction. This handle field can be used to speed firmware's detection of which transaction is being reported.

These examples of reading and writing have focused on full page transfers of 4K page NAND devices. Other device configurations can be specified by changing the ECCOUNT field in the GPMI registers and reprogramming the BCH's FLASHnLAYOUTm registers.

The BCH and GPMI blocks are designed to be very efficient at reading single 512 (or 1024)-byte pages in one transaction. With no errors, the transaction takes less than 20 HCLKs longer than the time to read the raw data from the NAND.

To summarize, the APBH DMA command chain for a BCH decode operation is shown in Figure 9-19. Seven DMA command structures must be present for each NAND read transaction decoded by the BCH. The seven DMA command structures for multiple NAND read transaction blocks can be chained together to make larger units of work for the BCH, and each will produce an appropriate error report in the BCH PIO space. Multiple NAND devices can have such multiple chains scheduled. The results can come back out of order with respect to the multiple chains.

9.5.4.3 Interrupts

There are two interrupt sources used in processing BCH protected NAND read and write transfers.

Since all BCH operations are initiated by GPMI DMA command structures, the DMA completion interrupt for the GPMI is an important ISR. Both of the flow charts of Figure 9-15 and Figure 9-18 show the GPMI DMA complete ISR skeleton. In both reads and writes, the GPMI DMA completion interrupt is used to schedule work ***INTO*** the error correction pipeline. As the front end processing completes, the DMA interrupt is

generated and additional work, such as DMA chains, are passed to the GPMI DMA to keep it *fed*. For write operations, this is the only interrupt that is generated for processing the NAND write transfer.

For reads, however, two interrupts are needed. Every read is started by a GPMI DMA command chain and the front end queue is fed as described above. The back end of the read pipeline is drained by monitoring the BCH completion interrupt found int HW_BCH_CTRL_COMPLETE_IRQ.

An BCH transaction consists of reading or writing all of the blocks requested in the HW_GPMI_ECCCTRL_BUFFER_MASK bit field. As every read transaction completes, it posts the status of all of the blocks to the HW_BCH_STATUS0 and HW_BCH_STATUS1 registers and sets the completion interrupt. The five stages of the BCH read pipeline completes, one in the GPMI and four in the BCH, are independently stalled as they complete and try to deliver to the next stage in the data flow. Several of these stages can be skipped if no-errors are found or once an uncorrectable error is found in a block.

In any case, the final stage will stall if the status register is busy waiting for the CPU to take status register results. The hardware monitors the state of the HW_BCH_CTRL_COMPLETE_IRQ bit. If it is still set when the last pipeline stage is ready to post data, then the stage will stall. It follows that the next previous stage will stall when it is ready to hand off work to the final stage, and so on up the pipeline.

CAUTION

It is important that firmware read the STATUS0/1 results and save them before clearing the interrupt request bit. Otherwise, a transaction and its results could be completely lost.

9.5.4.4 Randomizer

BCH ECC has a Randomizer module that is interfaced through the GPMI APBHDMA chain. The Randomizer can generate random data based on BCH ECC encoded/decoded data. It can be employed to reduce the disturbances caused by a neighboring cell in the NAND chip, thus reducing bit errors.

To enable the Randomizer module, set GPMI_ECCCTRL[RANDOMIZER_ENABLE] to 1, then set GPMI_ECCCOUNT[RANDOMIZER_PAGE] to select randomizer page number needed to be randomized. All these registers can be programmed by the DMA chain. The randomized data should start from the zero column address and be the size of the whole NAND page. If the randomizer function is enabled,

GPMI_ECCCTRL[ENABLE_ECC] should also be enabled. To bypass BCH error correction function, set BCH_FLASHxLAYOUT0[ECC0] and BCH_FLASHxLAYOUT1[ECCN] to 0.

9.5.5 Behavior During Reset

A soft reset (SFTRST) can take multiple clock periods to complete, so do NOT set CLKGATE when setting SFTRST.

The reset process gates the clocks automatically. The example code is shown below.

```
// A soft reset can take multiple clocks to complete, so do NOT gate the
// clock when setting soft reset. The reset process will gate the clock
// automatically. Poll until this has happened before subsequently
// preparing soft-reset and clock gate
  BCH_CTRL_CLR(BM_BCH_CTRL_SFTRST);
  BCH_CTRL_CLR(BM_BCH_CTRL_CLKGATE);
// asserting soft-reset
  BCH_CTRL_SET(BM_BCH_CTRL_SFTRST);
// waiting for confirmation of soft-reset
  while (!BCH_CTRL.B.CLKGATE)
  {
// busy wait
  }
// Done.
  BCH_CTRL_CLR(BM_BCH_CTRL_SFTRST);
  BCH_CTRL_CLR(BM_BCH_CTRL_CLKGATE);
```

9.5.6 BCH Memory Map/Register Definition

BCH Hardware Register Format Summary

BCH memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3300_4000	Hardware BCH ECC Accelerator Control Register (BCH_CTRL)	32	R/W	C000_0000h	9.5.6.1/ 2261
3300_4004	Hardware BCH ECC Accelerator Control Register (BCH_CTRL_SET)	32	R/W	C000_0000h	9.5.6.1/ 2261
3300_4008	Hardware BCH ECC Accelerator Control Register (BCH_CTRL_CLR)	32	R/W	C000_0000h	9.5.6.1/ 2261
3300_400C	Hardware BCH ECC Accelerator Control Register (BCH_CTRL_TOG)	32	R/W	C000_0000h	9.5.6.1/ 2261
3300_4010	Hardware ECC Accelerator Status Register 0 (BCH_STATUS0)	32	R	0000_0010h	9.5.6.2/ 2263

Table continues on the next page...

BCH memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3300_4020	Hardware ECC Accelerator Mode Register (BCH_MODE)	32	R/W	0000_0000h	9.5.6.3/ 2265
3300_4030	Hardware BCH ECC Loopback Encode Buffer Register (BCH_ENCODEPTR)	32	R/W	0000_0000h	9.5.6.4/ 2266
3300_4040	Hardware BCH ECC Loopback Data Buffer Register (BCH_DATAPTR)	32	R/W	0000_0000h	9.5.6.5/ 2266
3300_4050	Hardware BCH ECC Loopback Metadata Buffer Register (BCH_METAPTR)	32	R/W	0000_0000h	9.5.6.6/ 2267
3300_4070	Hardware ECC Accelerator Layout Select Register (BCH_LAYOUTSELECT)	32	R/W	E4E4_E4E4h	9.5.6.7/ 2267
3300_4080	Hardware BCH ECC Flash 0 Layout 0 Register (BCH_FLASH0LAYOUT0)	32	R/W	070A_4080h	9.5.6.8/ 2268
3300_4090	Hardware BCH ECC Flash 0 Layout 1 Register (BCH_FLASH0LAYOUT1)	32	R/W	10DA_4080h	9.5.6.9/ 2270
3300_40A0	Hardware BCH ECC Flash 1 Layout 0 Register (BCH_FLASH1LAYOUT0)	32	R/W	070A_4080h	9.5.6.10/ 2271
3300_40B0	Hardware BCH ECC Flash 1 Layout 1 Register (BCH_FLASH1LAYOUT1)	32	R/W	10DA_4080h	9.5.6.11/ 2273
3300_40C0	Hardware BCH ECC Flash 2 Layout 0 Register (BCH_FLASH2LAYOUT0)	32	R/W	070A_4080h	9.5.6.12/ 2274
3300_40D0	Hardware BCH ECC Flash 2 Layout 1 Register (BCH_FLASH2LAYOUT1)	32	R/W	10DA_4080h	9.5.6.13/ 2275
3300_40E0	Hardware BCH ECC Flash 3 Layout 0 Register (BCH_FLASH3LAYOUT0)	32	R/W	070A_4080h	9.5.6.14/ 2276
3300_40F0	Hardware BCH ECC Flash 3 Layout 1 Register (BCH_FLASH3LAYOUT1)	32	R/W	10DA_4080h	9.5.6.15/ 2278
3300_4100	Hardware BCH ECC Debug Register0 (BCH_DEBUG0)	32	R/W	0000_0000h	9.5.6.16/ 2279
3300_4104	Hardware BCH ECC Debug Register0 (BCH_DEBUG0_SET)	32	R/W	0000_0000h	9.5.6.16/ 2279
3300_4108	Hardware BCH ECC Debug Register0 (BCH_DEBUG0_CLR)	32	R/W	0000_0000h	9.5.6.16/ 2279
3300_410C	Hardware BCH ECC Debug Register0 (BCH_DEBUG0_TOG)	32	R/W	0000_0000h	9.5.6.16/ 2279
3300_4110	KES Debug Read Register (BCH_DBGKESREAD)	32	R	0000_0000h	9.5.6.17/ 2281
3300_4120	Chien Search Debug Read Register (BCH_DBGCSFEREAD)	32	R	0000_0000h	9.5.6.18/ 2281
3300_4130	Syndrome Generator Debug Read Register (BCH_DBGSYNDGENREAD)	32	R	0000_0000h	9.5.6.19/ 2281
3300_4140	Bus Master and ECC Controller Debug Read Register (BCH_DBGAHBMREAD)	32	R	0000_0000h	9.5.6.20/ 2282
3300_4150	Block Name Register (BCH_BLOCKNAME)	32	R	2048_4342h	9.5.6.21/ 2282

Table continues on the next page...

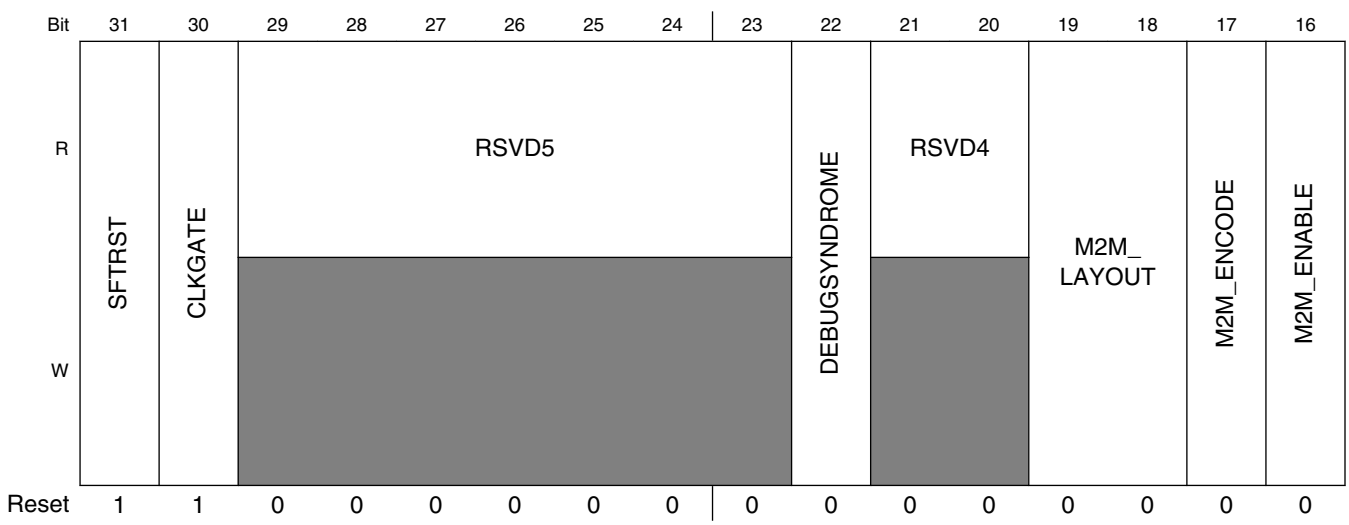
BCH memory map (continued)

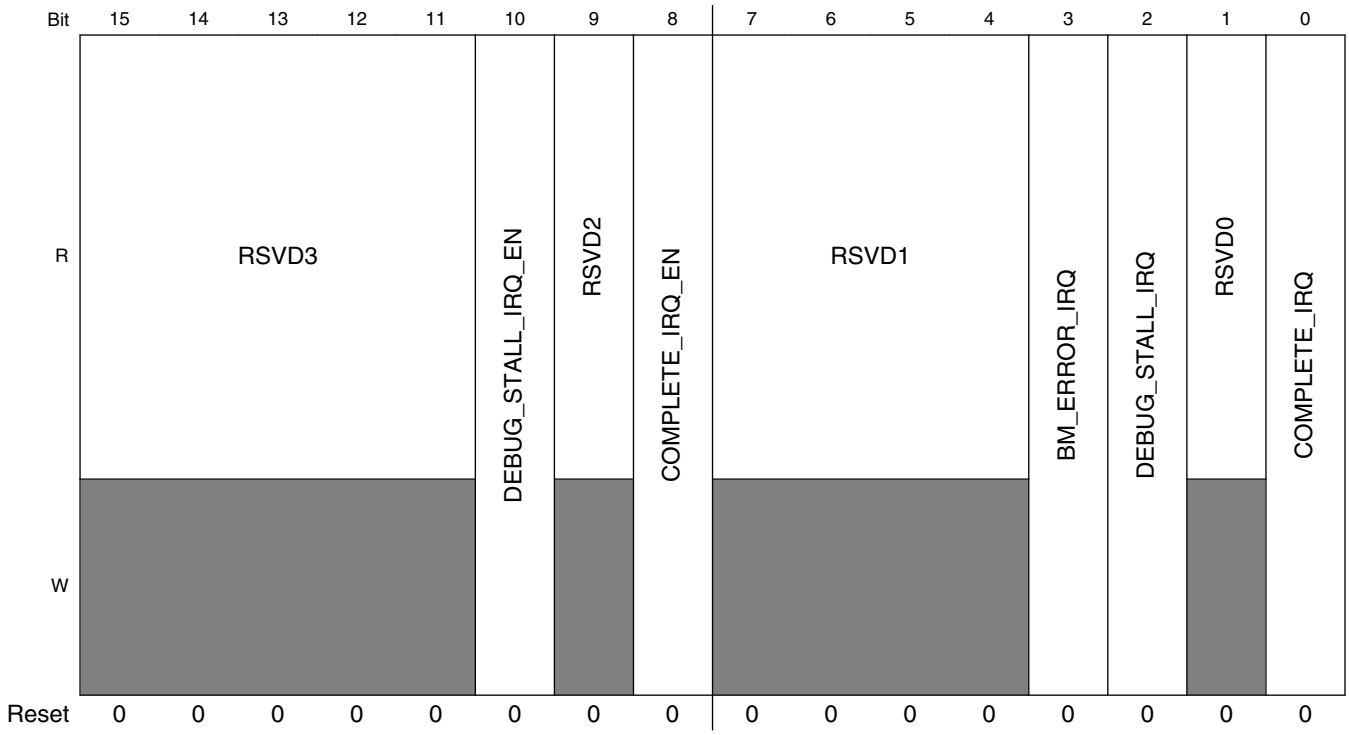
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3300_4160	BCH Version Register (BCH_VERSION)	32	R	0100_0000h	9.5.6.22/ 2283
3300_4170	Hardware BCH ECC Debug Register 1 (BCH_DEBUG1)	32	R/W	0000_0000h	9.5.6.23/ 2284

9.5.6.1 Hardware BCH ECC Accelerator Control Register (BCH_CTRL*n*)

The BCH CTRL provides overall control of the hardware ECC accelerator

Address: 3300_4000h base + 0h offset + (4d × i), where i=0d to 3d





BCH_CTRL*n* field descriptions

Field	Description
31 SFTRST	Set this bit to 0 to enable normal BCH operation. Set this bit to 1 (default) to disable clocking with the BCH and hold it in its reset (lowest power) state. This bit can be turned on and then off to reset the BCH block to its default state. This bit resets all state machines except for the AHB master state machine 0x0 **RUN** — Allow BCH to operate normally. 0x1 **RESET** — Hold BCH in reset.
30 CLKGATE	This bit must be set to 0 for normal operation. When set to 1 it gates off the clocks to the block. 0x0 **RUN** — Allow BCH to operate normally. 0x1 **NO_CLKS** — Do not clock BCH gates in order to minimize power consumption.
29–23 RSVD5	This field is reserved. This read-only field is reserved and always has the value 0.
22 DEBUGSYNDROME	(For debug purposes only). Enable write of computed syndromes to memory on BCH decode operations. Computed syndromes will be written to the auxiliary buffer after the status block. Syndromes will be written as padded 16-bit values.
21–20 RSVD4	This field is reserved. This read-only field is reserved and always has the value 0
19–18 M2M_LAYOUT	Selects the flash page format for memory-to-memory operations.
17 M2M_ENCODE	Selects encode (parity generation) or decode (correction) mode for memory-to-memory operations.
16 M2M_ENABLE	NOTE! WRITING THIS BIT INITIATES A MEMORY-TO-MEMORY OPERATION. The BCH module must be inactive (not processing data from the GPMI) when this bit is set. The M2M_ENCODE and

Table continues on the next page...

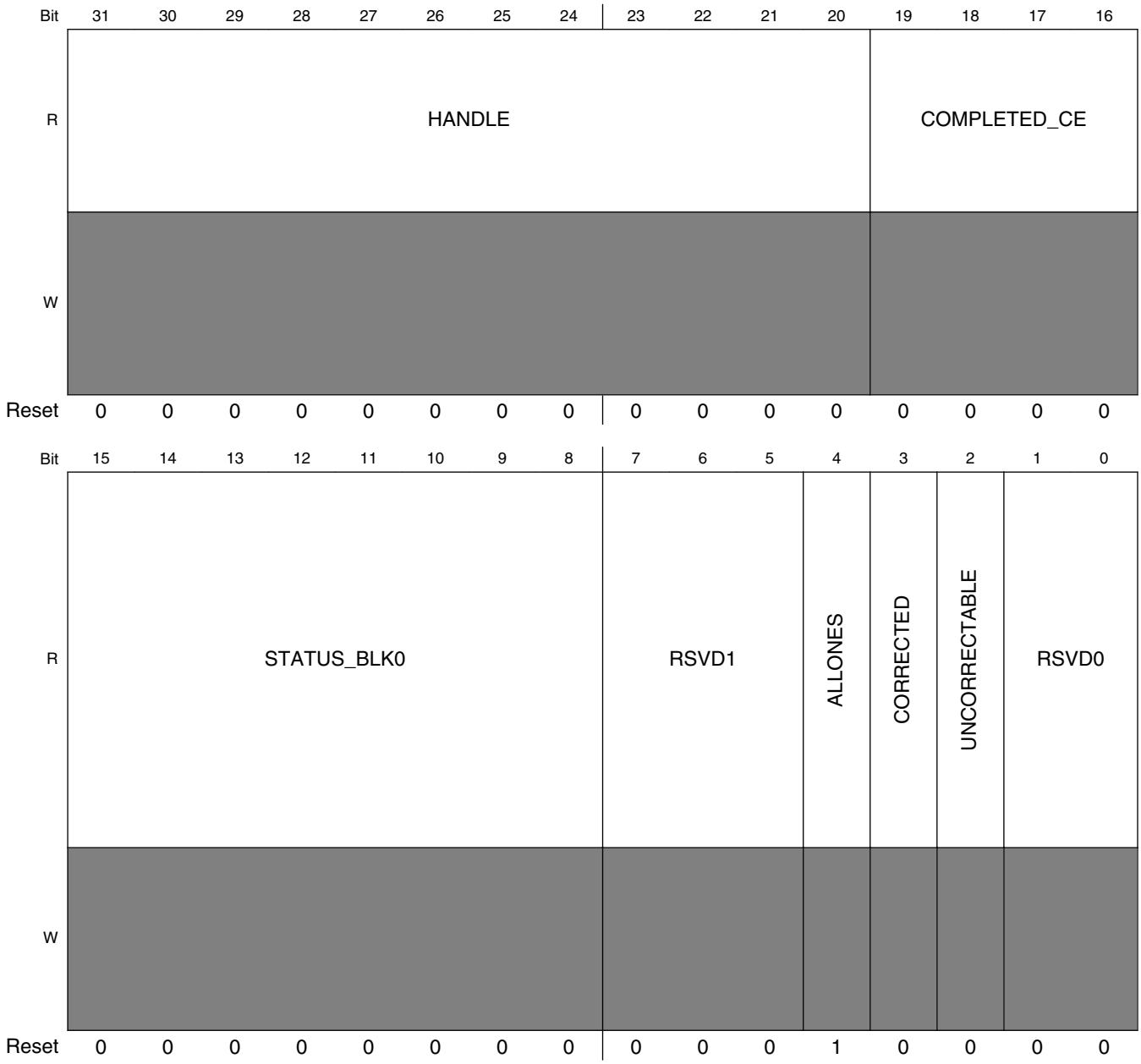
BCH_CTRL*n* field descriptions (continued)

Field	Description
	M2M_LAYOUT bits as well as the ENCODEPTR, DATAPTR, and METAPTR registers are used for memory-to-memory operations and must be correctly programmed before writing this bit.
15–11 RSVD3	This field is reserved. This read-only field is reserved and always has the value 0
10 DEBUG_STALL_ IRQ_EN	1 = interrupt on debug stall mode is enabled. The IRQ is raised on every block
9 RSVD2	This field is reserved. This read-only field is reserved and always has the value 0.
8 COMPLETE_IRQ_ EN	1 = interrupt on completion of correction is enabled.
7–4 RSVD1	This field is reserved. This read-only field is reserved and always has the value 0.
3 BM_ERROR_IRQ	AHB Bus interface Error Interrupt Status. Write a 1 to the SCT clear address to clear the interrupt status bit.
2 DEBUG_STALL_IRQ	DEBUG STALL Interrupt Status. Write a 1 to the SCT clear address to clear the interrupt status bit.
1 RSVD0	This field is reserved. This read-only field is reserved and always has the value 0.
0 COMPLETE_IRQ	This bit indicates the state of the external interrupt line. Write a 1 to the SCT clear address to clear the interrupt status bit. NOTE: subsequent ECC completions will be held off as long as this bit is set. Be sure to read the data from BCH_STATUS0, 1 before clearing this interrupt bit.

9.5.6.2 Hardware ECC Accelerator Status Register 0 (BCH_STATUS0)

The BCH STAT register provides visibility into the run-time status of the BCH and status information when processing is complete. It provides overall status of the hardware ECC accelerator.

Address: 3300_4000h base + 10h offset = 3300_4010h



BCH_STATUS0 field descriptions

Field	Description
31–20 HANDLE	Software supplies a 12 bit handle for this transfer as part of the GPMI DMA PIO operation that started the transaction. That handle passes down the pipeline and ends up here at the time the BCH interrupt is signaled.
19–16 COMPLETED_CE	This is the chip enable number corresponding to the NAND device from which this data came.
15–8 STATUS_BLK0	Count of symbols in error during processing of first block of flash (metadata block). The number of errors reported will be in the range of 0 to the ECC correction level for block 0. 0x00 **ZERO** — No errors found on block.

Table continues on the next page...

BCH_STATUS0 field descriptions (continued)

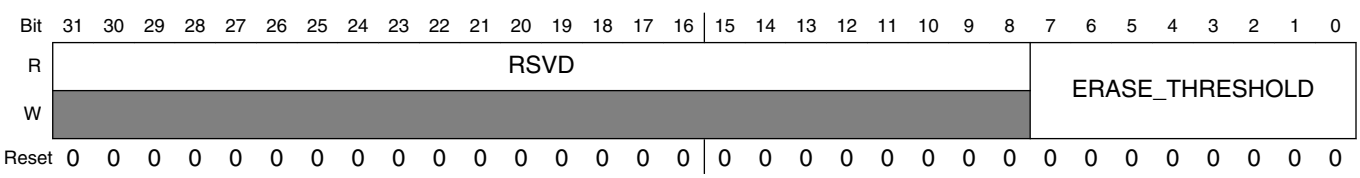
Field	Description
	0x01 **ERROR1** — One error found on block. 0x02 **ERROR2** — One errors found on block. 0x03 **ERROR3** — One errors found on block. 0x04 **ERROR4** — One errors found on block. 0xFE **UNCORRECTABLE** — Block exhibited uncorrectable errors. 0xFF **ERASED** — Page is erased.
7–5 RSVD1	This field is reserved. This read-only field is reserved and always has the value 0.
4 ALLONES	1 = All data bits of this transaction are ONE.
3 CORRECTED	1 = At least one correctable error encountered during last processing cycle.
2 UNCORRECTABLE	1 = Uncorrectable error encountered during last processing cycle.
RSVD0	This field is reserved. This read-only field is reserved and always has the value 0.

9.5.6.3 Hardware ECC Accelerator Mode Register (BCH_MODE)

The BCH MODE register provides additional mode controls.

Contains additional global mode controls for the BCH engine.

Address: 3300_4000h base + 20h offset = 3300_4020h



BCH_MODE field descriptions

Field	Description
31–8 RSVD	This field is reserved. This read-only field is reserved and always has the value 0.
ERASE_ THRESHOLD	This value indicates the maximum number of zero bits on a flash subpage for it to be considered erased. For SLC NAND devices, this value should be programmed to 0 (meaning that the entire page should consist of bytes of 0xFF. For MLC NAND devices, bit errors may occur on reads (even on blank pages), so this threshold can be used to tune the erased page checking algorithm.

9.5.6.4 Hardware BCH ECC Loopback Encode Buffer Register (BCH_ENCODEPTR)

When performing memory to memory operations, indicates the address of the encode buffer. This register should be programmed before writing a 1 to the M2M_ENABLE bit in the CTRL register.

For memory to memory operations, this register is used as the pointer to the encoded data, which is an output when encoding and an input while decoding.

Address: 3300_4000h base + 30h offset = 3300_4030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BCH_ENCODEPTR field descriptions

Field	Description
ADDR	Address pointer to encode buffer. This is the source for decode operations and the destination for encode operations. This value must be aligned on a 4 bytes boundary.

9.5.6.5 Hardware BCH ECC Loopback Data Buffer Register (BCH_DATAPTR)

When performing memory to memory operations, indicates the address of the data buffer.

For memory to memory operations, this register is used as the pointer to the data to encode or the destination buffer for decode operations.

Address: 3300_4000h base + 40h offset = 3300_4040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BCH_DATAPTR field descriptions

Field	Description
ADDR	Address pointer to data buffer. This is the source for encode operations and the destination for decode operations. This register should be programmed before writing a 1 to the M2M_ENABLE bit in the CTRL register. This value must be aligned on a 4 byte boundary.

9.5.6.6 Hardware BCH ECC Loopback Metadata Buffer Register (BCH_METAPTR)

When performing memory to memory operations, indicates the address of the metadata buffer.

For memory to memory operations, this register is used as the pointer to the metadata to encode or the extracted metadata for decode operations.

Address: 3300_4000h base + 50h offset = 3300_4050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BCH_METAPTR field descriptions

Field	Description
ADDR	Address pointer to metadata buffer. This is the source for encode metadata read operations and the destination for metadata decode operations. This register should be programmed before writing a 1 to the M2M_ENABLE bit in the CTRL register. This value must be aligned on a 4 bytes boundary.

9.5.6.7 Hardware ECC Accelerator Layout Select Register (BCH_LAYOUTSELECT)

The BCH LAYOUTSELECT register provides a mapping of chip selects to layout registers.

When the BCH engine receives a request to process a data block from the GPMI interface, it will use this register to map the incoming chip select to one of the four possible flash layout registers

Address: 3300_4000h base + 70h offset = 3300_4070h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CS15_SELECT		CS14_SELECT		CS13_SELECT		CS12_SELECT		CS11_SELECT		CS10_SELECT		CS9_SELECT		CS8_SELECT	
Reset	1	1	1	0	0	1	0	0	1	1	1	0	0	1	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CS7_SELECT		CS6_SELECT		CS5_SELECT		CS4_SELECT		CS3_SELECT		CS2_SELECT		CS1_SELECT		CS0_SELECT	
Reset	1	1	1	0	0	1	0	0	1	1	1	0	0	1	0	0

BCH_LAYOUTSELECT field descriptions

Field	Description
31–30 CS15_SELECT	Selects which layout is used for chip select 15.
29–28 CS14_SELECT	Selects which layout is used for chip select 14.
27–26 CS13_SELECT	Selects which layout is used for chip select 13.
25–24 CS12_SELECT	Selects which layout is used for chip select 12.
23–22 CS11_SELECT	Selects which layout is used for chip select 11.
21–20 CS10_SELECT	Selects which layout is used for chip select 10.
19–18 CS9_SELECT	Selects which layout is used for chip select 9.
17–16 CS8_SELECT	Selects which layout is used for chip select 8.
15–14 CS7_SELECT	Selects which layout is used for chip select 7.
13–12 CS6_SELECT	Selects which layout is used for chip select 6.
11–10 CS5_SELECT	Selects which layout is used for chip select 5.
9–8 CS4_SELECT	Selects which layout is used for chip select 4.
7–6 CS3_SELECT	Selects which layout is used for chip select 3.
5–4 CS2_SELECT	Selects which layout is used for chip select 2.
3–2 CS1_SELECT	Selects which layout is used for chip select 1.
CS0_SELECT	Selects which layout is used for chip select 0.

9.5.6.8 Hardware BCH ECC Flash 0 Layout 0 Register (BCH_FLASH0LAYOUT0)

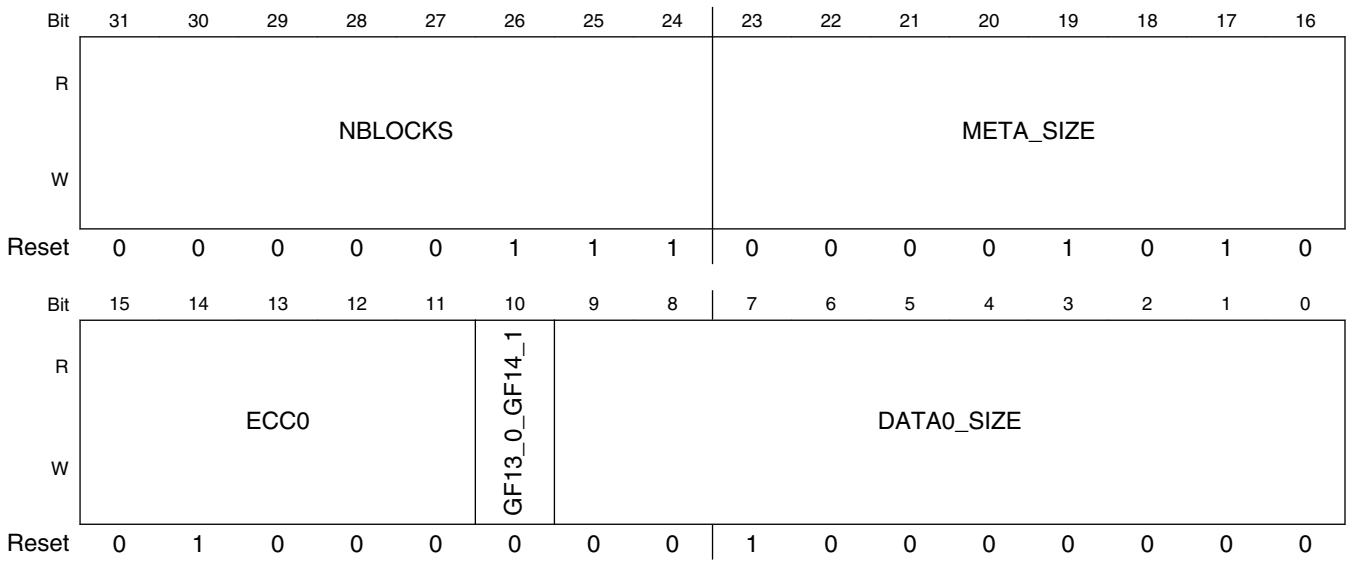
The flash format register contains a description of the logical layout of data on the flash device. This register is used in conjunction with the FLASH0LAYOUT1 register to control the format for the devices selecting layout 0 in the LAYOUTSELECT register.

Each pair of layout registers describes one of four supported flash configurations. Software should program the LAYOUTSELECT register for each supported GPMI chip select to select from one of the four layout values. Each pair of registers contains settings that are used by the BCH block while reading / writing the flash page to control data,

metadata, and flash page sizes as well as the ECC correction level. The first block written to flash can be programmed to have different ECC, metadata, and data sizes from subsequent data blocks on the device. In addition, the number of blocks stored on a page of flash is not fixed, but instead is determined by the number of bytes consumed by the initial (block 0) and subsequent data blocks.

See sections Flash Page Layout and Determining the ECC layout for a device for more detail information on setting up the flash layout registers.

Address: 3300_4000h base + 80h offset = 3300_4080h



BCH_FLASH0LAYOUT0 field descriptions

Field	Description
31–24 NBLOCKS	Number of subsequent blocks on the flash page (excluding the data0 block). A value of 0 indicates that only the DATA0 block is present and a value of 8 indicates that 8 subsequent blocks are present for a total of 9 blocks on the flash (including the DATA0 block). Any values from 0 to 255 are supported by the hardware.
23–16 META_SIZE	Indicates the size of the metadata (in bytes) to be stored on a flash page. The BCH design support from 0 to 255 bytes for metadata—if set to 0, no metadata will be stored. Metadata is stored before the associated data in block 0. If the DATA0_SIZE field is programmed to a 0, then metadata effectively be stored with its own parity. When both the metadata and data0 fields are programmed with non-zero values, the first block will contain both portions of data and covered by a single parity block.
15–11 ECC0	Indicates the ECC level for the first block on the flash page. The first block covers metadata plus the associated data from the DATA0_SIZE field. 0x0 **NONE** — No ECC to be performed 0x1 **ECC2** — ECC 2 to be performed 0x2 **ECC4** — ECC 4 to be performed : — 0x1E **ECC60** — ECC 60 to be performed 0x1F **ECC62** — ECC 62 to be performed
10 GF13_0_GF14_1	Select GF13 or GF14: 0-GF13; 1-GF14

Table continues on the next page...

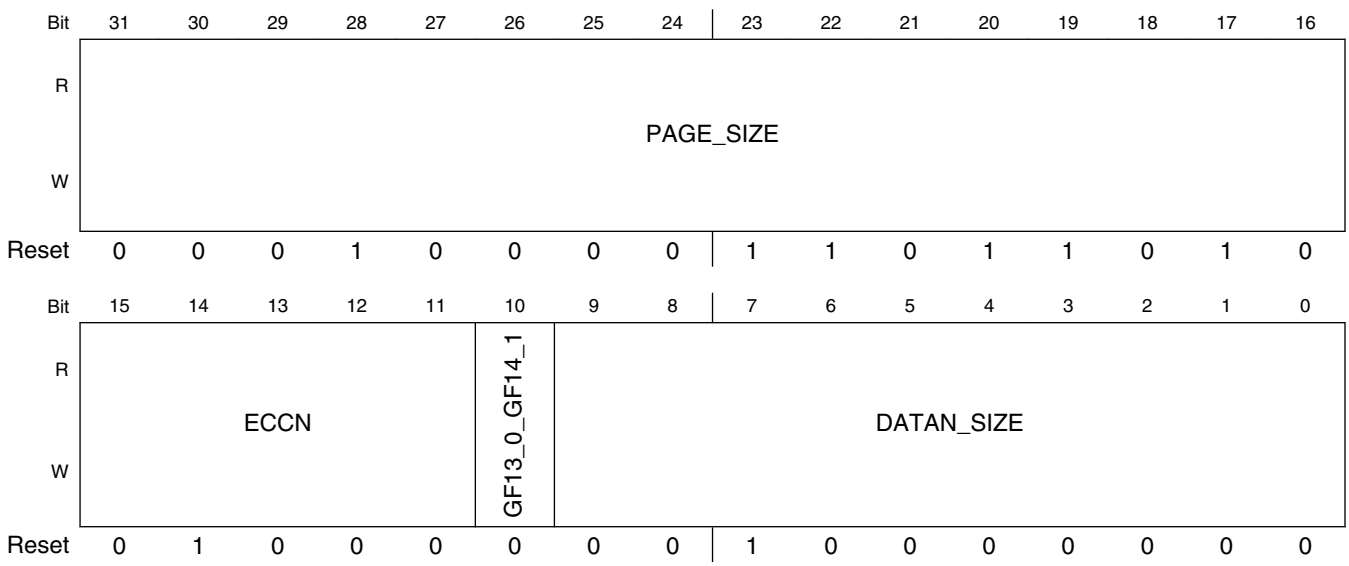
BCH_FLASH0LAYOUT0 field descriptions (continued)

Field	Description
DATA0_SIZE	Indicates the size of the data 0 block (in DWORDS / four bytes) to be stored on the flash page. If set to 0, the first block only contains metadata.

9.5.6.9 Hardware BCH ECC Flash 0 Layout 1 Register (BCH_FLASH0LAYOUT1)

The flash format register contains a description of the logical layout of data on the flash device. This register is used in conjunction with the FLASH0LAYOUT0 register to control the format for the device selecting layout 0 in the LAYOUTSELECT register.

Address: 3300_4000h base + 90h offset = 3300_4090h



BCH_FLASH0LAYOUT1 field descriptions

Field	Description
31–16 PAGE_SIZE	Indicates the total size of the flash page (in bytes). This should be set to the page size including spare area. The page size is programmable to accomodate different flash configurations that may be available in the future.
15–11 ECCN	Indicates the ECC level for the subsequent blocks on the flash page (blocks 1-n). Subsequent blocks only contain data (no metadata). 0x0 **NONE** — No ECC to be performed 0x1 **ECC2** — ECC 2 to be performed 0x2 **ECC4** — ECC 4 to be performed : —

Table continues on the next page...

BCH_FLASH0LAYOUT1 field descriptions (continued)

Field	Description
	0x1E **ECC60** — ECC 60 to be performed 0x1F **ECC62** — ECC 62 to be performed
10 GF13_0_GF14_1	Select GF13 or GF14: 0-GF13; 1-GF14
DATAN_SIZE	Indicates the size of the subsequent data blocks (in DWORDS / four bytes) to be stored on the flash page. The size of subsequent data blocks does not have to match the data size for block 0, which is important when metadata is stored separately or for balancing the amount of data stored in each block.

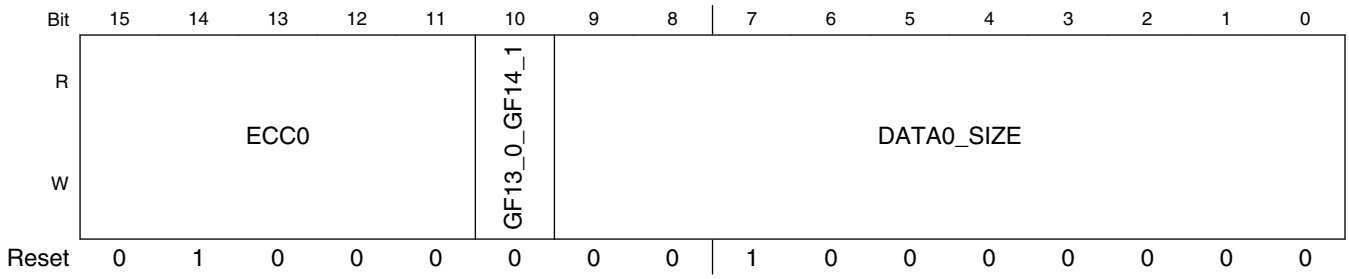
9.5.6.10 Hardware BCH ECC Flash 1 Layout 0 Register (BCH_FLASH1LAYOUT0)

The flash format register contains a description of the logical layout of data on the flash device. This register is used in conjunction with the FLASH1LAYOUT1 register to control the format for the devices selecting layout 1 in the LAYOUTSELECT register.

Each pair of layout registers describes one of four supported flash configurations. Software should program the LAYOUTSELECT register for each supported GPMI chip select to select from one of the four layout values. Each pair of registers contains settings that are used by the BCH block while reading / writing the flash page to control data, metadata, and flash page sizes as well as the ECC correction level. The first block written to flash can be programmed to have different ECC, metadata, and data sizes from subsequent data blocks on the device. In addition, the number of blocks stored on a page of flash is not fixed, but instead is determined by the number of bytes consumed by the initial (block 0) and subsequent data blocks.

See sections Flash Page Layout and Determining the ECC layout for a device for more detail information on setting up the flash layout registers.

Address: 3300_4000h base + A0h offset = 3300_40A0h



BCH_FLASH1LAYOUT0 field descriptions

Field	Description
31–24 NBLOCKS	Number of subsequent blocks on the flash page (excluding the data0 block). A value of 0 indicates that only the DATA0 block is present and a value of 8 indicates that 8 subsequent blocks are present for a total of 9 blocks on the flash (including the DATA0 block). Any values from 0 to 255 are supported by the hardware.
23–16 META_SIZE	Indicates the size of the metadata (in bytes) to be stored on a flash page. The BCH design supports from 0 to 255 bytes for metadata—if set to 0, no metadata will be stored. Metadata is stored before the associated data is in block 0. If the DATA0_SIZE field is programmed to a 0, then metadata effectively be stored with its own parity. When both the metadata and data0 fields are programmed with non-zero values, the first block will contain both portions of data and covered by a single parity block.
15–11 ECC0	Indicates the ECC level for the first block on the flash page. The first block covers metadata plus the associated data from the DATA0_SIZE field. 0x0 **NONE** — No ECC to be performed 0x1 **ECC2** — ECC 2 to be performed 0x2 **ECC4** — ECC 4 to be performed : — 0x1E **ECC60** — ECC 60 to be performed 0x1F **ECC62** — ECC 62 to be performed
10 GF13_0_GF14_1	Select GF13 or GF14: 0-GF13; 1-GF14
DATA0_SIZE	Indicates the size of the data 0 block (in DWORDS / four bytes) to be stored on the flash page. If set to 0, the first block will contains metadata.

9.5.6.11 Hardware BCH ECC Flash 1 Layout 1 Register (BCH_FLASH1LAYOUT1)

The flash format register contains a description of the logical layout of data on the flash device. This register is used in conjunction with the FLASH1LAYOUT0 register to control the format for the device selecting layout 1 in the LAYOUTSELECT register.

Address: 3300_4000h base + B0h offset = 3300_40B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PAGE_SIZE															
W																
Reset	0	0	0	1	0	0	0	0	1	1	0	1	1	0	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ECCN					GF13_0_GF14_1	DATAN_SIZE									
W																
Reset	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0

BCH_FLASH1LAYOUT1 field descriptions

Field	Description
31–16 PAGE_SIZE	Indicates the total size of the flash page (in bytes). This should be set to the page size including spare area. The page size is programmable to accomodate different flash configurations that may be available in the future.
15–11 ECCN	Indicates the ECC level for the subsequent blocks on the flash page (blocks 1-n). Subsequent blocks only contain data (no metadata). 0x0 **NONE** — No ECC to be performed 0x1 **ECC2** — ECC 2 to be performed 0x2 **ECC4** — ECC 4 to be performed : — 0x1E **ECC60** — ECC 60 to be performed 0x1F **ECC62** — ECC 62 to be performed
10 GF13_0_GF14_1	Select GF13 or GF14: 0-GF13; 1-GF14
DATAN_SIZE	Indicates the size of the subsequent data blocks (in DWORDS / four bytes) to be stored on the flash page. The size of subsequent data blocks does not have to match the data size for block 0, which is important when metadata is stored separately or for balancing the amount of data stored in each block.

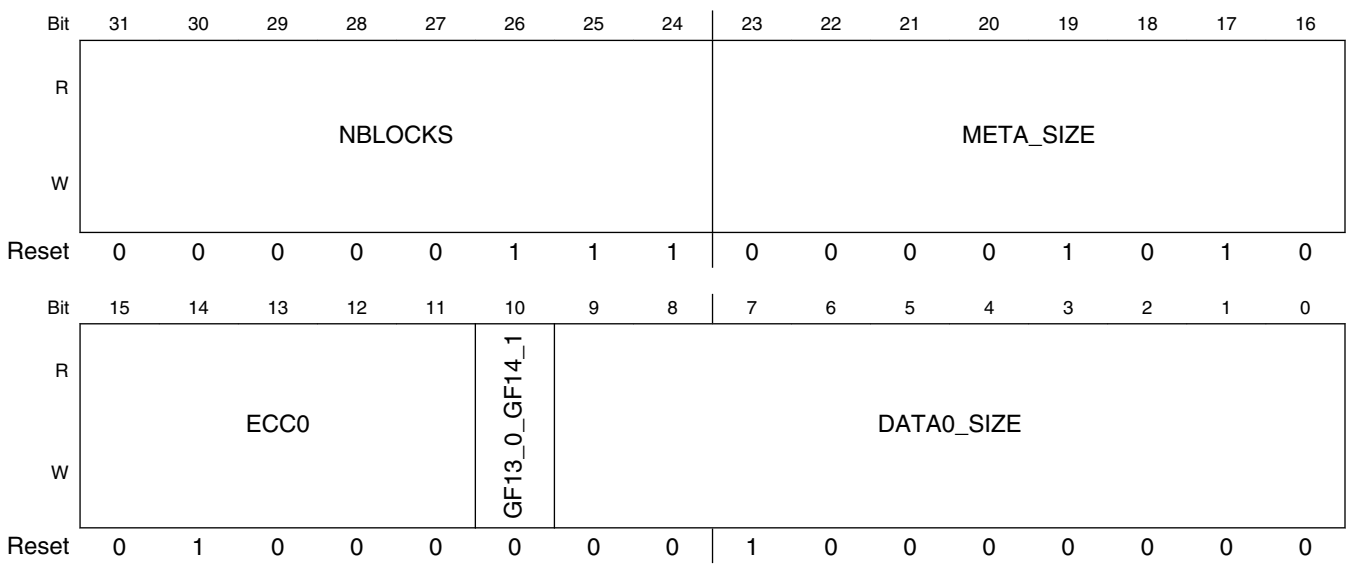
9.5.6.12 Hardware BCH ECC Flash 2 Layout 0 Register (BCH_FLASH2LAYOUT0)

The flash format register contains a description of the logical layout of data on the flash device. This register is used in conjunction with the FLASH2LAYOUT1 register to control the format for the devices selecting layout 2 in the LAYOUTSELECT register.

Each pair of layout registers describes one of four supported flash configurations. Software should program the LAYOUTSELECT register for each supported GPMI chip select to select from one of the four layout values. Each pair of registers contains settings that are used by the BCH block while reading / writing the flash page to control data, metadata, and flash page sizes as well as the ECC correction level. The first block written to flash can be programmed to have different ECC, metadata, and data sizes from subsequent data blocks on the device. In addition, the number of blocks stored on a page of flash is not fixed, but instead is determined by the number of bytes consumed by the initial (block 0) and subsequent data blocks.

See sections Flash Page Layout and Determining the ECC layout for a device for more detail information on setting up the flash layout registers.

Address: 3300_4000h base + C0h offset = 3300_40C0h



BCH_FLASH2LAYOUT0 field descriptions

Field	Description
31–24 NBLOCKS	Number of subsequent blocks on the flash page (excluding the data0 block). A value of 0 indicates that only the DATA0 block is present and a value of 8 indicates that eight subsequent blocks are present for a total of nine blocks on the flash (including the DATA0 block). Any values from 0 to 255 are supported by the hardware.

Table continues on the next page...

BCH_FLASH2LAYOUT0 field descriptions (continued)

Field	Description
23–16 META_SIZE	Indicates the size of the metadata (in bytes) to be stored on a flash page. The BCH design support from 0 to 255 bytes for metadata—if set to 0, no metadata will be stored. Metadata is stored before the associated data in block 0. If the DATA0_SIZE field is programmed to a 0, then metadata effectively be stored with its own parity. When both the metadata and data0 fields are programmed with non-zero values, the first block will contain both portions of data and will be covered by a single parity block.
15–11 ECC0	Indicates the ECC level for the first block on the flash page. The first block covers metadata plus the associated data from the DATA0_SIZE field. 0x0 **NONE** — No ECC to be performed 0x1 **ECC2** — ECC 2 to be performed 0x2 **ECC4** — ECC 4 to be performed : — 0x1E **ECC60** — ECC 60 to be performed 0x1F **ECC62** — ECC 62 to be performed
10 GF13_0_GF14_1	Select GF13 or GF14: 0-GF13; 1-GF14
DATA0_SIZE	Indicates the size of the data 0 block (in DWORDS / four bytes) to be stored on the flash page. If set to 0, the first block will only contain metadata.

9.5.6.13 Hardware BCH ECC Flash 2 Layout 1 Register (BCH_FLASH2LAYOUT1)

The flash format register contains a description of the logical layout of data on the flash device. This register is used in conjunction with the FLASH2LAYOUT0 register to control the format for the device selecting layout 2 in the LAYOUTSELECT register.

Address: 3300_4000h base + D0h offset = 3300_40D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PAGE_SIZE															
Reset	0	0	0	1	0	0	0	0	1	1	0	1	1	0	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ECCN					GF13_0_GF14_1	DATAN_SIZE									
Reset	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0

BCH_FLASH2LAYOUT1 field descriptions

Field	Description
31–16 PAGE_SIZE	Indicates the total size of the flash page (in bytes). This should be set to the page size including spare area. The page size is programmable to accomodate different flash configurations that may be available in the future.
15–11 ECCN	Indicates the ECC level for the subsequent blocks on the flash page (blocks 1-n). Subsequent blocks only contain data (no metadata). 0x0 **NONE** — No ECC to be performed 0x1 **ECC2** — ECC 2 to be performed 0x2 **ECC4** — ECC 4 to be performed : — 0x1E **ECC60** — ECC 60 to be performed 0x1F **ECC62** — ECC 62 to be performed
10 GF13_0_GF14_1	Select GF13 or GF14: 0-GF13; 1-GF14
DATAN_SIZE	Indicates the size of the subsequent data blocks (in DWORDS / four bytes) to be stored on the flash page. The size of subsequent data blocks does not have to match the data size for block 0, which is important when metadata is stored separately or for balancing the amount of data stored in each block.

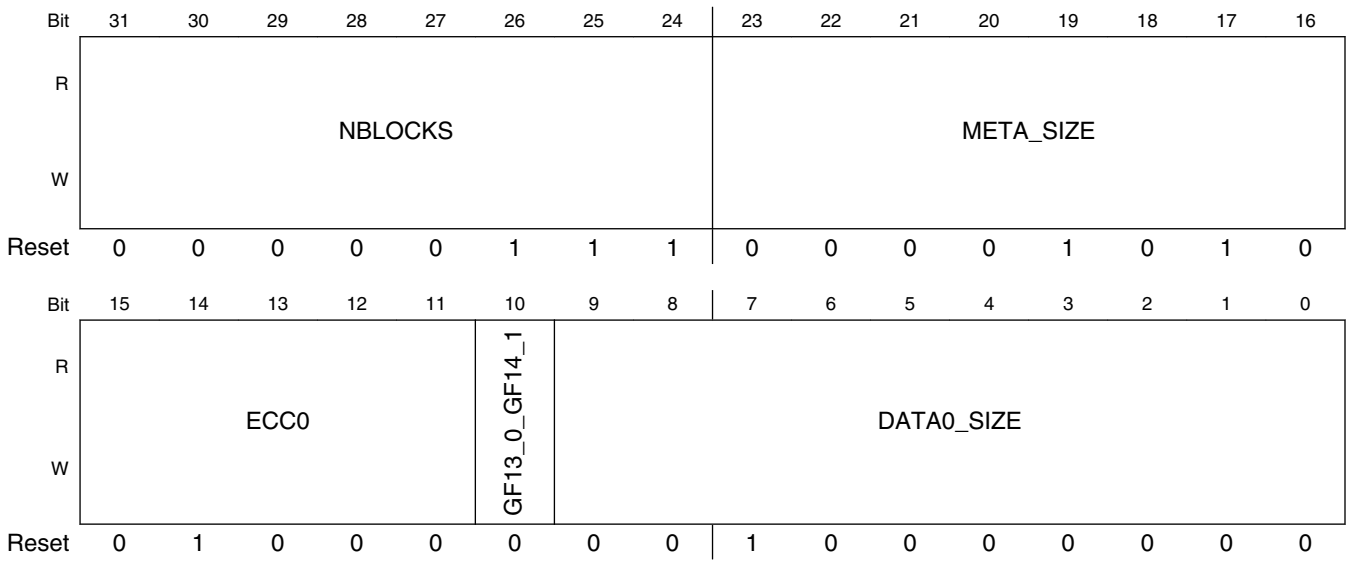
9.5.6.14 Hardware BCH ECC Flash 3 Layout 0 Register (BCH_FLASH3LAYOUT0)

The flash format register contains a description of the logical layout of data on the flash device. This register is used in conjunction with the FLASH3LAYOUT1 register to control the format for the devices selecting layout 3 in the LAYOUTSELECT register.

Each pair of layout registers describes one of four supported flash configurations. Software should program the LAYOUTSELECT register for each supported GPMI chip select to select from one of the four layout values. Each pair of registers contains settings that are used by the BCH block while reading / writing the flash page to control data, metadata, and flash page sizes as well as the ECC correction level. The first block written to flash can be programmed to have different ECC, metadata, and data sizes from subsequent data blocks on the device. In addition, the number of blocks stored on a page of flash is not fixed, but instead is determined by the number of bytes consumed by the initial (block 0) and subsequent data blocks.

See sections Flash Page Layout and Determining the ECC layout for a device for more detail information on setting up the flash layout registers.

Address: 3300_4000h base + E0h offset = 3300_40E0h



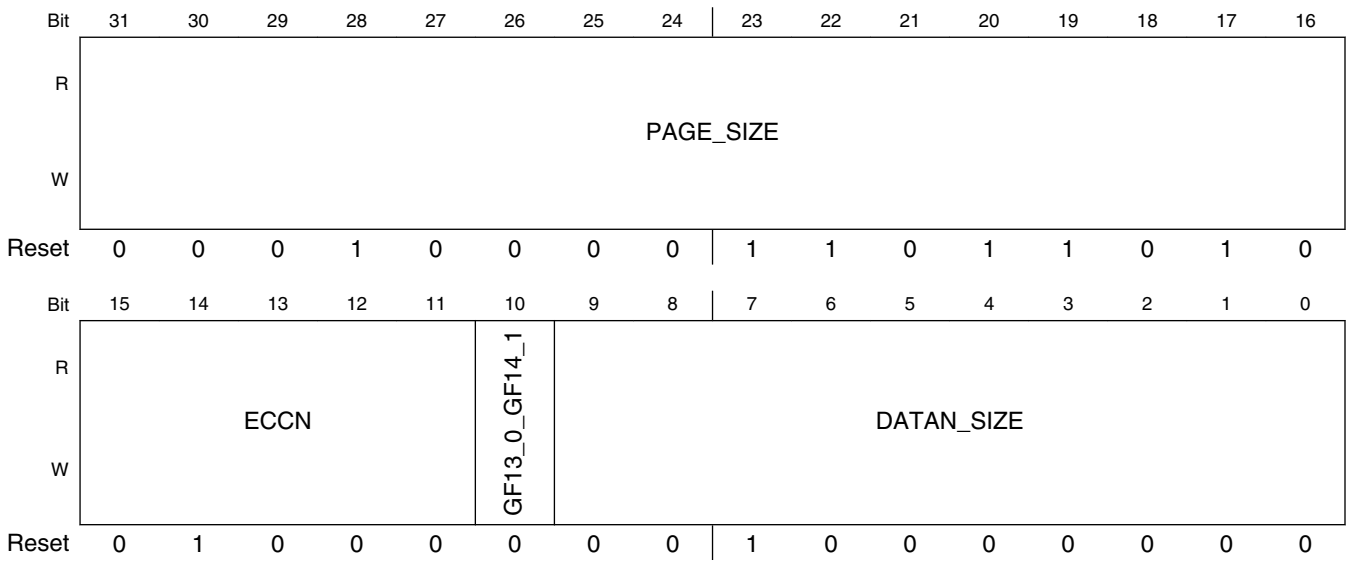
BCH_FLASH3LAYOUT0 field descriptions

Field	Description
31–24 NBLOCKS	Number of subsequent blocks on the flash page (excluding the data0 block). A value of 0 indicates that only the DATA0 block is present and a value of 8 indicates that 8 subsequent blocks are present for a total of 9 blocks on the flash (including the DATA0 block). Any values from 0 to 255 are supported by the hardware.
23–16 META_SIZE	Indicates the size of the metadata (in bytes) to be stored on a flash page. The BCH design support from 0 to 255 bytes for metadata—if set to 0, no metadata will be stored. Metadata is stored before the associated data in block 0. If the DATA0_SIZE field is programmed to a 0, then metadata effectively be stored with its own parity. When both the metadata and data0 fields are programmed with non-zero values, the first block will contain both portions of data and will be covered by a single parity block.
15–11 ECC0	Indicates the ECC level for the first block on the flash page. The first block covers metadata plus the associated data from the DATA0_SIZE field. 0x0 **NONE** — No ECC to be performed 0x1 **ECC2** — ECC 2 to be performed 0x2 **ECC4** — ECC 4 to be performed : — 0x1E **ECC60** — ECC 60 to be performed 0x1F **ECC62** — ECC 62 to be performed
10 GF13_0_GF14_1	Select GF13 or GF14: 0-GF13; 1-GF14
DATA0_SIZE	Indicates the size of the data 0 block (in DWORDS / four bytes) to be stored on the flash page. If set to 0, the first block will only contain metadata.

9.5.6.15 Hardware BCH ECC Flash 3 Layout 1 Register (BCH_FLASH3LAYOUT1)

The flash format register contains a description of the logical layout of data on the flash device. This register is used in conjunction with the FLASH3LAYOUT0 register to control the format for the device selecting layout 3 in the LAYOUTSELECT register.

Address: 3300_4000h base + F0h offset = 3300_40F0h



BCH_FLASH3LAYOUT1 field descriptions

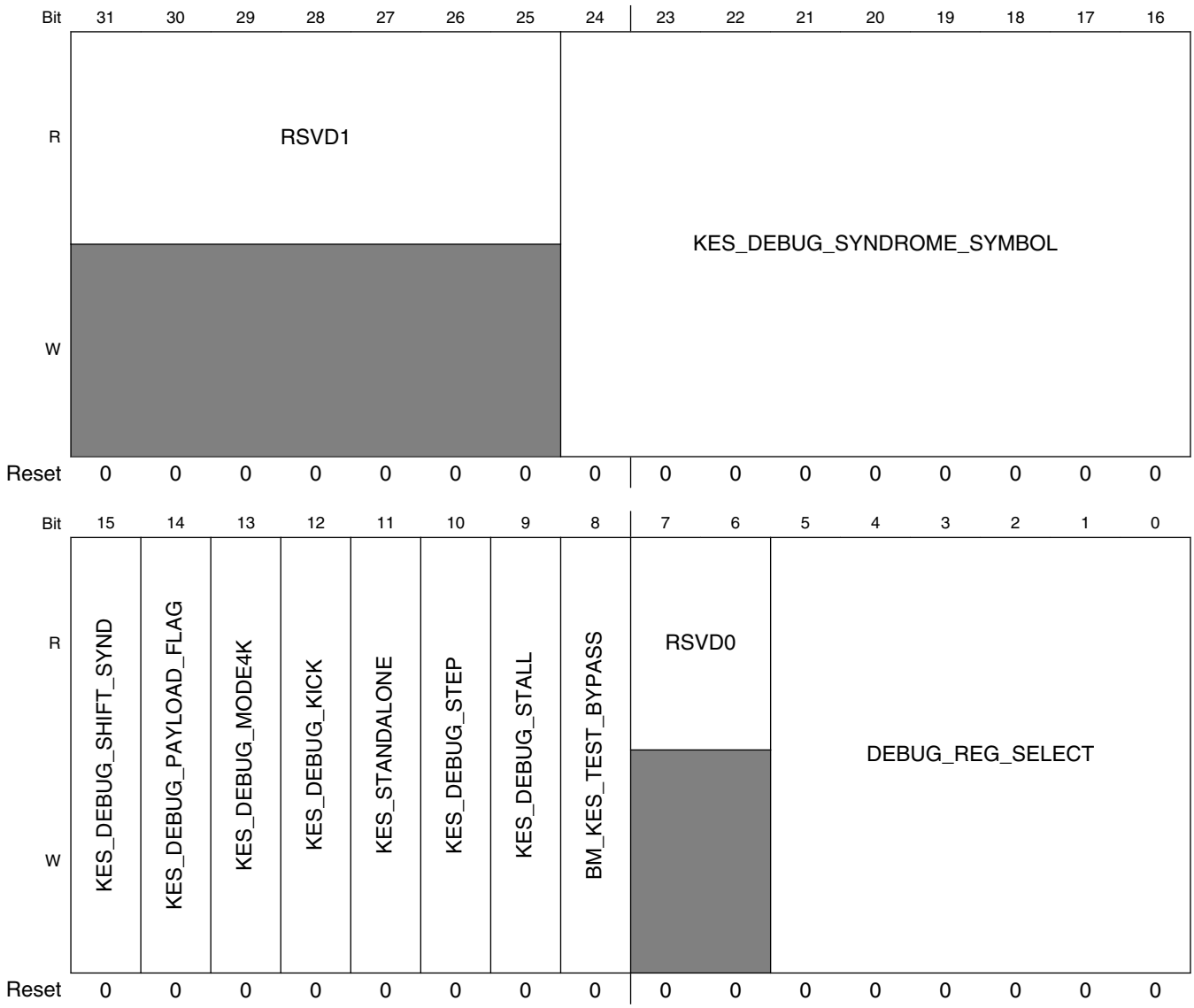
Field	Description
31–16 PAGE_SIZE	Indicates the total size of the flash page (in bytes). This should be set to the page size including spare area. The page size is programmable to accomodate different flash configurations that may be available in the future.
15–11 ECCN	Indicates the ECC level for the subsequent blocks on the flash page (blocks 1-n). Subsequent blocks only contain data (no metadata). 0x0 **NONE** — No ECC to be performed 0x1 **ECC2** — ECC 2 to be performed 0x2 **ECC4** — ECC 4 to be performed : — 0x1E **ECC60** — ECC 60 to be performed 0x1F **ECC62** — ECC 62 to be performed
10 GF13_0_GF14_1	Select GF13 or GF14: 0-GF13; 1-GF14
DATAN_SIZE	Indicates the size of the subsequent data blocks (in DWORDS / four bytes) to be stored on the flash page. The size of subsequent data blocks does not have to match the data size for block 0, which is important when metadata is stored separately or for balancing the amount of data stored in each block.

9.5.6.16 Hardware BCH ECC Debug Register0 (BCH_DEBUG0*n*)

The hardware BCH accelerator internal state machines and signals can be seen in the ECC debug register.

The BCH_DEBUG0 register provides access to various internal state information which might prove useful during hardware debug and validation.

Address: 3300_4000h base + 100h offset + (4d × i), where i=0d to 3d



BCH_DEBUG0*n* field descriptions

Field	Description
31–25 RSVD1	This field is reserved.

Table continues on the next page...

BCH_DEBUG0*n* field descriptions (continued)

Field	Description
	This read-only field is reserved and always has the value 0.
24–16 KES_DEBUG_ SYNDROME_ SYMBOL	The 9 bit value in this bit field shifts into the syndrome register array at the input of the KES engine whenever BCH_DEBUG0_KES_DEBUG_SHIFT_SYND is toggled. 0x0 **NORMAL** — Bus master address generator for SYND_GEN writes operates normally. 0x1 **TEST_MODE** — Bus master address generator always addresses last four bytes in Auxiliary block.
15 KES_DEBUG_ SHIFT_SYND	Toggling this bit causes the value in BCH_DEBUG0_KES_SYNDROME_SYMBOL to be shift into the syndrome register array at the input to the KES engine. After shifting in 16 symbols, one can kick off both KES and CF cycles by toggling BCH_DEBUG0_KES_DEBUG_KICK. Make sure that set KES_BCH_DEBUG0_KES_STANDALONE mode to 1 before kicking.
14 KES_DEBUG_ PAYLOAD_FLAG	When running the stand alone debug mode on the error calculator, the state of this bit is presented to the KES engine as the input payload flag. 0x1 **DATA** — Payload is set for 512 bytes data block. 0x1 **AUX** — Payload is set for 65 or 19 bytes auxiliary block.
13 KES_DEBUG_ MODE4K	When running the stand alone debug mode on the error calculator, the state of this bit is presented to the KES engine as the input mode (4K or 2K pages). 0x1 **4k** — Mode is set for 4K NAND pages. 0x1 **2k** — Mode is set for 2K NAND pages.
12 KES_DEBUG_ KICK	Toggling causes KES engine FSM to start as if kick by the Bus Master. This allows stand alone testing of the KES and Chien Search engines. Be sure to set KES_BCH_DEBUG0_KES_STANDALONE mode to 1 before kicking.
11 KES_ STANDALONE	Set to one, cause the KES engine to suppress toggling the KES_BM_DONE signal to the bus master and suppress toggling the CF_BM_DONE signal by the CF engine. 0x0 **NORMAL** — Bus master address generator for SYND_GEN writes operates normally. 0x1 **TEST_MODE** — Bus master address generator always addresses last four bytes in Auxiliary block.
10 KES_DEBUG_ STEP	Toggling this bit causes the KES FSM to skip passed the stall state if it is in DEBUG_STALL mode and completed processing a block.
9 KES_DEBUG_ STALL	Set to one to cause KES FSM to stall after notifying Chien search engine to start processing its block but before notifying the bus master that the KES computation is complete. This allows a diagnostic to stall the FSM after each blocks key equations are solved. This also has the effect of stalling the CSFE search engine so it's state can be examined after it finishes processing the KES stalled block. 0x0 **NORMAL** — KES FSM proceeds to next block supplied by bus master. 0x1 **WAIT** — KES FSM waits after current equations are solved and the search engine is started.
8 BM_KES_TEST_ BYPASS	1 = Point all SYND_GEN writes to dummy area at the end of the AUXILLIARY block so that diagnostics can preload all payload, parity bytes and computed syndrome bytes for test the KES engine. 0x0 **NORMAL** — Bus master address generator for SYND_GEN writes operates normally. 0x1 **TEST_MODE** — Bus master address generator always addresses last four bytes in Auxiliary block.
7–6 RSVD0	This field is reserved. This read-only field is reserved and always has the value 0.
DEBUG_REG_ SELECT	The value loaded in this bit field is used to select the internal register state view of KES engine or the Chien search engine.

9.5.6.17 KES Debug Read Register (BCH_DBGKESREAD)

The hardware BCH ECC accelerator key equation solver internal state machines and signals can be seen in the ECC debug registers.

Address: 3300_4000h base + 110h offset = 3300_4110h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VALUES																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BCH_DBGKESREAD field descriptions

Field	Description
VALUES	This register returns the ROM BIST CRC value after a BIST test.

9.5.6.18 Chien Search Debug Read Register (BCH_DBGCSFEREAD)

The hardware BCH ECC accelerator Chien Search internal state machines and signals can be seen in the ECC debug registers.

Address: 3300_4000h base + 120h offset = 3300_4120h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VALUES																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BCH_DBGCSFEREAD field descriptions

Field	Description
VALUES	Reserved

9.5.6.19 Syndrome Generator Debug Read Register (BCH_DBGSYNDGENREAD)

The hardware BCH ECC accelerator syndrome generator internal state machines and signals can be seen in the ECC debug registers.

Address: 3300_4000h base + 130h offset = 3300_4130h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VALUES																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BCH_DBGSYNDGENREAD field descriptions

Field	Description
VALUES	Reserved

9.5.6.20 Bus Master and ECC Controller Debug Read Register (BCH_DBGAHBMREAD)

The hardware BCH ECC accelerator bus master, ECC controller internal state machines, and signals can be seen in the ECC debug registers.

Address: 3300_4000h base + 140h offset = 3300_4140h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VALUES																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BCH_DBGAHBMREAD field descriptions

Field	Description
VALUES	Reserved

9.5.6.21 Block Name Register (BCH_BLOCKNAME)

Read only view of the block name string BCH.

Fixed pattern read only value is for test purposes. It can be read as an ASCII string with the zero termination coming from the first byte of the BLOCKVERSION register.

Address: 3300_4000h base + 150h offset = 3300_4150h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NAME																															
W																																
Reset	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	1	1	0	1	0	0	0	0	1	0

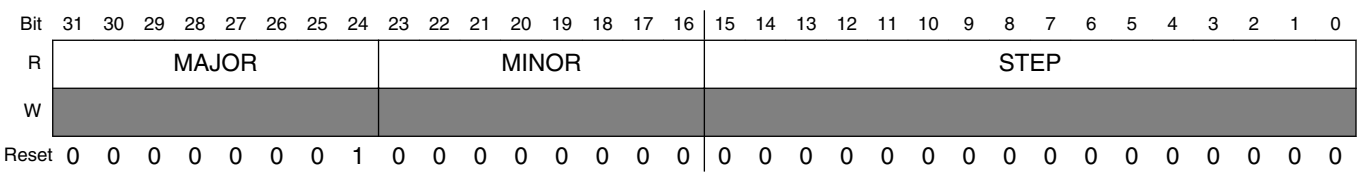
BCH_BLOCKNAME field descriptions

Field	Description
NAME	The name is in the ASCII characters BCH (0x20, H, C, B).

9.5.6.22 BCH Version Register (BCH_VERSION)

This register always returns a known read value for debug purposes and indicates the version of the block and RTL version in use.

Address: 3300_4000h base + 160h offset = 3300_4160h



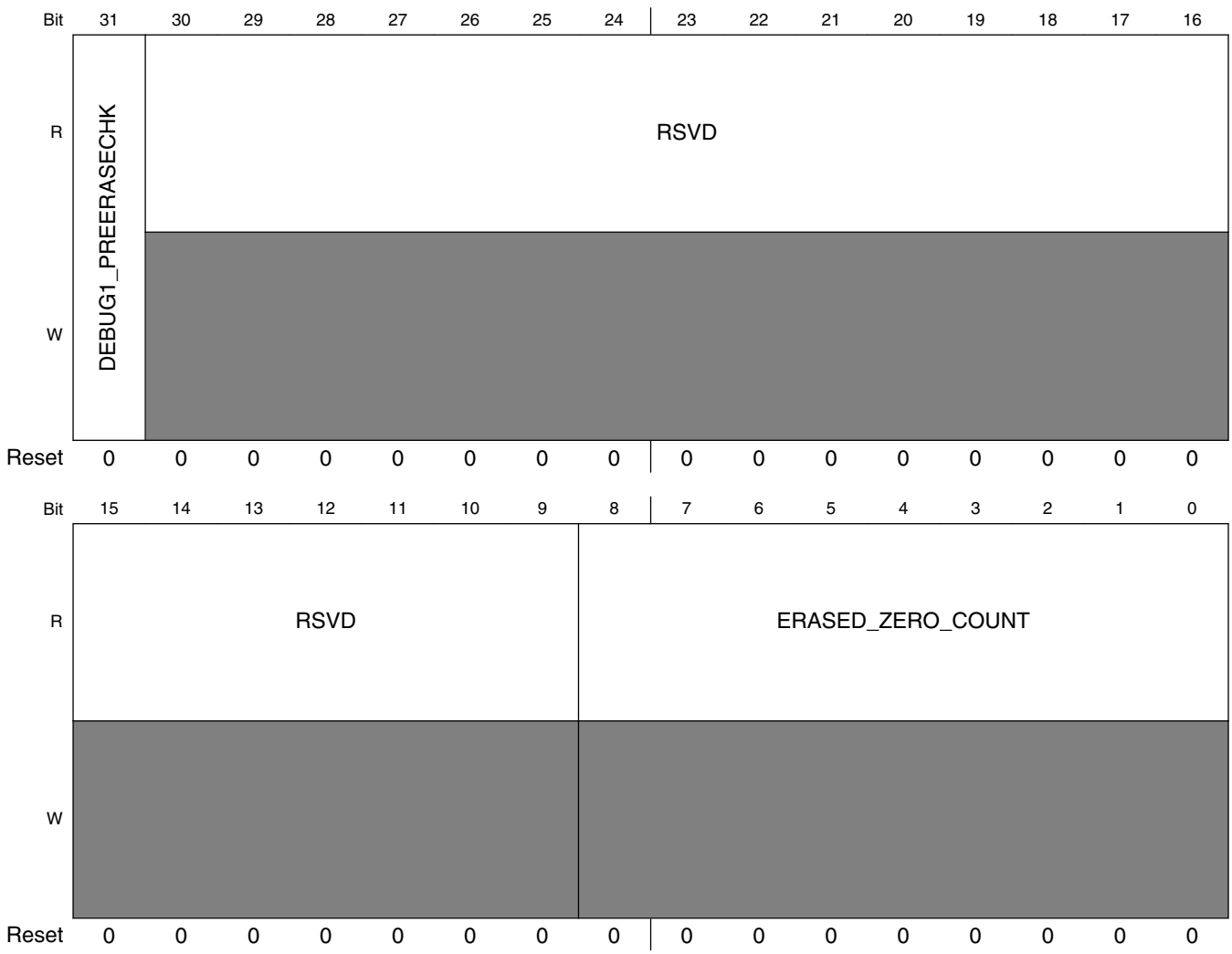
BCH_VERSION field descriptions

Field	Description
31–24 MAJOR	Fixed read-only value indicates the MAJOR field of the RTL version.
23–16 MINOR	Fixed read-only value indicates the MINOR field of the RTL version.
STEP	Fixed read-only value reflecting the stepping of the RTL version.

9.5.6.23 Hardware BCH ECC Debug Register 1 (BCH_DEBUG1)

The BCH_DEBUG1 register provides erased zero count information and pre-erase check.

Address: 3300_4000h base + 170h offset = 3300_4170h



BCH_DEBUG1 field descriptions

Field	Description
31 DEBUG1_ PREERASECHK	Blank page enables pre-erase check. 0x0 Turn off pre-erase check 0x1 Turn on pre-erase check
30–9 RSVD	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

BCH_DEBUG1 field descriptions (continued)

Field	Description
ERASED_ ZERO_COUNT	The zero counts on one page.

9.6 General Purpose Media Interface (GPMI)

9.6.1 Overview

The GPMI controller is a flexible interface to supporting up to four NAND flash chip selects.

- ONFI3.2, DDR Mode, Samsung / Toshiba Toggle NAND protocol compatible.
- Fully configurable address and command behavior, providing support for future devices not yet specified.

The GPMI resides on the APBH. The GPMI also provides an interface to the BCH module to allow direct parity processing.

Registers are clocked on the HCLK domain. The I/O and pin timing are clocked on a dedicated GPMICLK domain. GPMICLK can be set to maximize I/O performance.

The following figure shows a block diagram of the GPMI controller.

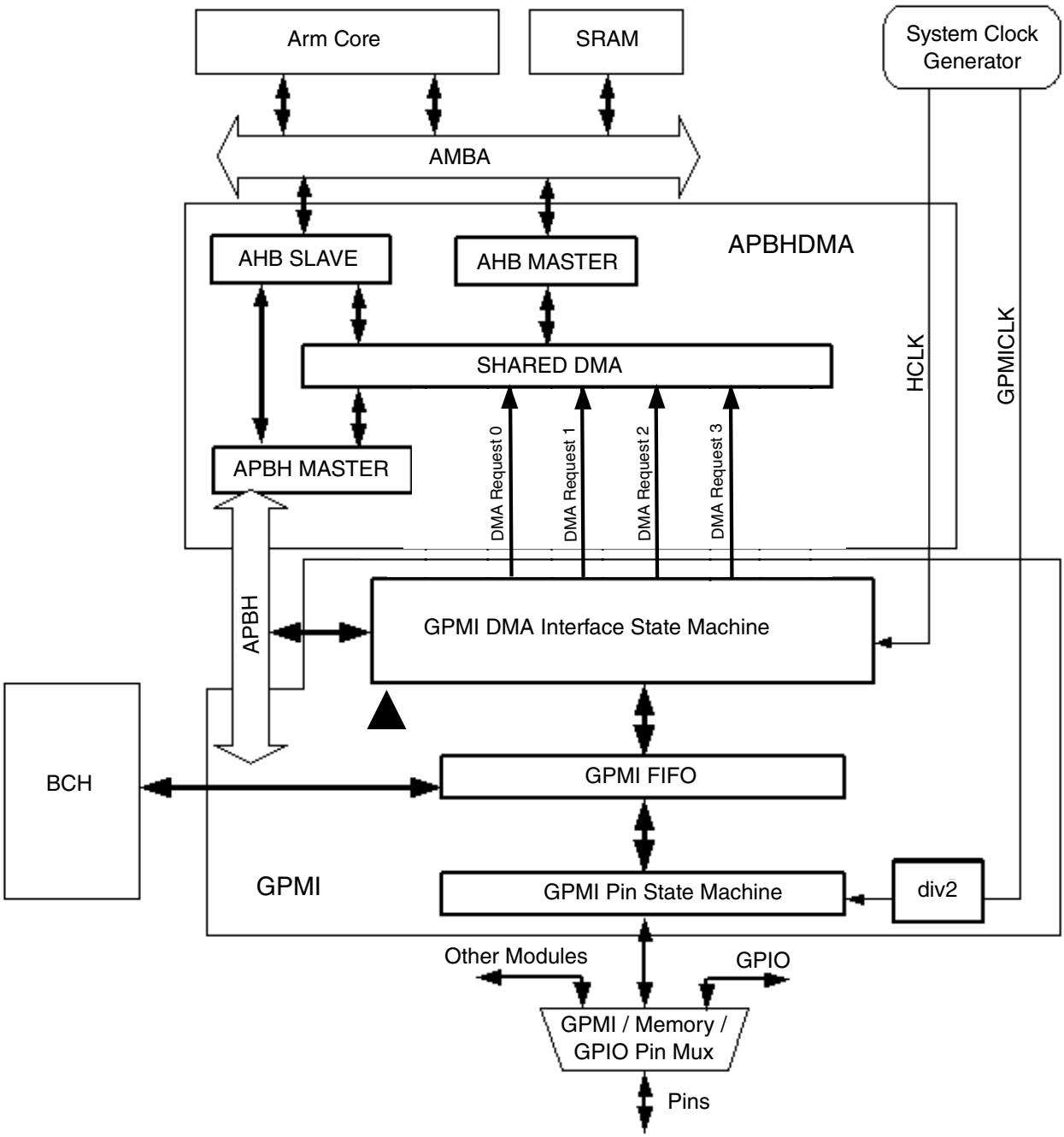


Figure 9-20. General-Purpose Media Interface Controller Block Diagram

9.6.2 GPMI NAND Mode

The general-purpose media interface has several features to efficiently support NAND:

- Individual chip select pins and ganged ready/busy pin for up to four NANDs.
- Individual state machine and DMA channel for each chip select.

- Special command modes work with DMA controller to perform all normal NAND functions without CPU intervention.
- Configurable timing based on a dedicated clock allows optimal balance of high NAND performance and low system power.

GPMI and DMA have been designed to handle complex multi-page operations without CPU intervention. The DMA uses a linked descriptor function with branching capability to automatically handle all of the operations needed to read/write multiple pages:

- **Data/Register Read/Write**-The GPMI can be programmed to read or write multiple cycles to the NAND address, command or data registers.
- **Wait for NAND Ready**-The GPMI's Wait-for-Ready mode can monitor the ready/busy signal of a single NAND flash and signal the DMA when the device has become ready. It also has a time-out counter and can indicate to the DMA that a time-out error has occurred. The DMAs can conditionally branch to a different descriptor in the case of an error.
- **Check Status**-The Read-and-Compare mode allows the GPMI to check NAND status against a reference. If an error is found, the GPMI can instruct the DMA to branch to an alternate descriptor, which attempts to fix the problem or asserts a CPU IRQ.

9.6.2.1 Multiple NAND Support

The GPMI supports up to four NAND chip selects, with ganged ready/busy pins. Since they share a data bus and control lines, the GPMI can only actively communicate with a single NAND at a time. However, all NANDs can concurrently perform internal read, write, or erase operations. With fast NAND flash and software support for concurrent NAND operations, this architecture allows the total throughput to approach the data bus speed, which can be as high as 50 MB/s (8-bit bus running at 50 MHz single clock edge) in asynchronous mode and 200MB/s (8-bit bus running at 100MHz both clock edges) in Source Synchronous mode.

There are two options for controlling the four NAND chip selects via the DMA interface. The first option is the one to one mapping, where the each DMA channel is tied to its own NAND chip select. For example DMA channel 'n' accesses only NAND attached to chip select 'n'. The second option is the decoupled mode where a DMA channel can access any or all NAND chip selects connected to the GPMI. A DMA channel will signify the NAND chip select it wants to access by writing its chip select value in the GPMI_CTRL0[CS] field and setting the GPMI_CTRL1[DECOUPLE_CS] to 1. This option is useful if software chooses to use only one DMA channel to access all the attached NAND devices.

9.6.2.2 GPMI NAND Timing and Clocking

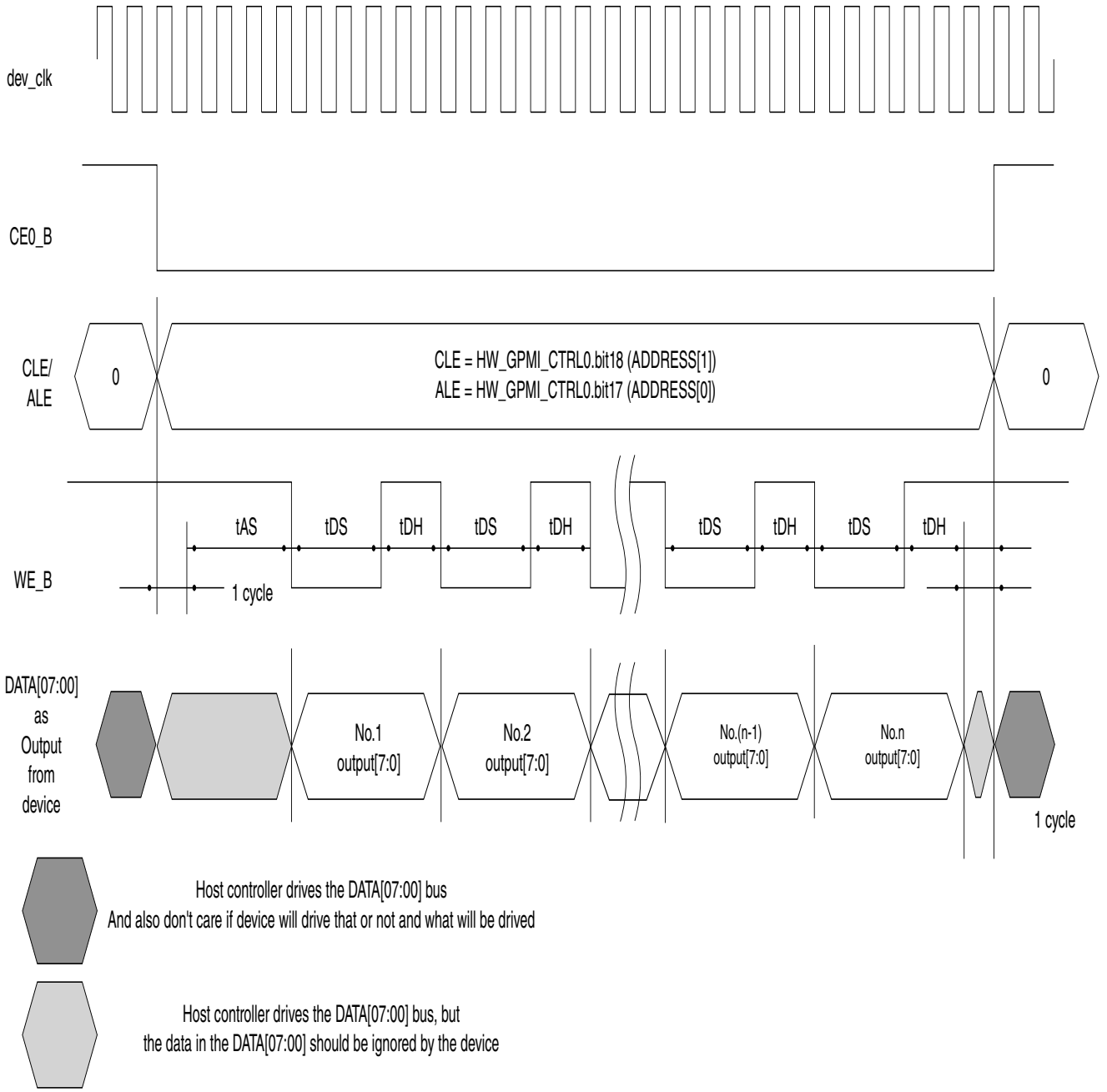
The dedicated clock, GPMICLK, is used as a timing reference for NAND flash I/O. Since various NANDs have different timing requirements, GPMICLK may need to be adjusted for each application.

While the actual pin timings are limited by the NAND chips used and the I/O pad configuration, the GPMI can support data bus speeds of up to 200 MHz x 8 bits. The actual read/write strobe timing parameters are adjusted as indicated in the register descriptions in Memory Map.

9.6.2.3 Basic NAND Timing

9.6.2.3.1 NAND Asynchronous Timing

Figure 9-22 and illustrates the operation of the output (from host to device) timing parameters in NAND ONFI asynchronous mode.



- tAS is configurable by programming HW_GPMI_TIMING0 Address_Setup: in this example, Address_Setup = 4, tAS is equal to 4 dev_clk cycles.
- tDS is configurable by programming HW_GPMI_TIMING0 Data_Setup; in this example, Data_Setup = 3, tDS is equal to 3 dev_clk cycles
- tDH is configuarble by programming HW_GPMI_TIMING0 Data_Hold: in this example, Data_Hold = 2, tDH is equal to 2 dev_clk cycles
- tAS/tDS/tDH will extend, if the output data is not ready in device fifo.

Figure 9-21. Asynchronous Mode Basic Write Timing Diagram (command write, address write, or data write

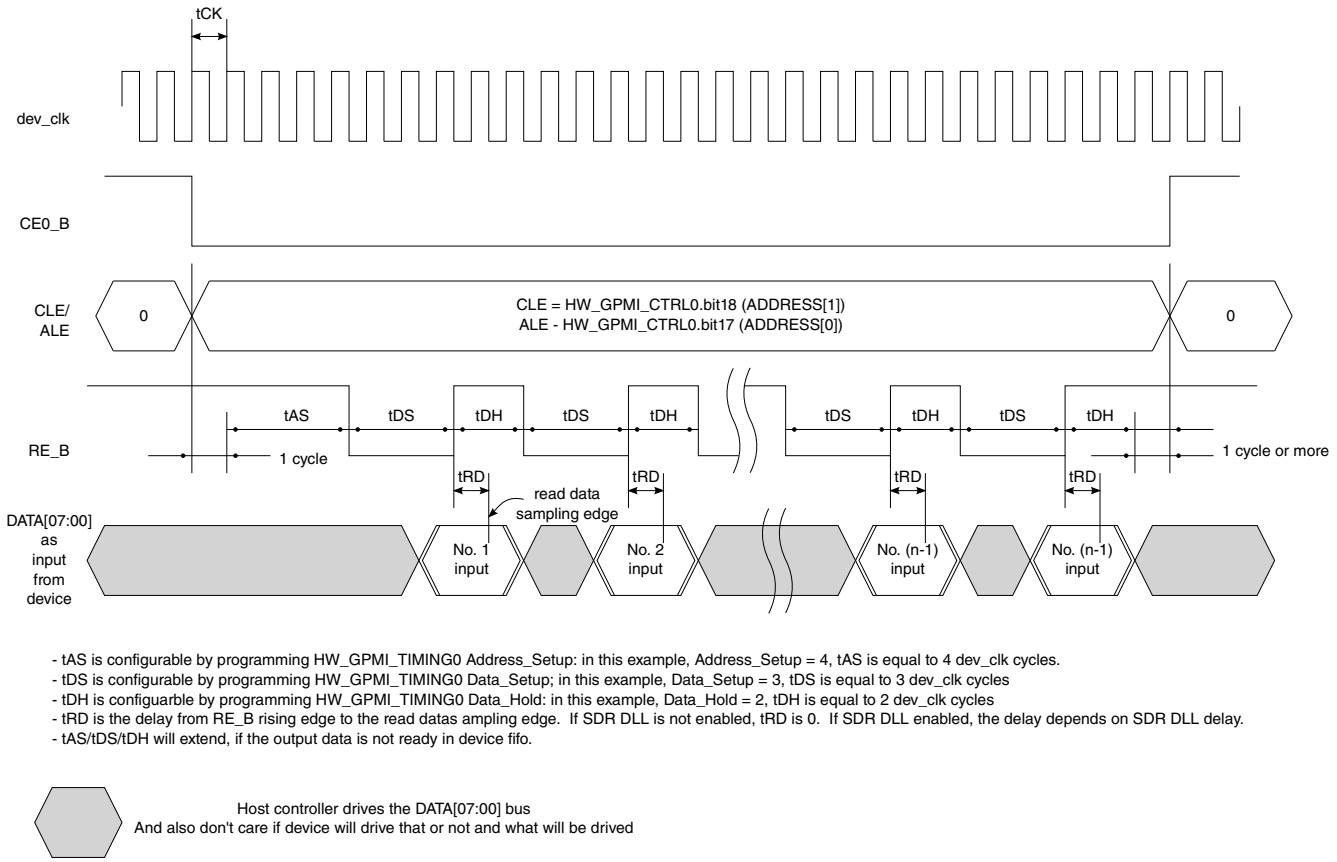


Figure 9-22. ONFI Asynchronous Mode Basic Read Timing Diagram (data read)

9.6.2.3.2 NAND Asynchronous EDO Mode Timing

In high-speed NANDS, the read data may not be valid until after the read strobe (NAND_RE_B) deasserts. This is the case when the minimum tDS is programmed to achieve higher bandwidth.

The GPMI implements a feedback read strobe to sample the read data. The feedback read strobe can be delayed to support fast nand EDO (Extended Data Out) timing where the read strobe may deassert before the read data is valid, and read data is valid for some time after read strobe. Nand EDO timings is applied typically for read cycle frequency above 33 MHz. See Figure 9-23.

The GPMI provides control over the amount of delay applied to the feedback read strobe. This delay depends on the maximum read access time (tREA) of the nand and the read pulse width (tRP) used to access the nand. tRP is specified by GPMI_TIMING0[DATA_SETUP] register. When (tREA + 4ns) is less than tRP, no

delay is required to sample to nand read data. (The 4ns provides adequate data setup time for the GPMI.) In this case set GPMI_CTRL1[HALF_PERIOD] = 0; GPMI_CTRL1[RDN_DELAY] = 0; GPMI_CTRL1[DLL_ENABLE] = 0.

When (tREA + 4ns) is greater than or equal to tRP, a delay of the feedback read strobe is required to sample to nand read data. This delay is equal to the difference between these two timings:

DELAY = tREA + 4ns - tRP.

Since the GPMI delay chain is limited to 6ns maximum, if DELAY > 6ns then increase tRP by increasing the value of GPMI_TIMING0[DATA_SETUP] until DELAY is less than or equal to 6ns.

The GPMI programming for this DELAY depends on the GPMICLK period. The GPMI DLL will not function properly if the GPMICLK period is greater than 12ns: disable the DLL if this is the case. If the GPMICLK period is greater than 6ns (and not greater than 12ns), set the GPMI_CTRL1[HALF_PERIOD]=1; This will cause the DLL reference period (RP) to be one-half of the GPMICLK period. If the GPMICLK period is 6ns or less then set the GPMI_CTRL1[HALF_PERIOD]=0; This will cause the DLL reference period (RP) to be equal to the GPMICLK period. DELAY is a multiple (0 to 1.875) of RP.

The GPMI_CTRL1[RDN_DELAY] is encoded as a 1-bit integer and 3-bit fraction delay factor. DELAY is a multiple of the delay factor and the reference period. See table below for details.

Table 9-11. RDN DELAY

HW_GPMI_CTRL1[RDN_DELAY]	Delay Factor
0	0.000
1	0.125
2	0.250
3	0.375
4	0.500
5	0.625
6	0.750
7	0.875
8	1.000
9	1.125
10	1.250
11	1.375
12	1.500
13	1.625

Table continues on the next page...

Table 9-11. RDN DELAY (continued)

HW_GPMI_CTRL1[RDN_DELAY]	Delay Factor
14	1.750
15	1.875

DELAY = DelayFactor x RP or DELAY = GPMI_CTRL1[RDN_DELAY] x 0.125 x RP.

Use this equation to calculate the value for GPMI_CTRL1[RDN_DELAY]. Then set GPMI_CTRL1[DLL_ENABLE]=1.

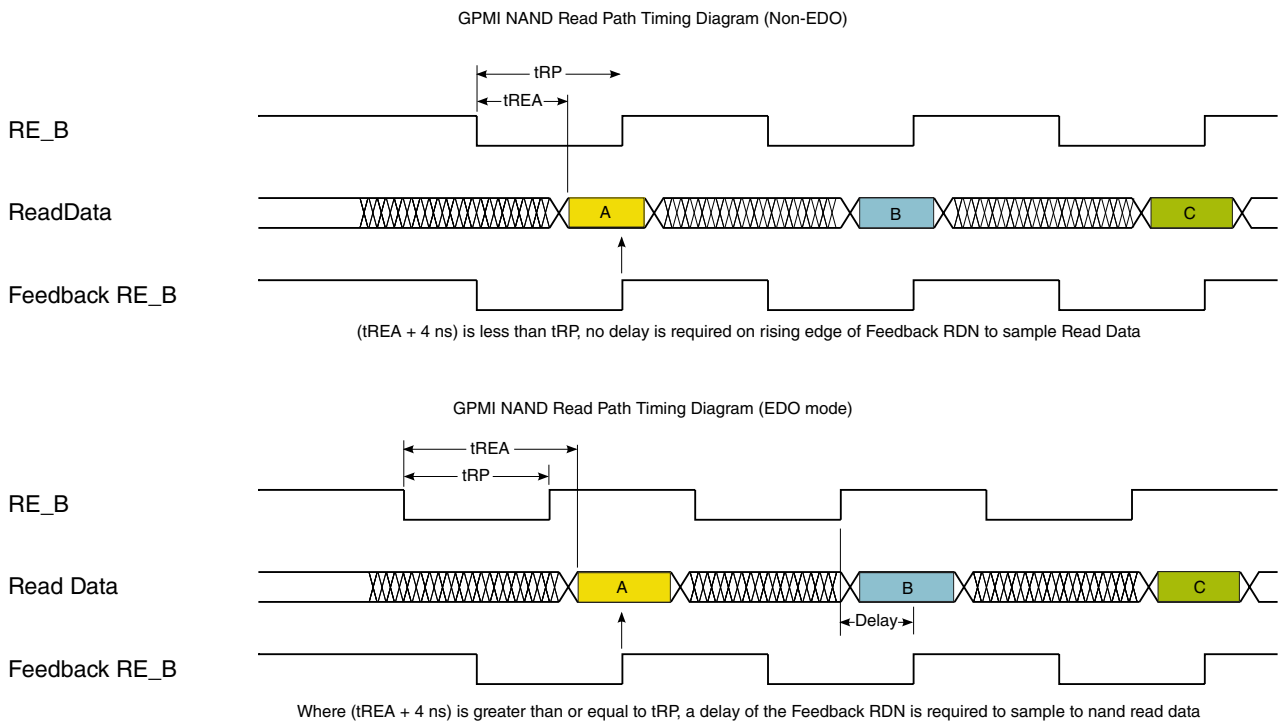


Figure 9-23. NAND Read Path Timing

For example, a NAND with tREAmax = 20 ns, tRPmin = 12 ns, and tRCmin = 25 ns (read cycle time) may be programmed as follows:

- GPMICLK clock frequency: Consider 480/6 = 80 MHz which is 12.5 ns clock period. This is too close to the minimum NAND spec if we program the data setup and hold to 1 GPMICLK cycle. Consider 480/7=68.57 MHz which is 14.58 ns clock period. With data setup and hold set to 1, we have a tRP of 14.58ns and a tRC of 29.16 ns (good margins).
- Since (tREA +4ns) is greater than tRP, required DELAY = tREA + 4ns - tRP = 20 + 4 - 14.58ns = 9.42 ns.

- GPMI_CTRL1[HALF_PERIOD] =, since GPMICLK period is ns. So RP=GPMICLK period = ns.
- DELAY = GPMI_CTRL1[RDN_DELAY] x 0.125 x RP. 9.42 ns = GPMI_CTRL1[RDN_DELAY] x 0.125 x ns. GPMI_CTRL1[RDN_DELAY] =

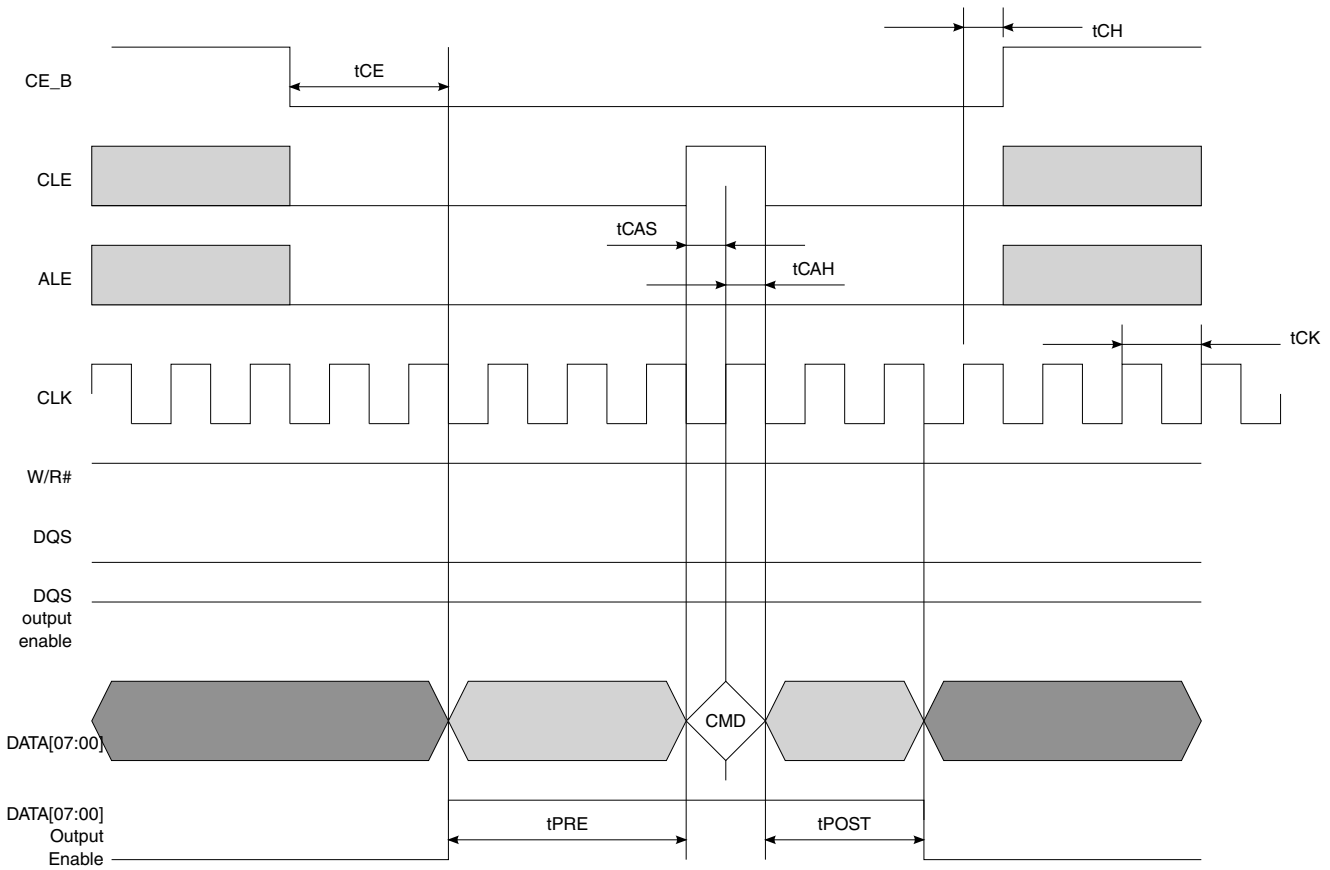
NOTE

It is recommended that the drive strength of NAND_RE_B and NAND_WE_B output pins be set to 8 mA. This will reduce the transition time under heavy loads. Low transition times will be important when NAND interface read and write cycle times are below 30 ns. The other GPMI pins may remain at 4 mA, since their frequency is only up to half that of NAND_RE_B and NAND_WE_B.

9.6.2.3.3 NAND ONFI Source Synchronous Mode Timing

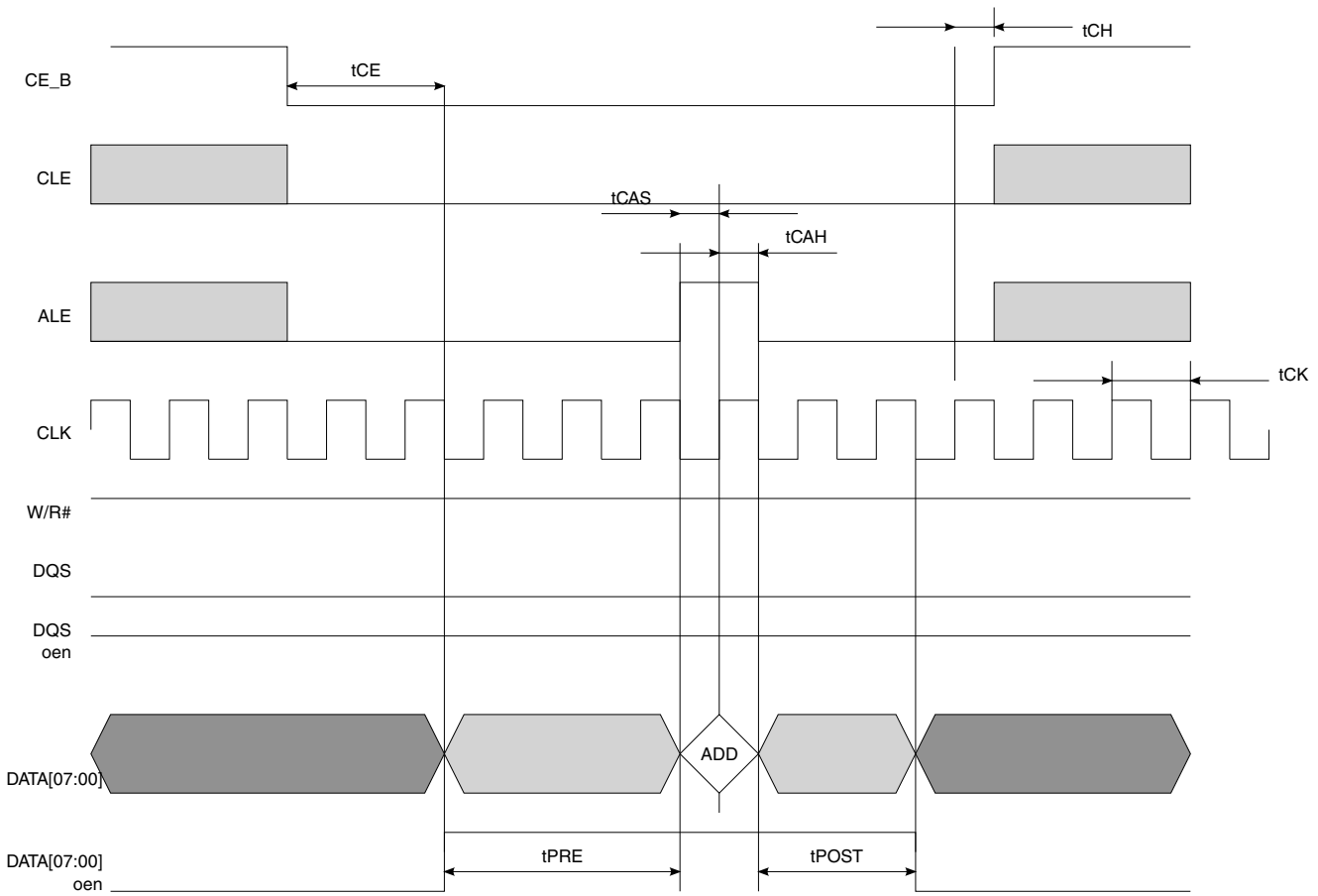
NOTE

In the following figures, CLK shares the same pin as WE_B in Async Mode. And W/R# shares the same pin as RE_B in Async Mode.



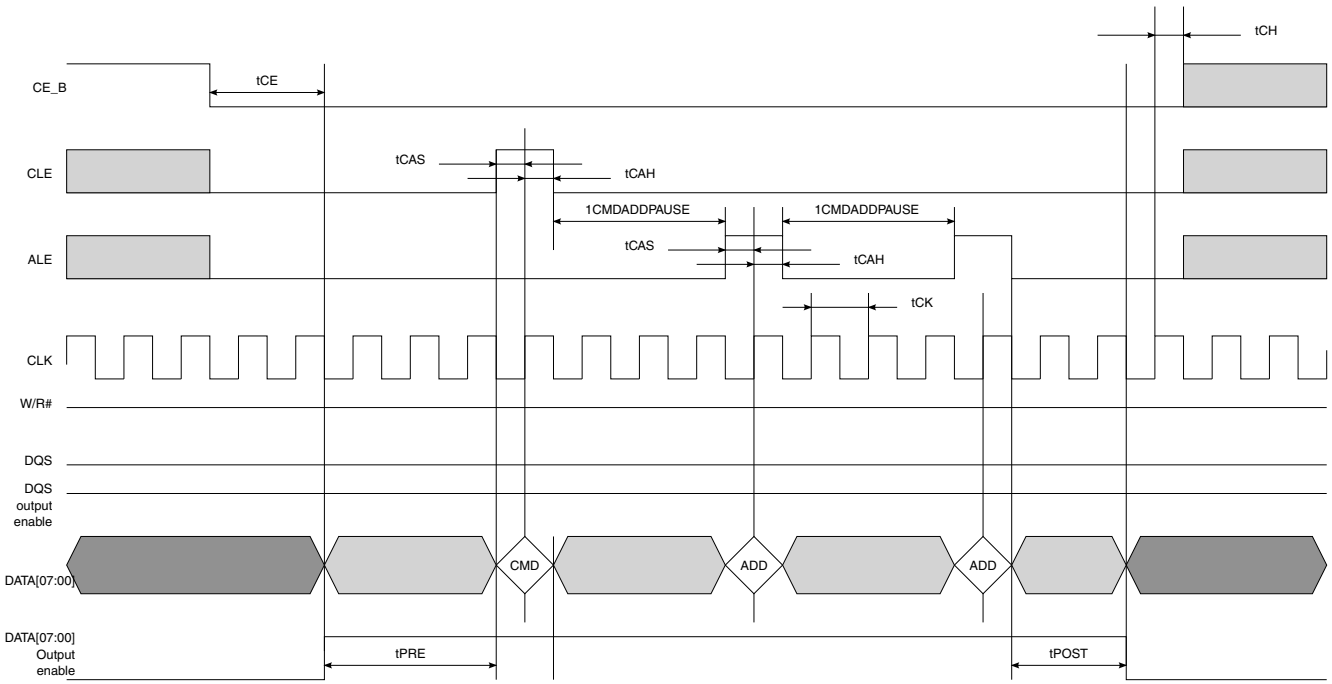
tCK is equal to device clock period
tCAS = 0.5 * tCK
tCAH = 0.5 * tCK
tCE = HW_GPMI_TIMING2.CE_DELAY * tCK
tCH = 0.5 * tCK
tPRE = HW_GPMI_TIMING2.PREAMBLE_DELAY * tCK
tPOST = HW_GPMI_TIMING2.POST_DELAY * tCK

Figure 9-24. ONFI Source Synchronous Mode Basic Command Write Timing Diagram



tCK is equal to device clock period
tCAS = 0.5 * tCK
tCAH = 0.5 * tCK
tCE = HW_GPMI_TIMING2.CE_DELAY * tCK
tCH = 0.5 * tCK
tPRE = HW_GPMI_TIMING2.PREAMBLE_DELAY * tCK
tPOST = HW_GPMI_TIMING2.POST_DELAY * tCK

Figure 9-25. ONFI Source Synchronous Mode Basic Address Write Timing Diagram

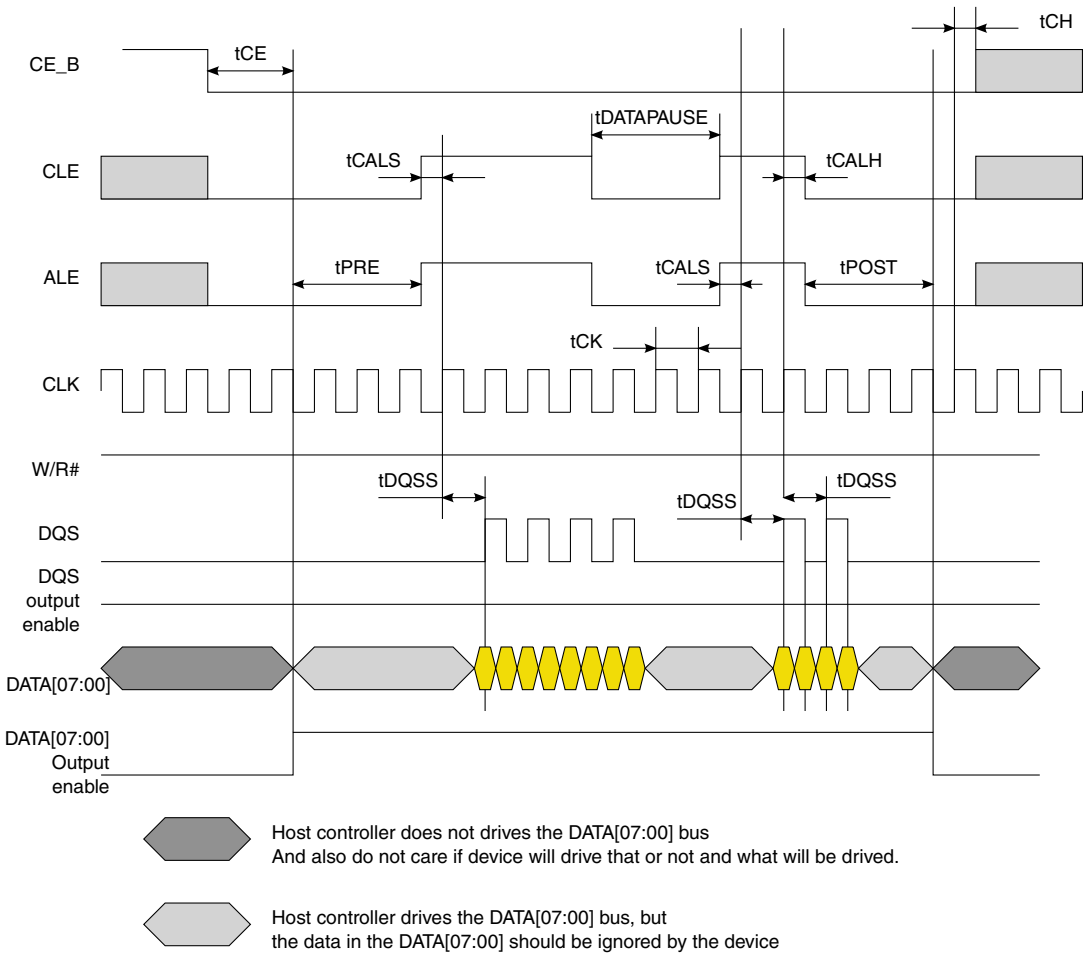


Host controller does not drives the DATA[07:00] bus
And also do not care if device will drive that or not and what will be drived.

Host controller drives the DATA[07:00] bus, but
the data in the DATA[07:00] should be ignored by the device

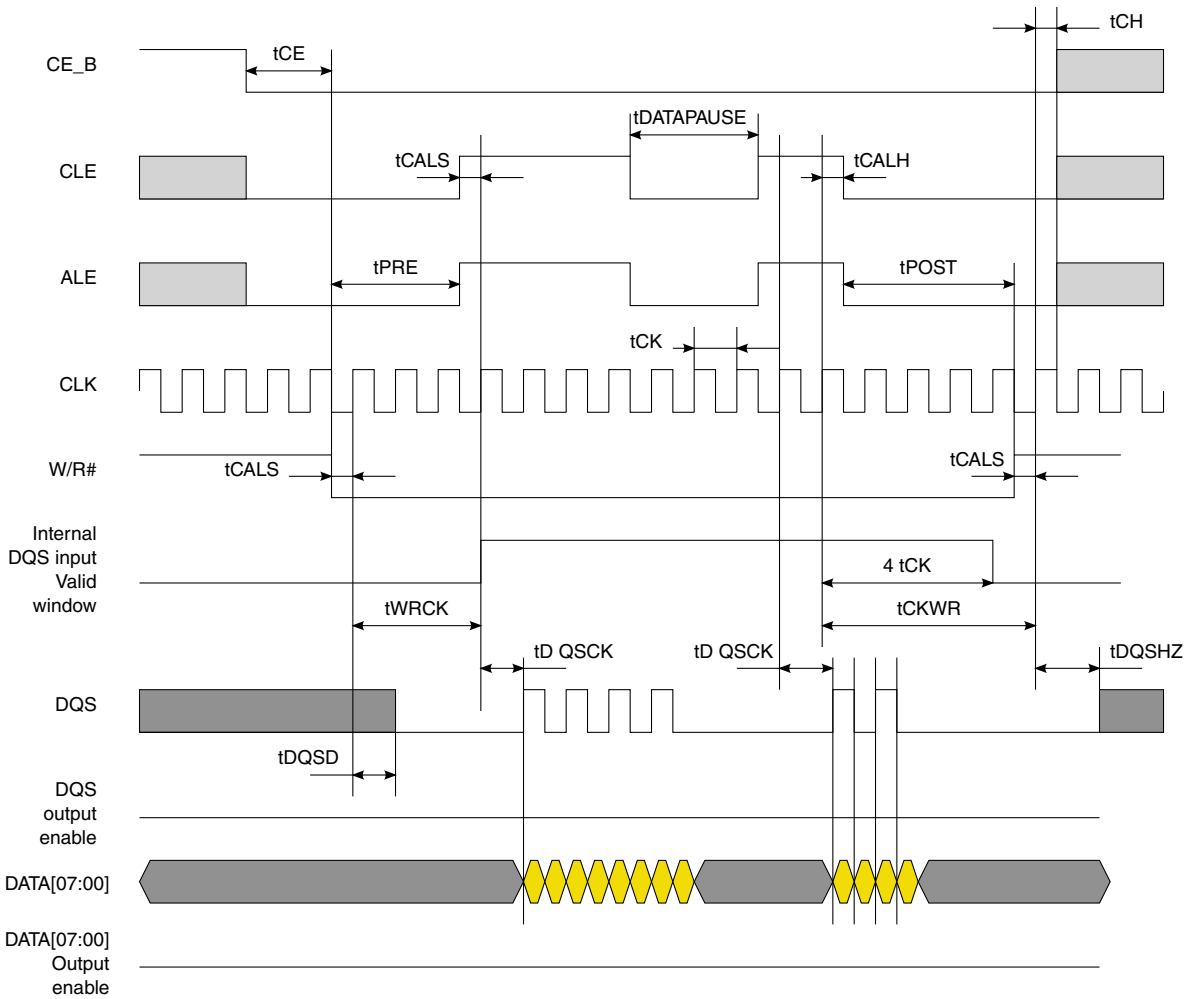
tCK is equal to device clock period
tCAS = 0.5 * tCK
tCAH = 0.5 *tCK
tCE = HW_GPMI_TIMING2.CE_DELAY * tCK
tCH = 0.5 * tCK
tPRE = HW_GPMI_TIMING2.PREAMBLE_DELAY * tCK
tPOST = HW_GPMI_TIMING2.POST_DELAY * tCK
tCMDADDPAUSE = HW_GPMI_TIMING2.CMDADD_PAUSE * tCK

Figure 9-26. ONFI Source Synchronous Mode Command + Address Write Timing Diagram



tCK is equal to device clock period
tCE = HW_GPI_TIMING2.CE_DELAY * tCK
tCH = 0.5 * tCK
tCALS = 0.5 * tCK
tCALH = 0.5 * tCK
tPRE = HW_GPMI_TIMING2.PREAMBLE_DELAY *tCK
tPOST = HW_GPMI_TIMING2.POST_DELAY * tCK
tDATAPAUSE = HW_GPMI_TIMING2.DATA_PAUSE * tCK
tDQSS = tCK

Figure 9-27. ONFI Source Synchronous Mode Data Write Timing Diagram



tCK is equal to device clock period
tCE = HW_GPI_TIMING2.CE_DELAY * tCK
tCH = 0.5 * tCK
tCALS = 0.5 * tCK
tCALH = 0.5 * tCK
tPRE = HW_GPMI_TIMING2.PREAMBLE_DELAY *tCK
tPOST = HW_GPMI_TIMING2.POST_DELAY * tCK
tDATAPAUSE = HW_GPMI_TIMING2.DATA_PAUSE * tCK
tWRCK/tCKWR/tDQSD/tDASCK/tDASHZ are device parameters

Figure 9-28. ONFI Source Synchronous Mode Data Read Timing Diagram

9.6.2.3.4 NAND Toggle Mode Timing

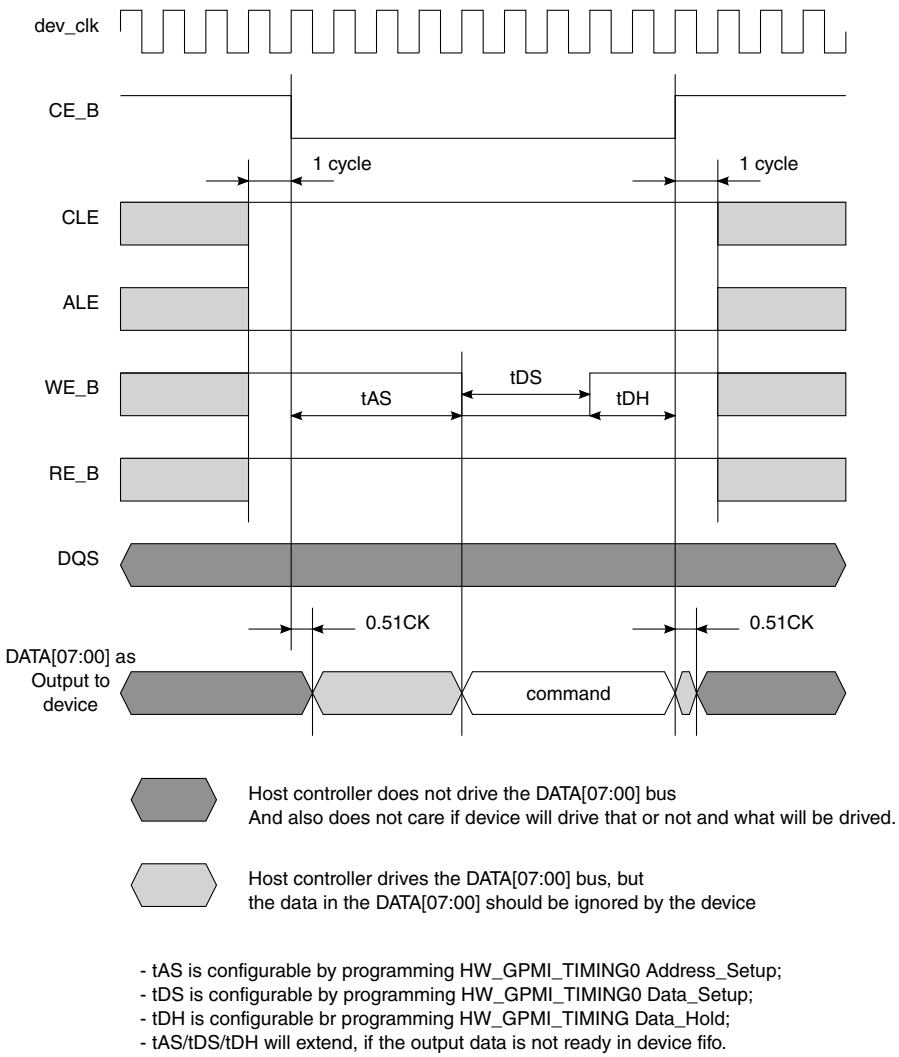


Figure 9-29. Samsung Toggle Mode Basic Command Write Timing Diagram

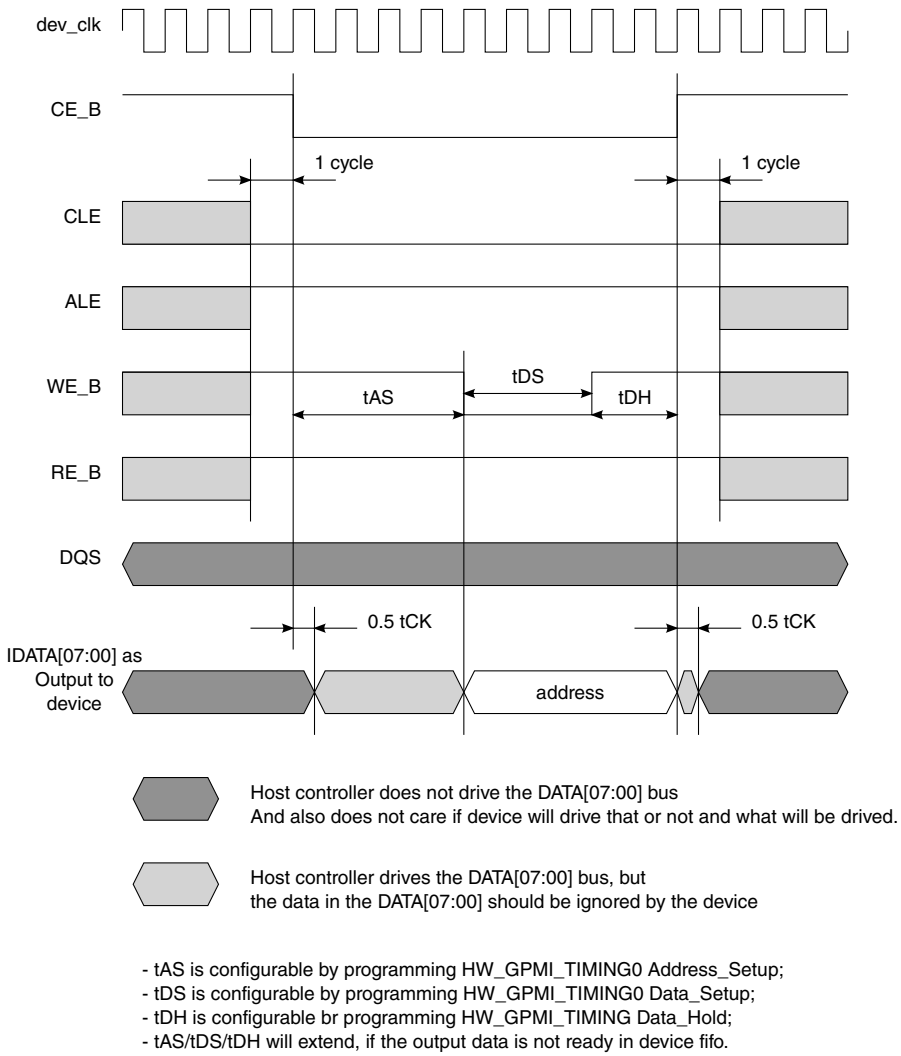


Figure 9-30. Samsung Toggle Mode Basic Address Write Timing Diagram

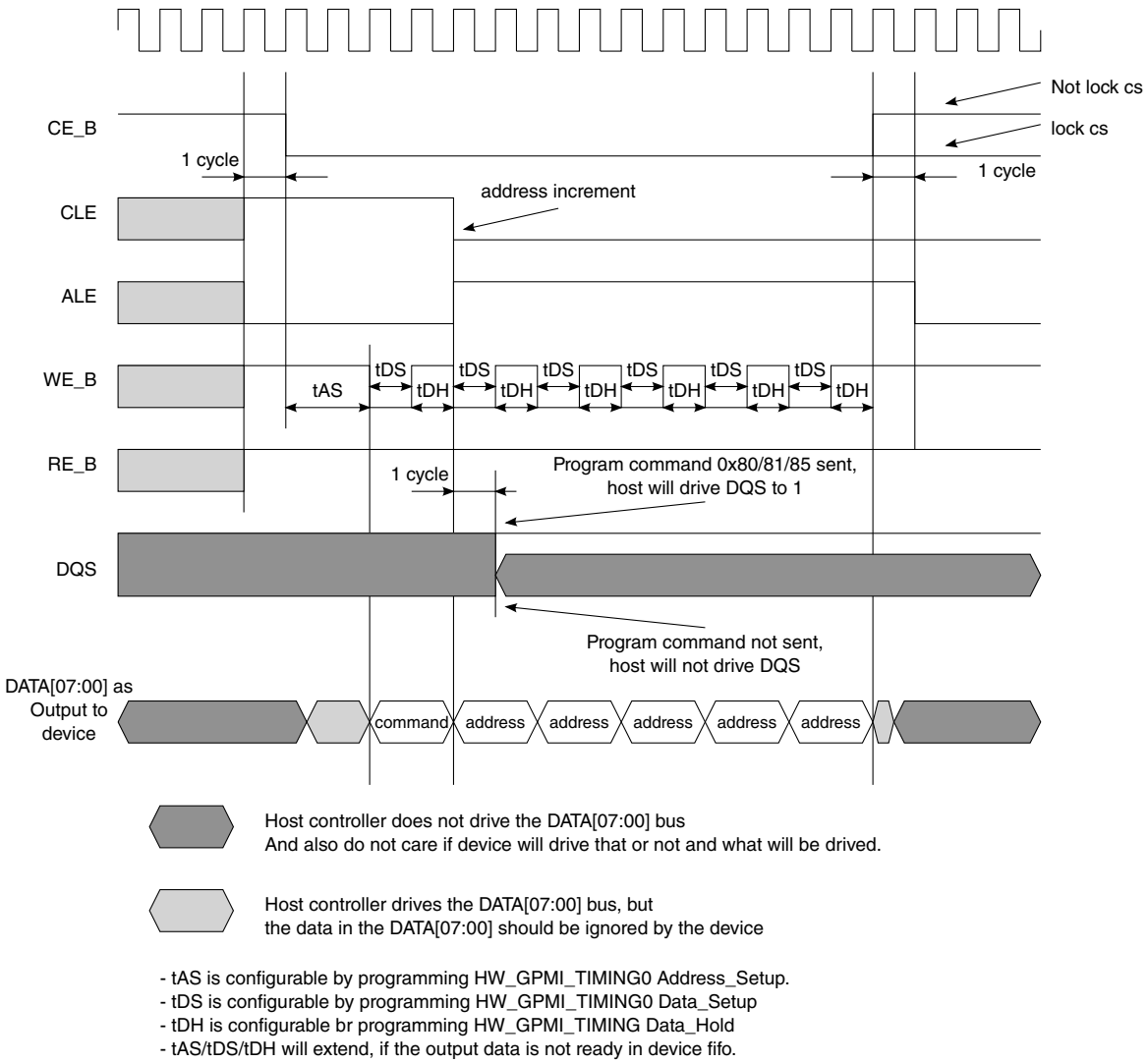


Figure 9-31. Samsung Toggle Mode Basic Command + Address Timing Diagram

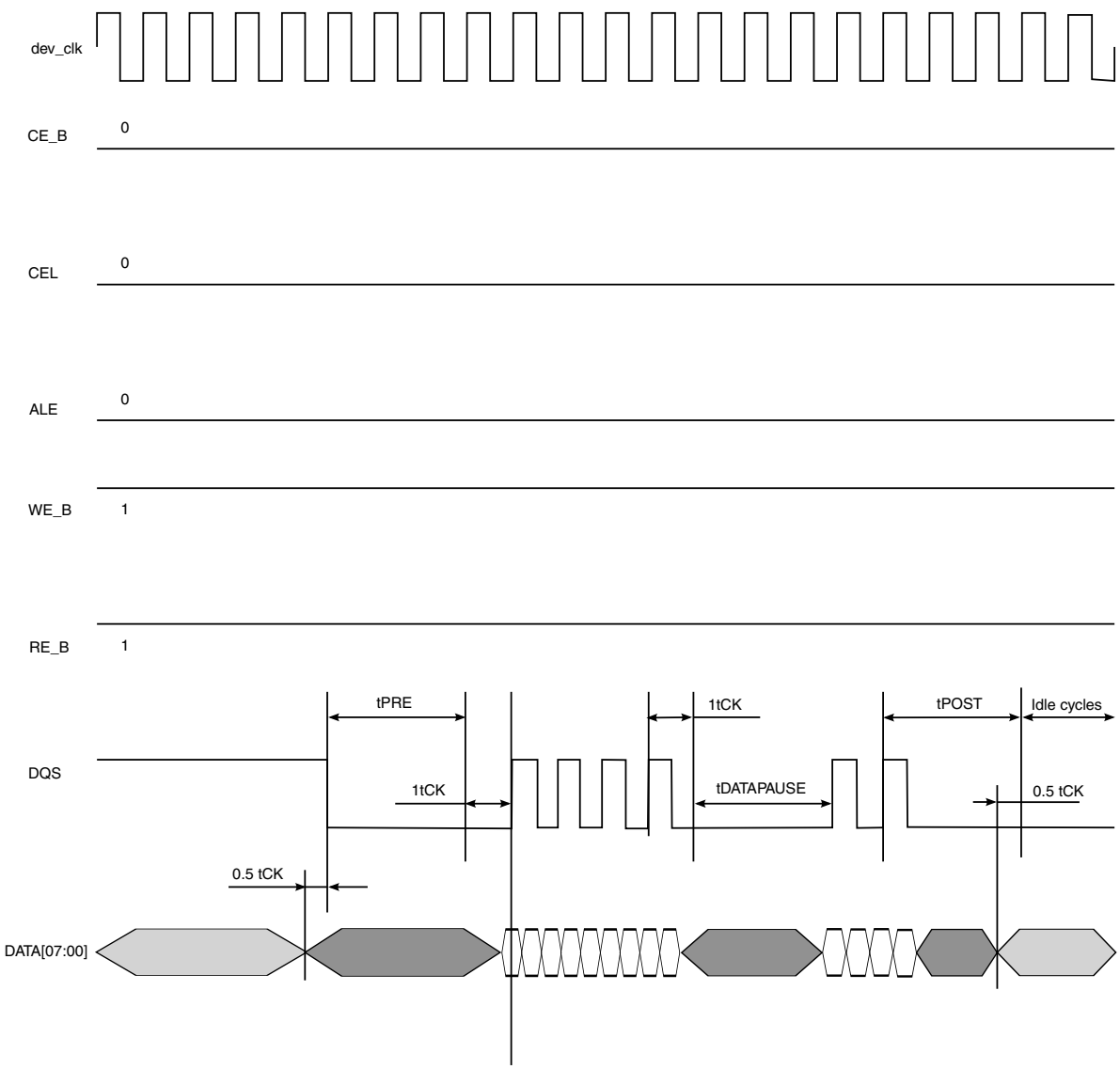
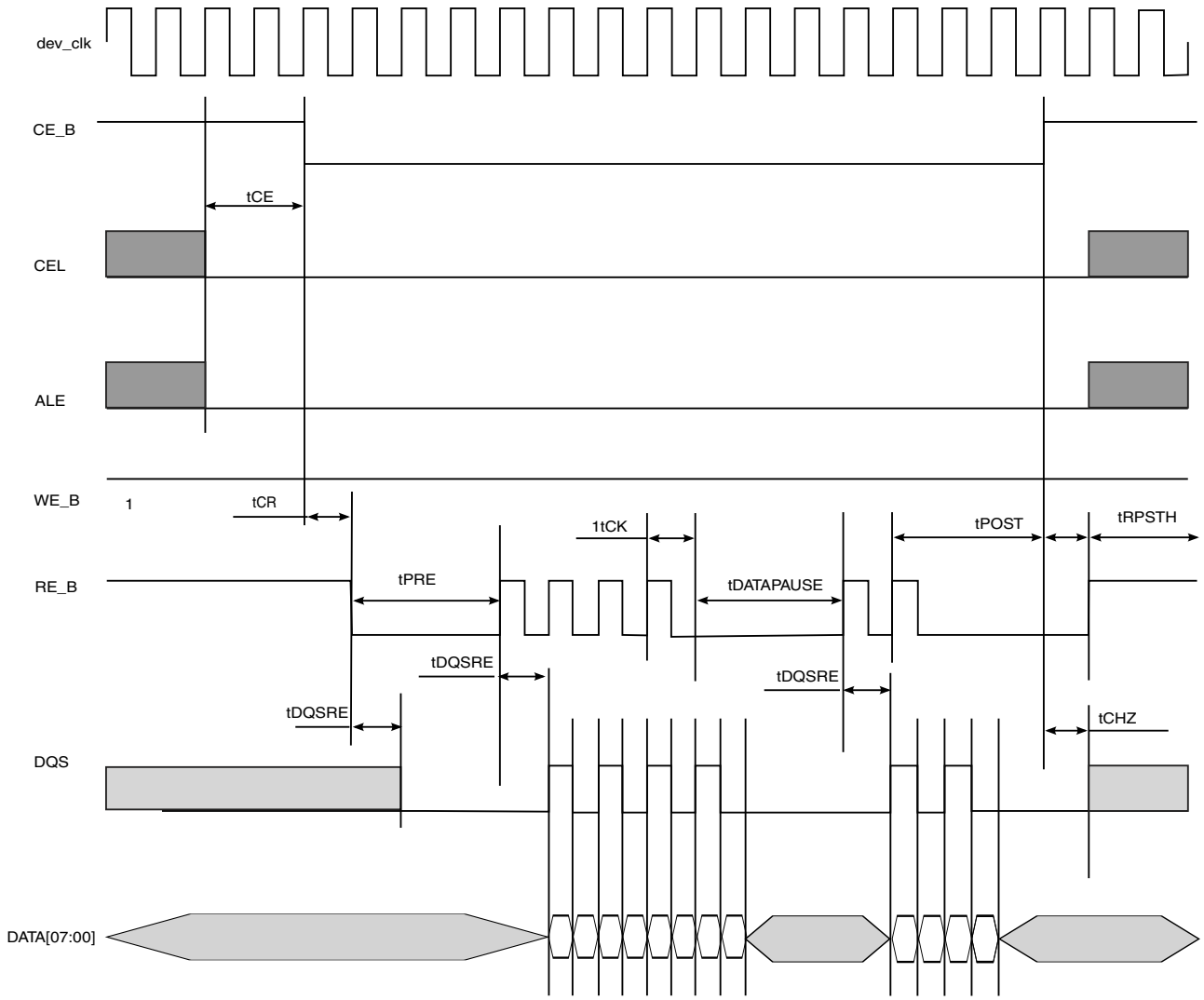


Figure 9-32. Toggle Mode Data Write Timing Diagram



Host controller does not drive the DATA[07:00] bus.
It also does not care if the device will drive or what will be driven.

Host controller drives the DATA[07:00] bus, but
the data in the DATA[07:00] should be ignored by device.

tCK is equal to device clock period

tCE = HW_GPMI_TIMING2.CE_DELAY* tCK

tPRE = (HW_GPMI_TIMING2.PREAMBLE_DELAY - HW_GPMI_TIMING2.TCR) * tCK

tPOST = HW_GPMI_TIMING2.POSTAMBLE_DELAY* tCK

tDATAPAUSE = HW_GPMI_TIMING2.DATA_PAUSE* tCK

tCR = (HW_GPMI_TIMING2.TCR + 1) * tCK

tRPSTH = HW_GPMI_TIMING2.TRPSTH * tCK

Figure 9-33. Toggle Mode Data Read Timing Diagram

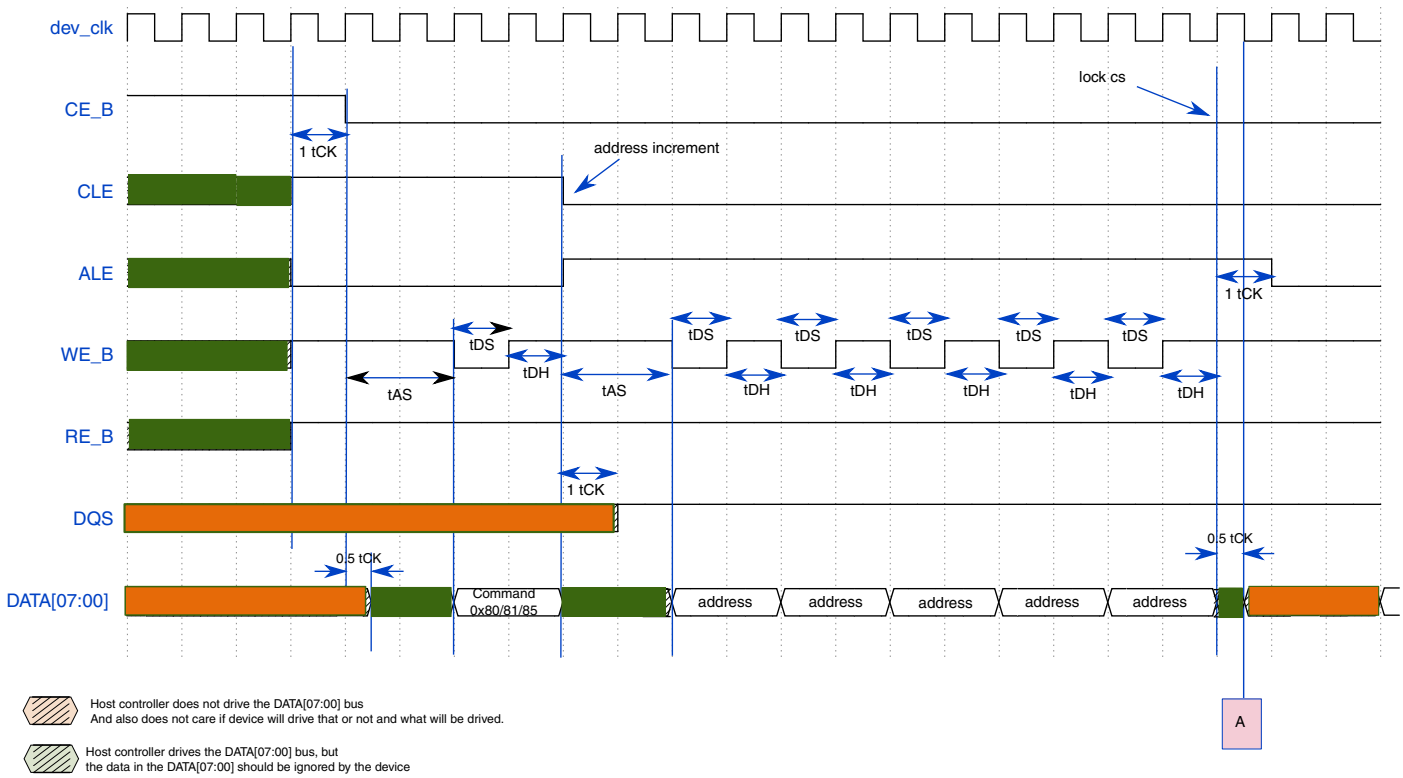


Figure 9-34. Toggle Mode Program Timing Diagram (A)

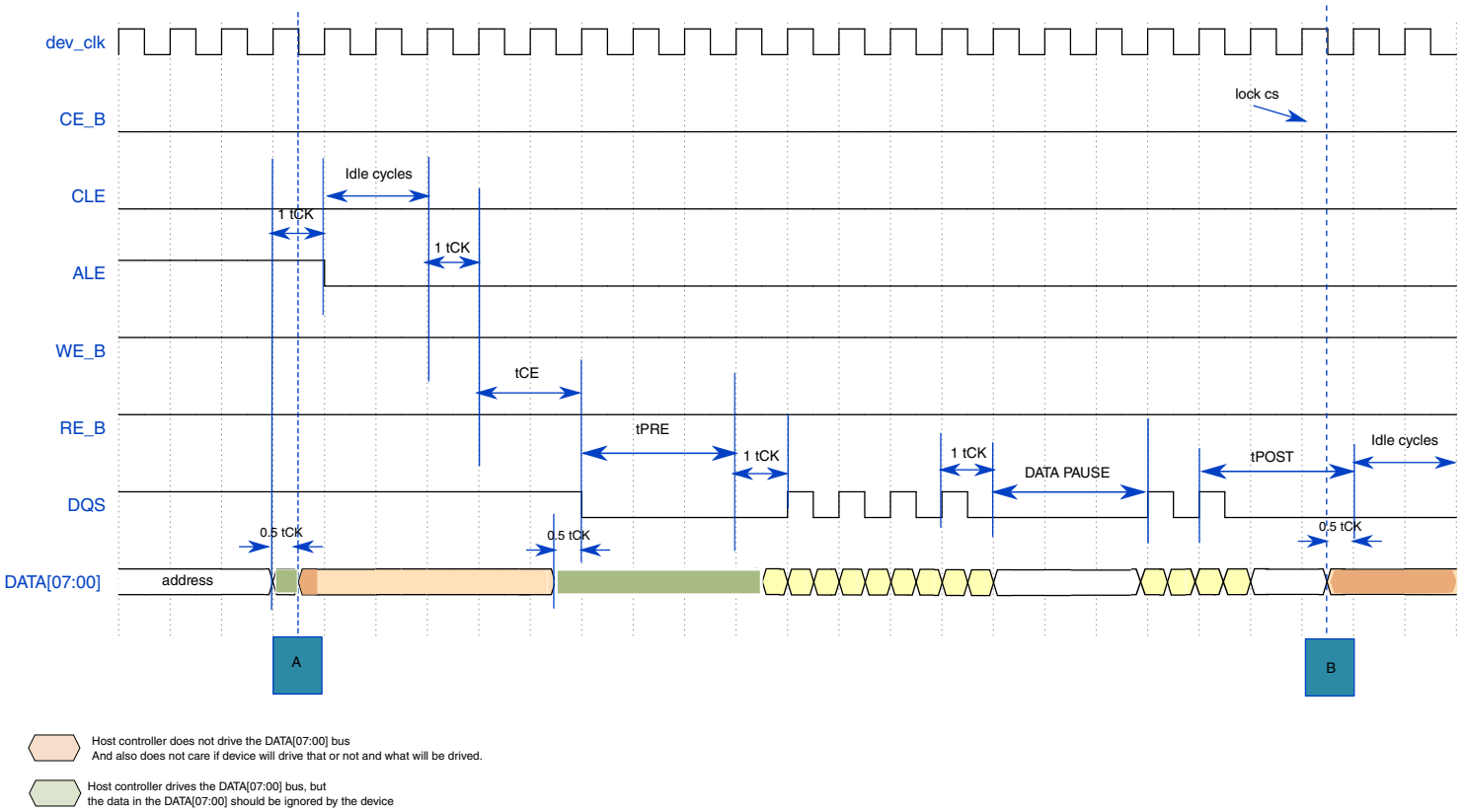


Figure 9-35. Toggle Mode Program Timing Diagram (B)

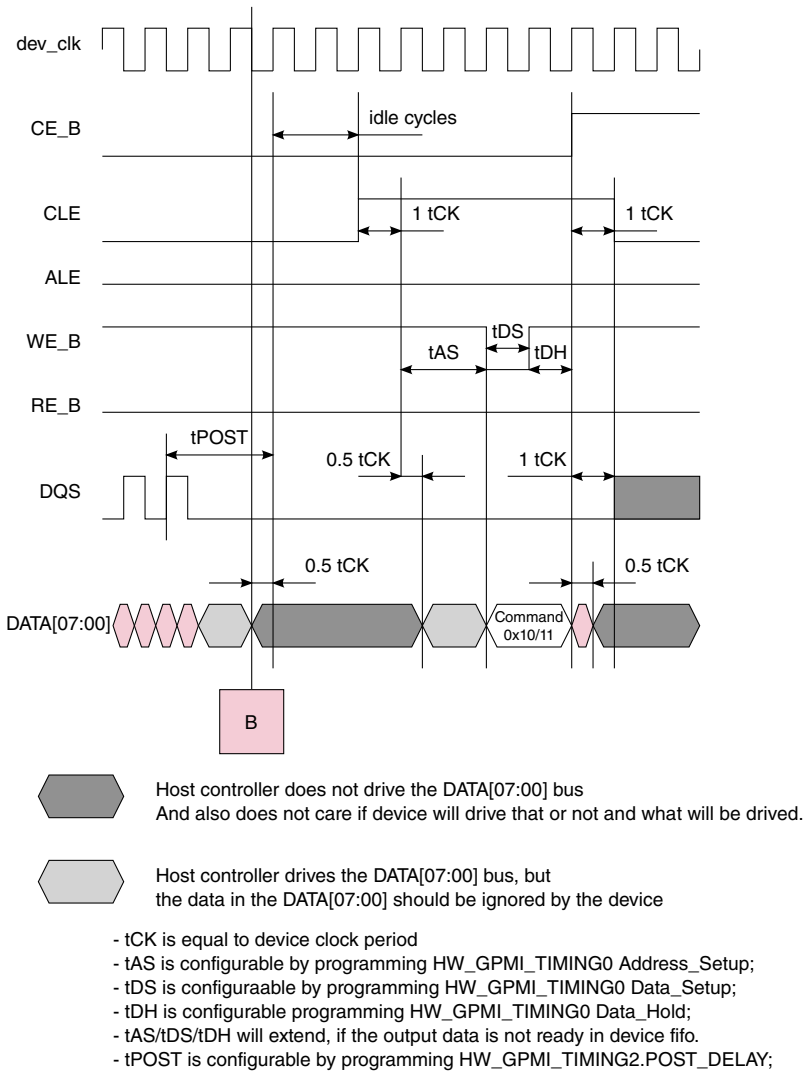


Figure 9-36. Toggle Mode Program Timing Diagram (C)

9.6.2.4 Hardware BCH Interface

The GPMI provides an interface to the BCH module. This reduces the SOC bus traffic and the software involvement.

The GPMI also provides an interface to the Randomizer module. The Randomizer can generate random data based on BCH ECC encoded / decoded data. It can be employed to reduce the disturbances caused by neighboring cells in the NAND chip, thus reducing bit errors. To enable the Randomizer module, set GPMI_ECCCTRL[RANDOMIZER_ENABLE] to 1, then set GPMI_ECCCOUNT[RANDOMIZER_PAGE] to select randomizer page number needed

to be randomized. All these registers can be programmed by the DMA chain. The randomized data should start from the zero column address and be the size of the whole NAND page. If the randomizer function is enabled, GPMI_ECCCTRL[ENABLE_ECC] should also be enabled. To bypass BCH error correction function, set BCH_FLASHxLAYOUT0[ECC0] and BCH_FLASHxLAYOUT1[ECCN] to 0.

When BCH ECC is enable, parity information is inserted on-the-fly during writes to 8-bit NAND devices. The BCH will supply payload and parity to the GPMI to write to the NAND. During NAND reads, parity is checked and ECC processing is performed after each read block. In this case the GPMI reads the NAND device and redirects the data and parity to the BCH module for ECC processing.

To program the BCH for NAND writes, remove the soft reset and clock gates from BCH_CTRL[SFTRST] and BCH_CTRL[CLKGATE]. The bulk of BCH programming is actually applied to the GPMI via PIO operations embedded in its DMA command structures. This has a subtle implication when writing to the GPMI ECC registers: access to the these registers must be written in progressive register order. Thus, to write to the GPMI_ECCCOUNT register, write first (in order) to registers GPMI_CTRL0, GPMI_COMPARE, and GPMI_ECCCTRL before writing to GPMI_ECCCOUNT. These additional register writes need to be accounted for in the CMDWORDS field of the respective DMA channel command register.

NOTE

Note that the GPMI_PAYLOAD and GPMI_AUXILIARY pointers need to be word-aligned for proper ECC operation. If those pointers are non-word-aligned, then the BCH engine will not operate properly and could possibly corrupt system memory in the adjoining memory regions.

9.6.3 Behavior During Reset

A soft reset (SFTRST) can take multiple clock periods to complete, so do NOT set CLKGATE when setting SFTRST. The reset process gates the clocks automatically.

9.6.4 GPMI Memory Map/Register Definition

The following registers provide control for programmable elements of the GPMI module.

NOTE

All ATA or UDMA features are not supported for the chip.

NOTE

GPMI does not support the Set feature command in Toggle mode. The NANDF_DQS output is only enabled in program operation for Toggle mode, but the Set feature command also needs to use the NANDF_DQS signal to write data to Toggle NAND flash. So the Set feature command in Toggle mode is not supported.

GPMI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3300_2000	GPMI Control Register 0 Description (GPMI_CTRL0)	32	R/W	C000_0000h	9.6.4.1/ 2310
3300_2004	GPMI Control Register 0 Description (GPMI_CTRL0_SET)	32	R/W	C000_0000h	9.6.4.1/ 2310
3300_2008	GPMI Control Register 0 Description (GPMI_CTRL0_CLR)	32	R/W	C000_0000h	9.6.4.1/ 2310
3300_200C	GPMI Control Register 0 Description (GPMI_CTRL0_TOG)	32	R/W	C000_0000h	9.6.4.1/ 2310
3300_2010	GPMI Compare Register Description (GPMI_COMPARE)	32	R/W	0000_0000h	9.6.4.2/ 2312
3300_2020	GPMI Integrated ECC Control Register Description (GPMI_ECCCTRL)	32	R/W	0000_0000h	9.6.4.3/ 2313
3300_2024	GPMI Integrated ECC Control Register Description (GPMI_ECCCTRL_SET)	32	R/W	0000_0000h	9.6.4.3/ 2313
3300_2028	GPMI Integrated ECC Control Register Description (GPMI_ECCCTRL_CLR)	32	R/W	0000_0000h	9.6.4.3/ 2313
3300_202C	GPMI Integrated ECC Control Register Description (GPMI_ECCCTRL_TOG)	32	R/W	0000_0000h	9.6.4.3/ 2313
3300_2030	GPMI Integrated ECC Transfer Count Register Description (GPMI_ECCCOUNT)	32	R/W	0000_0000h	9.6.4.4/ 2314
3300_2040	GPMI Payload Address Register Description (GPMI_PAYLOAD)	32	R/W	0000_0000h	9.6.4.5/ 2315
3300_2050	GPMI Auxiliary Address Register Description (GPMI_AUXILIARY)	32	R/W	0000_0000h	9.6.4.6/ 2315

Table continues on the next page...

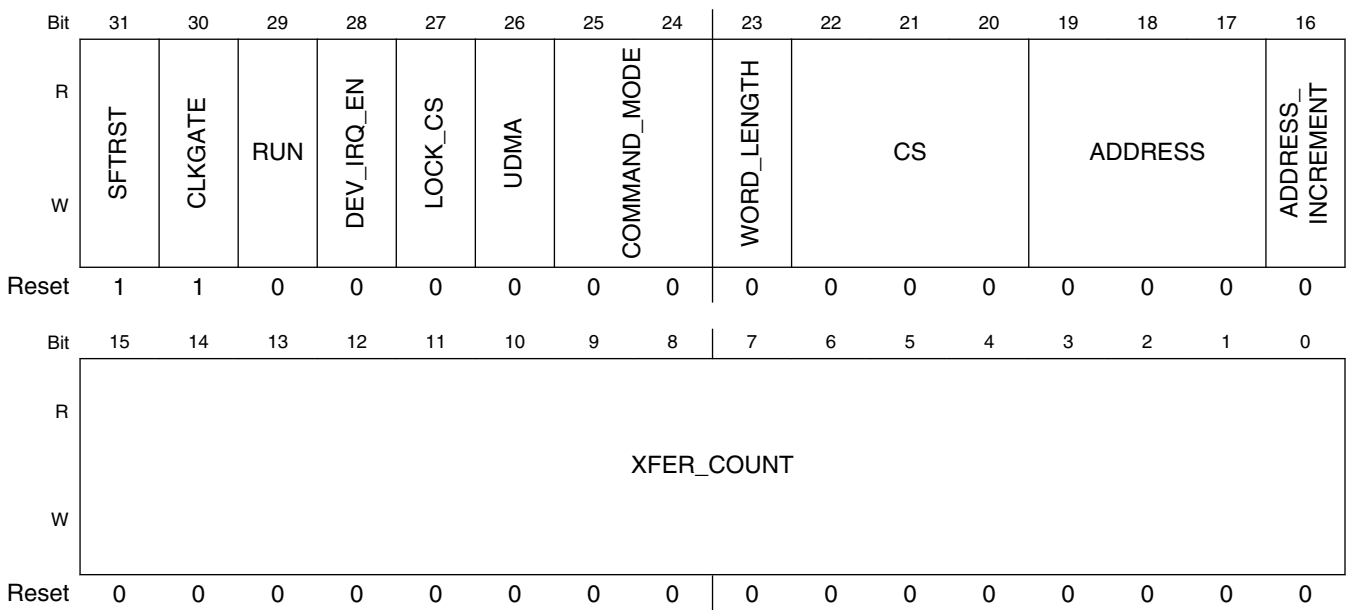
GPMI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3300_2060	GPMI Control Register 1 Description (GPMI_CTRL1)	32	R/W	0004_0004h	9.6.4.7/ 2316
3300_2064	GPMI Control Register 1 Description (GPMI_CTRL1_SET)	32	R/W	0004_0004h	9.6.4.7/ 2316
3300_2068	GPMI Control Register 1 Description (GPMI_CTRL1_CLR)	32	R/W	0004_0004h	9.6.4.7/ 2316
3300_206C	GPMI Control Register 1 Description (GPMI_CTRL1_TOG)	32	R/W	0004_0004h	9.6.4.7/ 2316
3300_2070	GPMI Timing Register 0 Description (GPMI_TIMING0)	32	R/W	0001_0203h	9.6.4.8/ 2319
3300_2080	GPMI Timing Register 1 Description (GPMI_TIMING1)	32	R/W	0000_0000h	9.6.4.9/ 2319
3300_2090	GPMI Timing Register 2 Description (GPMI_TIMING2)	32	R/W	0302_3336h	9.6.4.10/ 2320
3300_20A0	GPMI DMA Data Transfer Register Description (GPMI_DATA)	32	R/W	0000_0000h	9.6.4.11/ 2321
3300_20B0	GPMI Status Register Description (GPMI_STAT)	32	R	0000_0005h	9.6.4.12/ 2321
3300_20C0	GPMI Debug Information Register Description (GPMI_DEBUG)	32	R	0000_0000h	9.6.4.13/ 2324
3300_20D0	GPMI Version Register Description (GPMI_VERSION)	32	R	0502_0000h	9.6.4.14/ 2325
3300_20E0	GPMI Debug2 Information Register Description (GPMI_DEBUG2)	32	R/W	0000_F100h	9.6.4.15/ 2325
3300_20F0	GPMI Debug3 Information Register Description (GPMI_DEBUG3)	32	R	0000_0000h	9.6.4.16/ 2328
3300_2100	GPMI Double Rate Read DLL Control Register Description (GPMI_READ_DDR_DLL_CTRL)	32	R/W	0000_0038h	9.6.4.17/ 2329
3300_2110	GPMI Double Rate Write DLL Control Register Description (GPMI_WRITE_DDR_DLL_CTRL)	32	R/W	0000_0038h	9.6.4.18/ 2330
3300_2120	GPMI Double Rate Read DLL Status Register Description (GPMI_READ_DDR_DLL_STS)	32	R	0000_0000h	9.6.4.19/ 2332
3300_2130	GPMI Double Rate Write DLL Status Register Description (GPMI_WRITE_DDR_DLL_STS)	32	R	0000_0000h	9.6.4.20/ 2333

9.6.4.1 GPMI Control Register 0 Description (GPMI_CTRL0*n*)

The GPMI control register 0 specifies the GPMI transaction to perform for the current command chain item.

Address: 3300_2000h base + 0h offset + (4d × i), where i=0d to 3d



GPMI_CTRL0*n* field descriptions

Field	Description
31 SFTRST	Set to zero for normal operation. When this bit is set to one (default), then the entire block is held in its reset state. This will not work if the CLKGATE bit is already set to '1'. CLKGATE must be cleared to '0' before issuing a soft reset. Also the GPMICLK must be running for this to work properly. RUN = 0x0 Allow GPMI to operate normally. RESET = 0x1 Hold GPMI in reset.
30 CLKGATE	Set this bit zero for normal operation. Setting this bit to one (default), gates all of the block level clocks off for miniminizing AC energy consumption. RUN = 0x0 Allow GPMI to operate normally. NO_CLKS = 0x1 Do not clock GPMI gates in order to minimize power consumption.
29 RUN	The GPMI is busy running a command whenever this bit is set to '1'. The GPMI is idle whenever this bit set to zero. This can be set to one by a CPU write. In addition, the DMA sets this bit each time a DMA command has finished its PIO transfer phase. IDLE = 0x0 The GPMI is idle. BUSY = 0x1 The GPMI is busy running a command.

Table continues on the next page...

GPMI_CTRL0*n* field descriptions (continued)

Field	Description
28 DEV_IRQ_EN	When set to '1' and ATA_IRQ pin is asserted, the GPMI_IRQ output will assert.
27 LOCK_CS	For ATA/NAND mode: 0= Deassert chip select (CS) after RUN is complete. 1= Continue to assert chip select (CS) after RUN is complete. For Camera Mode: 0= Dont wait for VSYNC rising edge before capturing data. 1= Wait for VSYNC rising edge before capturing data (Camera mode only). DISABLED = 0x0 Deassert chip select (CS) after RUN is complete. ENABLED = 0x1 Continue to assert chip select (CS) after RUN is complete.
26 UDMA	DISABLED = 0x0 Use ATA-PIO mode on the external bus. ENABLED = 0x1 Use ATA-Ultra DMA mode on the external bus. 0 Use ATA-PIO mode on the external bus. 1 Use ATA-Ultra DMA mode on the external bus.
25–24 COMMAND_ MODE	WRITE = 0x0 Write mode. READ = 0x1 Read mode. READ_AND_COMPARE = 0x2 Read and Compare mode (setting sense flop). WAIT_FOR_READY = 0x3 Wait for Ready mode. For ATA WAIT_FOR_READY command set CS=01. 00 Write mode. 01 Read Mode. 10 Read and Compare Mode (setting sense flop). 11 Wait for Ready.
23 WORD_LENGTH	This bit should only be changed when RUN==0. Reserve = 0x0 Reserved. 8_BIT = 0x1 8-bit Data Bus mode. 0 Reserved. 1 8-bit Data Bus mode.
22–20 CS	Selects which chip select is active for this command. For ATA WAIT_FOR_READY command, this must be set to b01.
19–17 ADDRESS	Specifies the three address lines for ATA mode. In NAND mode, use A0 for CLE and A1 for ALE. NAND_DATA = 0x0 In NAND mode, this address is used to read and write data bytes. NAND_CLE = 0x1 In NAND mode, this address is used to write command bytes. NAND_ALE = 0x2 In NAND mode, this address is used to write address bytes.
16 ADDRESS_ INCREMENT	In ATA mode, the address will increment with each cycle. In NAND mode, the address will increment once, after the first cycle (going from CLE to ALE). DISABLED = 0x0 Address does not increment. ENABLED = 0x1 Increment address. 0 Address does not increment. 1 Increment address.
XFER_COUNT	Number of bytes to transfer for this command. A value of zero will transfer 64K bytes.

9.6.4.2 GPMI Compare Register Description (GPMI_COMPARE)

The GPMI compare register specifies the expect data and the xor mask for comparing to the status values read from the device. This register is used by the Read and Compare command.

GPMI_COMPARE 0x010

Address: 3300_2000h base + 10h offset = 3300_2010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MASK																REFERENCE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

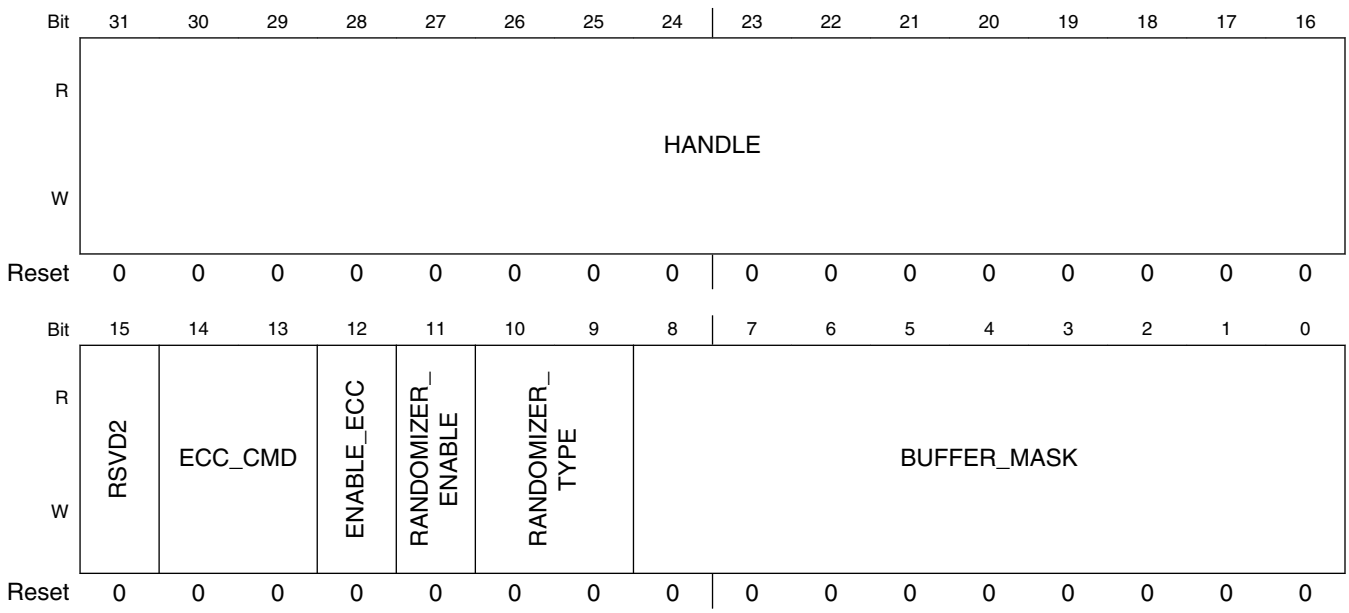
GPMI_COMPARE field descriptions

Field	Description
31–16 MASK	16-bit mask which is applied after the read data is XORed with the REFERENCE bit field.
REFERENCE	16-bit value which is XORed with data read from the NAND device.

9.6.4.3 GPMI Integrated ECC Control Register Description (GPMI_ECCCTRL*n*)

The GPMI ECC control register handles configuration of the integrated ECC / Randomizer accelerator.

Address: 3300_2000h base + 20h offset + (4d × i), where i=0d to 3d



GPMI_ECCCTRL*n* field descriptions

Field	Description
31–16 HANDLE	This is a register available to software to attach an identifier to a transaction in progress. This handle will be available from the ECC register space when the completion interrupt occurs.
15 RSVD2	Always write zeroes to this bit field.
14–13 ECC_CMD	ECC Command information. DECODE = 0x0 Decode. ENCODE = 0x1 Encode. RESERVE2 = 0x2 Reserved. RESERVE3 = 0x3 Reserved.
12 ENABLE_ECC	Enable ECC processing of GPMI transfers. ENABLE = 0x1 Use integrated ECC for read and write transfers. DISABLE = 0x0 Integrated ECC remains in idle.
11 RANDOMIZER_ ENABLE	Enable randomizer function. If this bit is set to enable, ENABLE_ECC should be also enable.

Table continues on the next page...

GPMI_ECCCTRL*n* field descriptions (continued)

Field	Description
	0x0 disable 0x1 enable
10–9 RANDOMIZER_TYPE	Set randomizer type 0x00 Type 0 0x01 Type 1 0x10 Type 2
BUFFER_MASK	ECC buffer information. The BCH error correction only allows two configurations of the buffer mask - software may either read just the first block on the flash page or the entire flash page. Write operations must be for the entire flash page. Invalid buffer mask values will cause the DMA descriptor command to be terminated. BCH_AUXONLY = 0x100 Set to request transfer from only the auxiliary buffer (block 0 on flash). BCH_PAGE = 0x1FF Set to request transfer to/from the entire page.

9.6.4.4 GPMI Integrated ECC Transfer Count Register Description (GPMI_ECCCOUNT)

The GPMI ECC Transfer Count Register contains the count of bytes that flow through the ECC / Randomizer subsystem.

GPMI_ECCCOUNT 0x030

Address: 3300_2000h base + 30h offset = 3300_2030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RANDOMIZER_PAGE								COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPMI_ECCCOUNT field descriptions

Field	Description
31–24 -	Always write zeroes to this bit field.
23–16 RANDOMIZER_PAGE	Set NAND page number needed to be randomized. The value is between 0-255
COUNT	Number of bytes to pass through ECC. This is the GPMI transfer count plus the syndrome count that will be inserted into the stream by the ECC. In DMA2ECC_MODE this count must match the GPMI_CTRL0_XFER_COUNT. A value of zero will transfer 64K words.

9.6.4.5 GPMI Payload Address Register Description (GPMI_PAYLOAD)

The GPMI payload address register specifies the location of the data buffers in system memory. This value must be word aligned.

GPMI_PAYLOAD 0x040

Address: 3300_2000h base + 40h offset = 3300_2040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ADDRESS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDRESS														RSVD0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPMI_PAYLOAD field descriptions

Field	Description
31–2 ADDRESS	Pointer to an array of one or more 512 byte payload buffers.
RSVD0	Always write zeroes to this bit field.

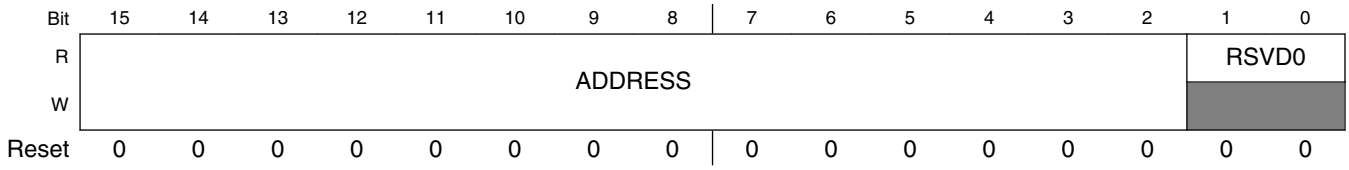
9.6.4.6 GPMI Auxiliary Address Register Description (GPMI_AUXILIARY)

The GPMI auxiliary address register specifies the location of the auxiliary buffers in system memory. This value must be word aligned.

GPMI_AUXILIARY 0x050

Address: 3300_2000h base + 50h offset = 3300_2050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ADDRESS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



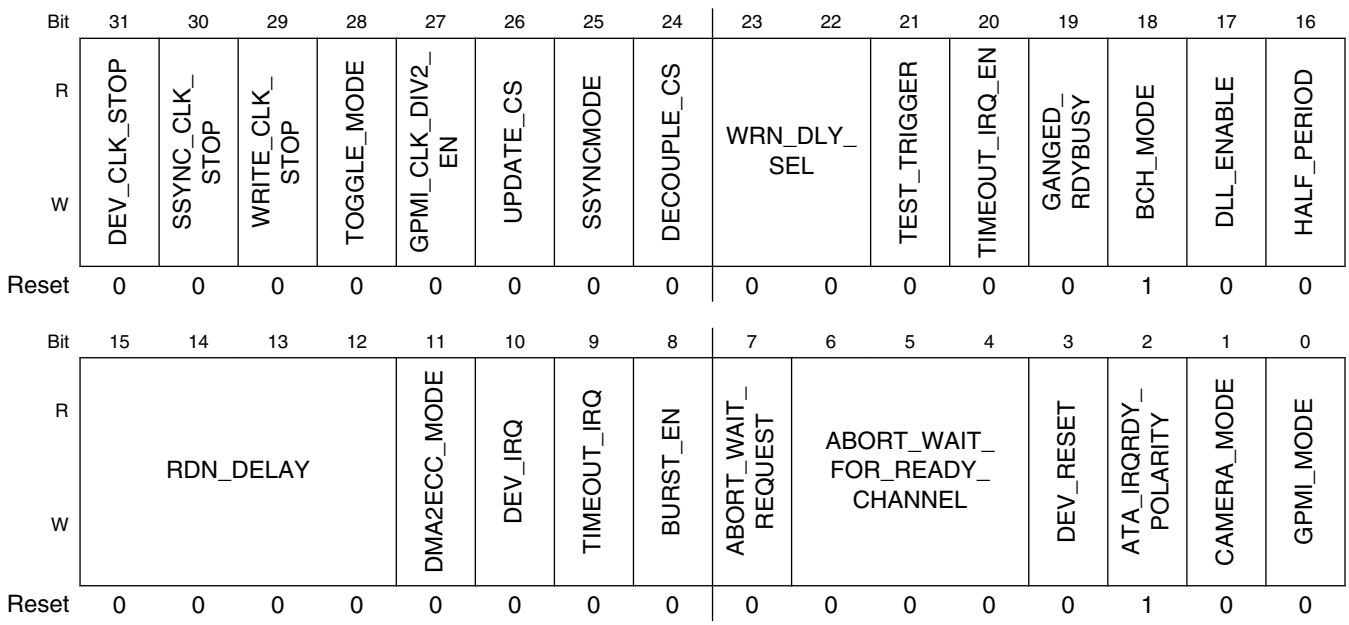
GPMI_AUXILIARY field descriptions

Field	Description
31–2 ADDRESS	Pointer to ECC control structure and meta-data storage.
RSVD0	Always write zeroes to this bit field.

9.6.4.7 GPMI Control Register 1 Description (GPMI_CTRL1*n*)

The GPMI control register 1 specifies additional control fields that are not used on a per-transaction basis.

Address: 3300_2000h base + 60h offset + (4d × i), where i=0d to 3d



GPMI_CTRL1*n* field descriptions

Field	Description
31 DEV_CLK_STOP	set this bit to 1 will stop gpmi io working clk.
30 SSYNC_CLK_STOP	set this bit to 1 will stop the source synchronous mode clk.

Table continues on the next page...

GPMI_CTRL1*n* field descriptions (continued)

Field	Description
29 WRITE_CLK_STOP	In onfi source synchronous mode, host may save power during the data write cycles by holding the CLK signal high (i.e. stopping the CLK). The host may only stop the CLK during data write, by setting this bit to 1, if the device supports this feature as indicated in the parameter page.
28 TOGGLE_MODE	enable samsung toggle mode.
27 GPMI_CLK_DIV2_EN	This bit should be reset to 0 in asynchronous mode. The frequence ratio of (device clock : ccm gpmi clock) will be (1 : 1). This bit should be set to 1, in source synchronous mode or toggle mode. The frequence raito of (device clock : ccm gpmi clock) will be (1 : 2). enable the gpmi clk divider. 0x0 internal factor-2 clock divider is disabled 0x1 internal factor-2 clock divider is enabled.
26 UPDATE_CS	force the CS value is be updated to external chip select pin, even GPMI is idle.
25 SSYNCMODE	source synchronouse mode 1 or asynchrous mode 0. ASYNC = 0x0 Asynchronous mode. SSYNC = 0x1 Source Synchronous mode.
24 DECOUPLE_CS	Decouple Chip Select from DMA Channel. Setting this bit to 1 will allow a DMA channel to specify any value in the CTRL0_CS register field. Software can use one DMA channel to access all 8 Nand devices.
23–22 WRN_DLY_SEL	Since the GPMI write strobe (WRN) is a fast clock pin, the delay on this signal can be programmed to match the load on this pin. 0 = ~2ns; 1 = ~4ns; 2 = ~6ns; 3 = no delay.
21 TEST_TRIGGER	Test Trigger Enable 0 Disable 1 Enable
20 TIMEOUT_IRQ_EN	Setting this bit to '1' will enable timeout IRQ for transfers in ATA mode only, and for WAIT_FOR_READY commands in both ATA and Nand mode. The Device_Busy_Timeout value is used for this timeout.
19 GANGED_RDYBUSY	Set this bit to 1 will force all Nand RDY_BUSY inputs to be sourced from (tied to) RDY_BUSY0. This will free up all, except one, RDY_BUSY input pins.
18 BCH_MODE	This bit selects which error correction unit will access GPMI. This bit must always be set to '1', since only the BCH unit is available in this design.
17 DLL_ENABLE	Set this bit to 1 to enable the GPMI DLL. This is required for fast NAND reads (above 30 MHz read strobe). After setting this bit, wait 64 GPMI clock cycles for the DLL to lock before performing a NAND read.
16 HALF_PERIOD	Set this bit to 1 if the GPMI clock period is greater than 16ns for proper DLL operation. DLL_ENABLE must be zero while changing this field.
15–12 RDN_DELAY	This variable is a factor in the calculated delay to apply to the internal read strobe for correct read data sampling. The applied delay (AD) is between 0 and 1.875 times the reference period (RP). RP is one half of the GPMI clock period if HALF_PERIOD=1 otherwise it is the full GPMI clock period. The equation is: AD = RDN_DELAY x 0.125 x RP. This value must not exceed 16ns.

Table continues on the next page...

GPMI_CTRL1*n* field descriptions (continued)

Field	Description
	This variable is used to achieve faster NAND access. For example if the Read Strobe is asserted from time 0 to 13ns but the read access time is 20ns, then choose AD=12ns will cause the data to be sampled at time 25ns (13+12) giving a 5ns data setup time. If RP=13ns then RDN_DELAY = 12/(0.125 x 13ns) = 7.38 (0111b). DLL_ENABLE must be zero while changing this field.
11 DMA2ECC_ MODE	This is mainly for testing HWECC without involving the Nand device. Setting this bit will cause DMA write data to redirected to HWECC module (instead of Nand Device) for encoding or decoding.
10 DEV_IRQ	This bit is set when an Interrupt is received from the ATA device. Write 0 to clear.
9 TIMEOUT_IRQ	This bit is set when a timeout occurs using the Device_Busy_Timeout value. Write 0 to clear.
8 BURST_EN	When set to 1 each DMA request will generate a 4-transfer burst on the APB bus.
7 ABORT_WAIT_ REQUEST	Request to abort "wait for ready" command on channel indicated by ABORT_WAIT_FOR_READY_CHANNEL. Hardware will clear this bit when abort is done.
6–4 ABORT_WAIT_ FOR_READY_ CHANNEL	Abort a wait for ready command on selected channel. Set the ABORT_WAIT_REQUEST to kick of operation.
3 DEV_RESET	ENABLED = 0x0 NANDF_WP_B(WPN) pin is held low (asserted). DISABLED = 0x1 NANDF_WP_B(WPN) pin is held high (de-asserted). 0 NANDF_WP_B pin is held low (asserted). 1 NANDF_WP_B pin is held high (de-asserted).
2 ATA_IRQRDY_ POLARITY	For ATA MODE: Note NAND_RDY_BUSY[3:2] are not affected by this bit. ACTIVELOW = 0x0 ATA IORDY and IRQ are active low, or NAND_RDY_BUSY[1:0] are active low ready. ACTIVEHIGH = 0x1 ATA IORDY and IRQ are active high, or NAND_RDY_BUSY[1:0] are active high ready. 0 External ATA IORDY and IRQ are active low. 1 External ATA IORDY and IRQ are active high. For NAND MODE: 0 External RDY_BUSY[1] and RDY_BUSY[0] pins are ready when low and busy when high. 1 External RDY_BUSY[1] and RDY_BUSY[0] pins are ready when high and busy when low.
1 CAMERA_MODE	When set to 1 and ATA UDMA is enabled the UDMA interface becomes a camera interface.
0 GPMI_MODE	ATA mode is only supported on channel zero. If ATA mode is selected, then only channel three is available for NAND use. NAND = 0x0 NAND mode. ATA = 0x1 ATA mode.

Table continues on the next page...

GPMI_CTRL1*n* field descriptions (continued)

Field	Description
	0 NAND mode. 1 ATA mode.

9.6.4.8 GPMI Timing Register 0 Description (GPMI_TIMING0)

The GPMI timing register 0 specifies the timing parameters that are used by the cycle state machine to guarantee the various setup, hold and cycle times for the external media type.

GPMI_TIMING0 0x070

Address: 3300_2000h base + 70h offset = 3300_2070h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									ADDRESS_SETUP								DATA_HOLD								DATA_SETUP							
W	0																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1

GPMI_TIMING0 field descriptions

Field	Description
31–24 RSVD1	Always write zeroes to this bit field.
23–16 ADDRESS_ SETUP	Number of GPMICLK cycles that the CE/ADDR signals are active before a strobe is asserted. A value of zero is interpreted as 0. For ATA PIO modes this is known in the ATA7 specification as "Address valid to DIOR-/DIOW- setup"
15–8 DATA_HOLD	Data bus hold time in GPMICLK cycles. Also the time that the data strobe is de-asserted in a cycle. A value of zero is interpreted as 256. For ATA PIO modes this is known in the ATA7 specification as "DIOR-/DIOW- recovery time"
DATA_SETUP	Data bus setup time in GPMICLK cycles. Also the time that the data strobe is asserted in a cycle. This value must be greater than 2 for ATA devices that use IORDY to extend transfer cycles. A value of zero is interpreted as 256. For ATA PIO modes this is known in the ATA7 specification as ""DIOR-/DIOW-"

9.6.4.9 GPMI Timing Register 1 Description (GPMI_TIMING1)

The GPMI timing register 1 specifies the timeouts used when monitoring the NAND READY pin or the ATA IRQ and IOWAIT signals.

GPMI_TIMING1 0x080

Address: 3300_2000h base + 80h offset = 3300_2080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DEVICE_BUSY_TIMEOUT																RSVD1																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPMI_TIMING1 field descriptions

Field	Description
31–16 DEVICE_BUSY_TIMEOUT	Timeout waiting for NAND Ready/Busy or ATA IRQ. Used in WAIT_FOR_READY mode. This value is the number of GPMI_CLK cycles multiplied by 4096.
RSVD1	Always write zeroes to this bit field.

9.6.4.10 GPMI Timing Register 2 Description (GPMI_TIMING2)

The GPMI timing register 2 specifies the double data rate timing parameters that are used by the cycle state machine to guarantee the various cs delay, pre-amble delay, post-amble delay, command/address delay, data delay, TCR, TRPSTH, and read latency cycle times for the external media type.

GPMI_TIMING2 0x090

Address: 3300_2000h base + 90h offset = 3300_2090h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TRPSTH			TCR		READ_LATENCY			RSVD0			CE_DELAY					PREAMBLE_DELAY				POSTAMBLE_DELAY				CMDADD_PAUSE				DATA_PAUSE				
W																																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	1	1	0

GPMI_TIMING2 field descriptions

Field	Description
31–29 TRPSTH	Only for Toggle NAND timing control delay TRPSTH GPMICLK cycles for CEn_B high to RE_B high, A value of zero is interpreted as 8
28–27 TCR	Only for Toggle NAND timing control delay (TCR+1) GPMICLK cycles for CEn_B low to RE_B low, 0 is less than or equal to TCR, which is less than the PREAMBLE_DELAY
26–24 READ_LATENCY	This field is for double data rate read latency configuration. others READ LATENCY is 3 000 READ LATENCY is 0 001 READ LATENCY is 1 010 READ LATENCY is 2 011 READ LATENCY is 3 100 READ LATENCY is 4 101 READ LATENCY is 5

Table continues on the next page...

GPMI_TIMING2 field descriptions (continued)

Field	Description
23–21 RSVD0	Always write zeroes to this bit field.
20–16 CE_DELAY	GPMI dealy from CEn assert to W/Rn changing edge. value of zero is interpreted as 32.
15–12 PREAMBLE_ DELAY	GPMI pre-amble delay in GPMICLK cycles. A value of zero is interpreted as 16.
11–8 POSTAMBLE_ DELAY	GPMI post-amble delay in GPMICLK cycles. A value of zero is interpreted as 16.
7–4 CMDADD_ PAUSE	GPMI delay time from command or addres pause to command or address resume in GPMICLK cycles. A value of zero is interpreted as 16.
DATA_PAUSE	GPMI delay time from data pause to data resume in GPMICLK cycles. A value of zero is interpreted as 16.

9.6.4.11 GPMI DMA Data Transfer Register Description (GPMI_DATA)

The GPMI DMA data transfer register is used by the DMA to read or write data to or from the ATA/NAND control state machine.

GPMI_DATA 0x0A0

Address: 3300_2000h base + A0h offset = 3300_20A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPMI_DATA field descriptions

Field	Description
DATA	In 8-bit mode, one, two, three or four bytes can can be accessed to send the same number of bus cycles.

9.6.4.12 GPMI Status Register Description (GPMI_STAT)

The GPMI control and status register provides a read back path for various operational states of the GPMI controller.

GPMI_STAT 0x0B0

Address: 3300_2000h base + B0h offset = 3300_20B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	READY_BUSY								RDY_TIMEOUT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DEV7_ERROR	DEV6_ERROR	DEV5_ERROR	DEV4_ERROR	DEV3_ERROR	DEV2_ERROR	DEV1_ERROR	DEV0_ERROR	RSVD1			ATA_IRQ	INVALID_BUFFER_MASK	FIFO_EMPTY	FIFO_FULL	PRESENT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

GPMI_STAT field descriptions

Field	Description
31–24 READY_BUSY	Read-only view of NAND Ready_Busy Input pins.
23–16 RDY_TIMEOUT	State of the RDY/BUSY Timeout Flags. When any bit is set to '1' in this field, it indicates that a time out has occurred while waiting for the ready state of the requested NAND device. Multiple bits may be set simultaneously. When GPMI_CTRL1_DECOUPLE_CS = 0, RDY_TIMEOUT[n] is associated with the NAND device on chip_select[n]. When GPMI_CTRL1_DECOUPLE_CS = 1, these flags become associated to a DMA channel instead of a NAND device. For example if DMA channel 6 sends a WAIT_FOR_READY command for NAND Device 2, and a timeout occured on READY_BUSY2, then READY_TIMEOUT[6] will be set instead of READY_TIMEOUT[2].
15 DEV7_ERROR	DMA channel 7 (Timeout or compare failure, depending on COMMAND_MODE). 0 No error condition present on ATA/NAND Device accessed by DMA channel 7. 1 An Error has occurred on ATA/NAND Device accessed by
14 DEV6_ERROR	DMA channel 6 (Timeout or compare failure, depending on COMMAND_MODE). 0 No error condition present on ATA/NAND Device accessed by DMA channel 6. 1 An Error has occurred on ATA/NAND Device accessed by
13 DEV5_ERROR	DMA channel 5 (Timeout or compare failure, depending on COMMAND_MODE). 0 No error condition present on ATA/NAND Device accessed by DMA channel 5. 1 An Error has occurred on ATA/NAND Device accessed by
12 DEV4_ERROR	DMA channel 4 (Timeout or compare failure, depending on COMMAND_MODE). 0 No error condition present on ATA/NAND Device accessed by DMA channel 4. 1 An Error has occurred on ATA/NAND Device accessed by
11 DEV3_ERROR	DMA channel 3 (Timeout or compare failure, depending on COMMAND_MODE). 0 No error condition present on ATA/NAND Device accessed by DMA channel 3. 1 An Error has occurred on ATA/NAND Device accessed by
10 DEV2_ERROR	DMA channel 2 (Timeout or compare failure, depending on COMMAND_MODE). 0 No error condition present on ATA/NAND Device accessed by DMA channel 2. 1 An Error has occurred on ATA/NAND Device accessed by
9 DEV1_ERROR	DMA channel 1 (Timeout or compare failure, depending on COMMAND_MODE). 0 No error condition present on ATA/NAND Device accessed by DMA channel 1. 1 An Error has occurred on ATA/NAND Device accessed by
8 DEV0_ERROR	DMA channel 0 (Timeout or compare failure, depending on COMMAND_MODE). 0 No error condition present on ATA/NAND Device accessed by DMA channel 0. 1 An Error has occurred on ATA/NAND Device accessed by
7–5 RSVD1	Always write zeroes to this bit field.

Table continues on the next page...

GPMI_STAT field descriptions (continued)

Field	Description
4 ATA_IRQ	Status of the ATA_IRQ input pin.
3 INVALID_ BUFFER_MASK	Buffer Mask Validity bit. 0 ECC Buffer Mask is not invalid. 1 ECC Buffer Mask is invalid.
2 FIFO_EMPTY	NOT_EMPTY = 0x0 FIFO is not empty. EMPTY = 0x1 FIFO is empty. 0 FIFO is not empty. 1 FIFO is empty.
1 FIFO_FULL	NOT_FULL = 0x0 FIFO is not full. FULL = 0x1 FIFO is full. 0 FIFO is not full. 1 FIFO is full.
0 PRESENT	UNAVAILABLE = 0x0 GPMI is not present in this product. AVAILABLE = 0x1 GPMI is present in this product. 0 GPMI is not present in this product. 1 GPMI is present is in this product.

9.6.4.13 GPMI Debug Information Register Description (GPMI_DEBUG)

The GPMI debug information register provides a read back path for diagnostics to determine the current operating state of the GPMI controller.

GPMI_DEBUG 0x0C0

Address: 3300_2000h base + C0h offset = 3300_20C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WAIT_FOR_READY_END								DMA_SENSE								DMAREQ								CMD_END							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPMI_DEBUG field descriptions

Field	Description
31–24 WAIT_FOR_ READY_END	Read Only view of the Wait_For_Ready End toggle signals to DMA. One per channel

Table continues on the next page...

GPMI_DEBUG field descriptions (continued)

Field	Description
23–16 DMA_SENSE	Read-only view of sense state of the 8 DMA channels. A value of "1" in any bit position indicates that a read and compare command failed or a timeout occured for the corresponding channel.
15–8 DMAREQ	Read-only view of DMA request line for 8 DMA channels. A toggle on any bit position indicates a DMA request for the corresponding channel.
CMD_END	Read Only view of the Command End toggle signals to DMA. One per channel

9.6.4.14 GPMI Version Register Description (GPMI_VERSION)

This register reflects the version number for the GPMI.

GPMI_VERSION 0x0D0

Address: 3300_2000h base + D0h offset = 3300_20D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								STEP															
W																																
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPMI_VERSION field descriptions

Field	Description
31–24 MAJOR	Fixed read-only value reflecting the MAJOR field of the RTL version.
23–16 MINOR	Fixed read-only value reflecting the MINOR field of the RTL version.
STEP	Fixed read-only value reflecting the stepping of the RTL version.

9.6.4.15 GPMI Debug2 Information Register Description (GPMI_DEBUG2)

The GPMI Debug2 information register provides a read back path for diagnostics to determine the current operating state of the GPMI controller.

GPMI_DEBUG2 0x0E0

Address: 3300_2000h base + E0h offset = 3300_20E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD1				UDMA_STATE				BUSY	PIN_STATE			MAIN_STATE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SYND2GPMI_BE				GPMI2SYND_VALID	GPMI2SYND_READY	SYND2GPMI_VALID	SYND2GPMI_READY	VIEW_DELAYED_RDN	UPDATE_WINDOW	RDN_TAP					
W																
Reset	1	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0

GPMI_DEBUG2 field descriptions

Field	Description
31–28 RSVD1	Always write zeroes to this bit field.
27–24 UDMA_STATE	USM_IDLE = 4'h0, idle USM_DMARQ = 4'h1, DMA req USM_ACK = 4'h2, DMA ACK USM_FIFO_E = 4'h3, Fifo empty USM_WPAUSE = 4'h4, WR DMA Paused by device USM_TSTRB = 4'h5, Toggle HSTROBE USM_CAPTUR = 4'h6, Capture Stage, (data sampled with DSTROBE is valid) USM_DATOUT = 4'h7, Change Burst DATAOUT USM_CRC = 4'h8, Source CRC to Device USM_WAIT_R = 4'h9, Waiting for DDMARDY- USM_END = 4'ha; Negate DMAACK (end of DMA) USM_WAIT_S = 4'hb, Waiting for DSTROBE USM_RPAUSE = 4'hc, Rd DMA Paused by Host USM_RSTOP = 4'hd, Rd DMA Stopped by Host USM_WTERM = 4'he, Wr DMA Termination State USM_RTERM = 4'hf, Rd DMA Termination state
23 BUSY	When asserted the GPMI is busy. Undefined results may occur if any registers are written when BUSY is asserted. DISABLED = 0x0 The GPMI is not busy. ENABLED = 0x1 The GPMI is busy.
22–20 PIN_STATE	parameter PSM_IDLE = 3'h0, PSM_BYTCNT = 3'h1, PSM_ADDR = 3'h2, PSM_STALL = 3'h3, PSM_STROBE = 3'h4, PSM_ATARDY = 3'h5, PSM_DHOLD = 3'h6, PSM_DONE = 3'h7. PSM_IDLE = 0x0 PSM_BYTCNT = 0x1 PSM_ADDR = 0x2 PSM_STALL = 0x3 PSM_STROBE = 0x4 PSM_ATARDY = 0x5 PSM_DHOLD = 0x6 PSM_DONE = 0x7
19–16 MAIN_STATE	parameter MSM_IDLE = 4'h0, MSM_BYTCNT = 4'h1, MSM_WAITFE = 4'h2, MSM_WAITFR = 4'h3, MSM_DMAREQ = 4'h4, MSM_DMAACK = 4'h5, MSM_WAITFF = 4'h6, MSM_LDFIFO = 4'h7, MSM_LDDMAR = 4'h8, MSM_RDCMP = 4'h9, MSM_DONE = 4'hA. MSM_IDLE = 0x0 MSM_BYTCNT = 0x1 MSM_WAITFE = 0x2 MSM_WAITFR = 0x3

Table continues on the next page...

GPMI_DEBUG2 field descriptions (continued)

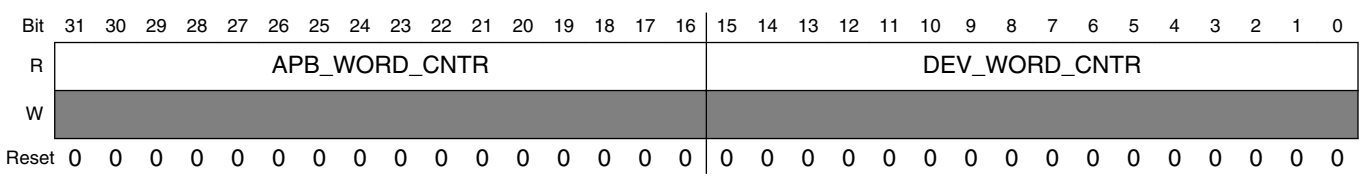
Field	Description
	MSM_DMAREQ = 0x4 MSM_DMAACK = 0x5 MSM_WAITFF = 0x6 MSM_LDFIFO = 0x7 MSM_LDDMAR = 0x8 MSM_RDCMP = 0x9 MSM_DONE = 0xA
15–12 SYND2GPMI_BE	Data byte enable Input from BCH.
11 GPMI2SYND_VALID	Data handshake output to BCH.
10 GPMI2SYND_READY	Data handshake output to BCH.
9 SYND2GPMI_VALID	Data handshake Input from BCH.
8 SYND2GPMI_READY	Data handshake Input from BCH.
7 VIEW_DELAYED_RDN	Set to a 1 to select the delayed feedback RE_B to drive the GPMI_ADDR[0] (Nand CLE) pin. For debug purposes, this will allow you see if DLL is functioning properly.
6 UPDATE_WINDOW	A 1 indicates that the DLL is busy generating the required delay.
RDN_TAP	This is the DLL tap calculated by the DLL controller. The selects the amount of delay form the DLL chain.

9.6.4.16 GPMI Debug3 Information Register Description (GPMI_DEBUG3)

The GPMI Debug3 information register provides a read back path for diagnostics to determine the current operating state of the GPMI controller.

GPMI_DEBUG3 0x0F0

Address: 3300_2000h base + F0h offset = 3300_20F0h



GPMI_DEBUG3 field descriptions

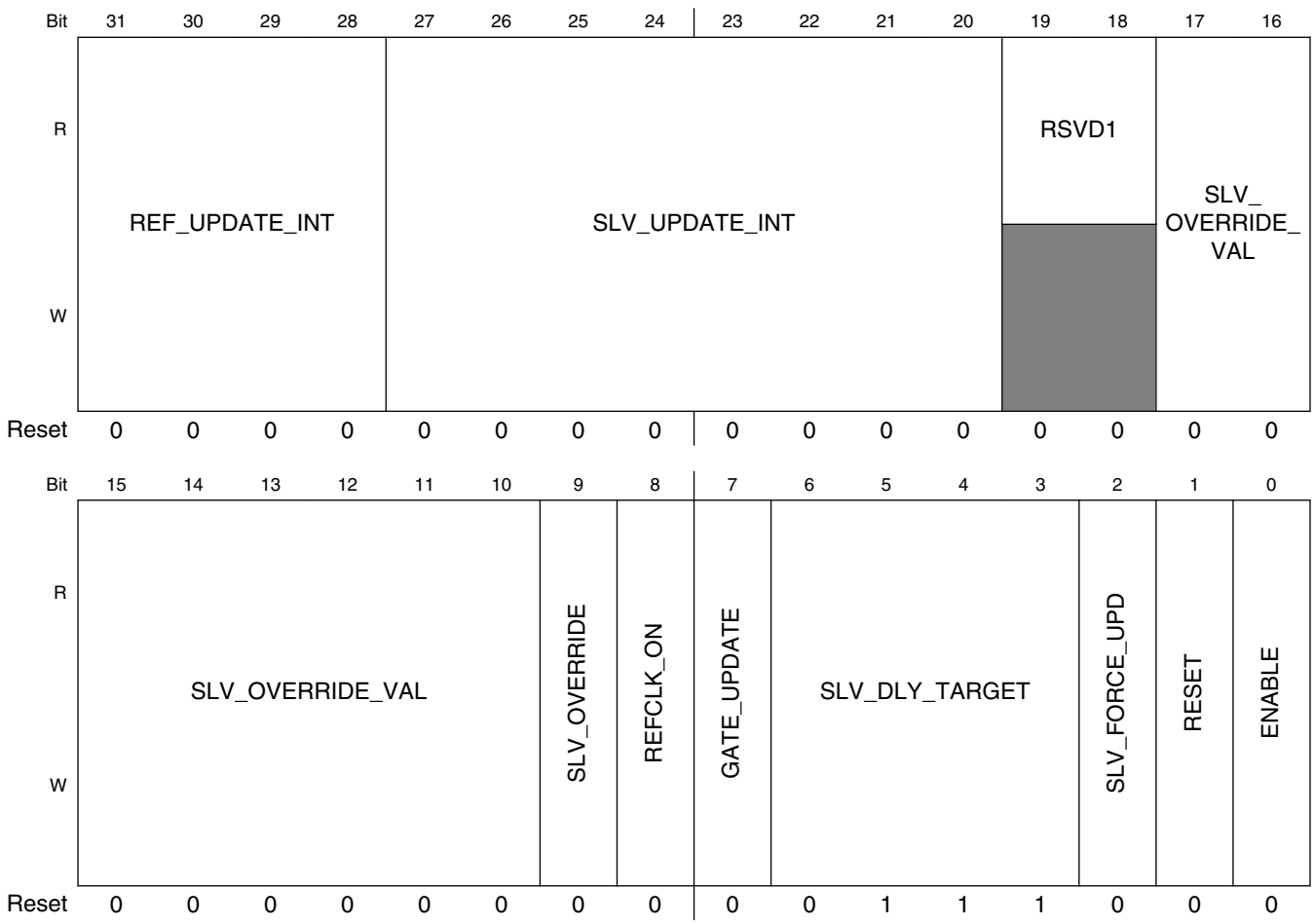
Field	Description
31–16 APB_WORD_ CNTR	Reflects the number of bytes remains to be transferred on the APB bus.
DEV_WORD_ CNTR	Reflects the number of bytes remains to be transferred on the ATA/Nand bus.

9.6.4.17 GPMI Double Rate Read DLL Control Register Description (GPMI_READ_DDR_DLL_CTRL)

GPMI DDR Read Delay Loop Lock Control Register. This register provides programmability in DDR mode for data input timing and data formats.

GPMI_READ_DDR_DLL_CTRL 0x100

Address: 3300_2000h base + 100h offset = 3300_2100h



GPMI_READ_DDR_DLL_CTRL field descriptions

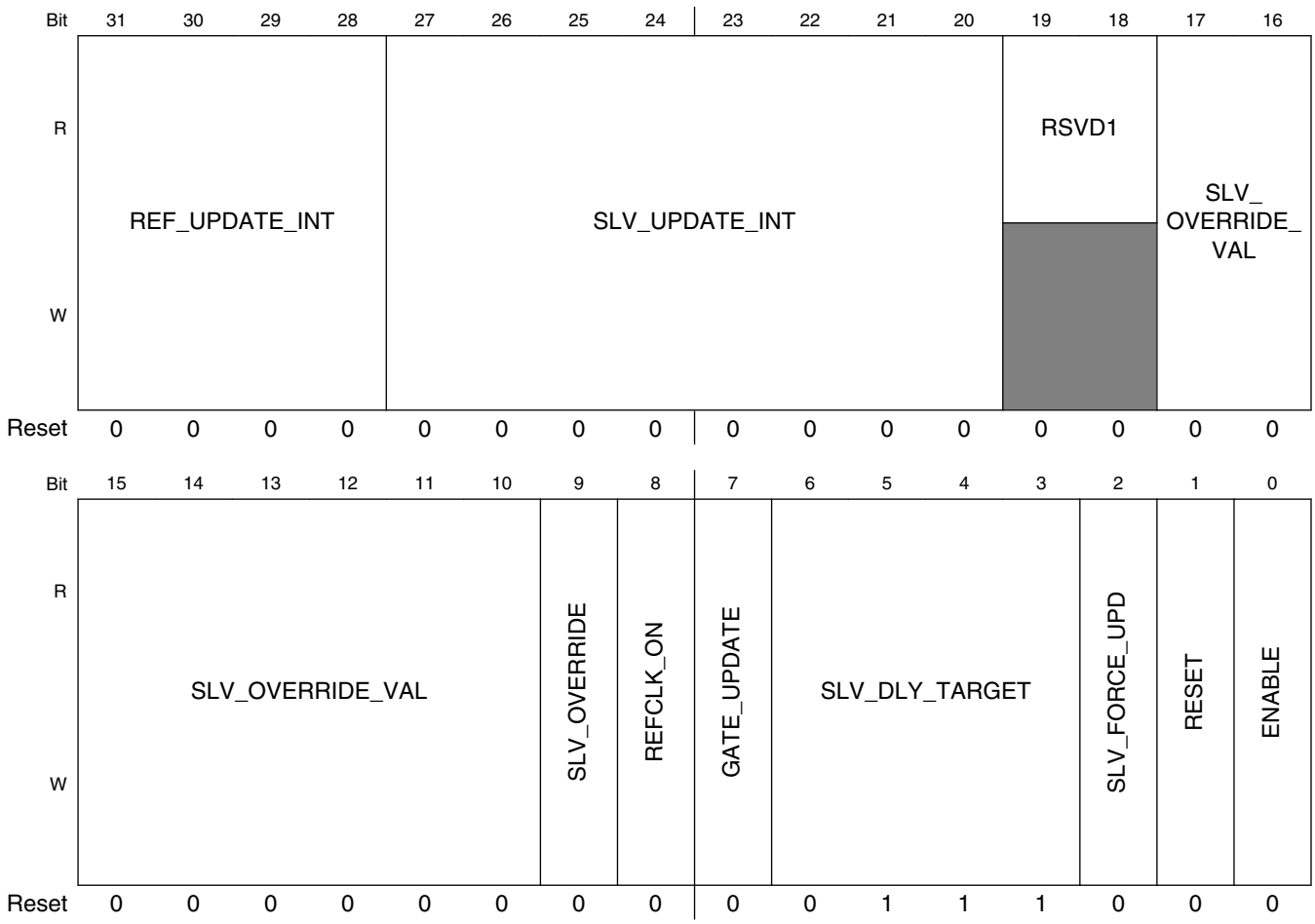
Field	Description
31–28 REF_UPDATE_INT	This field allows the user to add additional delay cycles to the DLL control loop (reference delay line control). By default, the DLL control loop shall update every two GPMICLK cycles. Programming this field results in a DLL control loop update interval of (2 + REF_UPDATE_INT) * GPMICLK. It should be noted that increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that may effect the delay (such as voltage and temperature)
27–20 SLV_UPDATE_INT	Setting a value greater than 0 in this field, shall over-ride the default slave delay-line update interval of 256 GPMICLK cycles. A value of 0 results in an update interval of 256 GPMICLK cycles (default setting). A value of 0x0f results in 15 cycles and so on. Note that software can always cause an update of the slave-delay line using the SLV_FORCE_UPDATE register. Note that the slave delay line will also update automatically when the reference DLL transitions to a locked state (from an un-locked state).
19–18 RSVD1	Reserved
17–10 SLV_OVERRIDE_VAL	When SLV_OVERRIDE=1 This field is used to select 1 of 256 physical taps manually. A value of 0 selects tap 1, and a value of 0x7f selects tap 256.
9 SLV_OVERRIDE	Set this bit to 1 to Enable manual override for slave delay chain using SLV_OVERRIDE_VAL; to set 0 to disable manual override. This feature does not require the DLL to tbe enabled using the ENABLE bit. In fact to reduce power, if SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE=0
8 REFCLK_ON	set this bit to 1 will turn on the reference clock
7 GATE_UPDATE	Setting this bit to 1, forces the slave delay line not update
6–3 SLV_DLY_TARGET	The delay target for the read clock is can be programmed in 1/16th increments of an GPMICLK half-period. So the input read-clock can be delayed relative input data from (GPMICLK/2)/16 to GPMICLK/2.
2 SLV_FORCE_UPD	Setting this bit to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line shall update automatically based on the SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered).
1 RESET	Setting this bit to 1 force a reset on DLL. This will cause the DLL to lose lock and re-calibrate to detect an GPMICLK half period phase shift. This signal is used by the DLL as edge-sensitive, so in order to create a subsequent reset, RESET must be taken low and then asserted again.
0 ENABLE	Set this bit to 1 to enable the DLL and delay chain; otherwise; set to 0 to bypasses DLL. Note that using the slave delay line override feature with SLV_OVERRIDE and SLV_OVERRIDE VAL, the DLL does not need to be enabled.

9.6.4.18 GPMI Double Rate Write DLL Control Register Description (GPMI_WRITE_DDR_DLL_CTRL)

GPMI DDR Write Delay Loop Lock Control Register. This register provides programmability in DDR mode for data output timing and data formats.

GPMI_WRITE_DDR_DLL_CTRL 0x110

Address: 3300_2000h base + 110h offset = 3300_2110h



GPMI_WRITE_DDR_DLL_CTRL field descriptions

Field	Description
31–28 REF_UPDATE_ INT	This field allows the user to add additional delay cycles to the DLL control loop (reference delay line control). By default, the DLL control loop shall update every two GPMICLK cycles. Programming this field results in a DLL control loop update interval of (2 + REF_UPDATE_INT) * GPMICLK. It should be noted that increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that may effect the delay (such as voltage and temperature)
27–20 SLV_UPDATE_ INT	Setting a value greater than 0 in this field, shall over-ride the default slave delay-line update interval of 256 GPMICLK cycles. A value of 0 results in an update interval of 256 GPMICLK cycles (default setting). A value of 0x0f results in 15 cycles and so on. Note that software can always cause an update of the slave-delay line using the SLV_FORCE_UPDATE register. Note that the slave delay line will also update automatically when the reference DLL transitions to a locked state (from an un-locked state).
19–18 RSVD1	Reserved
17–10 SLV_ OVERRIDE_VAL	When SLV_OVERRIDE=1 This field is used to select 1 of 256 physical taps manually. A value of 0 selects tap 1, and a value of 0x7f selects tap 256.
9 SLV_OVERRIDE	Set this bit to 1 to Enable manual override for slave delay chain using SLV_OVERRIDE_VAL; to set 0 to disable manual override. This feature does not require the DLL to tbe enabled using the ENABLE bit. In fact to reduce power, if SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE=0

Table continues on the next page...

GPMI_WRITE_DDR_DLL_CTRL field descriptions (continued)

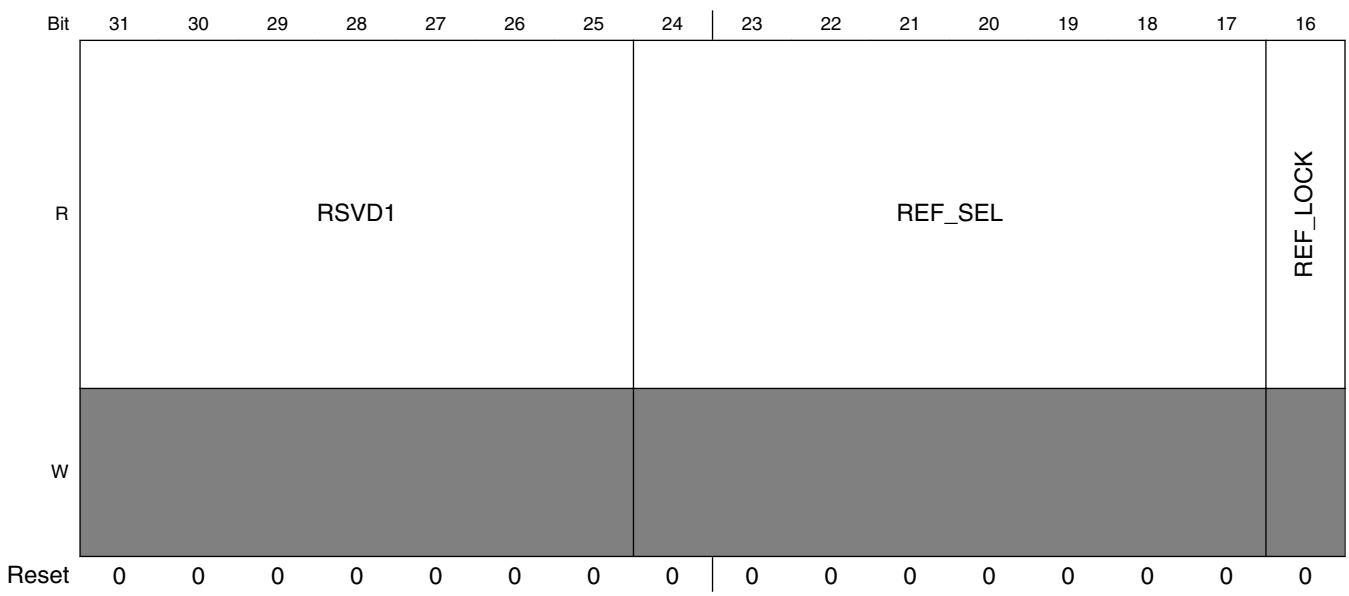
Field	Description
8 REFCLK_ON	set this bit to 1 will turn on the reference clock
7 GATE_UPDATE	Setting this bit to 1, forces the slave delay line not update
6–3 SLV_DLY_ TARGET	The delay target for the read clock can be programmed in 1/16th increments of an GPMICLK half-period. So the input read-clock can be delayed relative input data from (GPMICLK/2)/16 to GPMICLK/2.
2 SLV_FORCE_ UPD	Setting this bit to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line shall update automatically based on the SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered).
1 RESET	Setting this bit to 1 force a reset on DLL. This will cause the DLL to lose lock and re-calibrate to detect an GPMICLK half period phase shift. This signal is used by the DLL as edge-sensitive, so in order to create a subsequent reset, RESET must be taken low and then asserted again.
0 ENABLE	Set this bit to 1 to enable the DLL and delay chain; otherwise; set to 0 to bypasses DLL. Note that using the slave delay line override feature with SLV_OVERRIDE and SLV_OVERRIDE VAL, the DLL does not need to be enabled.

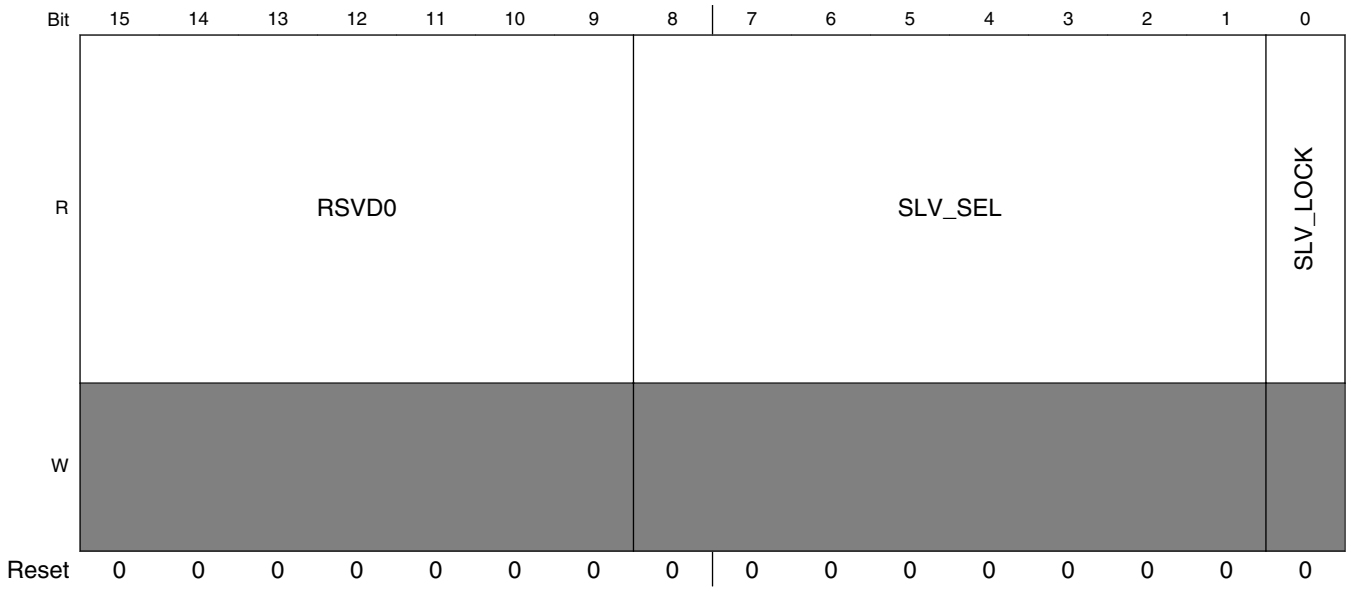
9.6.4.19 GPMI Double Rate Read DLL Status Register Description (GPMI_READ_DDR_DLL_STS)

GPMI Double Rate Read DLL Status Register, Read Only. GPMI DLL status fields are provided in this register.

GPMI_READ_DDR_DLL_STS 0x120

Address: 3300_2000h base + 120h offset = 3300_2120h





GPMI_READ_DDR_DLL_STS field descriptions

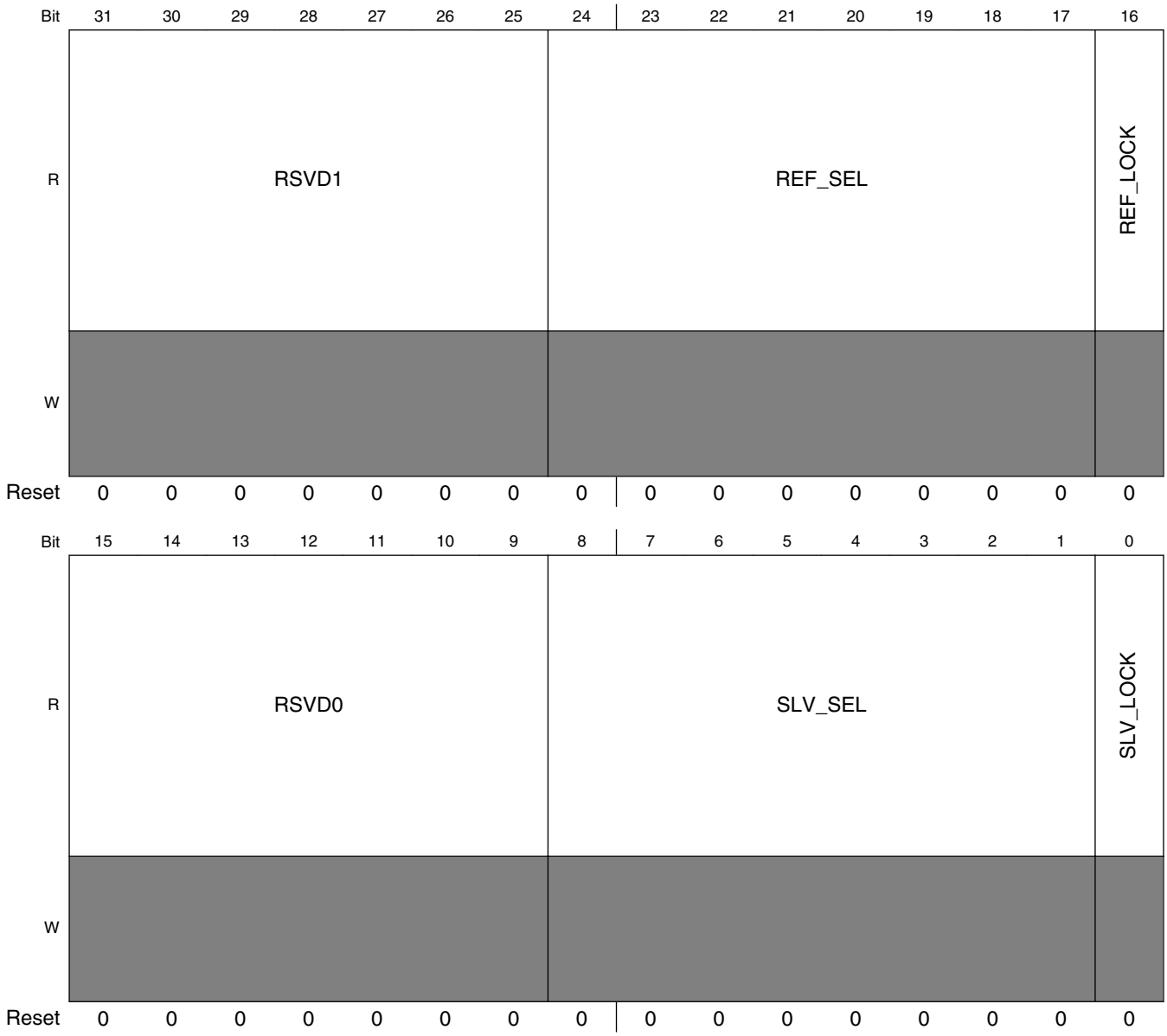
Field	Description
31–25 RSVD1	Reserved
24–17 REF_SEL	Reference delay line select status.
16 REF_LOCK	Reference DLL lock status. This signifies that the DLL has detected and locked to a half-phase GPMICLK shift, allowing the slave delay-line to perform programmed clock delays.
15–9 RSVD0	Reserved
8–1 SLV_SEL	Slave delay line select status
0 SLV_LOCK	Slave delay-line lock status. This signifies that a valid calibration has been set to the slave-delay line and that the slave-delay line is implementing the programmed delay value.

9.6.4.20 GPMI Double Rate Write DLL Status Register Description (GPMI_WRITE_DDR_DLL_STS)

GPMI Double Rate Write DLL Status Register, Read Only. GPMI DLL status fields are provided in this register.

GPMI_WRITE_DDR_DLL_STS 0x130

Address: 3300_2000h base + 130h offset = 3300_2130h



GPMI_WRITE_DDR_DLL_STS field descriptions

Field	Description
31–25 RSVD1	Reserved
24–17 REF_SEL	Reference delay line select status.
16 REF_LOCK	Reference DLL lock status. This signifies that the DLL has detected and locked to a half-phase GPMICLK shift, allowing the slave delay-line to perform programmed clock delays.
15–9 RSVD0	Reserved
8–1 SLV_SEL	Slave delay line select status

Table continues on the next page...

GPMI_WRITE_DDR_DLL_STS field descriptions (continued)

Field	Description
0 SLV_LOCK	Slave delay-line lock status. This signifies that a valid calibration has been set to the slave-delay line and that the slave-delay line is implementing the programmed delay value.

Chapter 10
Mass Storage

10.1 Enhanced Configurable SPI (ECSPI)

10.1.1 Overview

The Enhanced Configurable Serial Peripheral Interface (ECSPI) is a full-duplex, synchronous, four-wire serial communication block.

The ECSPI contains a 64 x 32 receive buffer (RXFIFO) and a 64 x 32 transmit buffer (TXFIFO). With data FIFOs, the ECSPI allows rapid data communication with fewer software interrupts. The figure below shows a block diagram of the ECSPI.

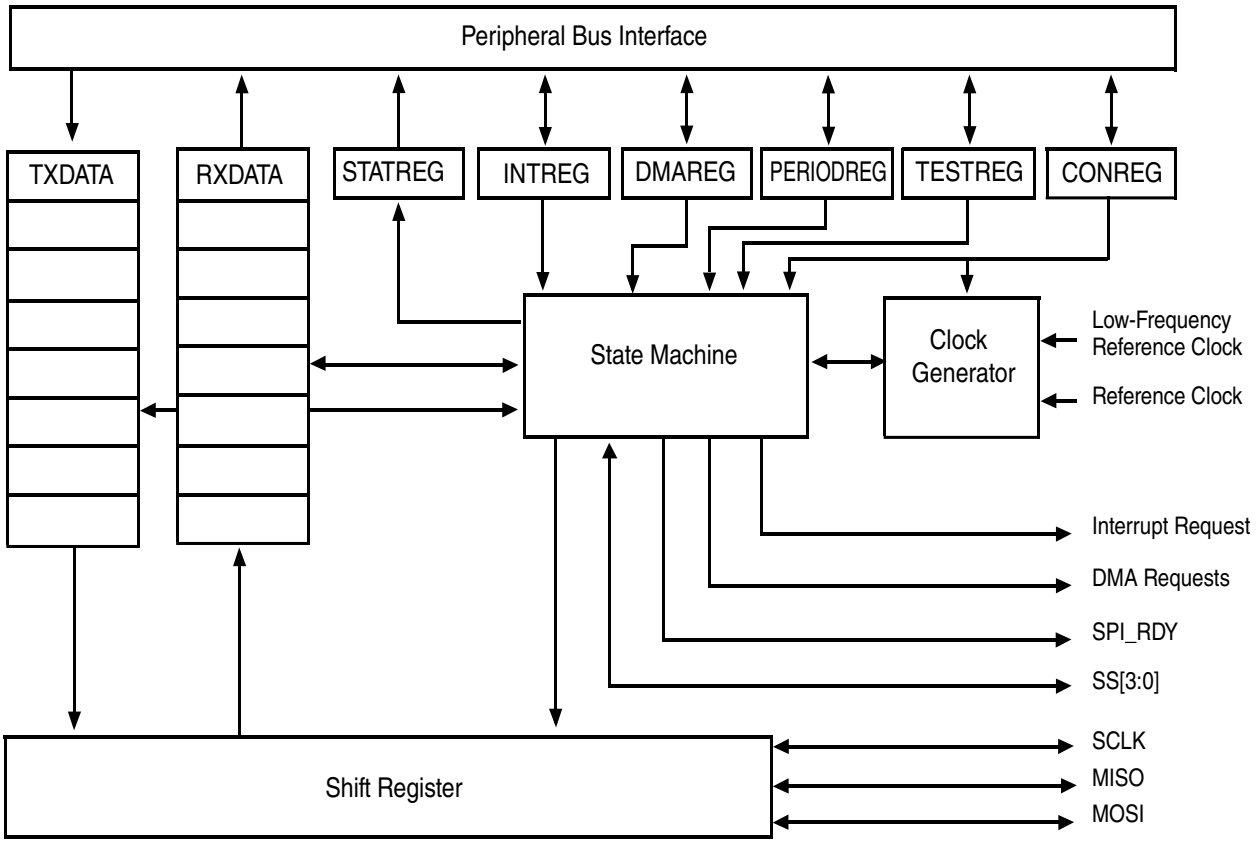


Figure 10-1. ECSPI Block Diagram

10.1.1.1 Features

Key features of the ECSPI include:

- Full-duplex synchronous serial interface
- Master/Slave configurable
- Four Chip Select (SS) signals to support multiple peripherals
- Transfer continuation function allows unlimited length data transfers
- 32-bit wide by 64-entry FIFO for both transmit and receive data
- Polarity and phase of the Chip Select (SS) and SPI Clock (SCLK) are configurable
- Direct Memory Access (DMA) support
- Refer to the product data sheet for the maximum operating frequency

10.1.1.2 Modes and Operations

The ECSPI supports the modes described in the indicated sections:

- Master Mode
- Slave Mode
- Low Power Modes

As described in Operations, the ECSPI supports the operations described in the indicated sections:

- Typical Master Mode
  - Master Mode with SPI_RDY
  - Master Mode with Wait States
  - Master Mode with SS_CTL[3:0] Control
  - Master Mode with Phase Control
- Typical Slave Mode

10.1.2 Functional Description

This section provides a complete functional description of the ECSPI. The figure found here shows the relationship of SCLK and data lines while ECSPI has been configured with different POL and PHA settings.

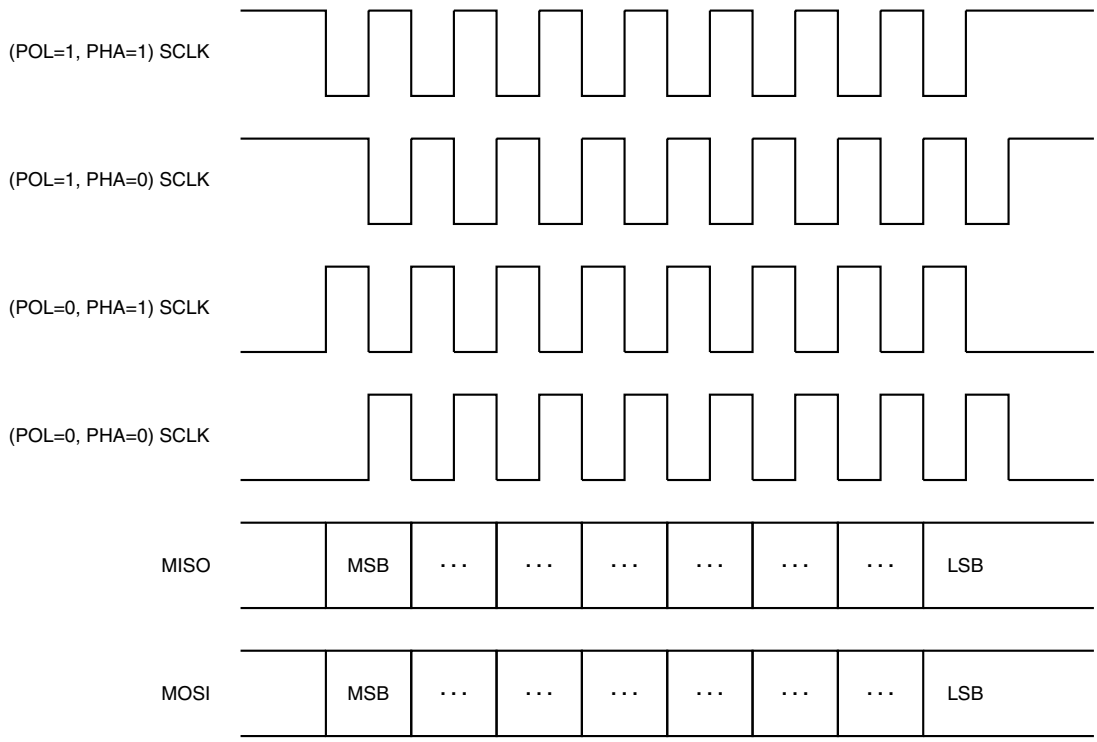


Figure 10-2. ECSPI SCLK, MISO, and MOSI Relationship

10.1.2.1 Master Mode

When the ECSPI is configured as a master, it uses a serial link to transfer data between the ECSPI and an external slave device.

One of the Chip Select (SS) signals and the clock signal (SCLK) are used to transfer data between two devices. If the external device is a transmit-only device, the ECSPI master's output port can be ignored and used for other purposes. In order to use the internal TXFIFO and RXFIFO, two auxiliary output signals, Chip Select (SS) and SPI_RDY, are used for data transfer rate control. Software can also configure the sample period control register to a fixed data transfer rate.

10.1.2.2 Slave Mode

When the ECSPI is configured as a slave, software can configure the ECSPI Control register to match the external SPI master's timing.

In this configuration, Chip Select (SS) becomes an input signal, and is used to control data transfers through the Shift register, as well as to load/store the data FIFO.

Slave mode only supports the case when ECSPIx_CONFIGREG[SS_CTL] is cleared. The accurate burst length should always be specified using the BURST_LENGTH parameter. ECSPIx_CONFIGREG[SS_CTL] set to 1 is not supported in slave mode.

10.1.2.3 Low Power Modes

The ECSPI does not operate under low power mode.

It holds its operation when its clock is gated off in master mode. In slave mode, the ECSPI does not respond when its clock is gated off.

10.1.2.4 Operations

The information found here describes the ECSPI's operations.

10.1.2.4.1 Typical Master Mode

The ECSPI master uses the Chip Select (SS) signal to enable an external SPI device, and uses the SCLK signal to transfer data in and out of the Shift register.

The SPI_RDY enables fast data communication with fewer software interrupts. By programming the ECSPI_PERIODREG register accordingly, the ECSPI can be used for a fixed data transfer rate.

When the ECSPI is in Master mode the SS, SCLK, and MOSI are output signals, and the MISO signal is an input.

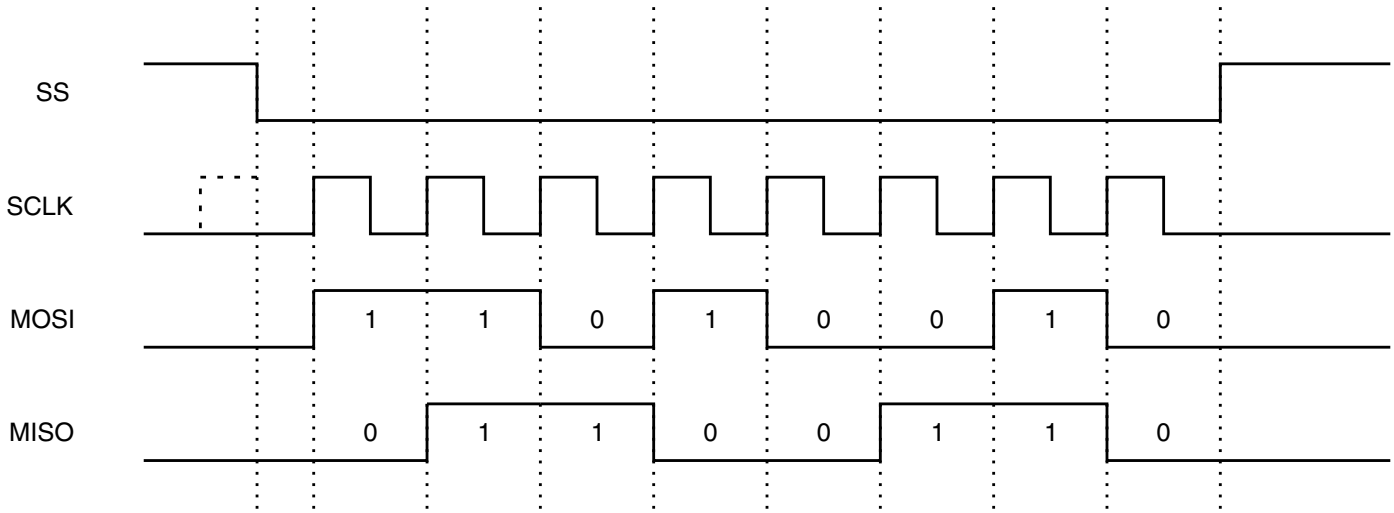


Figure 10-3. Typical SPI Burst (8-bit Transfer)

In the above figure, the Chip Select (SS) signal enables the selected external SPI device, and the SCLK synchronizes the data transfer. The MOSI and MISO signals change on rising edge of SCLK and the MISO signal is latched on the falling edge of the SCLK. The figure above shows a data of 0xD2 is shifted out, and a data of 0x66 is shifted in.

10.1.2.4.1.1 Master Mode with SPI_RDY

By default, the ECSPI does not use the SPI_RDY signal in master mode (MODE =1).

A SPI burst begins when the following events happen:

- The ECSPI is enabled, TXFIFO has data in it, and ECSPI_CONREG[XCH] bit or the ECSPI_CONREG[SMC] bit is set.
- When the SPI Data Ready Control (ECSPI_CONREG[DRCTL]) bits contains either 01 or 10, the SPI_RDY signal controls when a SPI burst starts.

A SPI burst is defined as a bus transaction that starts when the slave select is asserted and ends when the slave select is negated. The Chip Select (SS) signal will remain asserted until all the bits in a SPI burst are shifted out.

If ECSPI_CONREG[DRCTL] is set to 01, the SPI burst can be triggered only if a falling edge of the SPI_RDY signal has been detected.

The following figure shows the relationship between a SPI burst and the falling edge of SPI_RDY signal.

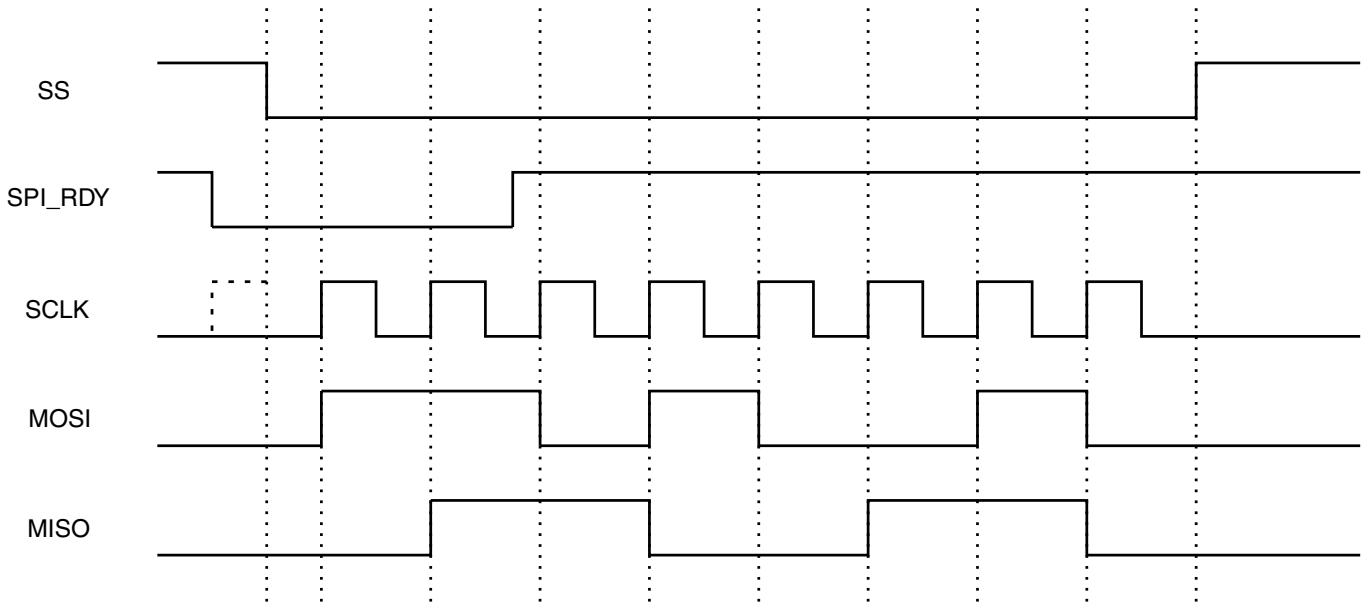


Figure 10-4. Relationship Between a SPI Burst and SPI_RDY: Falling-Edge Triggered

A SPI burst does not start until the falling edge of the SPI_RDY signal is detected. The next SPI burst starts when the next SPI_RDY falling edge is detected, after the last burst has finished.

If SPI Data Ready Control (ECSPI_CONREG[DRCTL]) is set to 10, the SPI burst can be triggered only if the SPI_RDY signal is low.

The following figure shows the relationship between a SPI burst and the SPI_RDY signal. The SPI burst does not begin until the SPI_RDY signal goes low. The ECSPI will keep transmitting SPI burst if the SPI_RDY signal remains low.

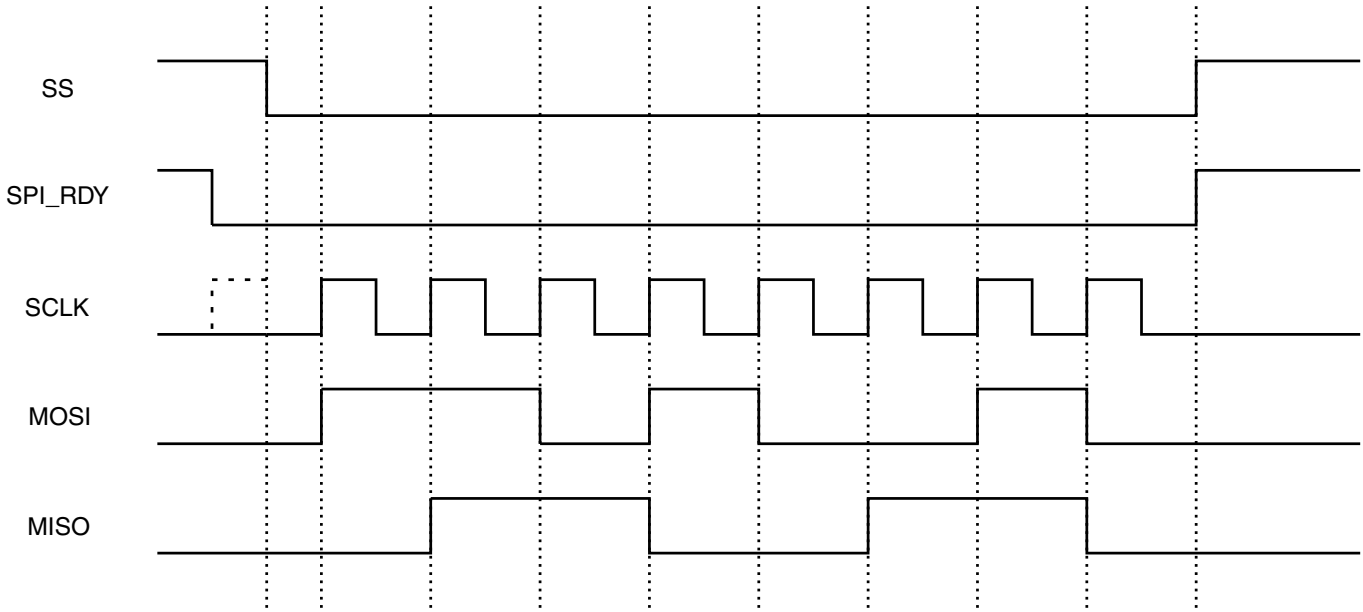


Figure 10-5. Relationship Between a SPI Burst and SPI_RDY: Low-Level Triggered

10.1.2.4.1.2 Master Mode with Wait States

Wait states can be inserted between SPI bursts. This provides a way for software to slow down the SPI burst to meet the timing requirements of a slower SPI device.

The following figure shows wait states inserted between SPI bursts.

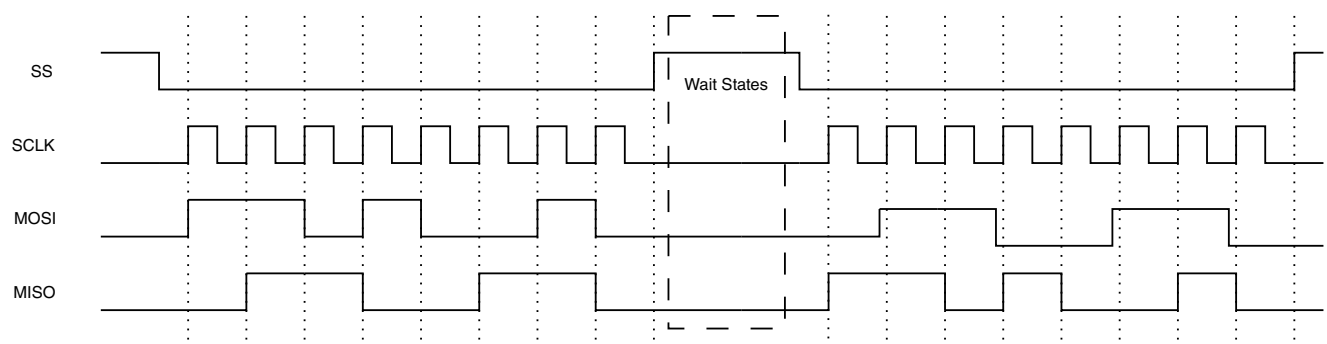


Figure 10-6. SPI Bursts with Wait States

In this case, the number of wait states is controlled by ECSPI_PERIODREG[SAMPLE PERIOD] and the wait states' clock source is selected by ECSPI_PERIODREG[CSRC].

10.1.2.4.1.3 Master Mode with SS_CTL[3:0] Control

The SPI SS Control (SS_CTL[3:0]) controls whether the current operation is single burst or multiple bursts.

When the SPI SS Wave Form Select (SS_CTL[3:0]) is set, the current operation is multiple bursts transfer. When the SPI SS Wave Form Select (SS_CTL[3:0]) bit is cleared, the current operation is single burst transfer. A SPI burst can contains multiple words as defined in the BURST LENGTH field of the ECSPI_CONREG register.

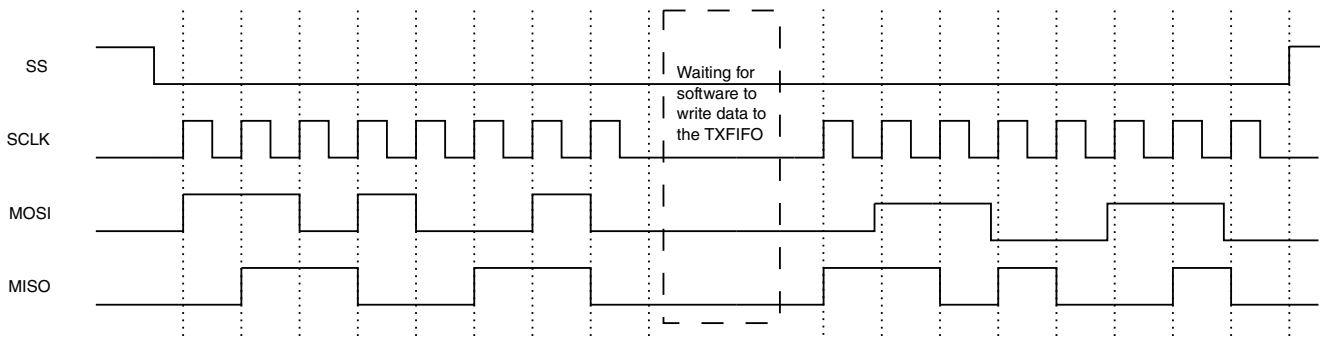


Figure 10-7. SPI Burst While SS_CTL[3:0] is Clear

In Figure 10-7, two 8-bit bursts in the TXFIFO have been combined and transmitted in one SPI burst. The maximum length of a single SPI burst is defined by the BURST LENGTH and limited by the FIFO size. (Figure 10-7 corresponds to a BURST LENGTH of 8.) This provides a way for transferring a longer SPI burst by writing data into TXFIFO while the ECSPI is transmitting.

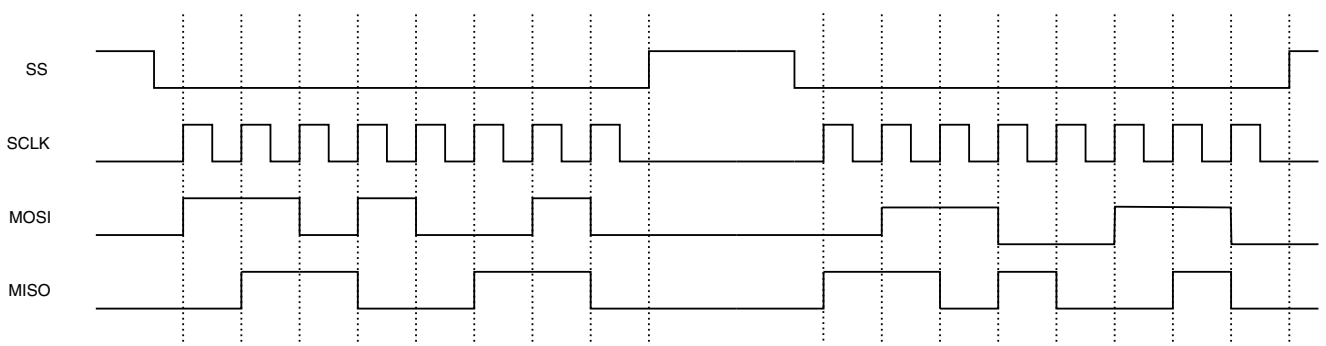


Figure 10-8. SPI Bursts While SS_CTL[3:0] is Set

In Figure 10-8, two FIFO entries are transmitted, one entry with each SPI burst. The ECSPI will continue to transmit SPI bursts until the TXFIFO is empty. When wait states can be inserted between SPI bursts, the SS will negate between SPI bursts until the wait states finish.

10.1.2.4.1.4 Master Mode with Phase Control

The Phase Control (ECSPI_CONREG[PHA]) bit controls how the transmit data shifts out and the receive data shifts in.

When the Phase control (ECSPI_CONREG[PHA]) bit is set, the transmit data will shift out on the rising edge of SCLK, and the receive data is latched on the falling edge of SCLK. The most-significant bit is output on the first rising SCLK edge.

When ECSPI_CONREG[PHA] is cleared, the transmit data is shifted out on the falling edge of SCLK and the receive data is latched on the rising edge of SCLK. The MSB is output when the host processor loads the transmitted data.

Inverting the SCLK polarity does not impact the edge-triggered operations because they are internal to the serial peripheral interface master. Figure 10-9 shows how SPI burst works with different POL and PHA configuration.

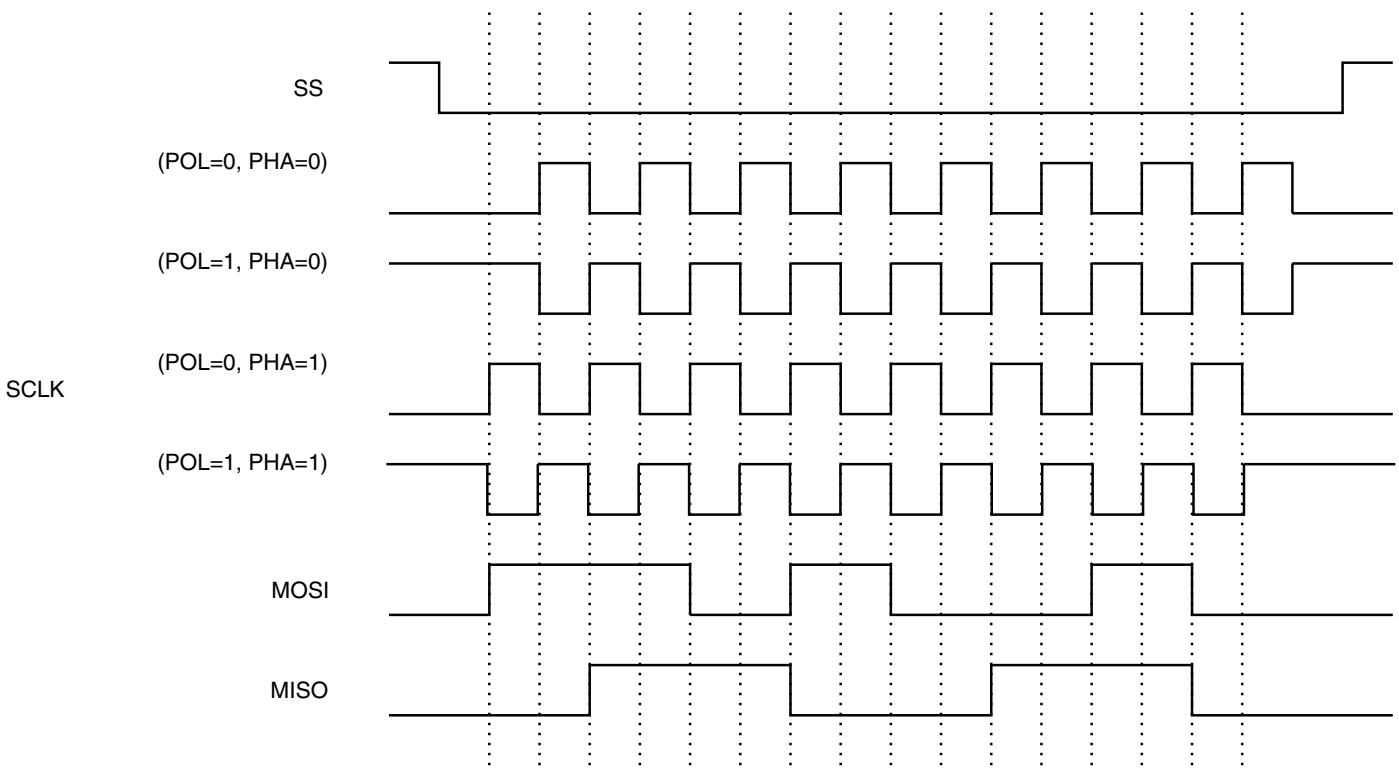


Figure 10-9. SPI Burst with Different POL and PHA Configurations

10.1.2.4.2 Typical Slave Mode

When the ECSPI is configured as a slave (Mode = 0), software can configure the ECSPI Control register to match the external SPI master's timing. In this configuration, SS becomes an input signal, and is used to latch data in and out of the internal data Shift registers, as well as to advance the data FIFO.

The SS, SCLK, and MOSI are inputs and MISO is output. Most of the timing diagrams are similar to the diagrams shown previously for the SPI in Master mode (Mode = 1), because the inputs come from a SPI master device.

However, the timing is different when SS is used to advance the data FIFO. When the SS_POL=0 is set while the ECSPI is configured in Slave mode, the data FIFO will advance on the rising edge of the SS signal. When the polarity is reversed (SS_POL = 1), the data FIFO will advance on the falling edge of the SS signal.

The figure below shows a SPI burst in which the data FIFO is advanced by the rising edge of the SS signal.

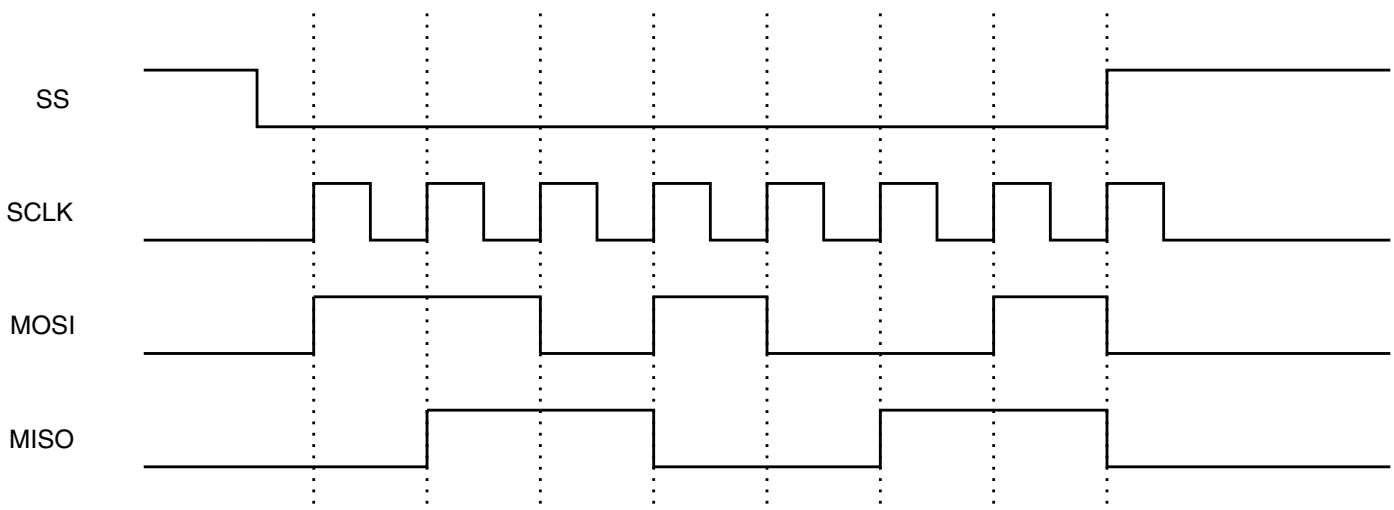


Figure 10-10. Advancing the Data FIFO on the Rising Edge of SS

In the above case, only the most significant 7 bits are loaded to the RXFIFO.

10.1.2.5 Reset

Whenever a device reset occurs, a reset is performed on the ECSPI, resetting all registers to their default values.

Software can reset the block using the CONREG[EN] bit; see ECSPI.

10.1.2.6 Interrupts

Interrupt control provides a way to manage the ECSPI FIFOs:

- For transmitting data, software can enable the TXFIFO empty, TXFIFO data request, and TXFIFO full interrupts to maintain the TXFIFO using an interrupt service routine.
- For receiving data, software can enable the RXFIFO ready, RXFIFO data request, and RXFIFO full interrupts to retrieve data from the RXFIFO using an interrupt service routine.

Other interrupt sources can be used to control or debug the SPI bursts:

- The transfer-completed interrupt means that there is no data left in the TXFIFO and that the data in the Shift register has been shifted out.
- The RXFIFO overflow interrupt means that the RXFIFO received more than 64 words and will not accept any other words.

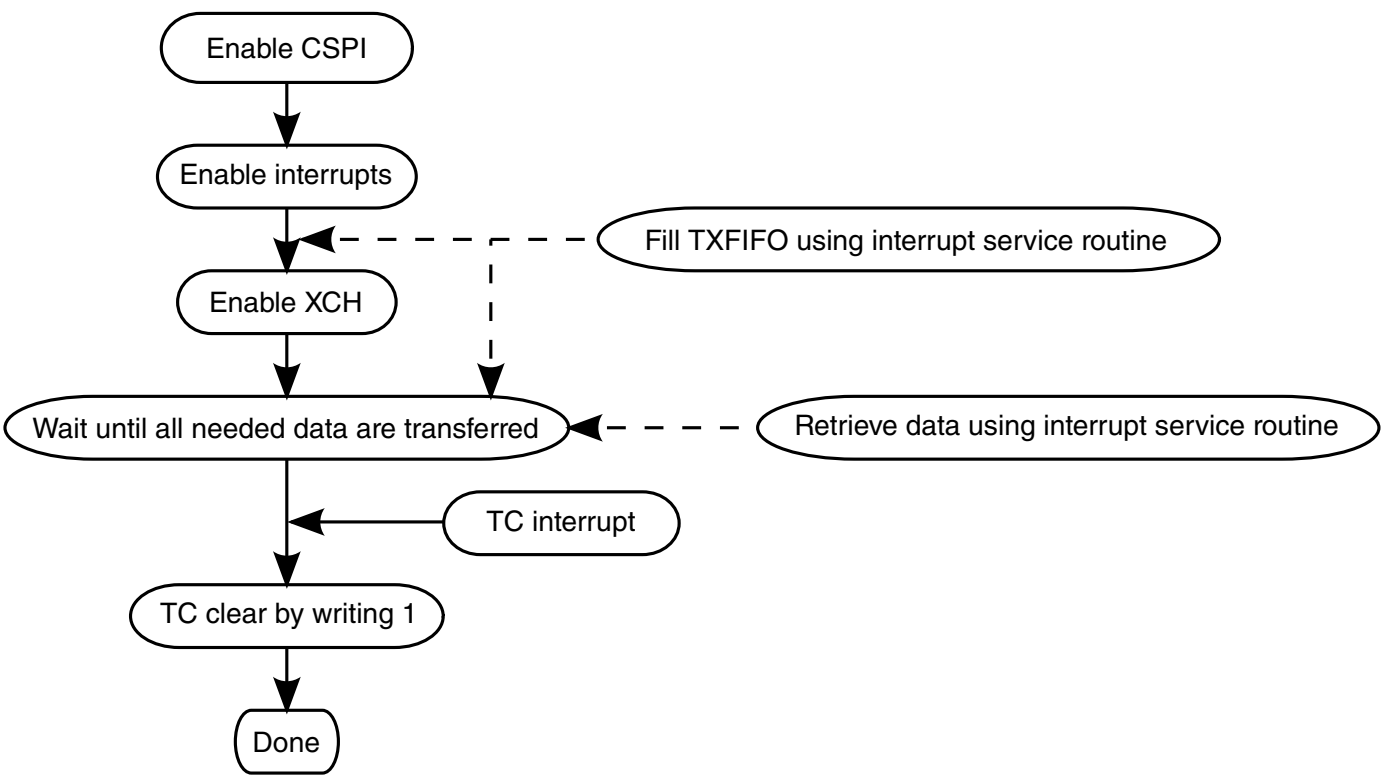


Figure 10-11. Program Sequence of SPI Burst Using Interrupt Control

NOTE

The TC bit does not provide a reliable indication that the transfer is complete. Under some conditions, the TC interrupt can occur before the transfer is completed. If the TC bit is used as an interrupt source, the XCH bit should be polled after the TC interrupt occurs to accurately confirm that the transfer is complete.

10.1.2.7 DMA

DMA control provides another method to utilize the FIFOs in the ECSPI. By using DMA request and acknowledge signals, larger amounts of data can be transferred, and will reduce interrupts and host processor loading. When the appropriate conditions are matched, the block will send out a DMA request.

The DMA can deal with the following conditions:

- TXFIFO empty
- TXFIFO data request
- RXFIFO data request
- RXFIFO full

The figure below shows a program sequence of SPI bursts using DMA control.

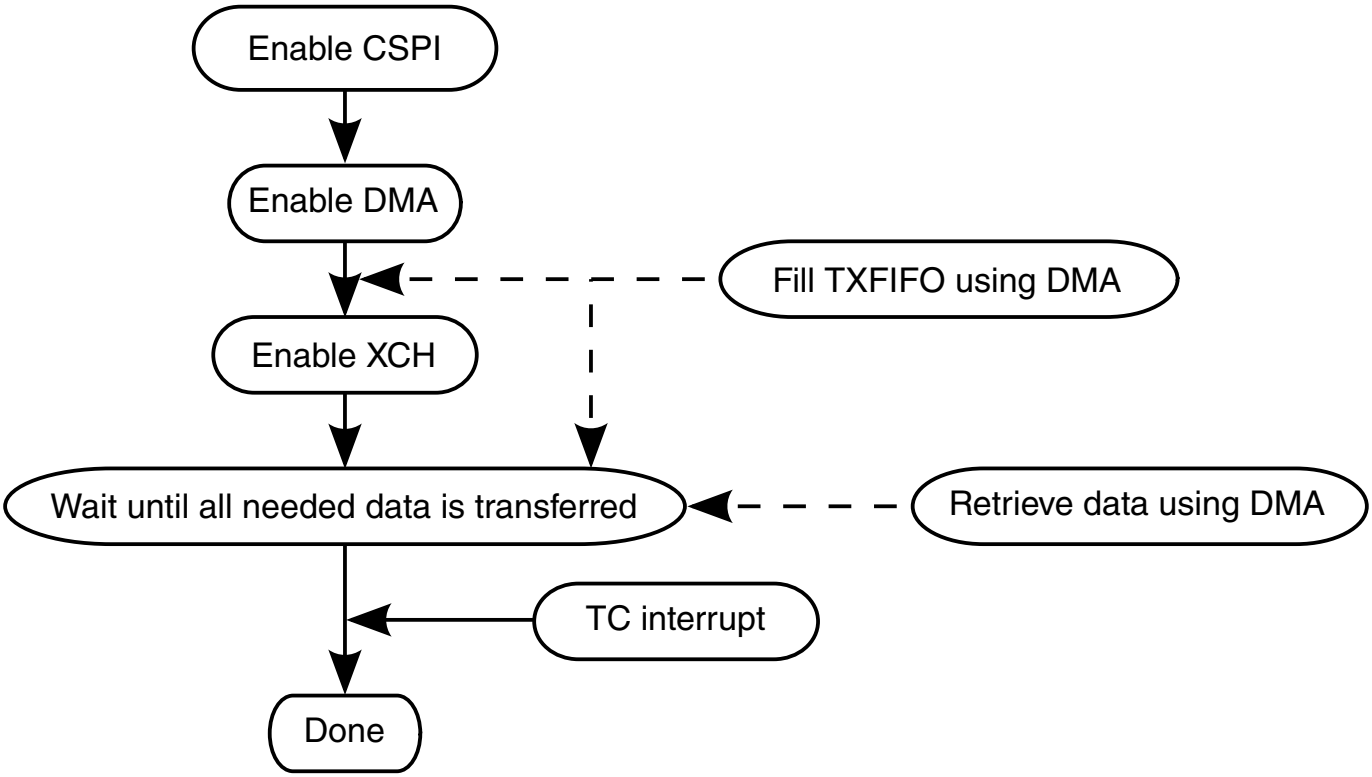


Figure 10-12. Program Sequence of SPI Burst Using DMA

NOTE

The TC bit does not provide a reliable indication that the transfer is complete. Under some conditions, the TC interrupt can occur before the transfer is completed. If the TC bit is used as an interrupt source, the XCH bit should be polled after the TC interrupt occurs to accurately confirm that the transfer is complete.

10.1.2.8 Byte Order

The ECSPI does not support byte re-ordering in hardware.

10.1.3 Initialization

This section provides initialization information for ECSPI.

To initialize the block:

1. Clear the EN bit in ECSPI_CONREG to reset the block.

2. Enable the clocks for ECSPI within the CCM.
3. Configure the Control Register and then set the EN bit in the ECSPI_CONREG to put ECSPI out of reset.
4. Configure corresponding IOMUX for ECSPI external signals.
5. Configure registers of ECSPI properly according to the specifications of the external SPI device.

10.1.4 Applications

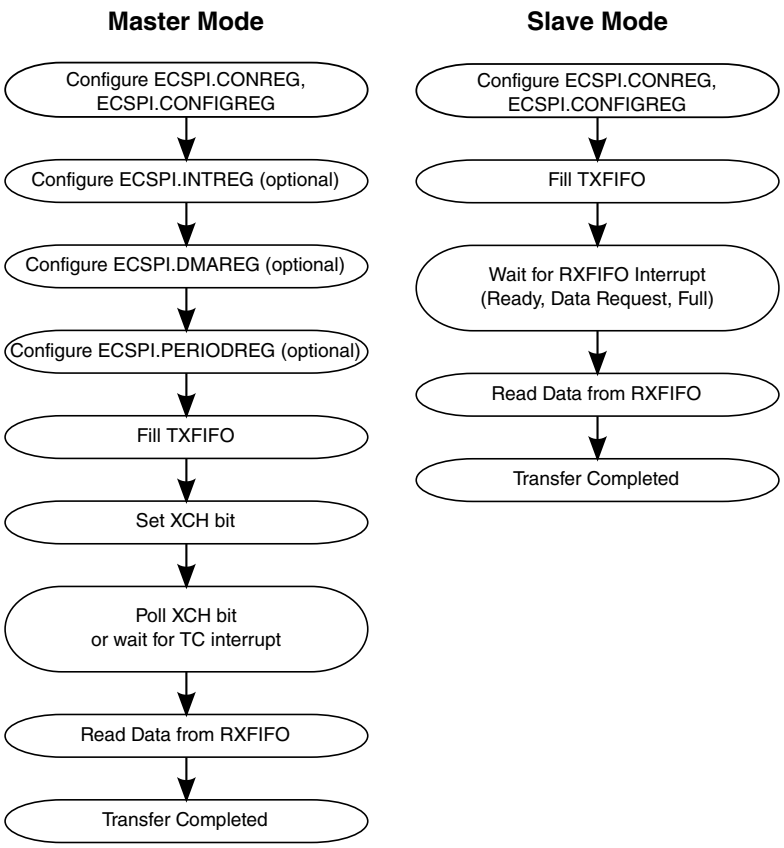


Figure 10-13. Flowchart of the ECSPI Operation

NOTE

The TC bit does not provide a reliable indication that the transfer is complete. Under some conditions, the TC interrupt can occur before the transfer is completed. If the TC bit is used as an interrupt source, the XCH bit should be polled after the TC interrupt occurs to accurately confirm that the transfer is complete.

10.1.5 ECSPI Memory Map/Register Definition

This section includes the block memory map and detailed descriptions of all registers. For the base address of a particular block instantiation, see the system memory map.

ECSPI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3082_0000	Receive Data Register (ECSPI1_RXDATA)	32	R	0000_0000h	10.1.5.1/ 2353
3082_0004	Transmit Data Register (ECSPI1_TXDATA)	32	W	0000_0000h	10.1.5.2/ 2354
3082_0008	Control Register (ECSPI1_CONREG)	32	R/W	0000_0000h	10.1.5.3/ 2355
3082_000C	Config Register (ECSPI1_CONFIGREG)	32	R/W	0000_0000h	10.1.5.4/ 2357
3082_0010	Interrupt Control Register (ECSPI1_INTREG)	32	R/W	0000_0000h	10.1.5.5/ 2359
3082_0014	DMA Control Register (ECSPI1_DMAREG)	32	R/W	0000_0000h	10.1.5.6/ 2360
3082_0018	Status Register (ECSPI1_STATREG)	32	R/W	0000_0003h	10.1.5.7/ 2362
3082_001C	Sample Period Control Register (ECSPI1_PERIODREG)	32	R/W	0000_0000h	10.1.5.8/ 2363
3082_0020	Test Control Register (ECSPI1_TESTREG)	32	R/W	0000_0000h	10.1.5.9/ 2365
3082_0040	Message Data Register (ECSPI1_MSGDATA)	32	W	0000_0000h	10.1.5.10/ 2366
3083_0000	Receive Data Register (ECSPI2_RXDATA)	32	R	0000_0000h	10.1.5.1/ 2353
3083_0004	Transmit Data Register (ECSPI2_TXDATA)	32	W	0000_0000h	10.1.5.2/ 2354
3083_0008	Control Register (ECSPI2_CONREG)	32	R/W	0000_0000h	10.1.5.3/ 2355
3083_000C	Config Register (ECSPI2_CONFIGREG)	32	R/W	0000_0000h	10.1.5.4/ 2357
3083_0010	Interrupt Control Register (ECSPI2_INTREG)	32	R/W	0000_0000h	10.1.5.5/ 2359
3083_0014	DMA Control Register (ECSPI2_DMAREG)	32	R/W	0000_0000h	10.1.5.6/ 2360
3083_0018	Status Register (ECSPI2_STATREG)	32	R/W	0000_0003h	10.1.5.7/ 2362

Table continues on the next page...

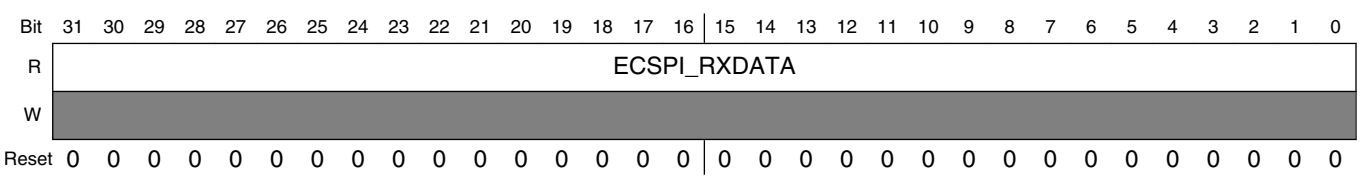
ECSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3083_001C	Sample Period Control Register (ECSPI2_PERIODREG)	32	R/W	0000_0000h	10.1.5.8/ 2363
3083_0020	Test Control Register (ECSPI2_TESTREG)	32	R/W	0000_0000h	10.1.5.9/ 2365
3083_0040	Message Data Register (ECSPI2_MSGDATA)	32	W	0000_0000h	10.1.5.10/ 2366
3084_0000	Receive Data Register (ECSPI3_RXDATA)	32	R	0000_0000h	10.1.5.1/ 2353
3084_0004	Transmit Data Register (ECSPI3_TXDATA)	32	W	0000_0000h	10.1.5.2/ 2354
3084_0008	Control Register (ECSPI3_CONREG)	32	R/W	0000_0000h	10.1.5.3/ 2355
3084_000C	Config Register (ECSPI3_CONFIGREG)	32	R/W	0000_0000h	10.1.5.4/ 2357
3084_0010	Interrupt Control Register (ECSPI3_INTREG)	32	R/W	0000_0000h	10.1.5.5/ 2359
3084_0014	DMA Control Register (ECSPI3_DMAREG)	32	R/W	0000_0000h	10.1.5.6/ 2360
3084_0018	Status Register (ECSPI3_STATREG)	32	R/W	0000_0003h	10.1.5.7/ 2362
3084_001C	Sample Period Control Register (ECSPI3_PERIODREG)	32	R/W	0000_0000h	10.1.5.8/ 2363
3084_0020	Test Control Register (ECSPI3_TESTREG)	32	R/W	0000_0000h	10.1.5.9/ 2365
3084_0040	Message Data Register (ECSPI3_MSGDATA)	32	W	0000_0000h	10.1.5.10/ 2366

10.1.5.1 Receive Data Register (ECSPIx_RXDATA)

The Receive Data register (ECSPI_RXDATA) is a read-only register that forms the top word of the 64 x 32 receive FIFO. This register holds the data received from an external SPI device during a data transaction. Only word-sized read operations are allowed.

Address: Base address + 0h offset



ECSPIx_RXDATA field descriptions

Field	Description
ECSPI_RXDATA	Receive Data. This register holds the top word of the receive data FIFO. The FIFO is advanced for each read of this register. The data read is undefined when the Receive Data Ready (RR) bit in the Interrupt Control/Status register is cleared. Zeros are read when ECSPI is disabled.

10.1.5.2 Transmit Data Register (ECSPIx_TXDATA)

The Transmit Data (ECSPI_TXDATA) register is a write-only data register that forms the bottom word of the 64 x 32 TXFIFO. The TXFIFO can be written to as long as it is not full, even when the SPI Exchange bit (XCH) in ECSPI_CONREG is set. This allows software to write to the TXFIFO during a SPI data exchange process. Writes to this register are ignored when the ECSPI is disabled (ECSPI_CONREG[EN] bit is cleared).

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	ECSPI_TXDATA																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ECSPIx_TXDATA field descriptions

Field	Description
ECSPI_TXDATA	Transmit Data. This register holds the top word of data loaded into the FIFO. Data written to this register must be a word operation. The number of bits actually transmitted is determined by the BURST_LENGTH field of the corresponding SPI Control register. If this field contains more bits than the number specified by BURST_LENGTH, the extra bits are ignored. For example, to transfer 10 bits of data, a 32-bit word must be written to this register. Bits 9-0 are shifted out and bits 31-10 are ignored. When the ECSPI is operating in Slave mode, zeros are shifted out when the FIFO is empty. Zeros are read when ECSPI is disabled.

10.1.5.3 Control Register (ECSPIx_CONREG)

The Control Register (ECSPI_CONREG) allows software to enable the ECSPI , configure its operating modes, specify the divider value, and SPI_RDY control signal, and define the transfer length.

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BURST_LENGTH												CHANNEL_ SELECT		DRCTL	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PRE_DIVIDER				POST_DIVIDER				CHANNEL_MODE				SMC	XCH	HT	EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ECSPIx_CONREG field descriptions

Field	Description
31–20 BURST_ LENGTH	Burst Length. This field defines the length of a SPI burst to be transferred. The Chip Select (SS) will remain asserted until all bits in a SPI burst are shifted out. A maximum of 2^12 bits can be transferred in a single SPI burst. In master mode, it controls the number of bits per SPI burst. Since the shift register always loads 32-bit data from transmit FIFO, only the n least-significant (n = BURST LENGTH + 1) will be shifted out. The remaining bits will be ignored. Number of Valid Bits in a SPI burst. 0x000 A SPI burst contains the 1 LSB in a word. 0x001 A SPI burst contains the 2 LSB in a word. 0x002 A SPI burst contains the 3 LSB in a word. ... 0x01F A SPI burst contains all 32 bits in a word. 0x020 A SPI burst contains the 1 LSB in first word and all 32 bits in second word. 0x021 A SPI burst contains the 2 LSB in first word and all 32 bits in second word. ... 0xFFE A SPI burst contains the 31 LSB in first word and 2^7 -1 words. 0xFFF A SPI burst contains 2^7 words.
19–18 CHANNEL_ SELECT	SPI CHANNEL SELECT bits. Select one of four external SPI Master/Slave Devices. In master mode, these two bits select the external slave devices by asserting the Chip Select (SSn) outputs. Only the selected Chip Select (SSn) signal can be active at a given time; the remaining three signals will be negated. 00 Channel 0 is selected. Chip Select 0 (SS0) will be asserted. 01 Channel 1 is selected. Chip Select 1 (SS1) will be asserted. 10 Channel 2 is selected. Chip Select 2 (SS2) will be asserted. 11 Channel 3 is selected. Chip Select 3 (SS3) will be asserted.

Table continues on the next page...

ECSPIx_CONREG field descriptions (continued)

Field	Description
17–16 DRCTL	SPI Data Ready Control. This field selects the utilization of the SPI_RDY signal in master mode. ECSPI checks this field before it starts an SPI burst. 00 The SPI_RDY signal is a don't care. 01 Burst will be triggered by the falling edge of the SPI_RDY signal (edge-triggered). 10 Burst will be triggered by a low level of the SPI_RDY signal (level-triggered). 11 Reserved.
15–12 PRE_DIVIDER	SPI Pre Divider. ECSPI uses a two-stage divider to generate the SPI clock. This field defines the pre-divider of the reference clock. 0000 Divide by 1. 0001 Divide by 2. 0010 Divide by 3. ... 1101 Divide by 14. 1110 Divide by 15. 1111 Divide by 16.
11–8 POST_DIVIDER	SPI Post Divider. ECSPI uses a two-stage divider to generate the SPI clock. This field defines the post-divider of the reference clock using the equation: 2^n . 0000 Divide by 1. 0001 Divide by 2. 0010 Divide by 4. ... 1110 Divide by 2^{14} . 1111 Divide by 2^{15} .
7–4 CHANNEL_ MODE	SPI CHANNEL MODE selects the mode for each SPI channel. CHANNEL MODE[3] is for SPI channel 3. CHANNEL MODE[2] is for SPI channel 2. CHANNEL MODE[1] is for SPI channel 1. CHANNEL MODE[0] is for SPI channel 0. 0 Slave mode. 1 Master mode.
3 SMC	Start Mode Control. This bit applies only to channels configured in Master mode (CHANNEL MODE = 1). It controls how the ECSPI starts a SPI burst, either through the SPI exchange bit, or immediately when the TXFIFO is written to. 0 SPI Exchange Bit (XCH) controls when a SPI burst can start. Setting the XCH bit will start a SPI burst or multiple bursts. This is controlled by the SPI SS Wave Form Select (SS_CTL). Refer to XCH and SS_CTL descriptions. 1 Immediately starts a SPI burst when data is written in TXFIFO.
2 XCH	SPI Exchange Bit. This bit applies only to channels configured in Master mode (CHANNEL MODE = 1). If the Start Mode Control (SMC) bit is cleared, writing a 1 to this bit starts one SPI burst or multiple SPI bursts according to the SPI SS Wave Form Select (SS_CTL). The XCH bit remains set while either the data exchange is in progress, or when the ECSPI is waiting for an active input if SPIRDY is enabled through DRCTL. This bit is cleared automatically when all data in the TXFIFO and the shift register has been shifted out.

Table continues on the next page...

ECSPIx_CONREG field descriptions (continued)

Field	Description
	0 Idle. 1 Initiates exchange (write) or busy (read).
1 HT	Hardware Trigger Enable. This bit is used in master mode only. It enables hardware trigger (HT) mode. Note, HT mode is not supported by this product. 0 Disable HT mode. 1 Enable HT mode.
0 EN	SPI Block Enable Control. This bit enables the ECSPI. This bit must be set before writing to other registers or initiating an exchange. Writing zero to this bit disables the block and resets the internal logic with the exception of the ECSPI_CONREG. The block's internal clocks are gated off whenever the block is disabled. 0 Disable the block. 1 Enable the block.

10.1.5.4 Config Register (ECSPIx_CONFIGREG)

The Config Register (ECSPI_CONFIGREG) allows software to configure each SPI channel, configure its operating modes, specify the phase and polarity of the clock, configure the Chip Select (SS), and define the HT transfer length. Note, HT mode is not supported by this product.

Address: Base address + Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved			HT_LENGTH					SCLK_CTL				DATA_CTL				SS_POL				SS_CTL				SCLK_POL				SCLK_PHA			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ECSPIx_CONFIGREG field descriptions

Field	Description
31–29 -	This field is reserved. Reserved
28–24 HT_LENGTH	HT LENGTH. This field defines the message length in HT Mode. Note, HT mode is not supported by this product. The length in bits of one message is (HT LENGTH + 1).
23–20 SCLK_CTL	SCLK CTL. This field controls the inactive state of SCLK for each SPI channel. SCLK CTL[3] is for SPI channel 3. SCLK CTL[2] is for SPI channel 2. SCLK CTL[1] is for SPI channel 1. SCLK CTL[0] is for SPI channel 0.

Table continues on the next page...

ECSPIx_CONFIGREG field descriptions (continued)

Field	Description
	0 Stay low. 1 Stay high.
19–16 DATA_CTL	DATA CTL. This field controls inactive state of the data line for each SPI channel. DATA CTL[3] is for SPI channel 3. DATA CTL[2] is for SPI channel 2. DATA CTL[1] is for SPI channel 1. DATA CTL[0] is for SPI channel 0. 0 Stay high. 1 Stay low.
15–12 SS_POL	SPI SS Polarity Select. In both Master and Slave modes, this field selects the polarity of the Chip Select (SS) signal. SS POL[3] is for SPI channel 3. SS POL[2] is for SPI channel 2. SS POL[1] is for SPI channel 1. SS POL[0] is for SPI channel 0. 0 Active low. 1 Active high.
11–8 SS_CTL	SPI SS Wave Form Select. In master mode, this field controls the output wave form of the Chip Select (SS) signal when the SMC (Start Mode Control) bit is cleared. The SS_CTL are ignored if the SMC bit is set. SS CTL[3] is for SPI channel 3. SS CTL[2] is for SPI channel 2. SS CTL[1] is for SPI channel 1. SS CTL[0] is for SPI channel 0. In slave mode, this bit controls when the SPI burst is completed. An SPI burst is completed by the Chip Select (SS) signal edges. (SSPOL = 0: rising edge; SSPOL = 1: falling edge) The RXFIFO is advanced whenever a Chip Select (SS) signal edge is detected or the shift register contains 32-bits of valid data. 0 In master mode - only one SPI burst will be transmitted. 1 In master mode - Negate Chip Select (SS) signal between SPI bursts. Multiple SPI bursts will be transmitted. The SPI transfer will automatically stop when the TXFIFO is empty. 0 In slave mode - an SPI burst is completed when the number of bits received in the shift register is equal to (BURST LENGTH + 1). Only the n least-significant bits (n = BURST LENGTH[4:0] + 1) of the first received word are valid. All bits subsequent to the first received word in RXFIFO are valid. 1 Reserved
7–4 SCLK_POL	SPI Clock Polarity Control. This field controls the polarity of the SCLK signal. See Figure 10-9 for more information. SCLK_POL[3] is for SPI channel 3. SCLK_POL[2] is for SPI channel 2. SCLK_POL[1] is for SPI channel 1. SCLK_POL[0] is for SPI channel 0.

Table continues on the next page...

ECSPIx_CONFIGREG field descriptions (continued)

Field	Description
	0 Active high polarity (0 = Idle). 1 Active low polarity (1 = Idle).
SCLK_PHA	SPI Clock/Data Phase Control. This field controls the clock/data phase relationship. See Figure 10-9 for more information. SCLK PHA[3] is for SPI channel 3. SCLK PHA[2] is for SPI channel 2. SCLK PHA[1] is for SPI channel 1. SCLK PHA[0] is for SPI channel 0. 0 Phase 0 operation. 1 Phase 1 operation.

10.1.5.5 Interrupt Control Register (ECSPIx_INTREG)

The Interrupt Control Register (ECSPI_INTREG) enables the generation of interrupts to the host processor. If the ECSPI is disabled, this register reads zero.

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								TCEN	ROEN	RFEN	RDREN	RRE N	TFEN	TDREN	TEEN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ECSPIx_INTREG field descriptions

Field	Description
31–8 -	This field is reserved. Reserved
7 TCEN	Transfer Completed Interrupt enable. This bit enables the Transfer Completed Interrupt. 0 Disable 1 Enable

Table continues on the next page...

ECSPIx_INTREG field descriptions (continued)

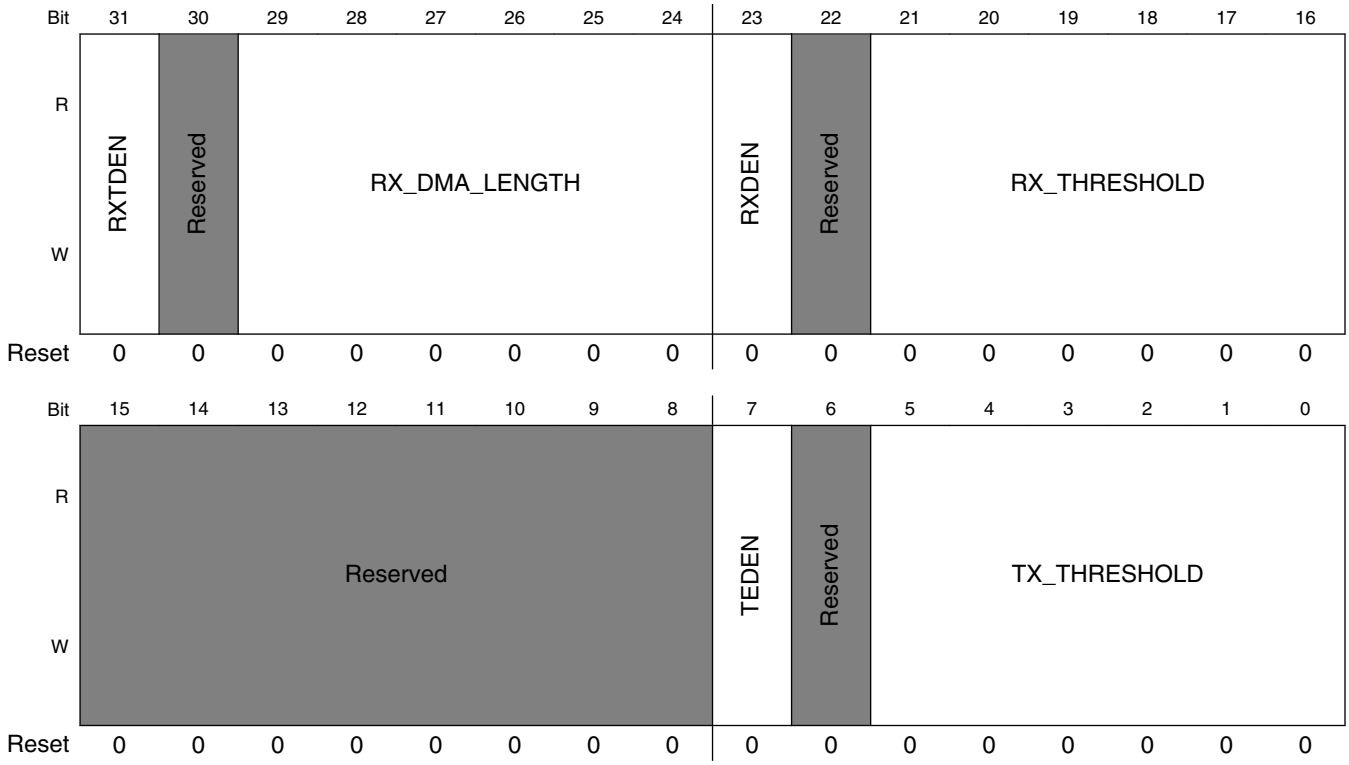
Field	Description
6 ROEN	RXFIFO Overflow Interrupt enable. This bit enables the RXFIFO Overflow Interrupt. 0 Disable 1 Enable
5 RFEN	RXFIFO Full Interrupt enable. This bit enables the RXFIFO Full Interrupt. 0 Disable 1 Enable
4 RDREN	RXFIFO Data Request Interrupt enable. This bit enables the RXFIFO Data Request Interrupt when the number of data entries in the RXFIFO is greater than RX_THRESHOLD. 0 Disable 1 Enable
3 RREN	RXFIFO Ready Interrupt enable. This bit enables the RXFIFO Ready Interrupt. 0 Disable 1 Enable
2 TFEN	TXFIFO Full Interrupt enable. This bit enables the TXFIFO Full Interrupt. 0 Disable 1 Enable
1 TDREN	TXFIFO Data Request Interrupt enable. This bit enables the TXFIFO Data Request Interrupt when the number of data entries in the TXFIFO is less than or equal to TX_THRESHOLD. 0 Disable 1 Enable
0 TEEN	TXFIFO Empty Interrupt enable. This bit enables the TXFIFO Empty Interrupt. 0 Disable 1 Enable

10.1.5.6 DMA Control Register (ECSPIx_DMAREG)

The Direct Memory Access Control Register (ECSPI_DMAREG) provides software a way to use an on-chip DMA controller for ECSPI data. Internal DMA request signals enable direct data transfers between the ECSPI FIFOs and system memory. The ECSPI sends out DMA requests when the appropriate FIFO conditions are matched.

If the ECSPI is disabled, this register is read as 0.

Address: Base address + 14h offset



ECSPIx_DMAREG field descriptions

Field	Description
31 RXTDEN	RXFIFO TAIL DMA Request/Interrupt Enable. This bit enables an internal counter that is increased at each read of the RXFIFO. This counter is cleared automatically when it reaches RX DMA LENGTH. If the number of words remaining in the RXFIFO is greater than or equal to RX DMA LENGTH, a DMA request/ interrupt is generated even if it is less than or equal to RX_THRESHOLD. 0 Disable 1 Enable
30 -	This field is reserved. Reserved
29–24 RX_DMA_ LENGTH	RX DMA LENGTH. This field defines the burst length of a DMA operation. Applies only when RXTDEN is set.
23 RXDEN	RXFIFO DMA Request Enable. This bit enables/disables the RXFIFO DMA Request. 0 Disable 1 Enable
22 -	This field is reserved. Reserved
21–16 RX_ THRESHOLD	RX THRESHOLD. This field defines the FIFO threshold that triggers a RX DMA/INT request. A RX DMA/INT request is issued when the number of data entries in the RXFIFO is greater than RX_THRESHOLD.
15–8 -	This field is reserved. Reserved

Table continues on the next page...

ECSPIx_DMAREG field descriptions (continued)

Field	Description
7 TEDEN	TXFIFO Empty DMA Request Enable. This bit enables/disables the TXFIFO Empty DMA Request. 0 Disable 1 Enable
6 -	This field is reserved. Reserved
TX_ THRESHOLD	TX THRESHOLD. This field defines the FIFO threshold that triggers a TX DMA/INT request. A TX DMA/INT request is issued when the number of data entries in the TXFIFO is not greater than TX_THRESHOLD.

10.1.5.7 Status Register (ECSPIx_STATREG)

The ECSPI Status Register (ECSPI_STATREG) reflects the status of the ECSPI's operating condition. If the ECSPI is disabled, this register reads 0x0000_0003.

Address: Base address + 18h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								TC	RO	RF	RDR	RR	TF	TDR	TE
W									w1c	w1c						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

ECSPIx_STATREG field descriptions

Field	Description
31–8 -	This field is reserved. Reserved
7 TC	Transfer Completed Status bit. Writing 1 to this bit clears it. **NOTE:** The TC bit does not provide a reliable indication that the transfer is complete. Under some conditions, the TC interrupt can occur before the transfer is completed. If the TC bit is used as an interrupt source, the XCH bit should be polled after the TC interrupt occurs to accurately confirm that the transfer is complete. 0 Transfer in progress. 1 Transfer completed.
6 RO	RXFIFO Overflow. When set, this bit indicates that RXFIFO has overflowed. Writing 1 to this bit clears it.

Table continues on the next page...

ECSPIx_STATREG field descriptions (continued)

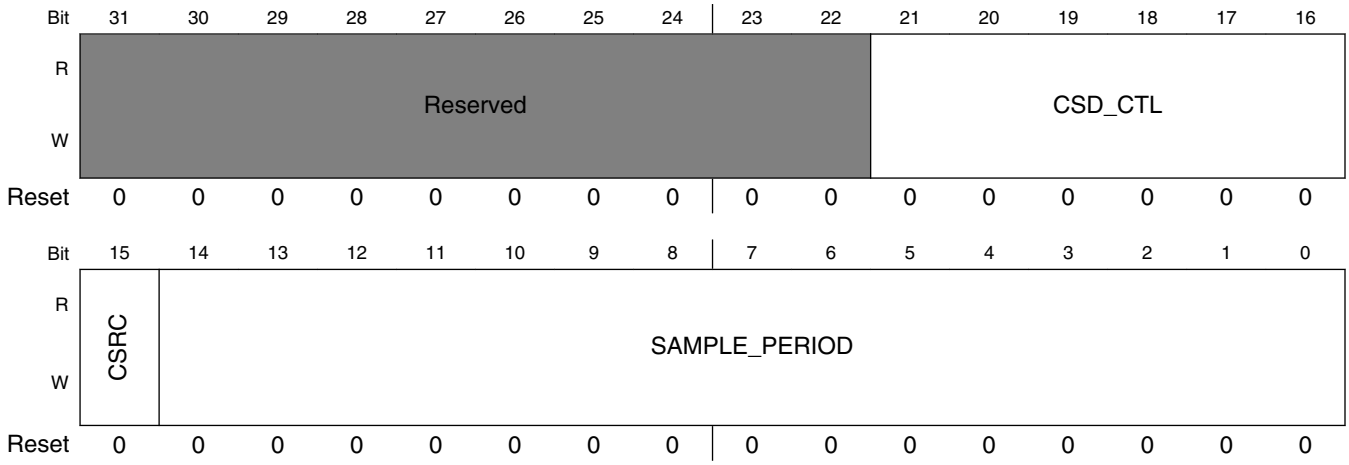
Field	Description
	0 RXFIFO has no overflow. 1 RXFIFO has overflowed.
5 RF	RXFIFO Full. This bit is set when the RXFIFO is full. 0 Not Full. 1 Full.
4 RDR	RXFIFO Data Request. 0 When RXTDE is set - Number of data entries in the RXFIFO is not greater than RX_THRESHOLD. 1 When RXTDE is set - Number of data entries in the RXFIFO is greater than RX_THRESHOLD or a DMA TAIL DMA condition exists. 0 When RXTDE is clear - Number of data entries in the RXFIFO is not greater than RX_THRESHOLD. 1 When RXTDE is clear - Number of data entries in the RXFIFO is greater than RX_THRESHOLD.
3 RR	RXFIFO Ready. This bit is set when one or more words are stored in the RXFIFO. 0 No valid data in RXFIFO. 1 More than 1 word in RXFIFO.
2 TF	TXFIFO Full. This bit is set when if the TXFIFO is full. 0 TXFIFO is not Full. 1 TXFIFO is Full.
1 TDR	TXFIFO Data Request. 0 Number of valid data slots in TXFIFO is greater than TX_THRESHOLD. 1 Number of valid data slots in TXFIFO is not greater than TX_THRESHOLD.
0 TE	TXFIFO Empty. This bit is set if the TXFIFO is empty. 0 TXFIFO contains one or more words. 1 TXFIFO is empty.

10.1.5.8 Sample Period Control Register (ECSPIx_PERIODREG)

The Sample Period Control Register (ECSPI_PERIODREG) provides software a way to insert delays (wait states) between consecutive SPI transfers. Control bits in this register select the clock source for the sample period counter and the delay count indicating the number of wait states to be inserted between data transfers.

The delay counts apply only when the current channel is operating in Master mode (ECSPI_CONREG[CHANNEL MODE] = 1).ECSPI_PERIODREG also contains the CSD CTRL field used to insert a delay between the Chip Select's active edge and the first SPI Clock edge.

Address: Base address + 1Ch offset



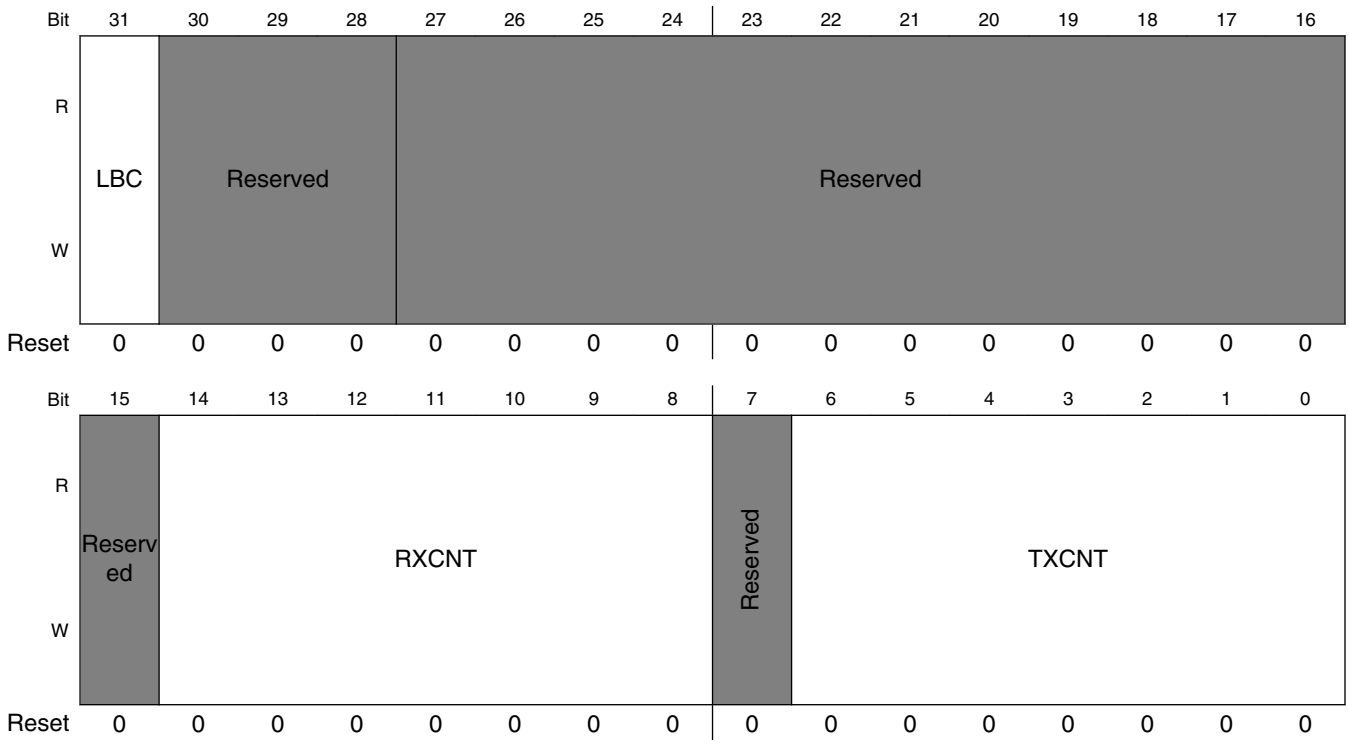
ECSPIx_PERIODREG field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–16 CSD_CTL	Chip Select Delay Control bits. This field defines how many SPI clocks will be inserted between the chip select's active edge and the first SPI clock edge. The range is from 0 to 63.
15 CSRC	Clock Source Control. This bit selects the clock source for the sample period counter. 0 SPI Clock (SCLK) 1 Low-Frequency Reference Clock (32.768 KHz)
SAMPLE_ PERIOD	Sample Period Control. These bits control the number of wait states to be inserted in data transfers. During the idle clocks, the state of the SS output will operate according to the SS_CTL control field in the ECSPI_CONREG register. 0x0000 0 wait states inserted 0x0001 1 wait state inserted ... ... 0x7FFE 32766 wait states inserted 0x7FFF 32767 wait states inserted

10.1.5.9 Test Control Register (ECSPIx_TESTREG)

The Test Control Register (ECSPI_TESTREG) provides software a mechanism to internally connect the receive and transmit devices of the ECSPI, and monitor the contents of the receive and transmit FIFOs.

Address: Base address + 20h offset



ECSPIx_TESTREG field descriptions

Field	Description
31 LBC	Loop Back Control. This bit is used in Master mode only. When this bit is set, the ECSPI connects the transmitter and receiver sections internally, and the data shifted out from the most-significant bit of the shift register is looped back into the least-significant bit of the Shift register. In this way, a self-test of the complete transmit/receive path can be made. The output pins are not affected, and the input pins are ignored. 0 Not connected. 1 Transmitter and receiver sections internally connected for Loopback.
30–28 -	This field is reserved. Reserved, all bits should be ignored.
27–15 -	This field is reserved. Reserved
14–8 RXCNT	RXFIFO Counter. This field indicates the number of words in the RXFIFO.

Table continues on the next page...

ECSPIx_TESTREG field descriptions (continued)

Field	Description
7 -	This field is reserved. Reserved
TXCNT	TXFIFO Counter. This field indicates the number of words in the TXFIFO.

10.1.5.10 Message Data Register (ECSPIx_MSGDATA)

The Message Data Register (ECSPI_MSGDATA) forms the top word of the 16 x 32 MSG Data FIFO. Only word-size accesses are allowed for this register. Reads to this register return zero, and writes to this register store data in the MSG Data FIFO.

Address: Base address + 40h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	ECSPI_MSGDATA																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ECSPIx_MSGDATA field descriptions

Field	Description
ECSPI_ MSGDATA	ECSPI_MSGDATA holds the top word of MSG Data FIFO. The MSG Data FIFO is advanced for each write of this register. The data read is zero. The data written to this register is stored in the MSG Data FIFO.

10.2 FlexSPI Controller (FlexSPI)

10.2.1 Overview

Flexible Serial Peripheral Interface (FlexSPI) host controller supports two SPI channels and up to 4 external devices. Each channel supports Single/Dual/Quad/Octal mode data transfer (1/2/4/8 bidirectional data lines).

NOTE

FlexSPI configuration is dependent on the chip configuration. Please refer to the system-level sections for boot and pinmux for chip specific information regarding the modes and number of devices supported.

10.2.1.1 Features

FlexSPI block supports following features:

- Flexible sequence engine (LUT table) to support various vendor devices
  - Serial NOR Flash or other device with similar SPI protocol as Serial NOR Flash
  - Serial NAND Flash
  - HyperBus device (HyperFlash/HyperRAM)
  - FPGA device
- Flash access mode
  - Single/Dual/Quad/Octal mode
  - SDR/DDR mode
  - Individual/Parallel mode
- Support sampling clock mode:
  - Internal dummy read strobe loopbacked internally
  - Internal dummy read strobe loopbacked from pad
  - Flash provided read strobe
- Automatic Data Learning to select correct sample clock phase
- Memory mapped read/write access by AHB Bus
  - AHB RX Buffer implemented to reduce read latency. Total AHB RX Buffer size: 256 * 64 Bits
  - 16 AHB masters supported with priority for read access
  - 8 flexible and configurable buffers in AHB RX Buffer
  - AHB TX Buffer implemented to buffer all write data from one AHB burst. AHB TX Buffer size: 8 * 64 Bits
  - All AHB masters share this AHB TX Buffer. No AHB master number limitation for Write Access.
- Software triggered Flash read/write access by IP Bus
  - IP RX FIFO implemented to buffer all read data from External device. FIFO size: 64 * 64 Bits
  - IP TX FIFO implemented to buffer all Write data to External device. FIFO size: 128 * 64 Bits
  - DMA support to fill IP TX FIFO
  - DMA support to read IP RX FIFO
  - SCLK stopped when reading flash data and IP RX FIFO is full
  - SCLK stopped when writing flash data and IP TX FIFO is empty

10.2.1.2 Block diagram

The block diagram of FlexSPI is indicated in following figure:

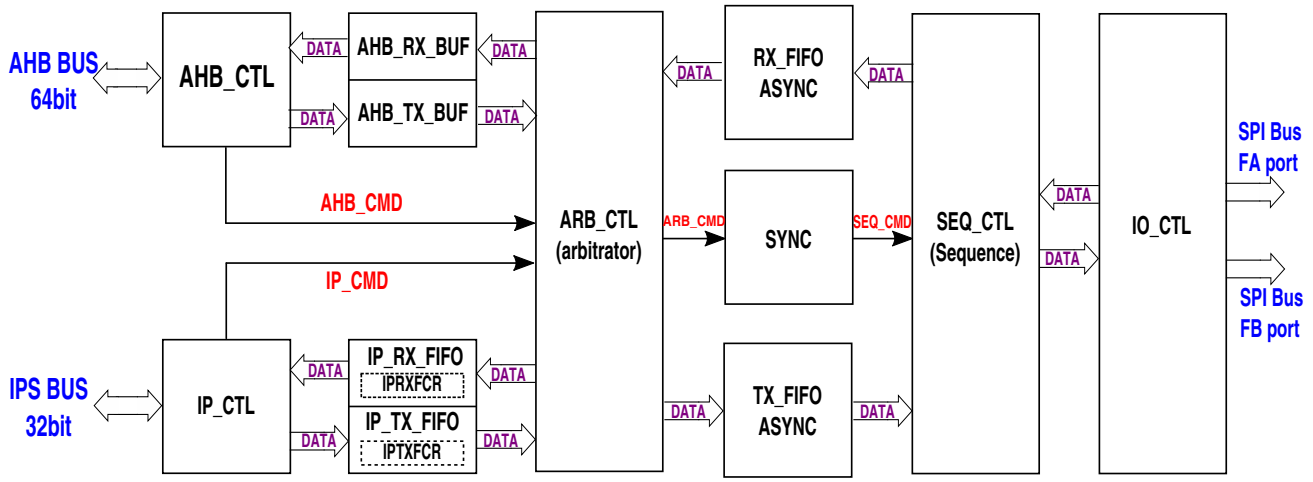


Figure 10-14. FlexSPI block diagram

10.2.1.3 Operation Modes

This section provides information about the modes FlexSPI could be used.

- Module Disable mode

  This mode is low power mode in FlexSPI module.

  In module disable mode, AHB clock and serial clock domain will be gated off internally, but IPS Bus clock is not gated off. Control and status register read/write access is available except LUT/IP RX FIFO/IP TX FIFO read/write access. Serial Flash Memory access is also not available.

  This mode is entered by setting MCR0[MDIS], and exited by clearing MCR0[MDIS].

- Doze mode

  This mode is low power mode in SOC system.

  When the system requires FlexSPI to enter doze mode and MCR0[DOZEEN] is set, the FlexSPI will wait for all transactions to complete (STS0[ARBIDLE]=0x1) and enter doze mode. In doze mode, the AHB clock and serial clock domain will be gated off internally, but IPS Bus clock is not gated off. Control and status register read/write access is available except LUT/IP RX FIFO/IP TX FIFO read/write access. Serial Flash Memory access is also not available.

  This mode is entered by system request, and exited by deasserting this system request.

- Stop mode

  This mode is low power mode in SOC system.

  When the system requires FlexSPI to enter stop mode, FlexSPI will wait for all transactions to complete (STS0[ARBIDLE]=0x1) and return ACK handshake to system. After ACK handshake returned, IP will gate off the AHB clock and serial clock domain internally, the system can gate the AHB Bus clock, IPS Bus clock and serial clock in system level.

  This mode is entered by system stop mode request. When all command sequences finish, FlexSPI enters stop mode and returns ACK handshake. This mode is exited by deasserting system stop mode request, the ACK handshake is also deasserted immediately.

- Normal mode

  In normal mode, all clocks are not gated internally. Normal register access and serial flash memory access is available.

10.2.2 Glossary for FlexSPI module

Table 10-1. Glossary

Term	Definition
AHB Command	AHB Command is one or several Serial Flash Memory access command sequences triggered by AHB read/write access to AHB address ranged mapped for Serial Flash Memory.
ASFM_BASE	Base address of AHB address space mapped to Serial Flash Memory.
ARDF_BASE	Base address of AHB address space mapped to IP RX FIFO.
ATDF_BASE	Base address of AHB address space mapped to IP TX FIFO.
Set	Write 1 to register bit to establish logic level one on the bit.
Clear	Write 0 to register bit to establish logic level zero on the bit.
Command Sequence	A command sequence is 4*32 bits sequence code consisting of up to 8 instructions. Command sequence should be programmed in LUT according to Flash command. When a command sequence executed, the chip selection signal becomes asserted. When a command sequence execution finished, the chip selection signal becomes negated.
DDR	Dual data transfer rate for flash access mode Flash receive data on both SCLK rise and fall edge and transmit data on both SCLK rise and fall edge.
Endianness	Byte Ordering scheme.
Field	Two or more register bits grouped together.
Fill	To add entries to a FIFO by software or hardware.

Table continues on the next page...

Table 10-1. Glossary (continued)

Term	Definition
Instruction Code	16 bits code defining the type of command to be executed on FlexSPI interface.
IP Command	IP Command is one or several Serial Flash Memory access command sequences triggered by set register bit IPCMD.TRG.
Negated	A signal that is negated is in its inactive state. An active low signal changes from logic level 0 to logic level 1 when negated, and an active high signal changes from logic level 1 to logic level 0.
SDR	Single data transfer rate for flash access mode Flash receive data on SCLK rise edge and transmit data on SCLK fall edge.
Set	To set a bit or bits means to establish logic level one on the bit or bits.
SFM	Serial Flash Memory
Individual Mode	Access to a single, individual serial flash device at a time.
Parallel Mode	Read/Program Access to two serial flash devices in parallel (Port A and Port B). FlexSPI will split flash program data before transmitting to Flash or merge flash read data before putting into IP RX FIFO or AHB RX Buffer automatically.
Memory Command	Serial access Command for reading/programming/configuring. A memory command may consist one or several command sequences, for each command sequence the Chip selection signal will be asserted and deasserted once.
LUT	Look-up table to preserve command sequence. LUT is programmed by software with command sequences which is used to issue memory commands.

10.2.3 External Signal Description

This section provides the external signal information of the FlexSPI module.

Table 10-2. External Signal List

Signal Name	Function	Direction	Description
A_SS0_B	Peripheral Chip Select Flash A1	O	This signal is the chip select for the serial flash device A1. Port name: PCSA1
A_SS1_B	Peripheral Chip Select Flash A2	O	This signal is the chip select for the serial flash device A2. Port name: PCSA2
B_SS0_B	Peripheral Chip Select Flash B1	O	This signal is the chip select for the serial flash device B1. Port name: PCSB1
B_SS1_B	Peripheral Chip Select Flash B2	O	This signal is the chip select for the serial flash device B2. Port name: PCSB2
A_SCLK	Serial Clock Flash A	O	This signal is the serial clock output to the serial flash device A.

Table continues on the next page...

Table 10-2. External Signal List (continued)

Signal Name	Function	Direction	Description
			Half clock frequency of serial clock root in DDR mode, and same frequency as serial clock root in SDR mode. Clock output toggles during the whole flash access sequence. Port name: SCKA
B_SCLK	Serial Clock Flash B	O	This signal is the serial clock output to the serial flash device B. Half clock frequency of serial clock root in DDR mode, and same frequency as serial clock root in SDR mode. Clock output toggles during the whole flash access sequence. **NOTE:** • B_SCLK may be optionally used as A_SCLK's differential clock output by setting Register Field MCR2[SCKBDIFFOPT]=1'b1 Port name: SCKB
A_SCLK2	Serial Clock 2 Flash A	O	This signal is the serial clock output 2, to the serial flash device A. A_SCLK2 is 90 degree phase shifted from A_SCLK in DDR mode. No clock output in SDR mode (Clock gated). Please refer to FlexSPI Input Timing for more detail. Port name: SCK2A
B_SCLK2	Serial Clock 2 Flash B	O	Similar to A_SCLK2. Port name: SCK2B
A_DATAn	Serial I/O Flash A	I/O	These signals are the data I/O lines to/from the serial flash device A. Port name: SIOAn
B_DATAn	Serial I/O Flash B	I/O	These signals are the data I/O lines to/from the serial flash device B. Port name: SIOBn
A_DQS	Data Strobe signal Flash A	I/O	Data strobe signal for port A. There are three functions for this signal: • Driven with Read Strobe by external device: Some flash devices provide the Read Strobe signal together with Read data. In this case, this pad may need a **pull down** resistor if external device drives this pad only when reading flash data. • Driven with Latency Information by external device:

Table continues on the next page...

Table 10-2. External Signal List (continued)

Signal Name	Function	Direction	Description
			Certain devices use this pin to indicate the dummy cycles needed (before Program/ Read data transfer) such as HyperRAM/ HyperFlash. • Loopback dummy read strobe: FlexSPI controller provides internal dummy read strobe for flash read data. Higher read frequency can be achieved by looping back this dummy read strobe from pad. This pin can be floated or put some cap loads on board level to compensate DATA/SCLK pins load. Port name: DQSA
B_DQS	Data Strobe signal Flash B	I/O	Similar to A_DQS. Port name: DQSB

10.2.4 Functional description

The following sections describe functional details of the FlexSPI module.

10.2.4.1 Clocks

This section describes clocks and special clocking requirements of the FlexSPI module.

Table 10-3. Clock Usage

Clock Name	Description	Comment
serial clock root (ipg_clk_sfck)	Root clock for Serial domain	-
ahb clock (hclk)	AHB Bus clock	-
ipg clock (ipg_clk)	IPS Bus clock	-
DQS_OUT	Dummy Read Strobe output	Same frequency as SCLK. Clock output toggles during READ/LEARN instructions.
DQS_IN	Sample clock for RX Data	Same frequency as SCLK. Sample clock comes from loopbacked dummy read strobe, loopbacked SCLK or Flash provided read strobe.

10.2.4.2 Interrupts

This section describes all the interrupts that the FlexSPI module generates.

- **IP command done interrupt**

  When IP command is finished, there will be interrupt generated if INTEN[IPCMDDONEEN] is set to 0x1.

- **IP command grant error interrupt**

  When IP command grant timeout (not grant after MCR0[IPGRANTWAIT] * 1024 ahb clock cycles), there will be interrupt generated if INTEN[IPCMDGEEN] is set to 0x1.

- **AHB command grant error interrupt**

  When AHB command grant timeout (not grant after MCR0[AHBGRANTWAIT] * 1024 ahb clock cycles), there will be interrupt generated if INTEN[AHBCMDGEEN] is set to 0x1.

- **IP command error interrupt**

  When there is command check error or command execution error for IP command, there will be interrupt generated if INTEN[IPCMDERREN] is set to 0x1. Refer Overview of Error Flags for more details.

- **AHB command error interrupt**

  When there is command check error or command execution error for AHB command, there will be interrupt generated if INTEN[AHBCMDERREN] is set to 0x1. Refer Overview of Error Flags for more details.

- **IP RX FIFO watermark available interrupt**

  When the fill level of IP RX FIFO is no less than watermark level (IPRXFCR[RXWMRK]), there will be interrupt generated if INTEN[IPRXWAEN] is set to 0x1.

- **IP TX FIFO watermark exceed interrupt**

  When the empty level of IP TX FIFO is no less than watermark level (IPTXFCR[TXWMRK]), there will be interrupt generated if INTEN[IPRXWAEN] is set to 0x1.

- **Data learning failed interrupt**

When there is no valid sample clock phase found after LEARN instruction executed, there will be interrupt generated if INTEN[DATALEARNFAILEN] is set to 0x1.

- **Sequence execution timeout interrupt**

  When a sequence execution time exceeds the timeout wait time (MCR1[SEQWAIT]), an interrupt will be generated if INTEN[SEQTIMEOUTEN] is set to 0x1. For example, the following flash read command sequence will lasts about 8000000 cycle (ipg_clk_sfck). If SEQWAIT is set 0xFFFF, there will be sequence timeout interrupt generated.

  - Triggered by IP command
  - Flash read data size is 0x1000000 bytes
  - Flash accessed in Single mode and SDR mode,

- **AHB Bus timeout interrupt**

  When AHB bus response timeout, there will be interrupt generated if INTEN[AHBBUSTIMEOUTEN] is set to 0x1. A typical case is AHB read sequence is not configured properly in LUT (such as without READ instruction). There will never be data read from external device, then FlexSPI will never hit the read data in AHB RX Buffers for AHB Read command.

- **SCLK stopped by write command interrupt**

  When IP TX FIFO is empty during write command sequence execution, FlexSPI will stop SCLK output clock toggling and wait for write data filling. At this time, this interrupt is generated if enabled by INTEN register.

- **SCLK stopped by read command interrupt**

  When IP RX FIFO is full during read command sequence execution, FlexSPI will stop SCLK output clock toggling and wait for read data read out from IP RX FIFO. At this time, this interrupt is generated if enabled by INTEN register.

10.2.4.3 Flash Connection

There are two FlexSPI interface ports (A port and B port). Each port supports 2 flash devices by providing 2 chip select outputs.

NOTE

FlexSPI configuration is dependent on the chip configuration. Please refer to the system-level sections for boot and pinmux for chip specific information regarding the modes and number of devices supported.

The connection diagram with 4 devices is as following:

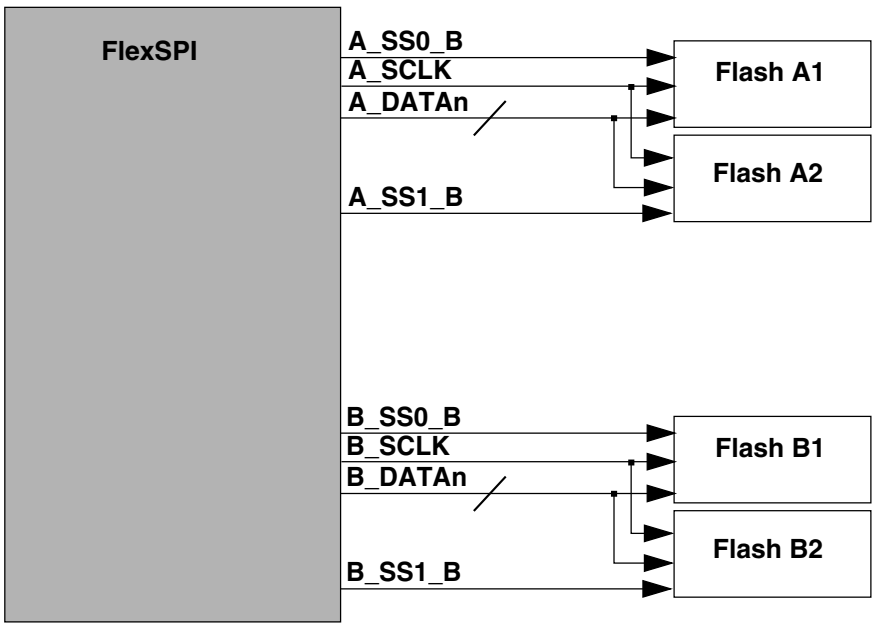


Figure 10-15. Flash connection diagram with four devices

NOTE

- Flash A1 and A2 could be two flash chip package or two flash die on the same package. There is no difference to FlexSPI. Same for Flash B1 and B2.
- Flash A1 and B1 could be accessed in parallel, using parallel mode. FlexSPI will merge/split the flash read/program data automatically. Same for A2 and B2.
- In parallel mode, A1 and A2 could not be accessed at the same time. Same for B1 and B2.
- In individual mode, A1, A2, B1 and B2 could not be accessed at the same time. But these four device could be accessed separately.

There is a combination mode to provide octal flash support by combining A port (A_DATA[3:0]) and B port (B_DATA[3:0]) together. The connection diagram for this combination mode is as following:

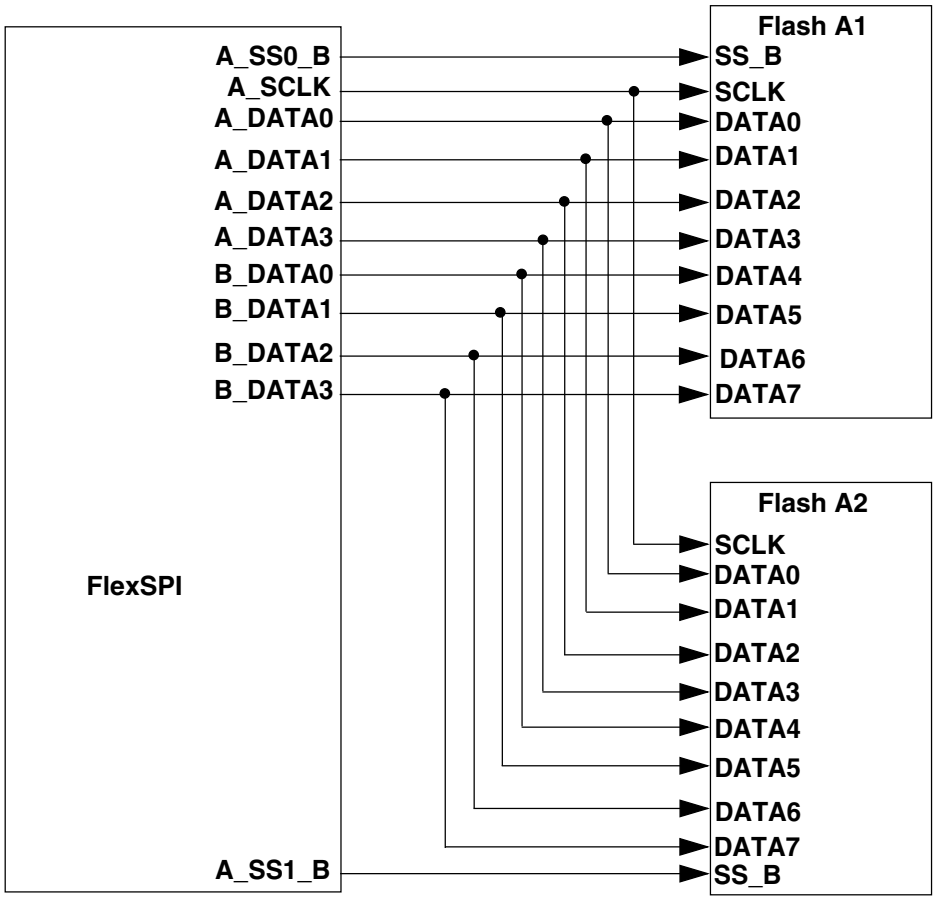


Figure 10-16. Flash connection diagram with combination mode

10.2.4.4 Flash Access mode

This section describes flash access mode.

10.2.4.4.1 SPI clock mode

FlexSPI supports only SPI clock mode 0: Clock polarity (CPOL)=0 and Clock Phase (CPHA)=0. SCLK will stay at logic low state when SPI bus is idle.

10.2.4.4.2 Flash Individual mode and Parallel mode

In individual mode, Flash read/write data is received/transmit on port A or port B.

In parallel mode, Flash read/write data is received/transmit on port A and B port parallely. FlexSPI will merge/split the flash read/program data automatically. Please note that only read/program data is merged/split (READ/WRITE instruction). For other

instructions (such as Command/Address/Mode/Data size), same command code/address/mode bits/data size information will be transmit to port A and port B device. For more detail, please refer to Instruction execution on SPI interface.

Individual mode and parallel mode is determined statically by register field IPCR1[IPAREN] (for IP command) or AHBCR[APAREN] (for AHB command).

10.2.4.4.3 SDR mode and DDR mode

In SDR (Single Data transfer Rate) mode, Flash receives data on SCLK rise edge and transmit data on SCLK fall edge.

In DDR (Dual Data transfer Rate) mode, Flash receives data on both SCLK rise and fall edges and transmit data on both SCLK rise and fall edges.

SDR and DDR mode is determined by instruction (opcode) in LUT sequence dynamically. There is no static configuration register field setting for SDR and DDR mode. For more details about input and output timing, please refer to FlexSPI Input Timing and FlexSPI Output Timing.

10.2.4.4.4 Single, Dual, Quad, and Octal mode

In Single mode, flash transmit/receive data on 1 Data pin (DATA0 for transmitting, DATA1 for receiving).

In Dual mode, flash transmit/receive data on 2 Data pin (DATA0~DATA1 for both transmitting and receiving).

In Quad mode, flash transmit/receive data on 4 Data pin (DATA0~DATA3 for both transmitting and receiving).

In Octal mode, flash transmit/receive data on 8 Data pin (DATA0~DATA7 for both transmitting and receiving).

Single, Dual, Quad and Octal mode is determined by instruction (num_pads) in LUT sequence dynamically. There is no static configuration register field setting for Single, Dual, Quad and Octal mode.

10.2.4.5 Flash memory map

Flash memory map in individual and parallel mode is as following:

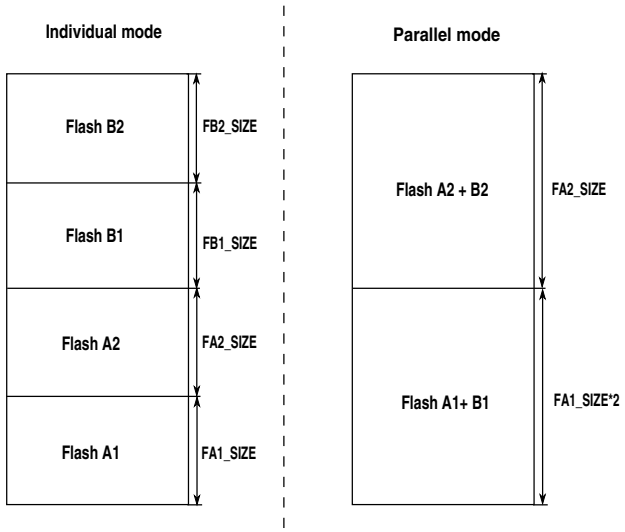


Figure 10-17. Flash memory map in individual and parallel mode

Flash memory map in individual mode:

- Flash A1 address range: 0x00000000 ~ FA1_SIZE
- Flash A2 address range: FA1_SIZE ~ (FA1_SIZE + FA2_SIZE)
- Flash B1 address range: (FA1_SIZE + FA2_SIZE) ~ (FA1_SIZE + FA2_SIZE + FB1_SIZE)
- Flash B2 address range: FA1_SIZE + FA2_SIZE + FB1_SIZE) ~ (FA1_SIZE + FA2_SIZE + FB1_SIZE + FB2_SIZE)

Flash memory map in parallel mode:

- Flash A1+B1 address range: 0x00000000 ~ FA1_SIZE*2
- Flash A2+B2 address range: FA1_SIZE*2 ~ (FA1_SIZE*2 + FA2_SIZE*2)

NOTE

- When MCR2[SAMEDEVICEEN] is set to 0x1, FA1_SIZE/FA2_SIZE/FB1_SIZE/FB2_SIZE = FLSHA1CR0[FLSHSZ] * 1KByte
- When MCR2[SAMEDEVICEEN] is set to 0x0, FA1_SIZE = FLSHA1CR0[FLSHSZ] * 1KByte; FA2_SIZE = FLSHA2CR0[FLSHSZ] * 1KByte; FB1_SIZE = FLSHB1CR0[FLSHSZ] * 1KByte; FB2_SIZE = FLSHB2CR0[FLSHSZ] * 1KByte
- Flash B1/B2 size setting are ignored in parallel mode (FLSHB1CR0[FLSHSZ], FLSHB2CR0[FLSHSZ]). To support parallel mode application, Flash B1 should be same device as A1 and Flash B2 should be same device as A2.

10.2.4.6 Flash address sent to Device

Flash access start address is determined by AHB address (AHB command) or IPCR0[SFAR] register (IP command). Refer Flash access by AHB Command and Flash access by IP Command for more details.

For AHB command, FlexSPI controller will remove flash base address automatically when sending flash address to devices. For IP command, the address in IPCR0[SFAR] should be the flash device's address without base address. Flash address is sent to devices in two part: Row Address and Column Address. For flash devices not supportting Column address, please set register field FLSHxCR1[CAS] to 0. Then all flash address bits will be sent to Flash device as Row address. For flash device supporting word-addressable feature, the last bit of flash address is not needed because flash is read/programmed in terms of 2 bytes. For parallel mode, Flash A1/B1 (or A2/B2) is accessed parallelly, so the flash address sent to flash device should be divided by 2. Following table indicates the relationship of Row/Column Address and flash address (FA)

Table 10-4. Flash Row/Column Address

Parallel mode	Word-addressable	Row Address	Column Address	Comment
0	0	FA[31:CAS]	FA[CAS-1:0]	There is no limitation on FA and data size alignment.
0	1	FA[31:CAS+1]	FA[CAS:1]	FA and data size should be 2 byte aligned.
1	0	FA[31:CAS+1]	FA[CAS:1]	FA and data size should be 2 byte aligned.
1	1	FA[31:CAS+2]	FA[CAS+1:2]	FA and data size should be 4 byte aligned.

NOTE

- FA is the flash address with flash base address removed.
- If the Row/Column Flash Address bit number to be sent to Flash device defined in Instructions is more than valid Row/Column Flash Address bit number, high posistion bits will be supplimented with zero. Refer Programmable Sequence Engine for more details about Row/Column Address instruction.

When parallel mode enabled or word-addressable flash used, there is limitation on flash start address and data size. This requirement could be meet by aligning AHB bus access address (For AHB command) or IP command address IPCR0[SFAR] (For IP command) in software. There are two ways to avoid these limitation in specified case.

1. For **AHB Read Command** only:

   When AHBCR[READADDROPT] and AHBCR[PREFETCHEN] are both set to 1, FlexSPI will guarantee flash access start address and data size are 64 bit aligned by hardware.

   When AHBCR[READADDROPT] is set to 1, FlexSPI will fetch redundant data to guarantee flash start address aligned with 8 bytes. When prefetch enabled (AHBCR[PREFETCHEN] is set to 1), flash read data size is determined by AHB RX Buffer size which is 64 bit aligned.

2. For **AHB Write Command and Individual mode** only:

   By default, FlexSPI will guarantee flash write access start address and data size are 16 bit aligned by using DQS as write mask.

   This feature is not applied in parallel mode and should be used only if external device supports write mask feature.

10.2.4.7 Look Up Table

The LUT (Look Up Table) is an internal memory to preserve a number of pre-programmed sequences. Each sequence consists of up to 8 instructions which are executed sequentially. When a flash access is triggered by an IP command or an AHB command, FlexSPI controller will fetch the sequence from LUT according to sequence index/number and execute it to generate a valid flash transaction on SPI interface.

Following figure indicates the structure of LUT, sequence and instruction.

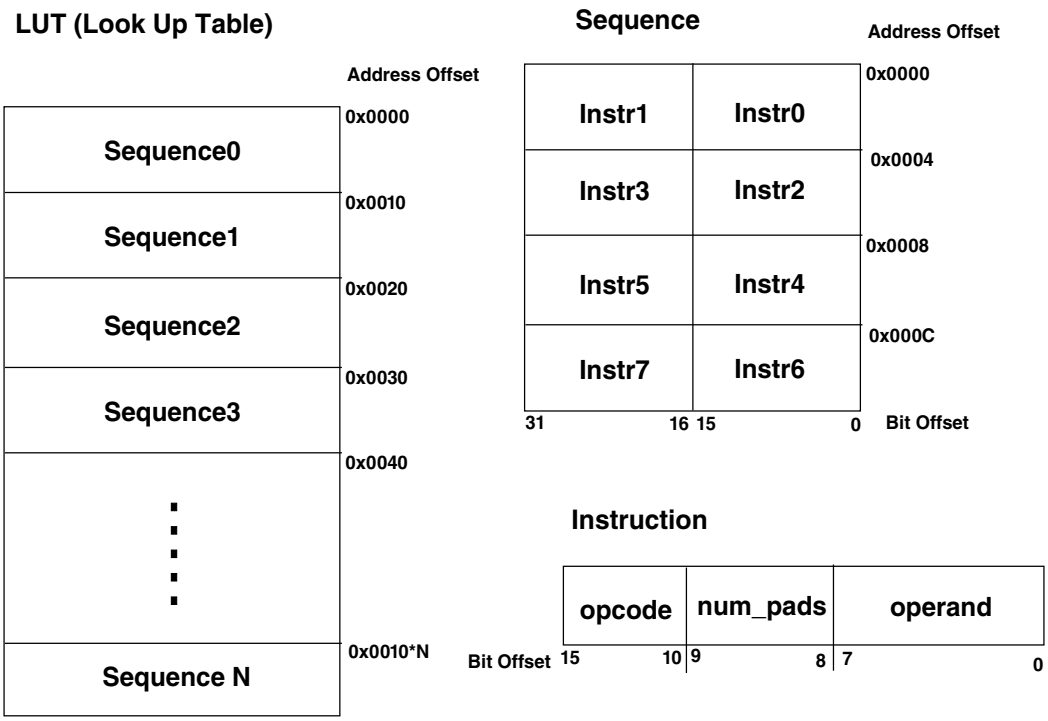


Figure 10-18. LUT and sequence structure

NOTE

If the instruction number needed is less than 8 for a flash transaction, STOP instruction should be programmed for the unneeded instructions (instruction code 8'h00).

For IP command and AHB write command, FlexSPI controller always executes from instruction pointer 0. For AHB read command, FlexSPI controller executes from a saved instruction start pointer. FlexSPI controller saves the instruction start pointers separately for each flash device. All these saved instruction pointer are zero before JMP_ON_CS instruction is executed,. When JMP_ON_CS instruction is executed, the operand in JMP_ON_CS instruction will be saved as instruction start pointer. Refer XIP Enhanced Mode for more details.

The reset value of LUT is unknown because it is implemented as internal memory. LUT should be programmed according to the device connected on board. In order to protect its contents during a code runover, the LUT could be locked/unlocked to avoid change by mistake after programmed. The key for locking or unlocking the LUT is **0x5AF05AF0**. The process for locking and unlocking the LUT is as follows:

Locking the LUT

1. Write the key **(0x5AF05AF0)** in to the LUT Key Register LUT Key Register (LUTKEY).

2. Write 1b1 to LUTCR[LOCK] and 1b0 to LUTCR[UNLOCK] fields of the LUT Control Register, immediately after the above KEY register writing. LUT is not successfully locked if there is any other register write access to FlexSPI between these two write accesses.

Unlocking the LUT

1. Write the key **(0x5AF05AF0)** in to the LUT Key Register LUT Key Register (LUTKEY).
2. Write 1b0 to LUTCR[LOCK] and 1b1 to LUTCR[UNLOCK] fields of the LUT Control Register, immediately after the above KEY register writing. LUT is not successfully unlocked if there is any other register write access to FlexSPI between these two write accesses.

The lock status of the LUT can be read from register field LUTCR[LOCK] and LUTCR[UNLOCK].

10.2.4.8 Programmable Sequence Engine

FlexSPI controller implements a programmable sequence engine that executes the sequence from LUT. FlexSPI controller executes the instructions sequentially and generates flash transaction on the SPI interface accordingly. The following table is a complete list of the supported instructions.

Table 10-5. Instruction set

Name	Opcode	Num_pads	Action on SPI interface	Transmit Data	Bits/Bytes/Cycle Number
CMD_SDR/ CMD_DDR	6'h01/ 6'h21	2'h0 - one pad (Single mode) 2'h1 - two pad (Dual mode) 2'h2 - four pad (Quad mode) 2'h3 - eight pad (Octal mode)	Transmit Command code to Flash	Command code: Operand[7:0]	Bit number: 8
RADDR_SDR/ RADDR_DDR	6'h02/ 6'h22		Transmit Row Address to Flash	Row_Address[31:0]	Bit number: operand[7:0]
CADDR_SDR/ CADDR_DDR	6'h03/ 6'h23		Transmit Column Address to Flash	Column_Address[31:0]	Bit number: operand[7:0]
MODE1_SDR/ MODE1_DDR	6'h04/ 6'h24		Transmit Mode bits to Flash	Mode bits: Operand[0]	Bit number: 1
MODE2_SDR/ MODE2_DDR	6'h05/ 6'25			Mode bits: Operand[1:0]	Bit number: 2
MODE4_SDR/ MODE4_DDR	6'h06/ 6'h26			Mode bits: Operand[3:0]	Bit number: 4
MODE8_SDR/ MODE8_DDR	6'h07/ 6'h27			Mode bits: Operand[7:0]	Bit number: 8

Table continues on the next page...

Table 10-5. Instruction set (continued)

Name	Opcode	Num_pads	Action on SPI interface	Transmit Data	Bits/Bytes/Cycle Number
WRITE_SDR/ WRITE_DDR	6'h08/ 6'h28		Transmit Programming Data to Flash	Programming Data in IP TX FIFO or AHB_TX_BUF	Byte number (data size) is determined by AHB burst size and burst type (AHB Command) or IPCR1[IDATSZ] (IP command). For more detail about flash read/ program data size, refer Flash access by AHB Command and Flash access by IP Command.
READ_SDR/ READ_DDR	6'h09/ 6'h29		Receive Read Data from Flash Read Data is put into AHB_RX_BUF or IP_RX_FIFO.	-	
LEARN_SDR/ LEARN_DDR	6'h0A/ 6'h2A		Receive Read Data or Preamble bit from Flash device FlexSPI Controller will compare the data line bits with DLPR register to determine a correct sampling clock phase.	-	Byte number: operand[7:0] Never set operand to zero for LEARN instruction.
DATSZ_SDR/ DATSZ_DDR	6'h0B/ 6'h2B		Transmit Read/ Program Data size (byte number) to Flash	Internal Logic Read/Program data size for current command sequence.	Bit number: operand[7:0] Please never set operand to zero or larger than 64 for DATSZ instruction.
DUMMY_SDR/ DUMMY_DDR	6'h0C/ 6'h2C		Leave data lines undriven by FlexSPI controller. Provide turnaround cycles from host driving to device driving. num_pads will determine the number of pads in input mode.	-	Dummy cycle number (in serial root clock): Operand[7:0] **Dummy cycle (N), described in the Flash device datasheet is in number of SCLK cycles and this number may be configurable.** **In SDR mode, SCLK cycle is same as serial root clock. The operand value should be set as N.** **In DDR mode, SCLK cycle is double the serial root clock cycle. The operand value should be set as 2*N, 2*N-1 or 2*N+1 depending on how dummy cycle defined in device datasheet. Please refer to Flash access sequence example and dummy cycle definition on device datasheet.**

Table continues on the next page...

Table 10-5. Instruction set (continued)

Name	Opcode	Num_pads	Action on SPI interface	Transmit Data	Bits/Bytes/Cycle Number
DUMMY_RWDS_SDR/ DUMMY_RWDS_DDR	6'h0D/ 6'h2D		This instruction is similar as DUMMY_SDR/ DUMMY_DDR instruction. But the dummy cycle number is different. DQS pins is called "RWDS" in HyperBus specification. Refer Dummy instruction for more details. **Set operand as "Latency count" for HyperBus devices.**	-	For read command, dummy cycle number (in serial root clock): (operand[7:0]*4-1) if RWDS (DQS pin) is high; (operand[7:0]*2-1) if RWDS (DQS pin) is low; For write command, dummy cycle number (in serial root clock): (operand[7:0]*4-2) if RWDS (DQS pin) is high; (operand[7:0]*2-2) if RWDS (DQS pin) is low;
JMP_ON_CS	6'h1F	Num_pads setting will be ignored. Always set num_pads to 2'h0.	Stop execution, deassert CS and save operand[7:0] as the instruction start pointer for next sequence. Normally this instruction is used to support Execute-In-Place enhance mode. Refer XIP Enhanced Mode for more details. This instruction is only allowed for AHB read command. If this instruction is used in IP command or AHB write command, there will be interrupt status bit set (INTR[IPCMDERR] or INTR[AHBCMDERR]).	-	No transaction on SPI interface.
STOP	6'h00		Stop execution, deassert CS. Next command sequence (to the same flash device)	-	

Table 10-5. Instruction set

Name	Opcode	Num_pads	Action on SPI interface	Transmit Data	Bits/Bytes/Cycle Number
			will started from instruction pointer 0.		

The programmable sequence engine allows the software to configure the FlexSPI LUT according to external serial device connected on board. The flexible structure is easily adaptable to new command/protocol changes from different vendors.

DDR sequence is a flash access sequence that contain DDR instruction which is not DUMMY, it may contain SDR instructions optionally. SDR sequence is a flash access sequence that don't contain any DDR instruction. FlexSPI controller determines instruction SDR or DDR mode by decoding bit 5 of instruction opcode. The output/input timing on FlexSPI is different for SDR and DDR sequences. Especially note that SDR instruction in SDR sequence and DDR sequence is executed differently. Refer FlexSPI Input Timing and FlexSPI Output Timing for more details.

10.2.4.8.1 Instruction execution on SPI interface

This section describes the detail of instruction execution on SPI interface. For all instructions transmitting/receiving data bits to/from flash, the bit order in one byte is higher on DATA3 than DATA0, and higher on B port than A port.

1. **Command Instruction**

   Command Instruction (CMD_SDR/CMD_DDR) is normally used to transmit command code to external flash device. Command code is the 8 bits operand in instruction. Command code will be send to both B port and A port in parallel mode. Refer Flash access sequence example for examples.

2. **Address Instruction**

   Address Instructions (RADDR_SDR/RADDR_DDR/CADDR_SDR/CADDR_DDR) are normally used to send Flash access start address (Row/Column Address) to external flash device. Row/Column Address bits are determined by FlexSPI according to AHB access address or IP command address. Refer Flash address sent to Device for more details. The bit number is the operand value in instruction code. Row/Column address bits will be send to both B port and A port in parallel mode. Refer Flash access sequence example for examples.

3. **Mode Instruction**

Mode Instructions (MODEx_SDR/MODEx_DDR) are normally used to send mode bits to external flash device. Mode bits are the (lower) bits value of the instruction operand. The transition bit number is 1 for MODE1_SDR/MODE1_DDR, 2 for MODE2_SDR/MODE2_DDR, 4 for MODE4_SDR/MODE4_DDR and 8 for MODE8_SDR/MODE8_DDR. Note that pad number should be no more than mode bit number. For example, it is not allowed to set num_pads to 2'b11 (Octal mode) for MODE4_* instructions. Mode bits will be send to both B port and A port in parallel mode. Refer Flash access sequence example for examples.

4. **Data Size Instruction**

   Data Size Instructions (DATSZ_SDR/DATSZ_DDR) are used to send program/read data size (byte number) to external device. This instruction is normally used in FPGA application when the memory space in external device acts similar as a FIFO. The device needs data size information to determine how much data will be popped from or pushed into internal FIFO. The bit number is the operation value in the instruction. Data size bits will be send to both B port and A port in parallel mode. Refer Flash access sequence example for examples.

5. **Write Instruction**

   Write Instructions (WRITE_SDR/WRITE_DDR) are normally used to send program data to external device. Programming data are fetched from IP TX FIFO (IP Command) or AHB TX Buffer (AHB command). For more details about flash program data size, refer Flash access by AHB Command and Flash access by IP Command. The byte order for programming date is always from low to high. Odd bytes are send on A port and Even bytes are send on B port in parallel mode. Refer Flash access sequence example for examples.

6. **Read Instruction**

   Read Instructions (READ_SDR/READ_DDR) are normally used to receive flash data from external device. Received data will be put into IP RX FIFO (IP Command) or AHB RX Buffer f(AHB command). For more detail about flash read data size, refer Flash access by AHB Command and Flash access by IP Command. The byte order for reading date is always from low to high. Odd bytes are received from A port and Even bytes are received from B port in parallel mode. Refer Flash access sequence example for examples.

7. **Dummy Instruction**

Dummy Instructions (DUMMY_SDR/DUMMY_DDR/DUMMY_RWDS_SDR/DUMMY_RWDS_DDR) are used to provide turnaround cycles on SPI interface. During dummy instruction, neither FlexSPI controller nor external device drives SPI interface. Refer Programmable Sequence Engine for more details about dummy cycle number.

DUMMY_RWDS_DDR could be used for HyperBus device which use "RWDS" pin to indicate whether extra latency count needed. DUMMY_RWDS_SDR is reserved for future. FlexSPI controller checks DQS pin input level at the 4th cycle after SCLK output toggling is enabled. DQS pins is called "RWDS" in HyperBus specification. Refer Flash access sequence example for examples.

NOTE

The FlexSPI releases the bus after at least one cycle. Therefore, to avoid data contention, number of dummy lines should be programmed more than 1 if data is transferred on more than 1 line.

8. **Learn Instruction**

   Learn Instructions (LEARN_SDR/LEARN_DDR) are used to determine the correct sample clock phase for flash read data sampling. External device drives read data (or data learning pattern) bits on FlexSPI interface. FlexSPI Controller will compare the data line bits with DLPR register to determine a correct sampling clock phase.

   Clock phase selection is automatically updated after learn instruction execution. Refer Data Learning Feature for more details. The byte number is the operand value in instruction. FlexSPI checks the same data pattern on ech data line.

   The byte order is byte 0, byte 1, byte 2, byte 3, byte0, byte 1, ... (byte 0 is DLPR register bit 7-0, byte 1 is DLPR register bit 15-8, byte 2 is DLPR register bit 23-16, byte 3 is DLPR register bit 31-24); The bit order is from high to low in each byte. Refer Flash access sequence example for examples.

10.2.4.8.2 Flash access sequence example

Following is an example for SDR single I/O Read sequence (Cypress Serial Nor Flash S25FS512S) in individual mode.

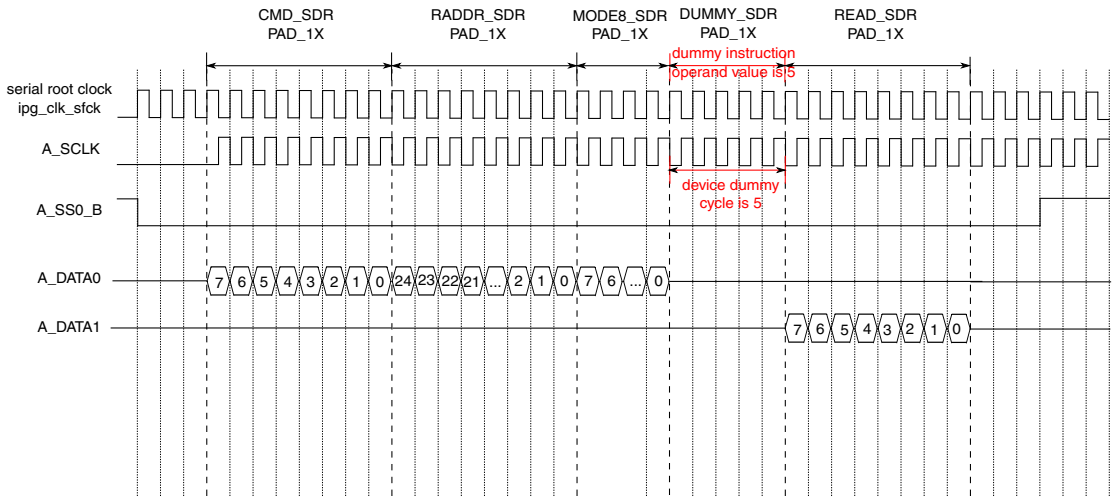


Figure 10-19. Flash access sequence example (SDR Single I/O Read sequence)

NOTE

- FlexSPI dummy instruction starts and ends at serial root clock rise edge.
- Device dummy cycle starts and ends at SCLK fall edge (on Cypress S25FS512S datasheet).

Following is an example for SDR Quad I/O Read sequence (Cypress Serial Nor Flash S25FS512S) in individual mode.

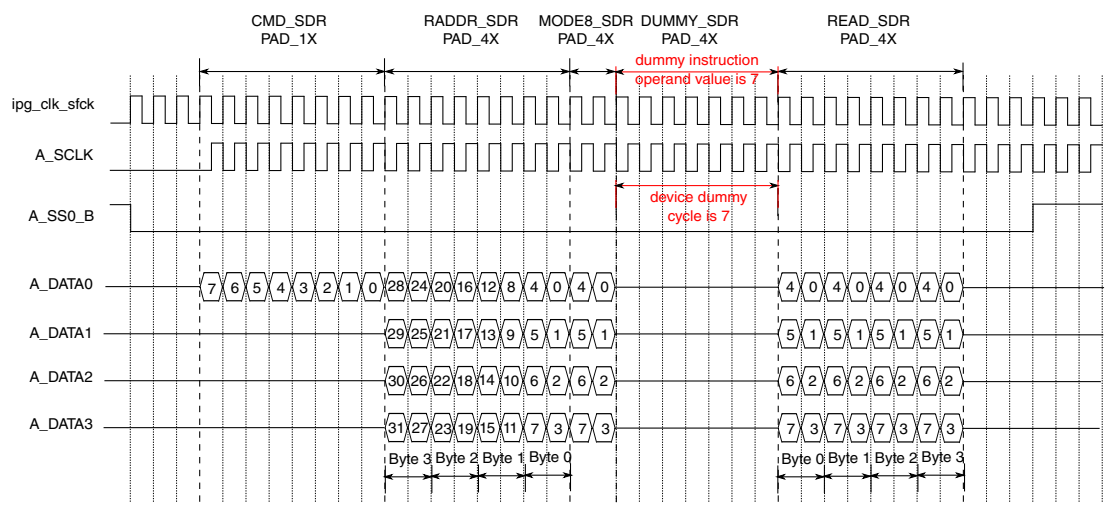


Figure 10-20. Flash access sequence example (SDR Quad I/O Read sequence)

NOTE

- FlexSPI dummy instruction starts and ends at serial root clock rise edge.
- Device dummy cycle starts and ends at SCLK fall edge (on Cypress S25FS512S datasheet).

Following is an example for DDR Quad I/O Read sequence (Cypress Serial Nor Flash S25FS512S) in parallel mode.

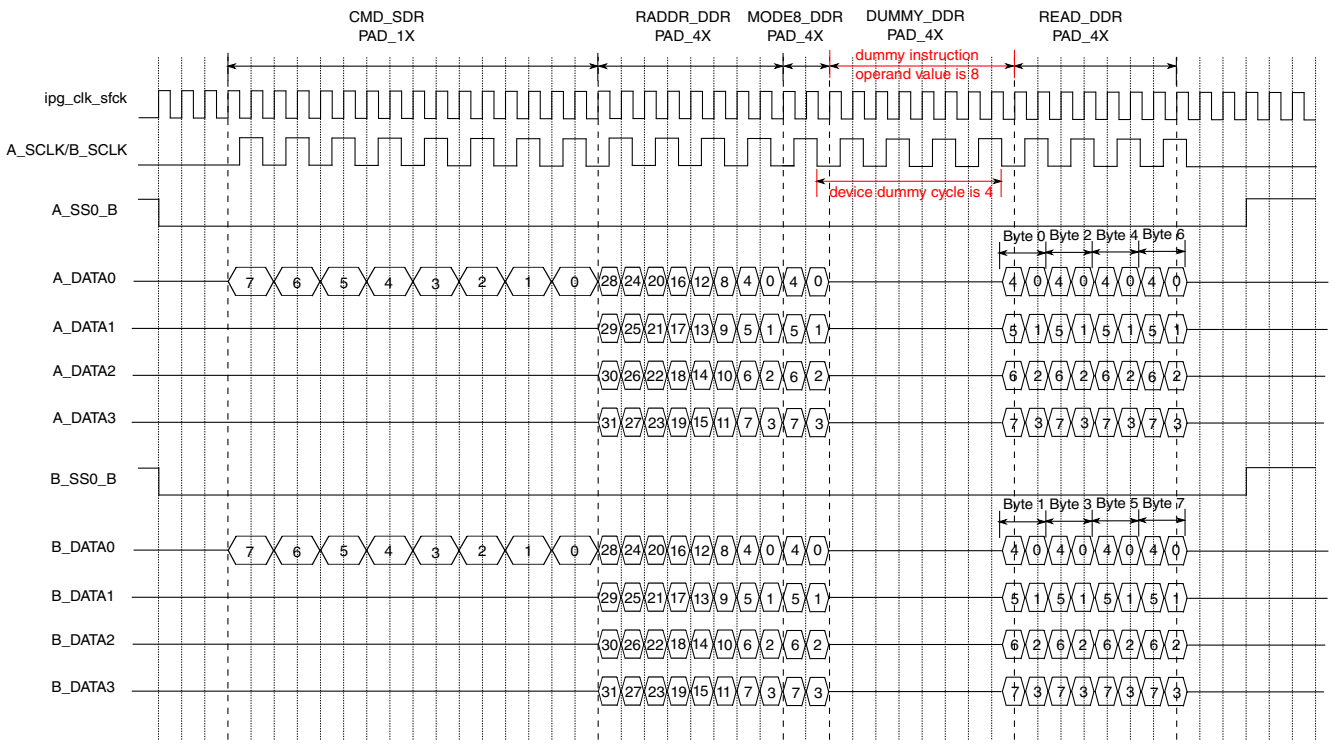


Figure 10-21. Flash access sequence example (DDR Quad I/O Read sequence)

Following is an example indicating Learning instruction (not for a specified flash device) in individual mode.

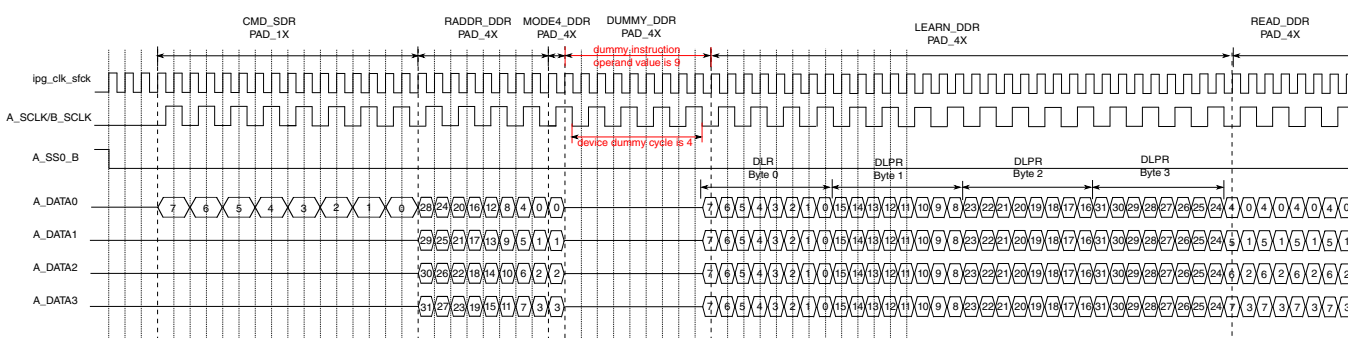


Figure 10-22. Flash access sequence example (Data Learning)

NOTE

- FlexSPI dummy instruction starts and ends at serial root clock rise edge.
- Device dummy cycle starts and ends at SCLK fall edge.
- DUMMY_DDR instruction operand value is odd because the total cycle number before DUMMY_DDR cycle is odd.

Following is an example for HyperBus device read transaction (Single latency count) in individual mode.

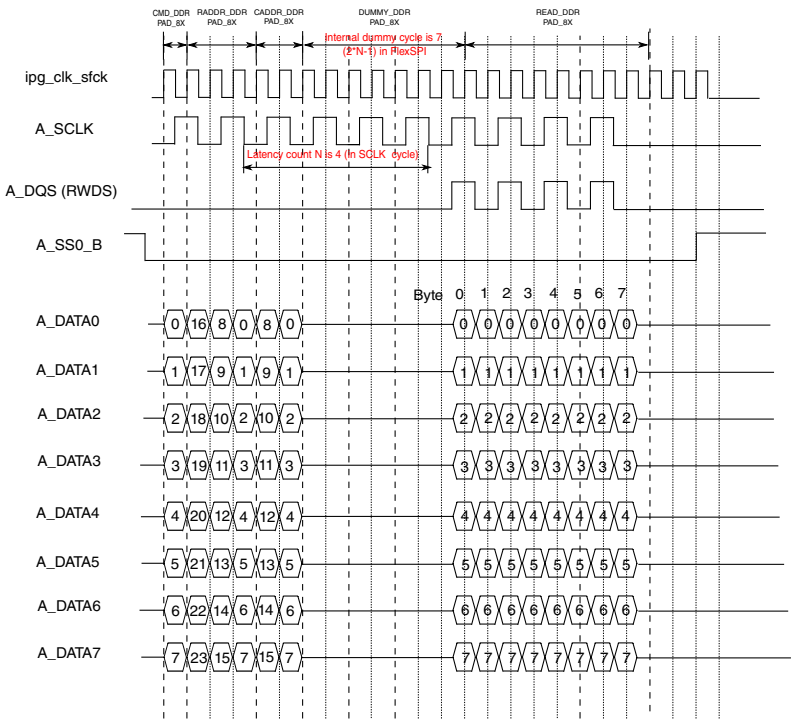


Figure 10-23. HyperBus device read transaction with single latency count

Following is an example for HyperBus device read transaction (Additional latency count) in individual mode.

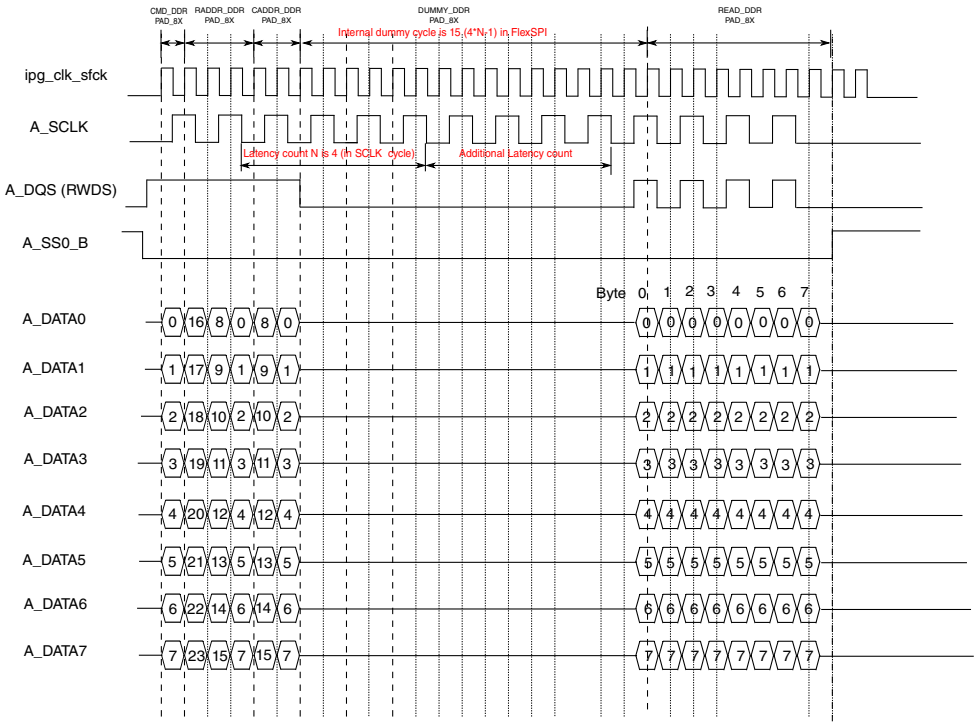


Figure 10-24. HyperBus device read transaction with additional latency count

Following is an example for HyperBus device write transaction (Single latency count) in individual mode.

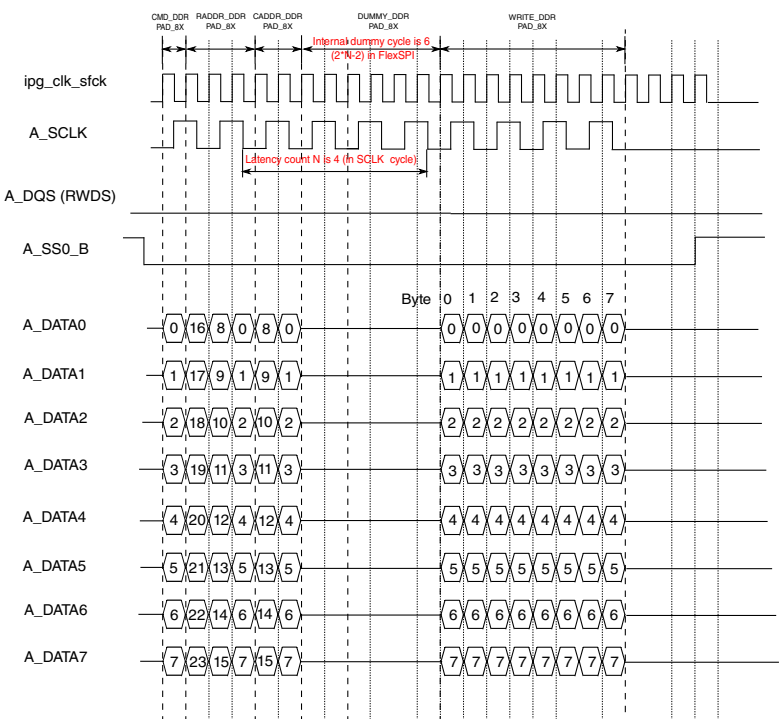


Figure 10-25. HyperBus device write transaction with single latency count

Following is an example for HyperBus device write transaction (Additional latency count) in individual mode.

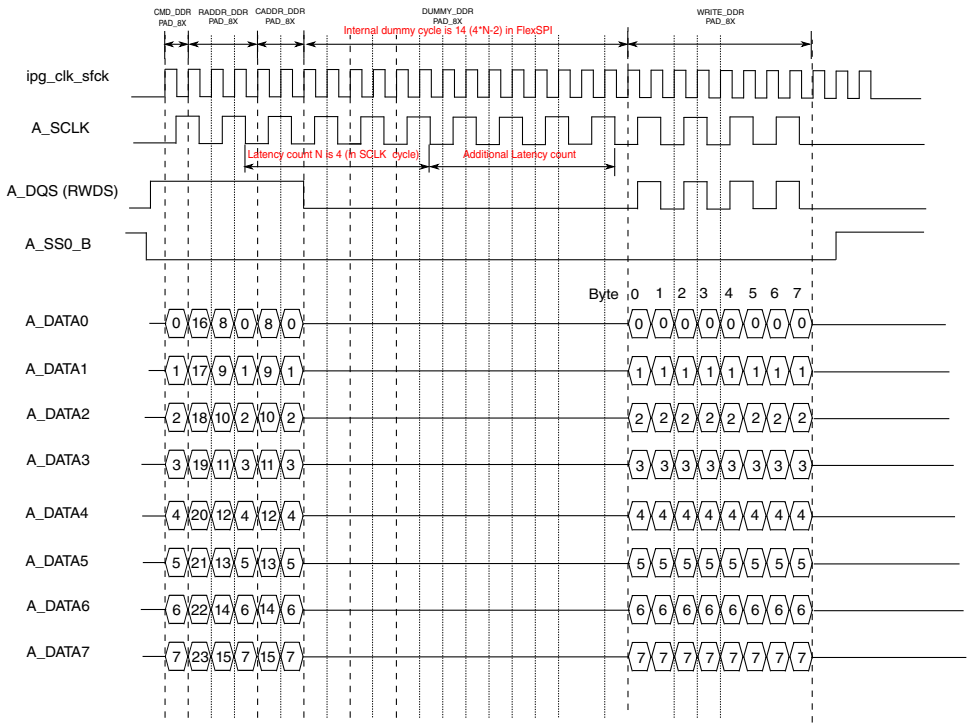


Figure 10-26. HyperBus device write transaction with additional latency count

10.2.4.9 Flash access by IP Command

Flash access could be triggered by IP command in following steps.

- Fill IP TX FIFO with programming data if this is a programing command (programming flash data, flash status registers etc.)
- Set flash access start address (IPCR0[SFAR]), read/program data size, sequence index in LUT and sequence number (IPCR1[ISEQNUM]).
- Trigger flash access command by writing 1 to register bit IPCMD[TRG]
- Polling register bit INTR[IPCMDDONE] to wait for this IP command to finish on FlexSPI interface.

NOTE

- IP TX FIFO could be filled before or after writing IPCR0/ IPCR1/IPCMD register. If SFM command is started with IP TX FIFO empty, FlexSPI will stop SCLK toggling to wait for TX data ready automatically.

- IPCMD register must be written after writing IPCR0/ IPCR1 register.
- Multiple Command sequences (8 at most) could be issued by one IP command.
- It is not allowed to issue another IP command before the previous IP command is finished. The behaviour is unknown in this case.

If this is a Reading command to Serial Flash Memory, all reading data from Flash will be put into IP RX FIFO. Software will need to read out data from IP RX FIFO by AHB bus or IP Bus. When IP RX FIFO is full and there is more data to be read from Flash device, FlexSPI will stop SCLK output clock toggling until IP RX FIFO is not full. Please refer to SCLK stop feature for more detail.

The detail of triggered Serial Flash Command is as following:

- Flash access start address:

  Determined by register field IPCR0[SFAR]

- Flash Chip Select:

  Determined by flash access address and Flash size setting (FLSHxCR0[FLSHSZ]).

- Flash command Sequence Index and Sequence Number:

  The sequences indexed from IPCR1[ISEQID] to (IPCR1[ISEQID] + IPCR1[ISEQNUM]) in LUT will be executed by FlexSPI sequentially.

- Flash Individual/Parallel access mode:

  Determined by IPCR1[IPAREN].

- Flash Read/Program Data Size:
  - If IPCR1[IDATSZ] value is non-zero, flash read/program data size (in byte) is IPCR1[IDATSZ].
  - If IPCR1[IDATSZ] value is zero, flash read/program data size (in byte) will the operand value in the READ/WRITE instruction.

NOTE

- Software should make sure the last sequence index never exceeds the LUT sequence number (IPCR1[ISEQID] + IPCR1[ISEQNUM] < 32).
- Data size is applied to every command sequence if sequence number is more than one.
- Data size is ignored if there is no WRITE/READ instruction in the command sequence .

IP command request is sent to arbitrator after triggered by software. It is not executed on FlexSPI Interface until granted by arbitrator. Please refer to Command Arbitration for more details.

10.2.4.9.1 Reading Data from IP RX FIFO

FlexSPI put the read data from external device into IP RX FIFO for IP command. These data could be read out by following two memory space access.

- 100 - 180 (by IPS Bus)
- 34000000 - 34000200 (by AHB Bus)

If MCR0[ARDFEN] is set to 0x1, read data in IP RX FIFO could only be read out by AHB Bus, IP Bus read access to IP RX FIFO will always return with data zero and no bus error occur.

If MCR0[ARDFEN] is set to 0x0, read data in IP RX FIFO could only be read out by IPS Bus, AHB Bus read access to IP RX FIFO will trigger bus error.

FlexSPI push read data into IP RX FIFO in terms of 64 bits every time it receives 64 bits data from external device. When read data bits number is not 64 bits aligned, FlexSPI will push additional zero bits into IP RX FIFO for the last push.

IP RX FIFO could be read by processor or DMA. Following is the detail flow for processor and DMA reading:

1. **Reading by processor**

   To read by processor, following register settings are needed:

   - Set register field IPRXFCR[RXDMAEN] to 0.
   - Set watermark level by IPRXFCR[RXWMRK], watermark level is (IPRXFCR[RXWMRK]+1)*8 bytes.
   - Set register field INTEN[IPRXWAEN] to enable IP RX FIFO watermark available interrupt (optional).

   Processor needs to poll register INTR[IPRXWA] or wait for IP RX FIFO Watermark Available interrupt before reading IP RX FIFO. This is to make sure there is a watermark level Data filled in IP RX FIFO before reading.

   After reading a watermark level data from IP RX FIFO, software need to set register bit INTR[IPRXWA]. This set action will pop out a watermark level data from IP RX FIFO.

   Following digram indicates the reading flow from IP RX FIFO by processor.

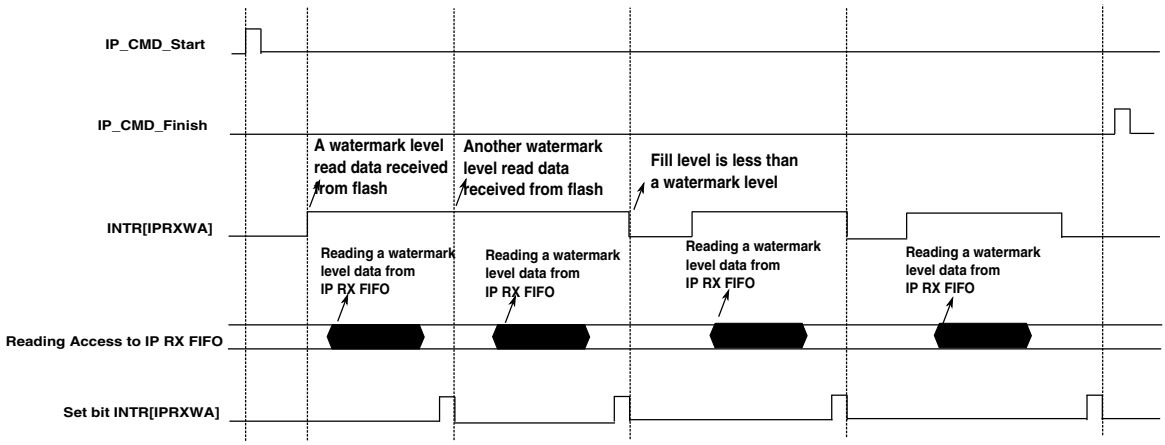


Figure 10-27. Reading IP RX FIFO by processor

NOTE

- Processor need to read out a watermark level data from IP RX FIFO each time before set register bit INTR[IPRXWA].
- It's supported that the total flash read/program data size is not multiple of watermark level. In this case, the reading data size from IP RX FIFO will be less than a watermark level for the last time, software should poll IPRXSTS[FILL] field instead of polling INTR[IPRXWA]. After copying all the data from IP RX FIFO and all command sequences to Flash are finished (INTR[IPCMDDONE]=1), software should clear IP RX FIFO by setting IPRXFCR[CLRIPRXF]. Otherwise, the reading data will be wrong for the next reading command to Flash.
- IP RX FIFO data is not popped out by each reading access to IP RX FIFO, but popped by writing 0x1 to register INTR[IPRXWA] bit.

2. **Reading by DMA**

To read IP RX FIFO by DMA, following setting is needed:

- Set register field IPRXFCR[RXDMAEN] to 1.
- Set watermark level by IPRXFCR[RXWMRK], watermark level is (IPRXFCR[RXWMRK]+1)*8 bytes.
- Set DMA transfer Minor loop size to same watermark level.

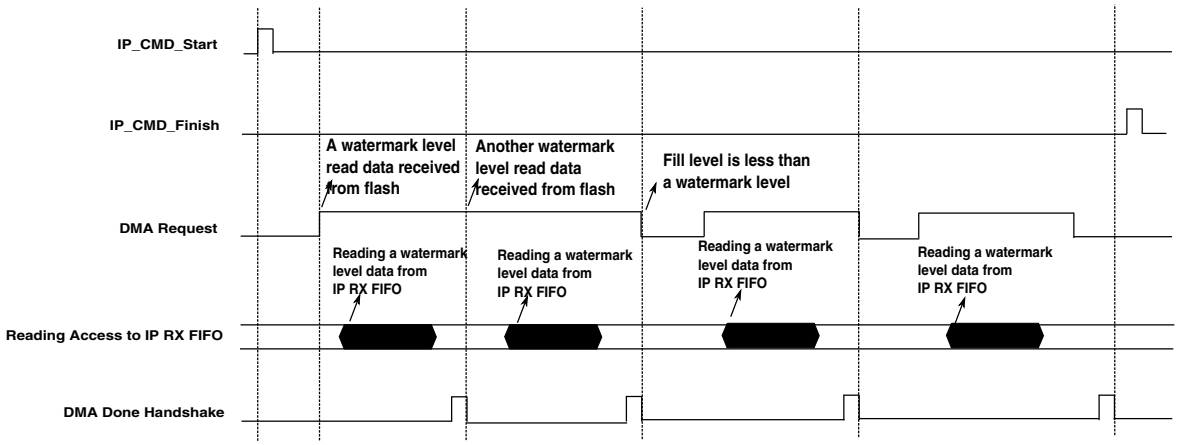


Figure 10-28. Reading from IP RX FIFO by DMA

NOTE

- DMA request is generated when the fill level of IP RX FIFO is higher than (or equal) watermark level. This request is not pulse valid but level valid.
- DMA should read out watermark level data from IP RX FIFO each time (set minor loop size with the same value as watermark level.
- DMA need to return a Done handshake (pulse valid) to FlexSPI each time it finished reading a watermark level data.
- IP RX FIFO data is not popped out by each reading access, but popped by DMA done handshake.
- It is not supported that the total read/write data size (Major loop size) is not multiple of watermark level. Because the DMA does not know when the data is ready for the last reading. It makes the DMA driver too complex to poll IPRXFSTS[FILL] field.

10.2.4.9.2 Filling Data to IP TX FIFO

The programming should be put into IP TX FIFO and then transmit to Flash by FlexSPI. It could be filled by two memory space.

- 0x180 - 0x200 (by IP Bus)
- 33008000 - 33008400 (by AHB Bus)

To fill by AHB Bus, need to set MCR0[ATDFEN] to 1, IP Bus write access to IP TX FIFO will be ignored, but no bus error.

To fill by IP Bus, need to set MCR0[ATDFEN] to 0, AHB Bus write access to IP TX FIFO will return Bus Error.

IP TX FIFO is popped with 64 bits data every time FlexSPI fetch data for transmitting. If the programming data size is not multiple of 64 bits, last popped valid bits will be less than 64 bits. But there is no problem because these invalid bits are not transmitted to Flash at all.

IP TX FIFO could be filled by processor or DMA.

To fill by processor, need following setting:

- Set register field IPTXFCR[TXDMAEN] to 0.
- Set watermark level by IPTXFCR[TXWMRK], watermark level is (IPTXFCR[TXWMRK]+1)*8 bytes.
- Set register field INTEN[IPTXWEEN] to enable IP TX FIFO watermark empty interrupt (optional).

Processor needs to poll register INTR[IPTXWE] or wait for IP TX FIFO Watermark empty interrupt before filling IP TX FIFO. This is to make sure there is enough space for a watermark level Data filling before filling.

After filling a watermark level data to IP TX FIFO, need to set register bit INTR[IPTXWE]. This will push a watermark level data into IP TX FIFO (write pointer is incremented).

NOTE

IP TX FIFO data is not pushed by each write access, only pushed by set INTR[IPTXWE] bit.

Following digram indicates the filling flow to IP TX FIFO by processor.

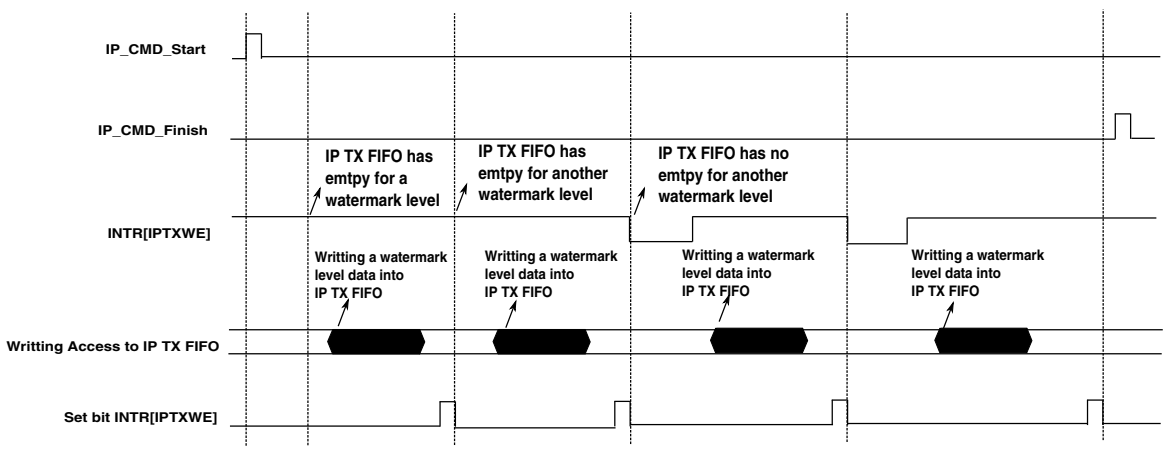


Figure 10-29. Filling IP TX FIFO by processor

NOTE

- Processor will need to fill a watermark level data to IP TX FIFO each time.
- It's allowed that total write data size is not multiple of watermark level. In this case, the writing size will be less than a watermark level for the last time. After filling all data into IP TX FIFO and all Command Sequences to Flash are finished (INTR[IPCMDDONE]=1), processor should clear IP TX FIFO by setting IPTXFCR[CLRIPTXF]. Otherwise, the programming data will be wrong for the next programming Command to Flash.

To fill IP TX FIFO by DMA, need following setting:

- Set register field IPTXFCR[TXDMAEN] to 1.
- Set watermark level by IPTXFCR[TXWMRK], watermark level is (IPTXFCR[TXWMRK]+1)*8 bytes.
- Set DMA transfer Minor loop size to same watermark level.

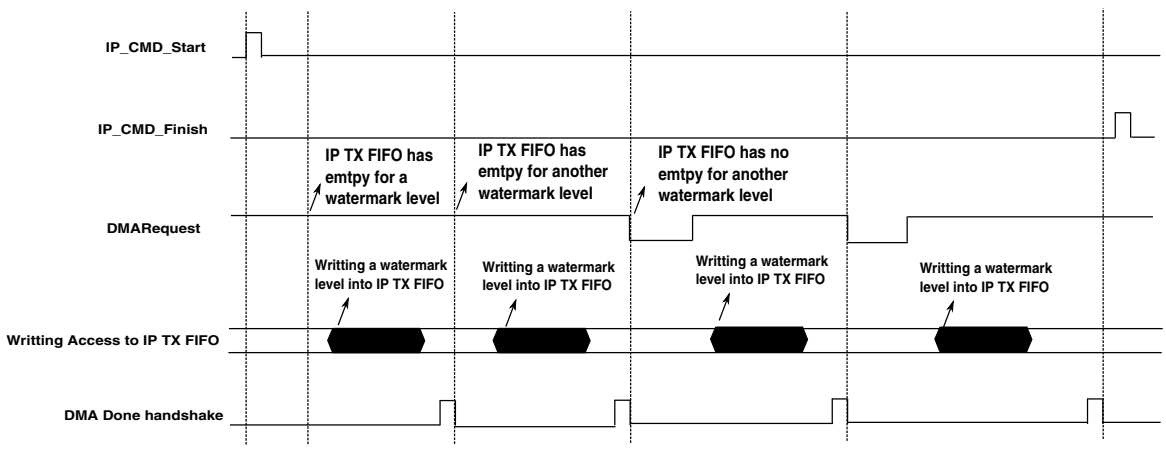


Figure 10-30. Filling from IP TX FIFO by DMA

NOTE

- DMA request is generated when there is empty space more than watermark level in IP TX FIFO. This request is not pulse valid but level valid.
- DMA should fill watermark level data into IP TX FIFO each time (set minor loop size with the same value as watermark level.
- DMA need to return a Done handshake (pulse valid) to FlexSPI each time it finished filling a watermark level data.

- IP TX FIFO data is not pushed in by each reading access, but pushed by DMA done handshake.
- It's allowed that total program data size (Major loop size) is not multiple of watermark level. After filling all data into IP TXFIFO and all command sequences to Flash are finished (INTR[IPCMDDONE]=1), need to clear IP TX FIFO by setting IPTXFCR[CLRIPTXF]. Otherwise, the programming data will be wrong for the next programming Command to Flash.

10.2.4.10 Flash access by AHB Command

Flash could be accessed by AHB bus directly on AHB address space:0~0x10000000. This address space is mapped to Serial Flash Memory in FlexSPI. AHB bus access to this address space may trigger Flash access command sequence as needed.

For AHB read access to Serial Flash Memory, FlexSPI will fetch data from flash into AHB RX Buffers and then return the data on AHB Bus. For AHB write access to Serial Flash Memory, FlexSPI will buffer AHB Bus write data into AHB TX Buffer and then transmit to Serial Flash memory.

There is no software configuration or polling need for AHB command except FlexSPI initialization. AHB master access external flash device transparently similar as normal AHB slave.

AHB command is normally used to access serial Flash memory space. IP command should be used to access the control and status registers or other spaces such as OTP in external flash device.

Following section described AHB command for read and write in more detail.

NOTE

When FlexSPI controller return AHB bus error for SFM access, AHB master should stop following access beats in current burst.

10.2.4.10.1 AHB write access to Flash

For AHB write access to Flash, FlexSPI will buffer the write data from AHB bus into internal AHB TX Buffer and then transmit them to Flash. FlexSPI only buffers write data for one AHB burst. Following diagram indicates the hardware operation in response to AHB write access to Flash.

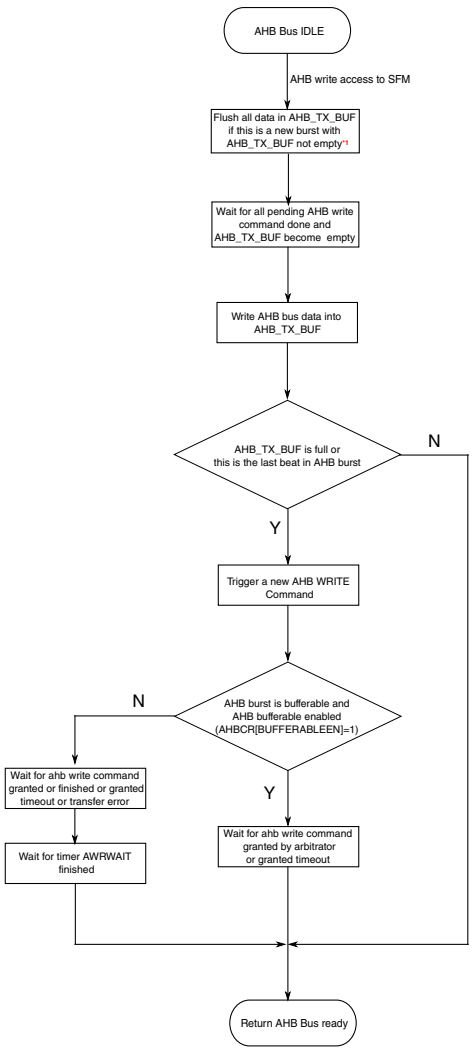


Figure 10-31. Hardware operation in responsee to AHB write access to Flash

FlexSPI triggers new AHB write command in following cases:

- This beat is the last one in current AHB burst (any burst type except INCR).
- AHB TX Buffer is full after buffering current beat data.
- AHB bus becomes IDLE or a new burst comes with AHB TX Buffer not emtpy.

The detail information about the triggered AHB write command:

- Flash Access Start Address:

  Determined by AHB burst address. FlexSPI will record the start address for the data in AHB TX Buffer and this address will be used as flash access start address

- Flash Chip Select:

FlexSPI determined the chip selection by flash access start address and flash size setting.

- Flash Command Sequence Index:

  Determined by FLSHxCR2[AWRSEQID].

- Flash Command Sequence Number:

  Determined by FLSHxCR2[AWRSEQNUM]. If AWRSEQNUM is not zero, multiple flash access command sequences will be triggered every time for AHB write command. The sequences indexed from AWRSEQID to (AWRSEQID + AWRSEQNUM) in LUT will be executed sequentially.

- Flash access mode (Individual/Parallel):

  Determined by AHBCR[APAREN].

- Flash Data Size:

  Determined by byte number of buffered data in AHB TX Buffer.

Following examples indicates internal logic for AHB write access to Flash. In these examples, AHB_TX_BUF is 64 Bytes (8*64bits) .

- AHB INCR/64bit/Bufferable burst with address sequence 0x8, 0x10, 0x18, 0x20, ..., 0x50 (10 beat totally):

  Two AHB write command will be triggered: the first command with flash start address 0x8 and data size 0x40; the second command with flash start address 0x48 and data size 0x10. See Figure 10-32.

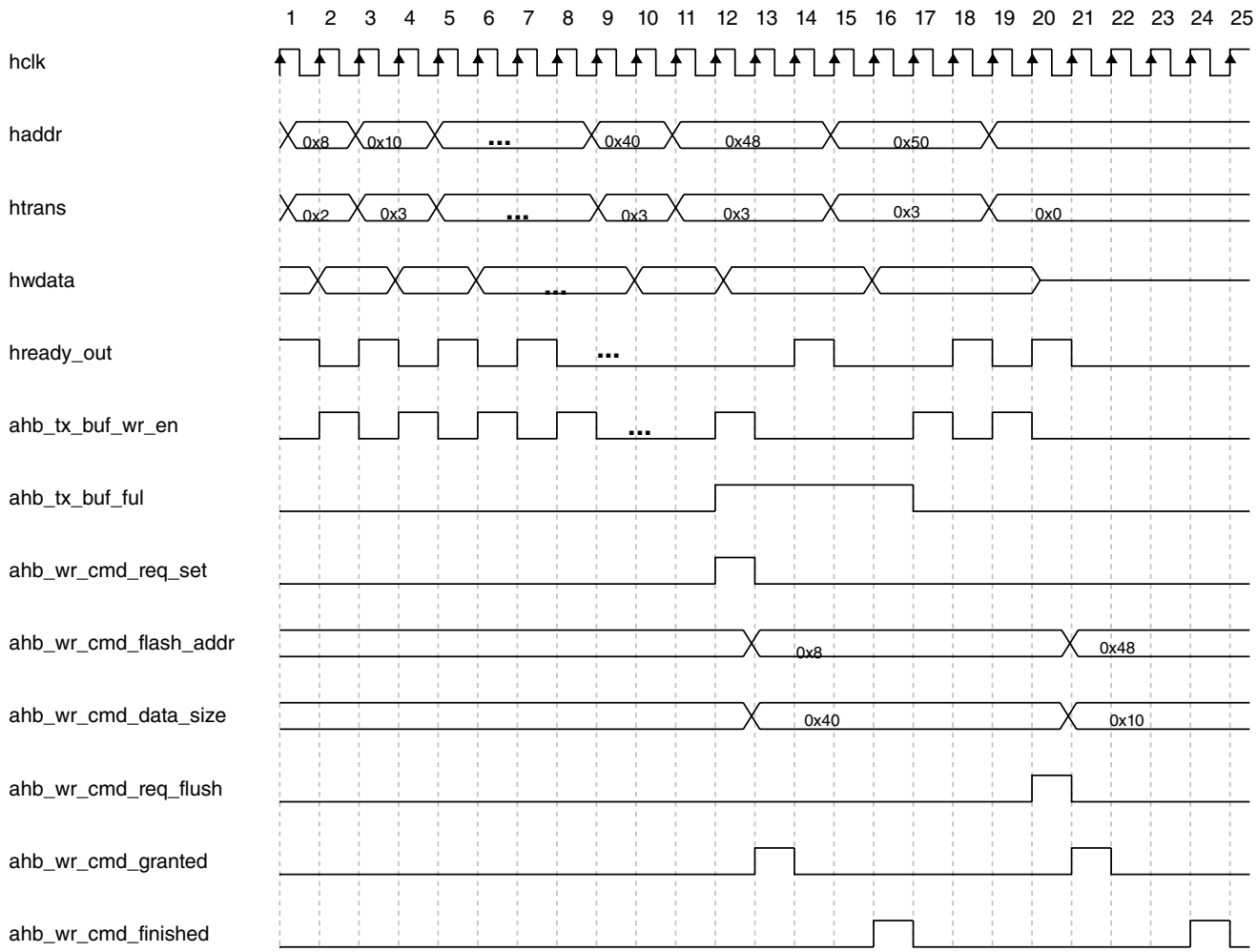


Figure 10-32. AHB write access (INCR/64bit/Bufferable)

- AHB WRAP8/64bit/Bufferable burst with address sequence 0x28, 0x30, 0x38, 0x0, 0x8, 0x10, 0x18, 0x20:

  One AHB write command will be triggered with flash start address 0x0 and data size 0x40. See Figure 10-33.

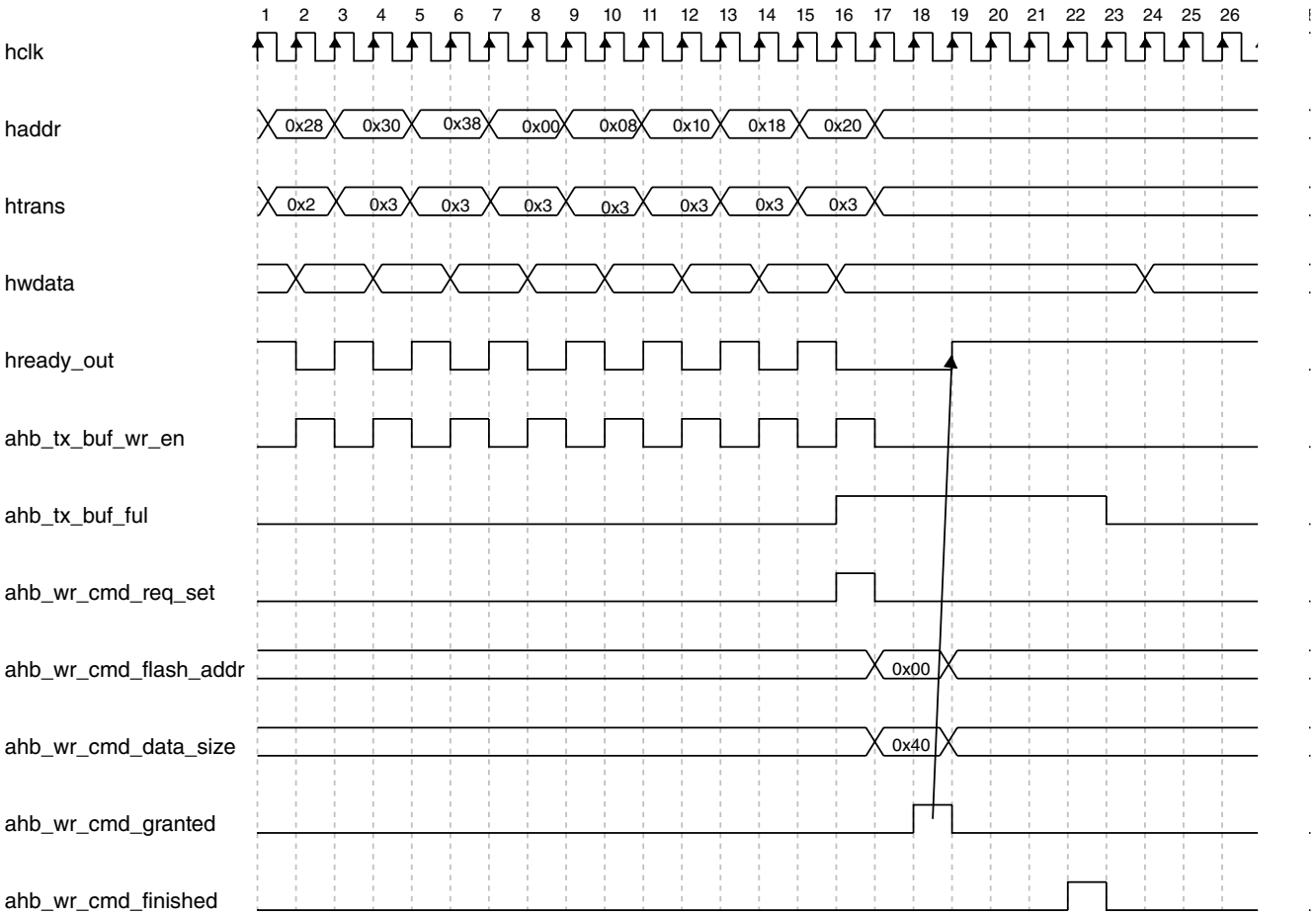


Figure 10-33. AHB write access (WRAP8/64bit/Bufferable)

- AHB WRAP8/64bit/Non-bufferable burst with address sequence 0x28, 0x30, 0x38, 0x0, 0x8, 0x10, 0x18, 0x20:

  One AHB write command will be triggered with flash start address 0x0 and data size 0x40;

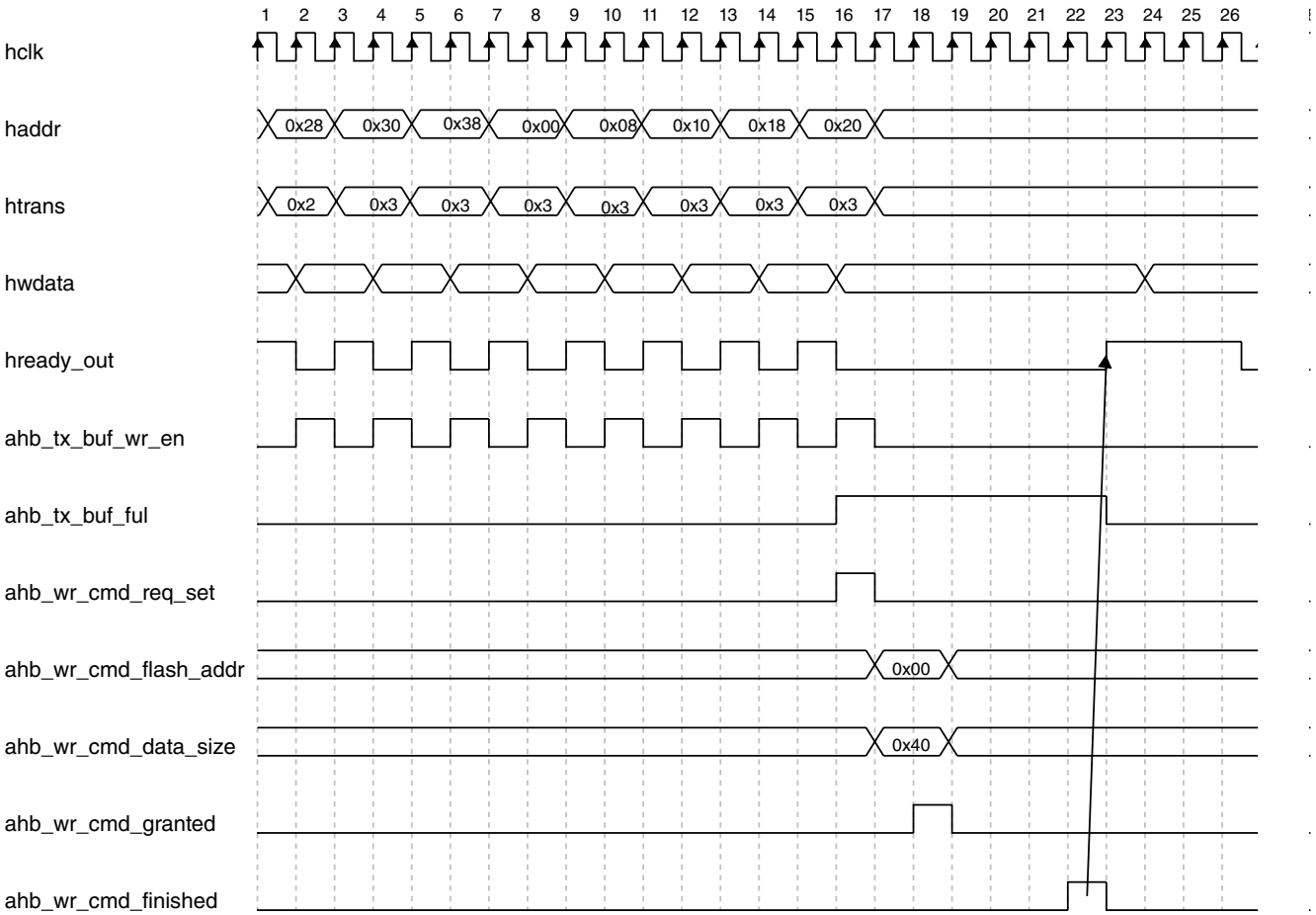


Figure 10-34. AHB write access (WRAP8/64bit/Non-Bufferable)

NOTE

The wrapper burst is not supported if burst data size (in byte) is larger than AHB TX Buffer size (in byte). For example, if AHB_TX_BUF is 64 Bytes (8*64bits), AHB WRAP16 * 64bit writet burst access is not supported.

10.2.4.10.2 AHB read access to Flash

For AHB read access to Flash, FlexSPI will check whether hit AHB TX Buffer/AHB RX Buffer/pending AHB read command according to the burst access type and register setting. If all these miss, FlexSPI trigger a new AHB read command to fetch data from Flash. Following diagram indicates the hardware operation in response to AHB read access to Flash.

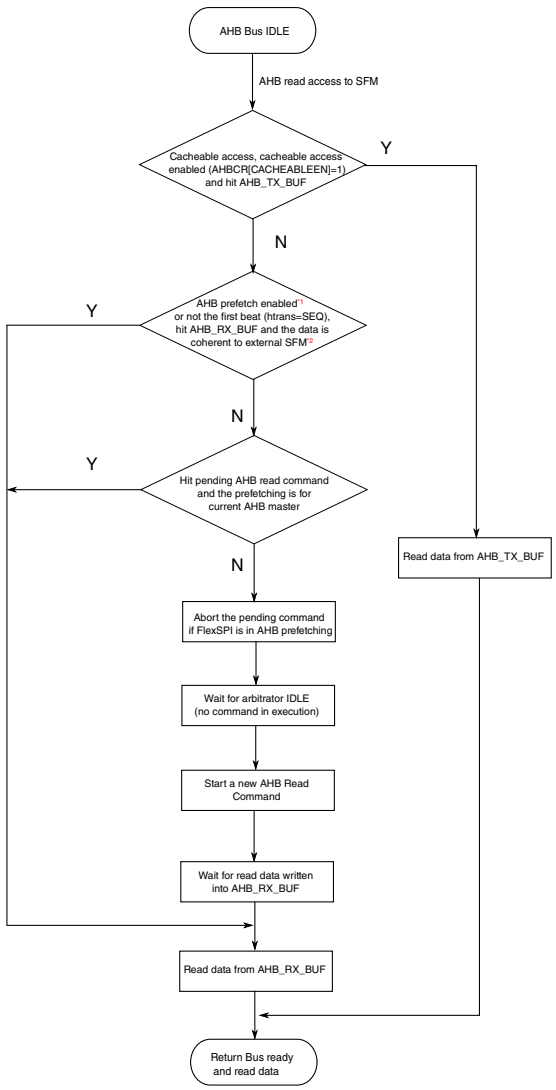


Figure 10-35. Hardware operation in response to AHB read access to Flash

NOTE

1. AHB prefetch is enabled when both AHBCR[PREFETCHEN]=0x1 and AHBRXBUFxCR0[PREFETCHEN]=0x1 (x is the ahb rx buffer ID for current AHB master).
2. AHB RX Buffer holds the data read from Flash. This data may become incoherent if the AHB writes to the same flash address.

The detail information about the triggered AHB read command:

- Flash Access Start Address and Data Size:

Determined by AHB address, burst type and burst size. FlexSPI will fetch read data from the start address for current burst or beat.

Table 10-6. AHB read command flash start address and data size

Prefetch Enable	Cross flash boundary	Burst Type	Flash start address [27:0]	Data Size
0	Never cross flash boundary because AHB burst never cross 1K Byte boundary.	SINGLE/INCR4/ INCR8/INCR16	hbeat_start_address	(hburst_end_address- hbeat_start_address)
		INCR	hbeat_start_address	byte size of current beat For INCR burst with prefetch disabled, each beat is handled same as SINGLE burst.
		WRAP4/ WRAP8/ WRAP16	hburst_start_address	(hburst_end_address- hburst_start_address)
1	No	INCR/SINGLE/ INCR4/INCR8/ INCR16	hbeat_start_address	ahb_rx_buf_sz
	No	WRAP4/ WRAP8/ WRAP16	hburst_start_address	ahb_rx_buf_sz
	Yes	INCR/SINGLE/ INCR4/INCR8/ INCR16	hbeat_start_address	(flash_top_address- hbeat_start_address)
	Yes	WRAP4/ WRAP8/ WRAP16	hburst_start_address	(flash_top_address- hburst_start_address)

NOTE

- hbeat_start_address is HADDR input from AHB master for current beat.
- hburst_start_address (for e.g.0x00) is the lowest address for current burst; hburst_end_address (for e.g. 0x10) is the highest address in current burst plus 1. For example, WRAP4 burst with HADDR 0x8,0xC, 0x0, 0x4.

- ahb_rx_buf_sz is the buffer size in byte of AHB RX Buffer which is used by current master.
- flash_top_address is the top address of currently accessed flash.

- Flash Chip Select:

  FlexSPI determined the chip selection by flash access start address and flash size setting.

- Flash Command Sequence Index:

  Determined by FLSHxCR2[ARDSEQID].

- Flash Command Sequence Number:

  Determined by FLSHxCR2[ARDSEQNUM]. If ARDSEQNUM is not zero, multiple flash access command sequences will be triggered every time for AHB read command. The sequences indexed from ARDSEQID to (ARDSEQID + ARDSEQNUM) in LUT will be executed sequentially.

- Flash access mode (Individual/Parallel):

  Determined by AHBCR[APAREN].

NOTE

- FlexSPI determines which ARDSEQNUM/ARDSEQID fields will be used as sequence ID by flash device chip selection automatically. Refer MCR2[SAMEDEVICEEN] for more detail.
- FlexSPI determines which AHB RX Buffer used for current AHB read access by master ID. For more details about the AHB RX buffer ID and AHB master ID mapping, refer AHB RX Buffer Management.
- It is not allowed to allocate AHB RX Buffer less than AHB Burst size. The behaviour is unknown for this case.

10.2.4.10.3 AHB RX Buffer Management

There are 8 buffers (Buffer 0 - Buffer 7) in AHB RX Buffer, which are transparent to AHB masters. FlexSPI fetch flash data and return on AHB Bus automatically. There is no status register polling needed for AHB read access to Serial Flash Memory.

AHB Rx Buffers total size is 2 KBytes . The buffer size is flexibly configurable for each buffer in AHB RX Buffers by register fields AHBRXBUF0CR0[BUFSZ]~AHBRXBUF6CR0[BUFSZ]. The buffer size for Buffer 0 to Buffer 7 could be set 0. If the buffer size is set to 0x0, the related MSTRID field setting (in same AHBRXBUFxCR0 register) is ignored by FlexSPI. Buffer 7 is used for all AHB masters which is not assigned to Buffer 0 - Buffer 6. Buffer 7 size setting field (AHBRXBUF7CR0[BUFSZ]) is ignored by FlexSPI, its buffer size is: AHB RX Buffer total size - sum of (Buffer 0 - Buffer 6 size).

When there is AHB read access to Serial Flash Memory, FlexSPI determines which AHB RX Buffer to use as following:

1. If master ID equal AHBRXBUF0CR0[MSTRID] and AHBRXBUF0CR0[BUFSZ] is not zero, Buffer 0 will be used.
2. If master ID equal AHBRXBUF1CR0[MSTRID] and AHBRXBUF1CR0[BUFSZ] is not zero, Buffer 1 will be used.
3. If master ID equal AHBRXBUF2CR0[MSTRID] and AHBRXBUF2CR0[BUFSZ] is not zero, Buffer 2 will be used.
4. If master ID equal AHBRXBUF3CR0[MSTRID] and AHBRXBUF3CR0[BUFSZ] is not zero, Buffer 3 will be used.
5. If master ID equal AHBRXBUF4CR0[MSTRID] and AHBRXBUF4CR0[BUFSZ] is not zero, Buffer 4 will be used.
6. If master ID equal AHBRXBUF5CR0[MSTRID] and AHBRXBUF5CR0[BUFSZ] is not zero, Buffer 5 will be used.
7. If master ID equal AHBRXBUF6CR0[MSTRID] and AHBRXBUF6CR0[BUFSZ] is not zero, Buffer 6 will be used.
8. If all above case not meet, Buffer 7 will be used

NOTE

- Software should make sure the buffer size of each buffer is no less than the max burst size of AHB Read access from the master using this buffer. Otherwise the behaviour is undefined.
- It is not supported to assign multiple buffers for single AHB master.
- When AHB read prefetch is enabled (AHBCR[PREFETCHEN] is set), the prefetch data size will be determined by buffer size. FlexSPI will fetch data from external Flash with buffer size if no flash boundary across.
- AHB master priority setting (register field AHBRXBUFxCR0[PRIORITY] is used only for the

suspending control of AHB prefetching. Refer Command Abort and Suspend.

10.2.4.11 Command Arbitration

There are four Flash access command sources:

1. AHB Command (triggered by AHB Write access to SFM space)
2. AHB Command (triggered by AHB Read access to SFM space)
3. IP command (triggered by writing IPCMD[TRG])
4. Suspended command (AHB Read prefetch sequence which is suspended)

NOTE

- An AHB bus access never triggers a write command and a read command at the same time.
- AHB prefetch sequence is an AHB Command sequence triggered by AHB Read access when AHB prefetch is enabled. After all read data fetched for current AHB read burst, FlexSPI will prefetch more data to reduce the read latency for next AHB read access. AHB command for read is never aborted while fetching read data for current AHB read burst. But AHB read command could be aborted by any new IP command or AHB command request when it's prefetching data (not for current read burst).

The granted priority of these 4 command source is as following when Arbitrator is idle (STS0[ARBIDLE]=1):

1. AHB command (Read/Write)
2. IP Command
3. Suspended Command

NOTE

Suspended command is not granted immediately when arbitrator is idle and no AHB/IP command request. Arbitrator will wait for n AHB clock cycle idle state before resuming the suspended command (n is the register field value in MCR2[RESUMEWAIT]). This intend to avoid AHB prefetch sequence being resumed and suspended frequently.

All command request are not granted if Arbitrator is busy in executing AHB/IP command (not suspended command), and there will AHB/IP Command granted error if the grant is timeout.

If new AHB/IP command request comes while Arbitrator is executing AHB read prefetch sequence, AHB read prefetch sequence will be aborted (but not immediately). Arbitrator will grant AHB/IP command request after AHB read prefetch sequence is aborted on FlexSPI interface and saved all internal data pointers.

10.2.4.11.1 Command Abort and Suspend

This section describes command abort and suspend mechanism.

1. **Command Abort**

   As mentioned above, AHB read prefetch sequence could be aborted if new AHB/IP command request comes.

2. **Command Suspend**

   When AHB read prefetch sequence is aborted on FlexSPI interface, FlexSPI will save this suspended sequence in following cases and resume this sequence if arbitrator is idle for enough time:

   - There is no valid suspended command (Register field AHBSPNDSTS[ACTIVE] is 0x0). This is possible if there is no suspended command yet or suspended command is resumed.
   - Aborted AHB read prefetch sequence is higher priority than current active suspend sequence.

   NOTE

   Original suspended sequence will be ignored and never resumed by FlexSPI.

3. **Suspended Command**

   The suspended command (internal status) turns active when there is any AHB prefetch command aborted and suspended. It turns inactive in following cases:

   - Suspended command is resumed by Arbitrator.
   - There is a new AHB read command request and it's triggered by AHB master using the same AHB RX Buffer (Buffer ID).

Following is an example indicating command abort/suspend/resume flow:

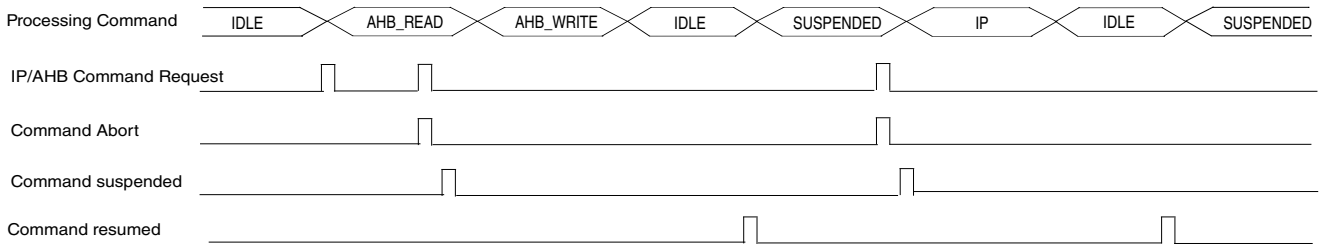


Figure 10-36. Command Instructions Execution on FlexSPI interface

10.2.4.12 SCLK stop feature

FlexSPI will stop SCLK output toggling when programming data is not ready for programming command sequence or there is no space (in internal FIFO) to receive data for reading command sequence.

There may be certain devices that do not support SCLK stopped during command sequence (chip select is valid). SCLK stopping could be avoid as following:

- For flash reading triggered by IP command
  - Never trigger a read command with data size larger than IP RX FIFO size.
  - Internal async FIFO for flash reading should never be full.

    FlexSPI pop data from this async FIFO in 64 bits per AHB clock cycle, and receiving data from FlexSPI interface in serial root clock. The receiving speed is determined by Flash access mode (Single/Dual/Quad/Octal mode and Individual/Parallel mode). For example, in Octal mode and Parallel mode, FlexSPI receives 16 bits per serial root clock cycle. This async FIFO is never full if AHB clock frequency is higher than 1/4 of serial root clock.

- For flash programming triggered by IP command
  - Never trigger a program command with data size larger than IP TX FIFO size.
  - Fill all programming data into IP TX FIFO before trigger the IP command.
  - Internal async FIFO for flash programming should never be empty

    FlexSPI fetch programming data into this async FIFO in 64 bits per AHB clock cycle, and transmitting data to FlexSPI interface in serial root clock. The transmitting speed is determined by Flash access mode (Single/Dual/Quad/Octal mode and Individual/Parallel mode). For example, in Octal mode and Parallel mode, FlexSPI transmits 16 bits per serial root clock cycle. This async FIFO is never empty if AHB clock frequency is higher than 1/4 of serial root clock.

- For flash reading/programming triggered by AHB command

- Internal async FIFO for flash reading/programming should never be full/empty. The frequency ratio limitation is same as flash reading/programming triggered by IP command.

NOTE

- FlexSPI never trigger a AHB read command with data size larger than internal AHB RX buffer size.
- FlexSPI never trigger a AHB program command with data size larger than internal AHB TX buffer size.
- All programming data is buffered into AHB TX Buffer before triggering AHB program command in FlexSPI.

10.2.4.13 FlexSPI Output Timing

This section describes the output timing in FlexSPI.

10.2.4.13.1 Output timing between Data and SCLK

This section describes the output timing relationship of data (on A_DATA/B_DATA) and SCLK. There are three cases for the data output timing:

- **SDR instruction in SDR sequence**
  SDR sequence is the sequence which contains only SDR instructions. In this case, all data bits last one serial root clock cycle on FlexSPI interface. Following diagram indicates the relationship of serial root clock, data and SCLK:

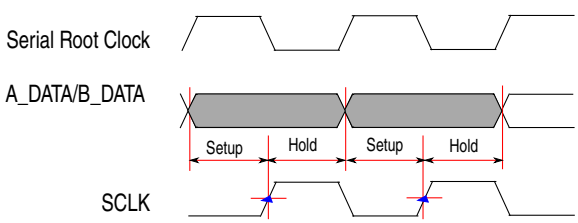


Figure 10-37. SDR instruction in SDR sequence

- **SDR instruction in DDR sequence**
  DDR sequence is a flash access command sequence that contain DDR instruction which is not DUMMY, it may contain SDR instructions optionally. In the case of SDR instruction in DDR sequence, all data bits last two serial root clock cycles on FlexSPI interface. Following diagram indicates the relationship of serial root clock, data and SCLK:

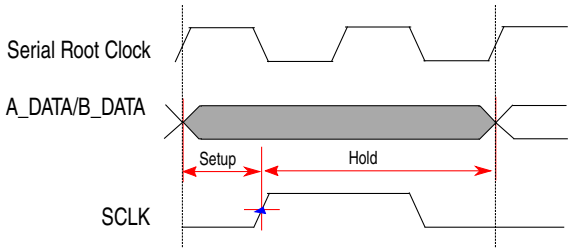


Figure 10-38. SDR instruction in DDR sequence

- **DDR instruction in DDR sequence**
  In the case of DDR instruction in DDR sequence, all data bits last one serial root clock cycle on FlexSPI interface. Following diagram indicates the relationship of serial root clock, data and SCLK:

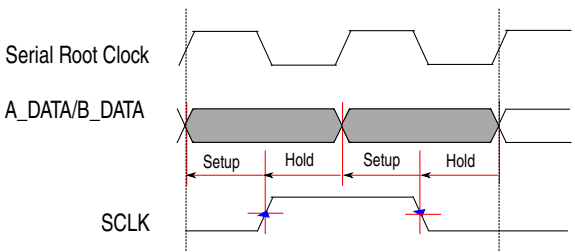


Figure 10-39. DDR instruction in DDR sequence

10.2.4.13.2 Output timing between Chip selection and SCLK

This section describes the output timing relationship of Chip select (on A_SS0_B/ A_SS1_B/B_SS0_B/B_SS1_B) and SCLK. The timing relationship is a little different for SDR sequence and DDR sequence.

- **Chip Select timing in SDR sequence**

  For SDR sequence, the delay from chip select assertion and the SCLK rising edge is **(FLSHxCR1[TCSS]+0.5)** cycles of serial root clock; The delay from SCLK falling edge and chip select deassertion is **FLSHxCR1[TCSH]** cycles of serial root clock. Following diagram indicates the timing relationship between chip selection and SCLK:

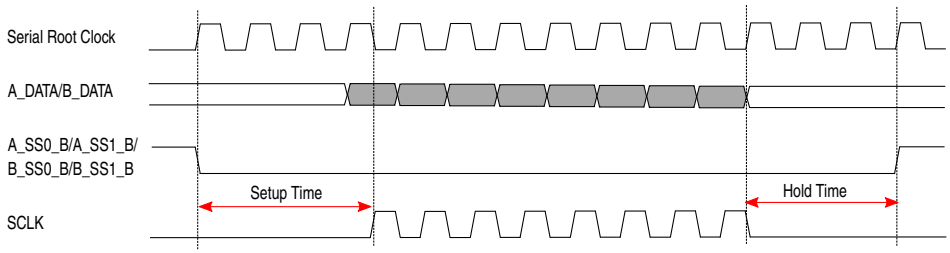


Figure 10-40. Chip selection output timing for SDR sequence

- **Chip Selection timing in DDR sequence**

  For DDR sequence, the delay from chip selection assertion and SCLK rise edge is **(TCSS+0.5)** cycles of serial root clock; The delay from SCLK fall edge and chip selection deassertion is **(TCSH+0.5)** cycles of serial root clock. Following diagram indicates the timing relationship between chip selection and SCLK:

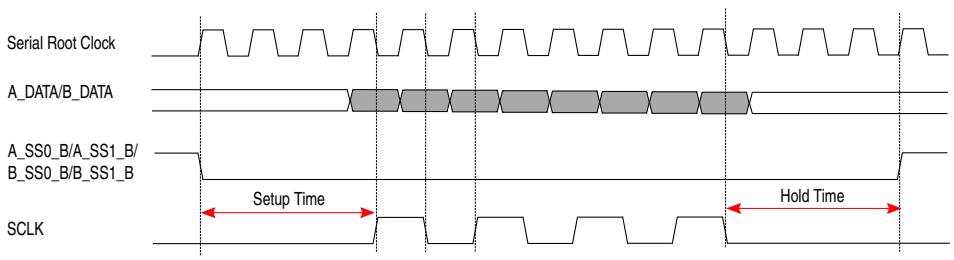


Figure 10-41. Chip selection output timing for DDR sequence

For certain device (such as FPGA device), there is limitation on the interval between Chip Selection valid. FlexSPI will ensure a delay time between chip selection valid if register field FLSHxCR1[CSINTERVAL] is set to non-zero value. The delay time is: CSINTERVAL*1024 cycle of serial root clock no matter SDR or DDR sequence. Please set this register field value to zero if there is no this limitation for external device. Following diagram indicates the timing of chip selection interval.

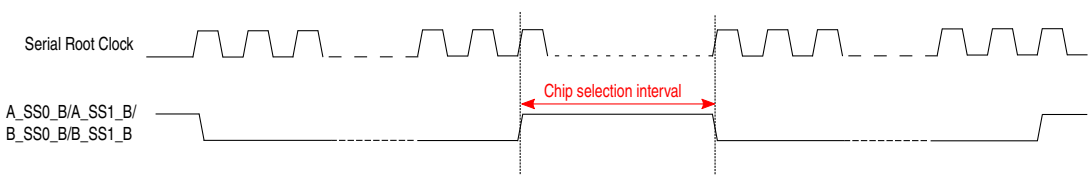


Figure 10-42. Chip Selection Valid interval

10.2.4.14 FlexSPI Input Timing

This section describes the input timing of FlexSPI.

10.2.4.14.1 RX Clock Source Features

This section describes the features of each RX clock source.

- **Internal dummy read strobe and loopbacked internally(MCR0[RXCLKSRC]==0)**

  Supporting legacy device with zero device output hold time.

  Saving one pad(DQS pad).

  Supporting low frequency clock for boot up usage.

- **Internal dummy read strobe and loopbacked from DQS pad(MCR0[RXCLKSRC]==1)**

  Supporting higher frequency than mode "MCR0[RXCLKSRC]==0".

  Supporting device doesn't provide read strobe.

- **Flash provided read strobe(MCR0[RXCLKSRC]==3)**

  Supporting the highest frequency.

  Supporting device provides read strobe.

10.2.4.14.2 Input timing for sampling with dummy read strobe

This section describes the input timing when sampling with internal dummy read strobe (MCR0[RXCLKSRC] is set to 0x0 or 0x1). The timing is very similar for sampling with dummy read strobe loopback internally and loopback from pad. But it could achieve higher read frequency by sampling with dummy read strobe loopback from DQS pad because it will compensate the delay of SCLK output path and Data pin input path. The input timing is different for SDR mode and DDR mode.

- **Input timing for sampling with dummy read strobe in SDR mode**

  For SDR Read/Learn instruction, FlexSPI samples input data pins with the falling edge of dummy read strobe. Following diagram indicates the input timing for sampling with dummy read strobe in SDR mode

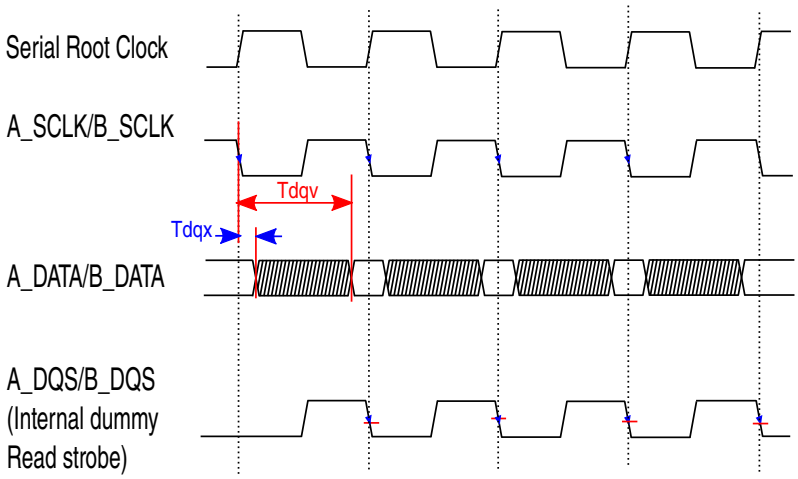


Figure 10-43. Input Timing for sampling with dummy read strobe in SDR mode

- **Input timing for sampling with dummy read strobe in DDR mode**

  For DDR Read/Learn instruction, FlexSPI sample input data pins with both rise and fall edge of dummy read strobe. Following diagram indicates the input timing for sampling with dummy read strobe in DDR mode

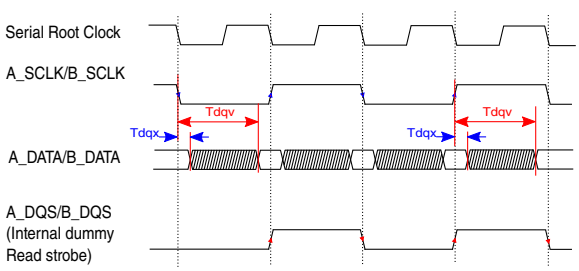


Figure 10-44. Input Timing for sampling with dummy read strobe in DDR mode

10.2.4.14.3 Input timing for sampling with flash provided read strobe

This section describes the input timing when sampling with flash provided read strobe (MCR0[RXCLKSRC] is set to 0x3). The input timing is different for SDR mode and DDR mode.

NOTE

There are no known devices that provide read strobe and support SDR mode operation.

There are two kinds of Flash provided read strobe:

- **Flash provide read strobe with SCLK2**

For certain flash devices, it provide read strobe with SCLK2 and provide read data with SCLK. Then read strobe edge is on the center with valid data window. FlexSPI controller should sample read data with read strobe directly (DLL is bypassed), refer DLL configuration for sampling for more details. Following diagrams indicates the input timing for sampling with flash read strobe in SDR mode and DDR mode:

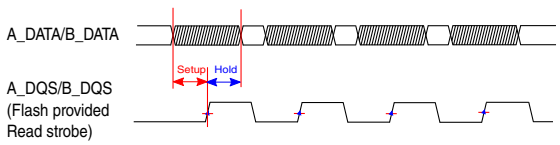


Figure 10-45. Input Timing 1 for Flash provided read strobe in SDR mode

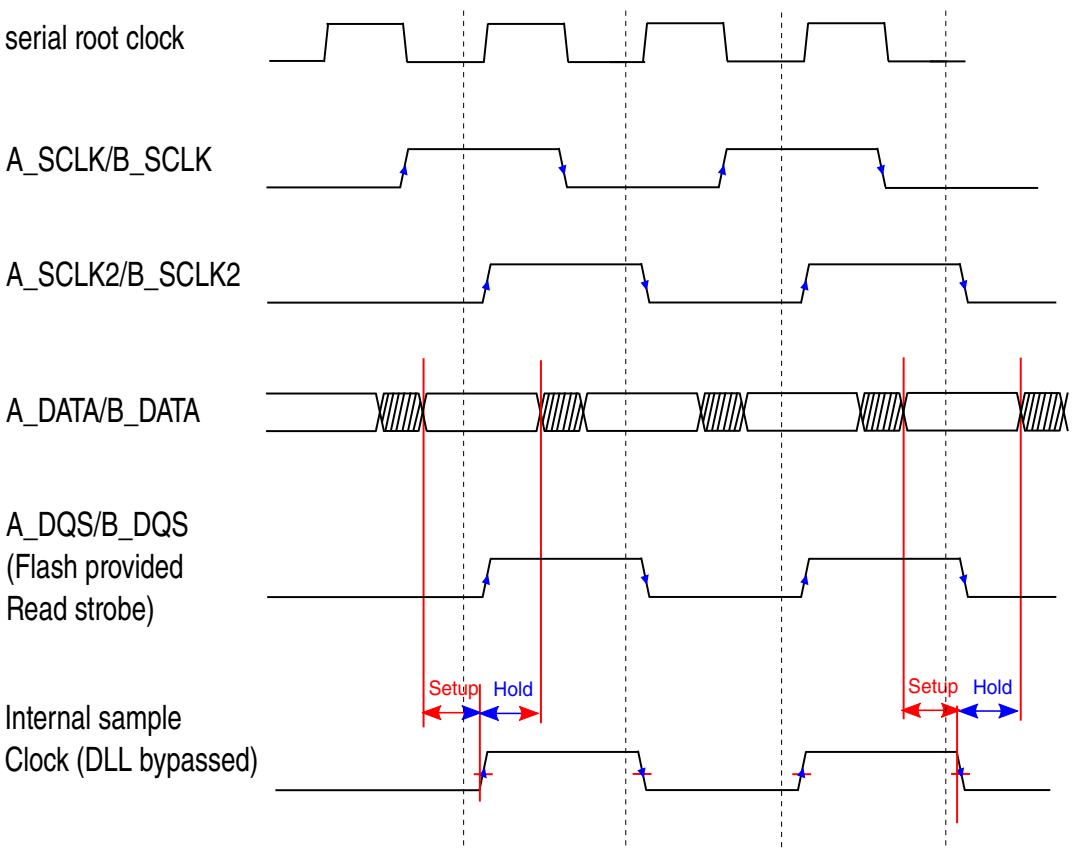


Figure 10-46. Input Timing 1 for Flash provided read strobe in DDR mode

- **Flash provide read strobe with SCLK**

  For certain flash devices, it provides both read data and read strobes with SCLK. Then the read strobe edge is aligned with read data change. FlexSPI controller should delay read strobe by half cycle in serial root clock (with DLL) and then sample read data with delayed strobe. Refer DLL configuration for sampling for more details. Following diagrams indicates the input timing for sampling with flash read strobe in SDR mode and DDR mode:

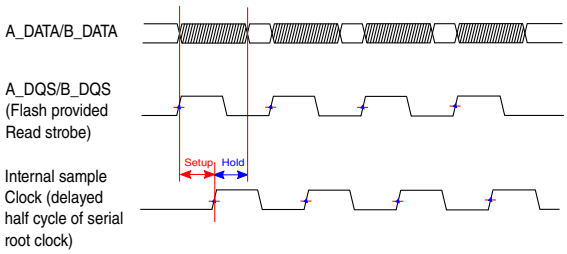


Figure 10-47. Input Timing 2 for Flash provided read strobe in SDR mode

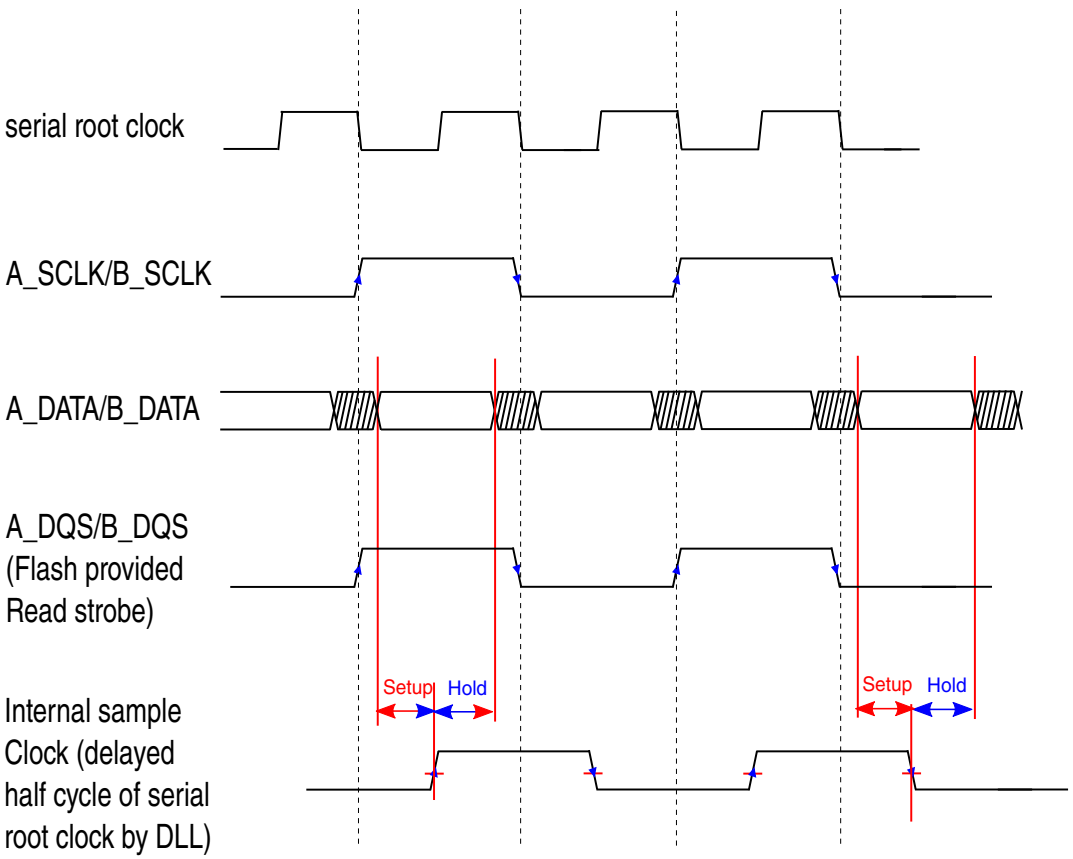


Figure 10-48. Input Timing 2 for Flash provided read strobe in DDR mode

10.2.4.14.4 DLL configuration for sampling

The input timing is different for those four sampling clock source. This is handled by setting register DLLxCR differently according to the sampling clock source mode. DLL is a delay line chain, which could be set to a fixed number of delay cells or auto-adjusted to lock on a certain phase delay to the reference clock.

- In following cases, DLLxCR should be set 0x00000100 (1 fixed delay cells in DLL delay chain) :

  - Sampling data with Dummy read strobe loopbacked internally(MCR0[RXCLKSRC]=0x0)
  - Sampling data with Dummy read strobe loopbacked from DQS pad(MCR0[RXCLKSRC]=0x1)
  - Sampling data with Flash provided read strobe (MCR0[RXCLKSRC]=0x3) and flash provides read strobe with SCLK2
- When data is sampled with Flash provided read strobe (MCR0[RXCLKSRC]=0x3) and flash provides read strobe with SCLK, DLL should be set as following to lock on half cycle of the reference clock (serial root clock)
  - SLVDLYTARGET=0xF
  - DLLEN=0x1
  - OVRDEN=0x0
  - Other fields in DLLxCR should be kept as reset value (all zero)

NOTE

If serial root clock is lower than 100 MHz, DLL is unable to lock on half cycle of serial root clock because the delay cell number is limited in delay chain. Then DLL should be configured as following instead:

- OVRDEN=0x1
- OVRDVAL=N; Each delay cell in DLL is about 75 ps~225 ps. The delay of DLL delay chain is (N * Delay_cell_delay), N should be set based on max. DDR frequency that current project supported, N = 17,please notice this is a recommended value. May need to adjust in real application if facing failure.
- Other fields in DLLxCR should be kept as reset value (all zero).

10.2.4.15 Data Learning Feature

FlexSPI controller generates 16 clock phases (Phase 0 ~ Phase 15) by delay cell line with the selected sample clock. Clock Phase 0 is actually the selected sample clock (DQS_IN) without any delay cell. There are 16 sampling blocks implemented for both Port A and 16 sampling blocks for Port B.

During Learn instruction, FlexSPI will compare the sampled data bits with internal data learn pattern (DLPR register) and determine the correct sampling clock phase. The phase selection is automatically updated after Learn instructions and applied to following Read instructions or sequences.

When the data learning feature is disabled (MCR0[LEARNEN]=0x0), FlexSPI always use clock phase 0 to sample FlexSPI data lines. If the data learning feature is disabled, then attempt to execute a LEARN instruction will result in an error flag (INTR[IPCMDERR] or INTR[AHBCMDERR]).

Data learning feature is supported in both SDR mode and DDR mode in FlexSPI. Data learning feature is not supported if sampling clock source is flash provided read strobe (MCR0[RXCLKSRC]=0x3).

Data learning feature would achieve high read frequency, but the highest frequency would be still limited by Flash input timing (Flash need to receive Command code and Flash address bits).

If data learning feature is enabled, FlexSPI will use Clock phase 0 after reset and before Learn instruction execution. The clock phase selection will be updated after Learn instruction executed by FlexSPI. The sample clock phase selected could be polled by register field STS0[DATALEARNPHASEA] and STS0[DATALEARNPHASEB]. If data learning failed, there will be interrupt bit set (INTR[DATALEARNFAIL]) and previous clock phase selection will be kept. Internal clock phase selection could be reset to Clock Phase 0 by write 0x1 to MCR2[CLRLEARNPHASE].

10.2.4.15.1 Data Learning with Flash providing preamble bit

Certain flash devices support driving with preamble bits (which may be also called DLP - Data learning pattern) before driving read data in each read command sequence. This data learning pattern is programmable by configuration register in flash device.

For specified Read Command sequence, flash will return preamble bit after dummy cycles and before returning read data.

For these flash devices, the operation flow with data learning is as following:

1. Set data learning pattern in DLPR register
2. Set same data learning pattern to flash device by triggering IP command.
3. Enable data learning feature in flash device by triggering IP command
4. Configure LUT sequence with valid Read command sequence which contains Learn instruction.
5. Trigger Flash read command by AHB/IP command as normal.

NOTE

The first 4 steps is not needed for every flash read command, only need to executed once.

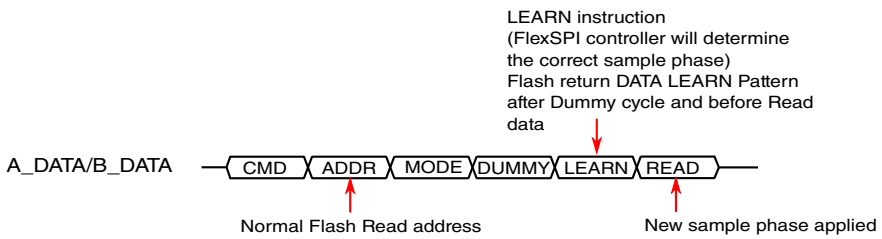


Figure 10-49. Data Learning flow with flash providing preamble bit

10.2.4.15.2 Data Learning with Flash not providing preamble bit

For flash devices not providing preamble bit, there is no way to return data learning pattern automatically by flash device for each read command sequence. Following sequence is data learning with flash not providing preamble bit:

1. Set data learning pattern in DLPR register
2. Reserve a certain Flash Memory area and program data (according to data learning pattern and flash access mode) to this area by IP command.
3. Configure LUT sequence with Read sequence (which using LEARN instruction instead of READ instruction). And trigger Read sequence to reserved flash area by IP command.
4. Trigger Flash read command by AHB/IP command as normal. No Learn instruction needed in this Read sequence.

NOTE

- The first 3 steps is not needed for every flash read command, only need to executed by once.
- Step 4 should be executed with a certain interval to make sure internal sampling clock phase is adjusted in time.

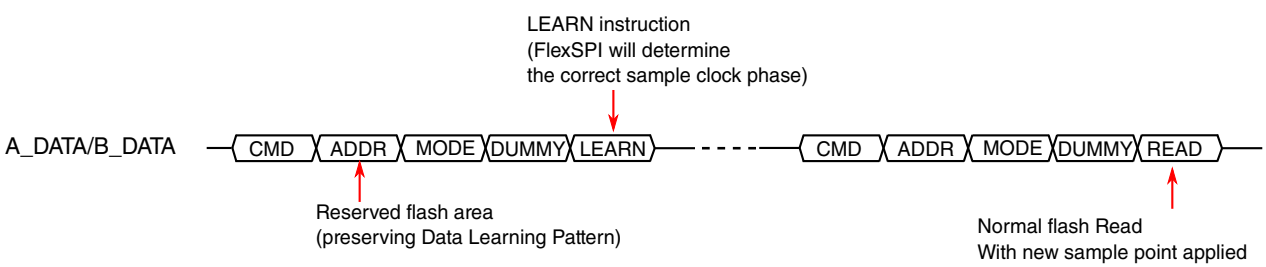


Figure 10-50. Data Learning flow with Flash not providing preamble bit

NOTE

- For Flash not providing preamble bit, there is software overhead because need to read the reserved flash area at interval.

The preprogramming data is determined by following items:

- Data Learning Pattern (DLPR register setting)
- Data Learning Pattern bit length
- Individual/Parallel mode for read command
- Octal/Quad/Dual/Single mode for read command

Following is an example for pre-programming data in case of DLP value is 0x43, 8 bits and flash is read in Quad mode and Parallel mode.

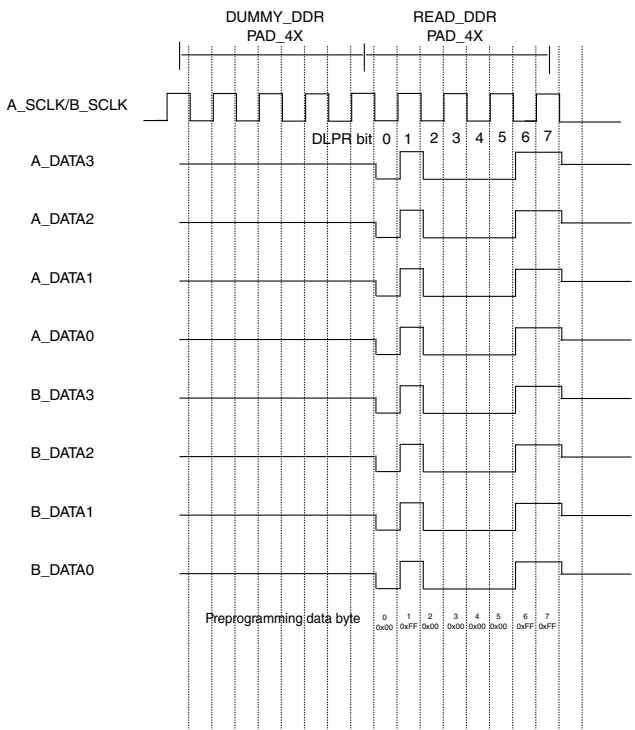


Figure 10-51. Preprogramming data for Data Learning

10.2.4.16 XIP Enhanced Mode

FlexSPI always supports Execute-In-Place (XIP), no matter external device provided XIP enhanced mode or not. Execute-In-Place is supported by putting program code on External device, then read/execute on external device directly by AHB read access to SFM space. There is no configuration or status polling needed during AHB read access to External device memory and AHB RX buffer is fully transparent to software.

Certain devices provide XIP enhanced mode to improve code execution. In this mode, there is no need to provide Command code for Read Sequence. It saves many cycles for Command Instructions and greatly improves code execution. This XIP enhanced mode is entered/exit by a special sequence which is device specified. Please refer to external device datasheet for more detail.

Normally, the XIP enhanced mode is entered by following sequence:

1. Enable XIP enhanced mode in External Flash by IP command
2. Send the first Read Sequence to External Flash device with correct Mode bits. Command code is needed in this Read sequence.
3. Send following Read sequences to External Flash device with correct Mode bits. Command code is not needed in these Read Sequences. But Mode bits should be sent according to Flash specified. Otherwise Flash will exit Execute-In-Place Enhanced mode.

The instruction JMP_ON_CS in FlexSPI should be used to support external device XIP enhanced mode. This instruction is only allowed in AHB Read command, otherwise there will be IPCMDERR or AHBCDMERR error interrupt generated if this interrupt is enabled. JMP_ON_CS should never be used if device doesn't support XIP enhanced mode. To support XIP enhanced mode, the first instruction in the Read Sequence should be Command instruction and the last valid instruction should be JMP_ON_CS (with operand 0x1).

For the first AHB read Command triggered, FlexSPI will execute the instructions from the instruction pointer 0 in the sequence (which is Command instruction). After this sequence executed FlexSPI will save operand in JMP_ON_CS instruction as start pointer for next Command to current device internally. For the following AHB read Command triggered, FlexSPI will execute from instruction pointer 0x1 and Command instruction is bypassed.

Following diagram indicates XIP operation with Flash XIP Enhanced mode:

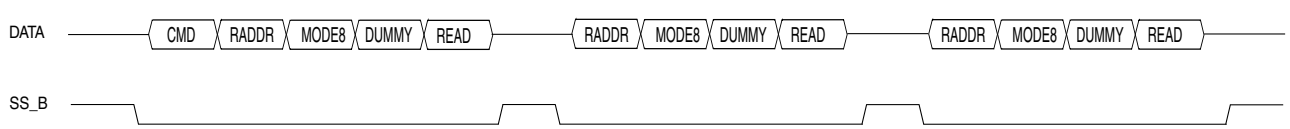


Figure 10-52. XIP Enhanced Mode operation

10.2.5 Application information

This section describes applications supported by the FlexSPI module.

10.2.5.1 FlexSPI Initialization

FlexSPI controller initialization sequence is as following:

- Enable controller clocks (AHB clock/IP Bus clock/Serial root clock) in System level.
- Set MCR0[MDIS] to 0x1 (Make sure controller is configured in module stop mode)
- Configure module control registers: MCR0, MCR1, MCR2. (Don't change MCR0[MDIS])
- Configure AHB bus control register (AHBCR) and AHB RX Buffer control register (AHBRXBUFxCR0) optionally, if AHB command will be used
- Configure Flash control registers (FLSHxCR0,FLSHxCR1,FLSHxCR2) according to external device type
- Configure DLL control register (DLLxCR) according to sample clock source selection
- set MCR0[MDIS] to 0x0 (Exit module stop mode)
- Configure LUT as needed (For AHB command or IP command)
- Reset controller optionally (by set MCR0[SWRESET] to 0x1)

External device needs configuration by IP command normally after controller initialization. For example, the device configuration is done by WRITE STATUS command for most serial NOR Flash.

10.2.5.2 Overview of Error Flags

The following table gives an overview of error category, flags and triggered source.

Table 10-7. Error category and flags in FlexSPI

Error Category	Triggered Source	Description	Error Flags
Command grant error	AHB write command	Command grant timeout	INTR[AHBCMDGE] will be set AHB bus error response
	AHB read command		INTR[AHBCMDGE] will be set AHB bus error response
	IP command		INTR[IPCMDGE] will be set
Command check error	AHB write command	• AHB write command with JMP_ON_CS instruction used in the sequence • There is unknown instruction opcode in the sequence.	INTR[AHBCMDERR] will be set

Table continues on the next page...

Table 10-7. Error category and flags in FlexSPI (continued)

Error Category	Triggered Source	Description	Error Flags
		• Instruction DUMMY_SDR/ DUMMY_RWDS_SDR used in DDR sequence. • Instruction DUMMY_DDR/ DUMMY_RWDS_DDR used in SDR sequence.	Command is not executed when error detected in command check
	AHB read command	• There is unknown instruction opcode in the sequence. • Instruction DUMMY_SDR/ DUMMY_RWDS_SDR used in DDR sequence. • Instruction DUMMY_DDR/ DUMMY_RWDS_DDR used in SDR sequence.	INTR[AHBCMDERR] will be set Command is not executed when error detected in command check
	IP command	• IP command with JMP_ON_CS instruction used in the sequence • There is unknown instruction opcode in the sequence. • Instruction DUMMY_SDR/ DUMMY_RWDS_SDR used in DDR sequence. • Instruction DUMMY_DDR/ DUMMY_RWDS_DDR used in SDR sequence. • Flash boundary across.	INTR[IPCMDERR] will be set Command is not executed when error detected in command check
Command execution error	AHB write command	Command timeout during execution	INTR[AHBCMDERR] will be set INTR[SEQTIMEOUT] will be set There will be AHB bus error response except following case: • AHB write command is triggered by flush (INCR burst ended with AHB_TX_BUF not empty) • AHB bufferable write access and bufferable enabled (AHBCR[BUFFERABLEEN]=0x1)
	AHB read command		INTR[AHBCMDERR] will be set INTR[SEQTIMEOUT] will be set There will be AHB bus error response

Table continues on the next page...

Table 10-7. Error category and flags in FlexSPI (continued)

Error Category	Triggered Source	Description	Error Flags
	IP command		INTR[IPCMDERR] will be set INTR[SEQTIMEOUT] will be set
AHB Bus timeout	AHB write command AHB read command	AHB bus timeout (no bus ready return)	INTR[AHBBUSTIMEOUT] will be set There will be AHB bus error response
Data Learning Failed	Any command	There is no valid sample clock phase found after LEARN instruction executed	INTR[DATALEARNFAIL] will be set

NOTE

- flash_top_address is the top address of currently accessed flash

10.2.5.3 Application on Serial NOR Flash device

This section provides the example sequences for serial NOR flash device (Cypress Flash S25FS128S).

10.2.5.3.1 Write Enable command

The following table shows WRITE ENABLE command sequence.

Table 10-8. WRITE ENABLE command

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x0	0x06	command name:WREN
1~7	STOP (0x00)	0x0	0x00	

10.2.5.3.2 Write Registers command

The following table shows Write Registers command sequence.

Table 10-9. Write Registers command

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x0 (Single)	0x01	command name:WRR

Table continues on the next page...

Table 10-9. Write Registers command (continued)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
1	WRITE_SDR	0x0 (Single)	0x1 or 0x2	Data size is 1 or 2byte. Byte 0 write data for Status Register 1; Byte 1 write data for Configuration Register 1 This value could be overridden by IPCR1[IDATSZ].
2~7	STOP (0x00)	0x0	0x00	

10.2.5.3.3 Page Program command

The following table shows Page Program command sequence.

Table 10-10. PAGE PROGRAM command (Cypress serial NOR flash)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x0 (Single)	0x02 or 0x12	command name: PP or 4PP PP is 3-Byte address mode. 4PP is 4-Byte address mode
1	ADDR_SDR	0x0 (Single)	0x18 or 0x20	Address bit number (operand value) should be 24 in 3-Byte address mode and 32 in 4-Byte address mode.
2	WRITE_SDR	0x0 (Single)	Any non-zero value	This operand value could be used as default programming data size if IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
3~7	STOP (0x00)	0x0	0x00	

The following table shows Page Program command sequence (QPI mode).

Table 10-11. PAGE PROGRAM (QPI mode) command (Cypress serial NOR flash)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x2 (Quad)	0x02 or 0x12	command name: PP or 4PP PP is 3-Byte address mode. 4PP is 4-Byte address mode
1	ADDR_SDR	0x2 (Quad)	0x18 or 0x20	Address bit number (operand value) should be 24 in 3-Byte address mode and 32 in 4-Byte address mode.
2	WRITE_SDR	0x2 (Quad)	Any non-zero value	This operand value could be used as default programming data size if

Table continues on the next page...

Table 10-11. PAGE PROGRAM (QPI mode) command (Cypress serial NOR flash) (continued)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
				IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
3~7	STOP (0x00)	0x0	0x00	

10.2.5.3.4 Read Status 1 command

The following table shows READ STATUS 1 command sequence.

Table 10-12. READ STATUS 1 command

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x0 (Single)	0x05	command name:RDSR1
1	READ_SDR	0x0 (Single)	0x1	1 Byte for Status register 1
2~7	STOP (0x00)	0x0	0x00	

10.2.5.3.5 Read command

The following table shows READ command sequence.

Table 10-13. READ command

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x0 (Single)	0x03 or 0x13	command name:READ or 4READ READ is 3-Byte address mode. 4READ is 4-Byte address mode
1	ADDR_SDR	0x0 (Single)	0x18 or 0x20	Address bit number (operand value) should be 24 in 3-Byte address mode and 32 in 4-Byte address mode.
2	READ_SDR	0x0 (Single)	Any non-zero value	This operand value could be used as default reading data size if IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
3~7	STOP (0x00)	0x0	0x00	

10.2.5.3.6 Fast Read command

The following table shows Fast Read command sequence (In the case of Cypress SPI Configuration Register bits CR2V[7]=0, CR2V[3:0]=0x8).

Table 10-14. FAST_READ command

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x0 (Single)	0x0B or 0x0C	command name:FAST_READ or 4FAST_READ FAST_READ is 3-Byte address mode. 4FAST_READ is 4-Byte address mode
1	ADDR_SDR	0x0 (Single)	0x18 or 0x20	Address bit number (operand value) should be 24 in 3-Byte address mode and 32 in 4-Byte address mode.
2	DUMMY_SDR	0x0 (Single)	0x08	Dummy cycle is 8 (in serial root clock) as CR2V[3:0]=0x8.
3	READ_SDR	0x0 (Single)	Any non-zero value	This operand value could be used as default reading data size if IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
4~7	STOP (0x00)	0x0	0x00	

10.2.5.3.7 Dual IO Fast Read command

The following table shows Dual IO FAST_READ command sequence (In the case of Cypress SPI Configuration Register bits CR2V[7]=0, CR2V[3:0]=0x8, Continuous Read mode).

Table 10-15. Dual IO FAST_READ command (Continuous Read mode)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x0 (Single)	0xBB or 0xBC	command name:DIOR or 4DIOR DIOR is 3-Byte address mode. 4DIOR is 4-Byte address mode
1	ADDR_SDR	0x1 (Dual)	0x18 or 0x20	Address bit number (operand value) should be 24 in 3-Byte address mode and 32 in 4-Byte address mode.
2	MODE8_SDR	0x1 (Dual)	0xAx	Enter continuous read mode or keep in continuous read mode.
3	DUMMY_SDR	0x1 (Dual)	0x08	Dummy cycle is 8 (in serial root clock) as CR2V[3:0]=0x8.
4	READ_SDR	0x1 (Dual)	Any non-zero value	This operand value could be used as default reading data size if IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
5	JMP_ON_CS	0x0 (or Dont care)	0x01	The CMD instruction will be bypassed after the first read access.
6~7	STOP (0x00)	0x0	0x00	

The following table shows Dual IO FAST_READ command sequence (In the case of Cypress SPI Configuration Register bits CR2V[7]=0, CR2V[3:0]=0x8, Non-continuous Read mode).

Table 10-16. Dual IO FAST_READ command (Non-continuous Read mode)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x0 (Single)	0xBB or 0xBC	command name:DIOR or 4DIOR DIOR is 3-Byte address mode. 4DIOR is 4-Byte address mode
1	ADDR_SDR	0x1 (Dual)	0x18 or 0x20	Address bit number (operand value) should be 24 in 3-Byte address mode and 32 in 4-Byte address mode.
2	MODE8_SDR	0x1 (Dual)	Any value other than 0xAx	Exit continuous read mode or keep in non-continuous read mode.
3	DUMMY_SDR	0x1 (Dual)	0x08	Dummy cycle is 8 (in serial root clock) as CR2V[3:0]=0x8.
4	READ_SDR	0x1 (Dual)	Any non-zero value	This operand value could be used as default reading data size if IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
5~7	STOP (0x00)	0x0	0x00	

10.2.5.3.8 Quad IO Fast Read command

The following table shows Quad IO FAST_READ command sequence (In the case of Cypress SPI Configuration Register bits CR2V[7]=0, CR2V[6]=0, CR2V[3:0]=0x8, Non-QPI mode, Non-continuous read mode).

Table 10-17. Quad IO FAST_READ command (Non-QPI mode, Non-continuous read mode)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x0 (Single)	0xEB or 0xEC	command name:QIOR or 4QIOR QIOR is 3-Byte address mode. 4QIOR is 4-Byte address mode
1	ADDR_SDR	0x2 (Quad)	0x18 or 0x20	Address bit number (operand value) should be 24 in 3-Byte address mode and 32 in 4-Byte address mode.
2	MODE8_SDR	0x2 (Quad)	Any value other than 0xAx	Exit continuous read mode or keep in non-continuous read mode.
3	DUMMY_SDR	0x2 (Quad)	0x08	Dummy cycle is 8 (in serial root clock) as CR2V[3:0]=0x8.
4	READ_SDR	0x2 (Quad)	Any non-zero value	This operand value could be used as default reading data size if

Table continues on the next page...

Table 10-17. Quad IO FAST_READ command (Non-QPI mode, Non-continuous read mode) (continued)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
				IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
5~7	STOP (0x00)	0x0	0x00	

The following table shows Quad IO FAST_READ command sequence (In the case of Cypress SPI Configuration Register bits CR2V[7]=0, CR2V[6]=0, CR2V[3:0]=0x8, Non-QPI mode, Continuous read mode).

Table 10-18. Quad IO FAST_READ command (Non-QPI mode, Continuous read mode)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x0 (Single)	0xEB or 0xEC	command name:QIOR or 4QIOR QIOR is 3-Byte address mode. 4QIOR is 4-Byte address mode
1	ADDR_SDR	0x2 (Quad)	0x18 or 0x20	Address bit number (operand value) should be 24 in 3-Byte address mode and 32 in 4-Byte address mode.
2	MODE8_SDR	0x2 (Quad)	0xA0	Enter continuous read mode or keep in continuous read mode.
3	DUMMY_SDR	0x2 (Quad)	0x08	Dummy cycle is 8 (in serial root clock) as CR2V[3:0]=0x8.
4	READ_SDR	0x2 (Quad)	Any non-zero value	This operand value could be used as default reading data size if IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
5	JMP_ON_CS	0x0 (or Dont care)	0x01	The CMD instruction will be bypassed after the first read access.
6~7	STOP (0x00)	0x0	0x00	

The following table shows Quad IO FAST_READ command sequence (In the case of Cypress SPI Configuration Register bits CR2V[7]=0, CR2V[6]=1, CR2V[3:0]=0x8, QPI mode, Continuous read mode).

Table 10-19. Quad IO FAST_READ command (QPI mode, Continuous read mode)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x2 (Quad)	0xEB or 0xEC	command name:QIOR or 4QIOR QIOR is 3-Byte address mode. 4QIOR is 4-Byte address mode

Table continues on the next page...

Table 10-19. Quad IO FAST_READ command (QPI mode, Continuous read mode) (continued)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
1	ADDR_SDR	0x2 (Quad)	0x18 or 0x20	Address bit number (operand value) should be 24 in 3-Byte address mode and 32 in 4-Byte address mode.
2	MODE8_SDR	0x2 (Quad)	0xA0	Enter continuous read mode or keep in continuous read mode.
3	DUMMY_SDR	0x2 (Quad)	0x08	Dummy cycle is 8 (in serial root clock) as CR2V[3:0]=0x8.
4	READ_SDR	0x2 (Quad)	Any non-zero value	This operand value could be used as default reading data size if IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
5	JMP_ON_CS	0x0 (or Dont care)	0x01	The CMD instruction will be bypassed after the first read access.
6~7	STOP (0x00)	0x0	0x00	

10.2.5.3.9 DDR Quad IO Fast Read command

The following table shows DDR Quad IO FAST_READ command sequence (In the case of Cypress SPI Configuration Register bits CR2V[7]=0, CR2V[6]=0, CR2V[3:0]=0x8, Non-QPI mode, Non-continuous read mode).

Table 10-20. DDR Quad IO FAST_READ command (Non-QPI mode, Non-continuous read mode)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x0 (Single)	0xED or 0xEE	command name:QIOR_DDR or 4QIOR_DDR QIOR_DDR is 3-Byte address mode. 4QIOR_DDR is 4-Byte address mode
1	ADDR_DDR	0x2 (Quad)	0x18 or 0x20	Address bit number (operand value) should be 24 in 3-Byte address mode and 32 in 4-Byte address mode.
2	MODE8_DDR	0x2 (Quad)	Any value other than 0xAx	Exit continuous read mode or keep in non-continuous read mode.
3	DUMMY_DDR	0x2 (Quad)	0x08	Dummy cycle is 8 (in serial root clock) as CR2V[3:0]=0x8.
4	LEARN_DDR	0x2 (Quad)	0x1	DLP (Data Learning Pattern is 8 bits).
5	READ_DDR	0x2 (Quad)	Any non-zero value	This operand value could be used as default reading data size if IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
6~7	STOP (0x00)	0x0	0x00	

The following table shows DDR Quad IO FAST_READ command sequence (In the case of Cypress SPI Configuration Register bits CR2V[7]=0, CR2V[6]=0, CR2V[3:0]=0x8, Non-QPI mode, Continuous read mode).

Table 10-21. DDR Quad IO FAST_READ command (Non-QPI mode, Continuous read mode)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x0 (Single)	0xED or 0xEE	command name:QIOR_DDR or 4QIOR_DDR QIOR_DDR is 3-Byte address mode. 4QIOR_DDR is 4-Byte address mode
1	ADDR_DDR	0x2 (Quad)	0x18 or 0x20	Address bit number (operand value) should be 24 in 3-Byte address mode and 32 in 4-Byte address mode.
2	MODE8_DDR	0x2 (Quad)	0xA0	Enter continuous read mode or keep in continuous read mode.
3	DUMMY_DDR	0x2 (Quad)	0x08	Dummy cycle is 8 (in serial root clock) as CR2V[3:0]=0x8.
4	LEARN_DDR	0x2 (Quad)	0x1	DLP (Data Learning Pattern is 8 bits).
5	READ_DDR	0x2 (Quad)	Any non-zero value	This operand value could be used as default reading data size if IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
6	JMP_ON_CS	0x0 (or Dont care)	0x01	The CMD instruction will be bypassed after the first read access.
7	STOP (0x00)	0x0	0x00	

The following table shows DDR Quad IO FAST_READ command sequence (In the case of Cypress SPI Configuration Register bits CR2V[7]=0, CR2V[6]=1, CR2V[3:0]=0x8, QPI mode, Continuous read mode).

Table 10-22. DDR Quad IO FAST_READ command (QPI mode, Continuous read mode)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x2 (Quad)	0xED or 0xEE	command name:QIOR_DDR or 4QIOR_DDR QIOR_DDR is 3-Byte address mode. 4QIOR_DDR is 4-Byte address mode
1	ADDR_DDR	0x2 (Quad)	0x18 or 0x20	Address bit number (operand value) should be 24 in 3-Byte address mode and 32 in 4-Byte address mode.
2	MODE8_DDR	0x2 (Quad)	0xA0	Enter continuous read mode or keep in continuous read mode.
3	DUMMY_DDR	0x2 (Quad)	0x08	Dummy cycle is 8 (in serial root clock) as CR2V[3:0]=0x8.

Table continues on the next page...

Table 10-22. DDR Quad IO FAST_READ command (QPI mode, Continuous read mode) (continued)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
4	LEARN_DDR	0x2 (Quad)	0x1	DLP (Data Learning Pattern is 8 bits).
5	READ_DDR	0x2 (Quad)	Any non-zero value	This operand value could be used as default reading data size if IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
6	JMP_ON_CS	0x0 (or Dont care)	0x01	The CMD instruction will be bypassed after the first read access.
7	STOP (0x00)	0x0	0x00	

10.2.5.4 Application on HyperBus device

This section provides the example sequences for HyperBus device (Cypress RPC flash/ HyperRam/HyperFlash).

10.2.5.4.1 HyperFlash

This section provides the example sequences for HyperFlash devices (Cypress S26KS series).

The following table shows Read Status command sequence.

Table 10-23. Read Status command

Seq No.	Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0 (Write - Addr=0x555, Data=0x70)	0	CMD_DDR	0x3 (Octal)	0x20	CA bit 47: (R/W#) = 0x0 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
	1	CMD_DDR	0x3 (Octal)	0x00	Row Address: 0x0000AA (24 bit)
	2	CMD_DDR	0x3 (Octal)	0x00	
	3	CMD_DDR	0x3 (Octal)	0xAA	
	4	CMD_DDR	0x3 (Octal)	0x00	Column Address: 0x05 (13 zero bits + 3 valid bits)
	5	CMD_DDR	0x3 (Octal)	0x05	
	6	CMD_DDR	0x3 (Octal)	0x00	Write Data: 0x0070
	7	CMD_DDR	0x3 (Octal)	0x70	
1	0	CMD_DDR	0x3 (Octal)	0xA0	CA bit 47: (R/W#) = 0x1 CA bit 46: (Target) = 0x0

Table continues on the next page...

Table 10-23. Read Status command (continued)

Seq No.	Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
(Read - Addr=xxx, Data= Status register data)					CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
	1	RADDR_DDR	0x3 (Octal)	0x18	Row Address: 24 bits
	2	CADDR_DDR	0x3 (Octal)	0x10	Column Address: (13 zero bits + 3 valid bits)
	3	DUMMY_RWDS_DDR	0x3 (Octal)	0x0B	In case of latency count=11
	4	READ_DDR	0x3 (Octal)	0x4	4 Byte read
	5~7	STOP (0x00)	0x0	0x00	

The following table shows Read (memory) command sequence.

Table 10-24. Read (memory) command

Seq No.	Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0 (Read - Addr=xxx, Data= memory data)	0	CMD_DDR	0x3 (Octal)	0xA0	CA bit 47: (R/W#) = 0x1 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
	1	RADDR_DDR	0x3 (Octal)	0x18	Row Address: 24 bits
	2	CADDR_DDR	0x3 (Octal)	0x10	Column Address: (13 zero bits + 3 valid bits)
	3	DUMMY_RWDS_DDR	0x3 (Octal)	0x0B	In case of latency count=11
	4	READ_DDR	0x3 (Octal)	Any non-zero value	This operand value could be used as default reading data size if IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
	5~7	STOP (0x00)	0x0	0x00	

The following table shows Word Program command sequence.

Table 10-25. Word Program command

Seq No.	Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0 (Write - Addr=0x555, Data=0xAA)	0	CMD_DDR	0x3 (Octal)	0x20	CA bit 47: (R/W#) = 0x0 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
	1	CMD_DDR	0x3 (Octal)	0x00	Row Address: 0x0000AA (24 bit)

Table continues on the next page...

Table 10-25. Word Program command (continued)

Seq No.	Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
	2	CMD_DDR	0x3 (Octal)	0x00	
	3	CMD_DDR	0x3 (Octal)	0xAA	
	4	CMD_DDR	0x3 (Octal)	0x00	Column Address: 0x05 (13 zero bits + 3 valid bits)
	5	CMD_DDR	0x3 (Octal)	0x05	
	6	CMD_DDR	0x3 (Octal)	0x00	Write Data: 0x00AA
	7	CMD_DDR	0x3 (Octal)	0xAA	
1 (Write - Addr=0x2AA, Data=0x55)	0	CMD_DDR	0x3 (Octal)	0x20	CA bit 47: (R/W#) = 0x0 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
	1	CMD_DDR	0x3 (Octal)	0x00	Row Address: 0x000055 (24 bit)
	2	CMD_DDR	0x3 (Octal)	0x00	
	3	CMD_DDR	0x3 (Octal)	0x55	
	4	CMD_DDR	0x3 (Octal)	0x00	Column Address: 0x02 (13 zero bits + 3 valid bits)
	5	CMD_DDR	0x3 (Octal)	0x02	
	6	CMD_DDR	0x3 (Octal)	0x00	Write Data: 0x0055
	7	CMD_DDR	0x3 (Octal)	0x55	
2 (Write - Addr=0x555, Data=0xA0)	0	CMD_DDR	0x3 (Octal)	0x20	CA bit 47: (R/W#) = 0x0 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
	1	CMD_DDR	0x3 (Octal)	0x00	Row Address: 0x0000AA (24 bit)
	2	CMD_DDR	0x3 (Octal)	0x00	
	3	CMD_DDR	0x3 (Octal)	0xAA	
	4	CMD_DDR	0x3 (Octal)	0x00	Column Address: 0x05 (13 zero bits + 3 valid bits)
	5	CMD_DDR	0x3 (Octal)	0x55	
	6	CMD_DDR	0x3 (Octal)	0x00	Write Data: 0x00A0
	7	CMD_DDR	0x3 (Octal)	0xA0	
3 (Word Program)	0	CMD_DDR	0x3 (Octal)	0x20	CA bit 47: (R/W#) = 0x0 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
	1	RADDR_DDR	0x3 (Octal)	0x18	Row Address: 24 bits
	2	CADDR_DDR	0x3 (Octal)	0x10	Column Address: (13 zero bits + 3 valid bits)
	3	WRITE_DDR	0x3 (Octal)	0x02	2 Byte written data
	4-7	STOP (0x0)	0x0	0x00	

The following table shows Written-to-Buffer and Program-Buffer-to-Flash command sequence.

Table 10-26. Written-to-Buffer and Program-Buffer-to-Flash command

Seq No.	Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0 (Write - Addr=0x555, Data=0xAA)	0	CMD_DDR	0x3 (Octal)	0x20	CA bit 47: (R/W#) = 0x0 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
	1	CMD_DDR	0x3 (Octal)	0x00	Row Address: 0x0000AA (24 bit)
	2	CMD_DDR	0x3 (Octal)	0x00	
	3	CMD_DDR	0x3 (Octal)	0xAA	
	4	CMD_DDR	0x3 (Octal)	0x00	Column Address: 0x05 (13 zero bits + 3 valid bits)
	5	CMD_DDR	0x3 (Octal)	0x05	
	6	CMD_DDR	0x3 (Octal)	0x00	Write Data: 0x00AA
	7	CMD_DDR	0x3 (Octal)	0xAA	
1 (Write - Addr=0x2AA, Data=0x55)	0	CMD_DDR	0x3 (Octal)	0x20	CA bit 47: (R/W#) = 0x0 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
	1	CMD_DDR	0x3 (Octal)	0x00	Row Address: 0x000055 (24 bit)
	2	CMD_DDR	0x3 (Octal)	0x00	
	3	CMD_DDR	0x3 (Octal)	0x55	
	4	CMD_DDR	0x3 (Octal)	0x00	Column Address: 0x02 (13 zero bits + 3 valid bits)
	5	CMD_DDR	0x3 (Octal)	0x02	
	6	CMD_DDR	0x3 (Octal)	0x00	Write Data: 0x0055
	7	CMD_DDR	0x3 (Octal)	0x55	
2 (Write - Addr=SA, Data=0x25)	0	CMD_DDR	0x3 (Octal)	0x20	CA bit 47: (R/W#) = 0x0 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
	1	RADDR_DDR	0x3 (Octal)	0x18	Row Address: SA (24 bit) SA is sector address. Please set IPCR0[SFAR]=SA
	2	CADDR_DDR	0x3 (Octal)	0x10	Column Address: 13 zero bits + 3 valid bits
	3	CMD_DDR	0x3 (Octal)	0x00	Write Data: 0x0025
	4	CMD_DDR	0x3 (Octal)	0x25	
2 (Write - Addr=SA, Data=WC)	0	CMD_DDR	0x3 (Octal)	0x20	CA bit 47: (R/W#) = 0x0 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1

Table continues on the next page...

Table 10-26. Written-to-Buffer and Program-Buffer-to-Flash command (continued)

Seq No.	Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
					CA bit 44-40: All reserved = 0x0
	1	RADDR_DDR	0x3 (Octal)	0x18	Row Address: SA (24 bit) SA is sector address. Please set IPCR0[SFAR]=SA
	2	CADDR_DDR	0x3 (Octal)	0x10	Column Address: 13 zero bits + 3 valid bits
	3	CMD_DDR	0x3 (Octal)	WC	Write Data: WC
	4	CMD_DDR	0x3 (Octal)		WC is word count
3 - N (Write - Addr=WBL, Data=PD) N is the word count + 2	0	CMD_DDR	0x3 (Octal)	0x20	CA bit 47: (R/W#) = 0x0 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
	1	RADDR_DDR	0x3 (Octal)	0x18	Row Address: WBL (24 bit) WBL is write buffer location. Please set IPCR0[SFAR]=WBL
	2	CADDR_DDR	0x3 (Octal)	0x10	Column Address: 13 zero bits + 3 valid bits
	3	WRITE_DDR	0x3 (Octal)	0x02	2 Byte write data
	4-7	STOP (0x0)	0x0	0x00	
N+1 (Write - Addr=SA, Data=29) Program Buffer to Flash	0	CMD_DDR	0x3 (Octal)	0x20	CA bit 47: (R/W#) = 0x0 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
	1	RADDR_DDR	0x3 (Octal)	0x18	Row Address: SA (24 bit) SA is sector address. Please set IPCR0[SFAR]=SA
	2	CADDR_DDR	0x3 (Octal)	0x10	Column Address: 13 zero bits + 3 valid bits
	3	CMD_DDR	0x3 (Octal)	0x00	Write Data: 0x29
	4	CMD_DDR	0x3 (Octal)	0x29	
	5-7	STOP (0x0)	0x0	0x00	

10.2.5.4.2 HyperRAM

This section provides the example sequences for HyperRAM (Cypress S27KL series).

Read (memory) command sequence is same as HyperFlash. The following table shows Write (memory) command sequence.

Table 10-27. Write (memory) command

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_DDR	0x3 (Octal)	0x20	CA bit 47: (R/W#) = 0x0 CA bit 46: (Target) = 0x0 CA bit 45: (Burst Type) = 0x1 CA bit 44-40: All reserved = 0x0
1	RADDR_DDR	0x3 (Octal)	0x18	Row Address: 24 bits
2	CADDR_DDR	0x3 (Octal)	0x10	Column Address: (13 zero bits + 3 valid bits)
3	DUMMY_RWDS_DDR	0x3 (Octal)	0x0B	In case of latency count=11
4	WRITE_DDR	0x3 (Octal)	Any non-zero value	This operand value could be used as default write data size if IPCR1[IDATSZ] is zero. This value is ignored for AHB command.
5~7	STOP (0x00)	0x0	0x00	

10.2.5.5 Application on Serial NAND Flash device

This section provides the example sequences for serial NAND flash device (Micron Flash MT29 series). The operation to serial NAND flash is quite similar to serial NOR flash.

READ operation sequence is as following:

- Page Read (Transfer the data from the NAND Flash array to the cache register)
- Get Feature to read the status
- Random Data Read

The following table shows Page Read command sequence.

Table 10-28. Page Read command

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x1 (Single)	0x13	Command code: 0x13
1	RADDR_SDR	0x1 (Single)	0x18	Row Address: 24 bit
2-7	STOP (0x0)	0x0	0x00	

The following table shows Get Feature command sequence.

Table 10-29. Get Feature command

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x1 (Single)	0x0F	Command code: 0x0F
1	CMD_SDR	0x1 (Single)	0xC0	Status register address (0xC0)
2	READ_SDR	0x1 (Single)	0x02	2 Byte read data
3-7	STOP (0x0)	0x0	0x00	

The following table shows Random Data Read command sequence.

Table 10-30. Read From Cache x4 command

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x1 (Single)	0x6B	Command code: 0x6B
1	MODE4_SDR	0x1 (Single)	0x0 or 0x1	Software should decode the flash address and set mode bits as 0x1 if plane selection is one, or 0x0 if plane selection is zero. Plane selection bit is the 18th bit of flash address. If NAND flash size is less than 4Gbit, plane selection will always be zero.
2	CADDR_SDR	0x1 (Single)	0x0C	Column address: 12 bit
3	DUMMY_SDR	0x2 (Quad)	0x08	Dummy cycle number: 8 (serial root clock)
4	READ_SDR	0x2 (Quad)	Any non-zero value	This operand value could be used as default reading data size if IPCR1[IDATSZ] is zero.
5-7	STOP (0x0)	0x0	0x00	

Program operation sequence is as following:

- Write Enable
- Program Load (Transfer the write data to the cache register)
- Program Execute
- Get Feature to read the status

The following table shows Program Load command sequence.

Table 10-31. Program Load command

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x1 (Single)	0x02	Command code: 0x02
1	MODE4_SDR	0x1 (Single)	0x0 or 0x1	Software should decode the flash address and set mode bits as 0x1 if

Table continues on the next page...

Table 10-31. Program Load command (continued)

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
				plane selection is one, or 0x0 if plane selection is zero. Plane selection bit is the 18th bit of flash address. If NAND flash size is less than 4Gbit, plane selection will always be zero.
2	CADDR_SDR	0x1 (Single)	0x0C	Column address: 12 bit
3	WRITE_SDR	0x1 (Single)	Any non-zero value	This operand value could be used as default write data size if IPCR1[IDATSZ] is zero.
4-7	STOP (0x0)	0x0	0x00	

The following table shows Program Execute command sequence.

Table 10-32. Program Execute command

Instruction No.	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0x1 (Single)	0x10	Command code: 0x10
1	RADDR_SDR	0x1 (Single)	0x18	Row Address: 24 bit
2-7	STOP (0x0)	0x0	0x00	

10.2.5.6 Application on FPGA device

FPGA device should be accessed by AHB command. All AHB accesses to FPGA will be transparent to SW driver (no SW intervention).There may be some special requirements from FPGA device.

1. **Device type may be different on A1/A2/B1/B2**

   For this case, clear the MCR2[SAMEDEVICEEN] bit and configure FLSHxCR0 and FLSHxCR1 register separately for up to four external devices.

2. **Device needs different wait cycle for Programming.**

   The AHB write wait cycle number could be set separately for these four external devices (by register field FLSHxCR2[AWRWAIT]). Software could configure the sequences in LUT with different DUMMY instructions (operand will determine dummy cycle). Note that FlexSPI will hold AHB bus ready for this wait time, so AHB Bus performance may become very low when this wait time is very long.

3. **Device needs different wait cycle for Reading.**

The AHB Read Sequence index and Sequence Number could be set separately for these four external devices (by register field FLSHxCR2[ARDSEQID] and FLSHxCR2[ARDSEQNUM]). Software could configure the sequences in LUT with different DUMMY instruction (operand will determine dummy cycle).

4. **Device may be sensitive to read instruction clock cycle number**

   Device will be sensitive to read instruction clock cycle number if its internal memory is implemented similar as FIFO. For this case, software could send the data size information to external device by DATSZ instruction. FPGA device should decode the data size information and determine how much data bytes should be popped.

5. **Device may needs interval time between Chip selection valid**

   This could be handled by register field FLSHxCR1[CSINTERVAL] setting.

6. **Device may use SCLK as reference clock for its internal PLL**

   In this case, SCLK should be free-running and clock frequency should be stable. This could be achieved by setting MCR0[SCKFREERUNEN] and use SDR sequence only.

10.2.6 Memory Map and register definition

This section includes the FlexSPI module memory map and detailed descriptions of all registers.

10.2.6.1 Register Access

All registers can be accessed with 8-bit, 16-bit, and 32-bit width operations. Never change the setting value of reserved fields in control registers. Changing the value of reserved fields may impact the normal functioning of the controller.

NOTE

For usage that FlexSPI is capable of accessing sensitive memory contents, protection should be done to FlexSPI controller using resource isolation (e.g. XRDC2, XRDC, RDC) or any kind of equivalent partition control via SCFW, to make sure only trusted software be allowed to access the FlexSPI controller register interface.

10.2.6.2 FlexSPI register descriptions

10.2.6.2.1 FlexSPI Memory map

FlexSPI base address: 30BB_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Control Register 0 (MCR0)	32	RW	FFFF_80C2h
4h	Module Control Register 1 (MCR1)	32	RW	FFFF_FFFFh
8h	Module Control Register 2 (MCR2)	32	RW	2000_81F7h
Ch	AHB Bus Control Register (AHBCR)	32	RW	0000_0018h
10h	Interrupt Enable Register (INTEN)	32	RW	0000_0000h
14h	Interrupt Register (INTR)	32	W1C	0000_0000h
18h	LUT Key Register (LUTKEY)	32	RW	5AF0_5AF0h
1Ch	LUT Control Register (LUTCR)	32	RW	0000_0002h
20h	AHB RX Buffer 0 Control Register 0 (AHBRXBUF0CR0)	32	RW	8000_0020h
24h	AHB RX Buffer 1 Control Register 0 (AHBRXBUF1CR0)	32	RW	8001_0020h
28h	AHB RX Buffer 2 Control Register 0 (AHBRXBUF2CR0)	32	RW	8002_0020h
2Ch	AHB RX Buffer 3 Control Register 0 (AHBRXBUF3CR0)	32	RW	8003_0020h
30h	AHB RX Buffer 4 Control Register 0 (AHBRXBUF4CR0)	32	RW	8004_0020h
34h	AHB RX Buffer 5 Control Register 0 (AHBRXBUF5CR0)	32	RW	8005_0020h
38h	AHB RX Buffer 6 Control Register 0 (AHBRXBUF6CR0)	32	RW	8006_0020h
3Ch	AHB RX Buffer 7 Control Register 0 (AHBRXBUF7CR0)	32	RW	8007_0020h
60h	Flash Control Register 0 (FLSHA1CR0)	32	RW	0001_0000h
64h	Flash Control Register 0 (FLSHA2CR0)	32	RW	0001_0000h
68h	Flash Control Register 0 (FLSHB1CR0)	32	RW	0001_0000h
6Ch	Flash Control Register 0 (FLSHB2CR0)	32	RW	0001_0000h
70h	Flash Control Register 1 (FLSHA1CR1)	32	RW	0000_0063h
74h	Flash Control Register 1 (FLSHA2CR1)	32	RW	0000_0063h
78h	Flash Control Register 1 (FLSHB1CR1)	32	RW	0000_0063h
7Ch	Flash Control Register 1 (FLSHB2CR1)	32	RW	0000_0063h
80h	Flash Control Register 2 (FLSHA1CR2)	32	RW	0000_0000h
84h	Flash Control Register 2 (FLSHA2CR2)	32	RW	0000_0000h
88h	Flash Control Register 2 (FLSHB1CR2)	32	RW	0000_0000h
8Ch	Flash Control Register 2 (FLSHB2CR2)	32	RW	0000_0000h
94h	Flash Control Register 4 (FLSHCR4)	32	RW	0000_0000h
A0h	IP Control Register 0 (IPCR0)	32	RW	0000_0000h
A4h	IP Control Register 1 (IPCR1)	32	RW	0000_0000h
B0h	IP Command Register (IPCMD)	32	RW	0000_0000h
B4h	Data Learn Pattern Register (DLPR)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
B8h	IP RX FIFO Control Register (IPRXFCR)	32	RW	0000_0000h
BCh	IP TX FIFO Control Register (IPTXFCR)	32	RW	0000_0000h
C0h	DLL Control Register 0 (DLLACR)	32	RW	0000_0100h
C4h	DLL Control Register 0 (DLLBCR)	32	RW	0000_0100h
E0h	Status Register 0 (STS0)	32	RO	0000_0002h
E4h	Status Register 1 (STS1)	32	RO	0000_0000h
E8h	Status Register 2 (STS2)	32	RO	0100_0100h
ECh	AHB Suspend Status Register (AHBSPNDSTS)	32	RO	0000_0000h
F0h	IP RX FIFO Status Register (IPRXFSTS)	32	RO	0000_0000h
F4h	IP TX FIFO Status Register (IPTXFSTS)	32	RO	0000_0000h
100h - 17Ch	IP RX FIFO Data Register a (RFDR0 - RFDR31)	32	RO	0000_0000h
180h - 1FCh	IP TX FIFO Data Register a (TFDR0 - TFDR31)	32	WO	0000_0000h
200h - 3FCh	LUT a (LUT0 - LUT127)	32	RW	Table 10-76

10.2.6.2.2 Module Control Register 0 (MCR0)

10.2.6.2.2.1 Offset

Register	Offset
MCR0	0h

10.2.6.2.2.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	AHBGRANTWAIT								IPGRANTWAIT							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LEARNEN	SCKFREERUNEN	COMBINATIONEN	DOZEEN	HSEN	SERCLKDIV			ATDFEN	ARDFEN	RXCLKSRC		Reserved		MDIS	SWRESET
Reset	1	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0

10.2.6.2.2.3 Fields

Field	Function
31-24 AHBGRANTWAIT	Timeout wait cycle for AHB command grant. If AHB Triggered Command is not granted by arbitrator, it will timeout after AHBGRANTTIMEOUT * 1024 AHB Clock cycles. This grant timeout may occur when the pending command sequence is IP triggered and the read/write data size is too large. When an AHB command grant time out occurs, there will be an interrupt generated (INTR[AHBCMDGE]) if this interrupt is enabled (INTEN[AHBCMDGEEN] is set) and AHB command is ignored by arbitrator. **NOTE:** This field is for debug only, please keep default value! It is not allowed to set this field to value 0x0.
23-16 IPGRANTWAIT	Time out wait cycle for IP command grant. If IP Triggered Command is not granted by arbitrator, it will timeout after IPGRANTTIMEOUT * 1024 AHB Clock cycles. This grant timeout maybe occur when pending command sequence is AHB triggered and read/write data size is too large. When IP command grant time out occurs, there will be an interrupt generated (INTR[IPCMDGE]) if this interrupt is enabled (INTEN[IPCMDGEEN] is set 0x1) and IP command is ignored by arbitrator. **NOTE:** This field is for debug only, please keep default value! It is not allowed to set this field to value 0x0.
15 LEARNEN	This bit is used to enable/disable data learning feature. When data learning is disabled, the sampling clock phase 0 is always used for RX data sampling even if LEARN instruction is correctly executed. 0b - Disable. 1b - Enable.
14 SCKFREERUNEN	This bit is used to force SCLK output free-running. For FPGA applications, external device may use SCLK as reference clock to its internal PLL. If SCLK free-running is enabled, data sampling with loopback clock from SCLK pad is not supported (MCR0[RXCLKSRC]=2). 0b - Disable. 1b - Enable.
13 COMBINATIONEN	This bit is to support Flash Octal mode access by combining Port A and B Data pins (A_DATA[3:0] and B_DATA[3:0]). **NOTE:** Combination mode is not supported if Port A and Port B are 8 bit data width. This bit should be set to zero in this case. 0b - Disable. 1b - Enable.
12 DOZEEN	Doze mode enable bit 0b - Doze mode support disabled. AHB clock and serial clock will not be gated off when there is doze mode request from system. 1b - Doze mode support enabled. AHB clock and serial clock will be gated off when there is doze mode request from system.
11 HSEN	Half Speed Serial Flash access Enable. This bit enables the divide by 2 of the clock to external serial flash devices (A_SCLK/B_SCLK) for all commands (for both SDR and DDR mode). FlexSPI need to be set into MDIS mode before changing value of HSEN. Otherwise, it is possible to cause issue on internal logic/state machine. 0b - Disable divide by 2 of serial flash clock for half speed commands. 1b - Enable divide by 2 of serial flash clock for half speed commands.
10-8 SERCLKDIV	The serial root clock could be divided inside FlexSPI . Refer Clocks chapter for more details on clocking. **NOTE:** Don't change this field during IP's normal operation mode. Alter the value after putting IP into stop mode. 000b - Divided by 1

Table continues on the next page...

Field	Function
	001b - Divided by 2 010b - Divided by 3 011b - Divided by 4 100b - Divided by 5 101b - Divided by 6 110b - Divided by 7 111b - Divided by 8
7 ATDFEN	Enable AHB bus Write Access to IP TX FIFO. 0b - IP TX FIFO should be written by IP Bus. AHB Bus write access to IP TX FIFO memory space will get bus error response. 1b - IP TX FIFO should be written by AHB Bus. IP Bus write access to IP TX FIFO memory space will be ignored but no bus error response.
6 ARDFEN	Enable AHB bus Read Access to IP RX FIFO. 0b - IP RX FIFO should be read by IP Bus. AHB Bus read access to IP RX FIFO memory space will get bus error response. 1b - IP RX FIFO should be read by AHB Bus. IP Bus read access to IP RX FIFO memory space will always return data zero but no bus error response.
5-4 RXCLKSRC	Sample Clock source selection for Flash Reading Refer RX Clock Source Features for more details 00b - Dummy Read strobe generated by FlexSPI Controller and loopback internally. 01b - Dummy Read strobe generated by FlexSPI Controller and loopback from DQS pad. 10b - Reserved 11b - Flash provided Read strobe and input from DQS pad
3-2 —	Reserved.
1 MDIS	Module Disable When module disabled, AHB/serial clock will be gated off internally to save power. Only register access (except LUT/IP RX FIFO/IP TX FIFO) is allowed.
0 SWRESET	Software Reset This bit is auto-cleared by hardware after software reset done. Configuration registers will not be reset.

10.2.6.2.3 Module Control Register 1 (MCR1)

10.2.6.2.3.1 Offset

Register	Offset
MCR1	4h

10.2.6.2.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQWAIT															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	AHBBUSWAIT															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

10.2.6.2.3.3 Fields

Field	Function
31-16 SEQWAIT	Command Sequence Execution will timeout and abort after SEQWAIT * 1024 Serial Root Clock cycles. When sequence execution time out occurs, there will be an interrupt generated (INTR[SEQTIMEOUT]) if this interrupt is enabled (INTEN[SEQTIMEOUTEN] is set 0x1) and AHB command is ignored by arbitrator. **NOTE:** It is not allowed to set this field to value 0x0.
15-0 AHBBUSWAIT	AHB Read/Write access to Serial Flash Memory space will timeout if not data received from Flash or data not transmitted after AHBBUSWAIT * 1024 ahb clock cycles, AHB Bus will get an error response. When AHB bus time out occurs, there will be an interrupt generated (INTR[AHBBUSTIMEOUT]) if this interrupt is enabled (INTR[AHBBUSTIMEOUT]) is set 0x1) and AHB command is ignored by arbitrator. **NOTE:** It is not allowed to set this field to value 0x0.

10.2.6.2.4 Module Control Register 2 (MCR2)

10.2.6.2.4.1 Offset

Register	Offset
MCR2	8h

10.2.6.2.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RESUMEWAIT								Reserved				SCKBDIFFOPT	Reserved		Reserved
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SAMEDEVICEEN	CLRLEARNPHASE	Reserved	Reserved	CLRAHBBUFOPT	Reserved		Reserved								
Reset	1	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1

10.2.6.2.4.3 Fields

Field	Function
31-24 RESUMEWAIT	Wait cycle (in AHB clock cycle) for idle state before suspended command sequence resumed.
23-20 —	Reserved.
19 SCKBDIFFOPT	B_SCLK pad can be used as A_SCLK differential clock output (inverted clock to A_SCLK). In this case, port B flash access is not available. After changing the value of this field, MCR0[SWRESET] should be set. 0b - B_SCLK pad is used as port B SCLK clock output. Port B flash access is available. 1b - B_SCLK pad is used as port A SCLK inverted clock output (Differential clock to A_SCLK). Port B flash access is not available.
18-17 —	Reserved.
16 —	Reserved.
15 SAMEDEVICEEN	All external devices are same devices (both in types and size) for A1/A2/B1/B2. 0b - In Individual mode, FLSHA1CRx/FLSHA2CRx/FLSHB1CRx/FLSHB2CRx register setting will be applied to Flash A1/A2/B1/B2 separately. In Parallel mode, FLSHA1CRx register setting will be applied to Flash A1 and B1, FLSHA2CRx register setting will be applied to Flash A2 and B2. FLSHB1CRx/FLSHB2CRx register settings will be ignored. 1b - FLSHA1CR0/FLSHA1CR1/FLSHA1CR2 register settings will be applied to Flash A1/A2/B1/B2. FLSHA2CRx/FLSHB1CRx/FLSHB2CRx will be ignored.
14	The sampling clock phase selection will be reset to phase 0 when this bit is written with 0x1. This bit will be auto-cleared immediately.

Table continues on the next page...

Field	Function
CLRLEARNPHASE	
13 —	Reserved.
12 —	Reserved.
11 CLRAHBBUFOPT	This bit determines whether AHB RX Buffer and AHB TX Buffer will be cleaned automatically when FlexSPI returns STOP mode ACK. Software should set this bit if AHB RX Buffer or AHB TX Buffer will be powered off in STOP mode. Otherwise AHB read access after exiting STOP mode may hit AHB RX Buffer or AHB TX Buffer but their data entries are invalid. 0b - AHB RX/TX Buffer will not be cleaned automatically when FlexSPI return Stop mode ACK. 1b - AHB RX/TX Buffer will be cleaned automatically when FlexSPI return Stop mode ACK.
10-9 —	Reserved.
8-0 —	Reserved.

10.2.6.2.5 AHB Bus Control Register (AHBCR)

10.2.6.2.5.1 Offset

Register	Offset
AHBCR	Ch

10.2.6.2.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				Reserved									Reserved	Reserved	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved		Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	READADDROPT	PREFETCHEN	BUFFERABLEEN	CACHABLEEN	Reserved		APAREN
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

10.2.6.2.5.3 Fields

Field	Function
31-28 —	Reserved.
27-19 —	Reserved.
18 —	Reserved.
17 —	Reserved.
16 —	Reserved.
15 —	Reserved.
14-13 —	Reserved.
12 —	Reserved.
11 —	Reserved.
10 —	Reserved.
9 —	Reserved.
8 —	Reserved.
7 —	Reserved.
6 READADDROPT	AHB Read Address option bit. This option bit is intend to remove AHB burst start address alignment limitation. When FlexSPI controller is used for FPGA application, there may be requirement that FlexSPI fetch exactly the byte number as AHB burst. In this case, FPGA device should be designed as non-wordaddressable and this option bit should be set 0. 0b - There is AHB read burst start address alignment limitation when flash is accessed in parallel mode or flash is wordaddressable. 1b - There is no AHB read burst start address alignment limitation. FlexSPI will fetch more data than AHB burst required to meet the alignment requirement.
5 PREFETCHEN	AHB Read Prefetch Enable. When AHB read prefetch is enabled, FlexSPI will fetch more flash read data than current AHB burst needed so that the read latency for next AHB read access will be reduced.
4	Enable AHB bus bufferable write access support. This field affects the last beat of AHB write access, refer for more details about AHB bufferable write.

Table continues on the next page...

Field	Function
BUFFERABLEEN	0b - Disabled. For all AHB write access (no matter bufferable or non-bufferable), FlexSPI will return AHB Bus ready after all data is transmitted to External device and AHB command finished. 1b - Enabled. For AHB bufferable write access, FlexSPI will return AHB Bus ready when the AHB command is granted by arbitrator and will not wait for AHB command finished.
3 CACHABLEEN	Enable AHB bus cachable read access support. 0b - Disabled. When there is AHB bus cachable read access, FlexSPI will not check whether it hit AHB TX Buffer. 1b - Enabled. When there is AHB bus cachable read access, FlexSPI will check whether it hit AHB TX Buffer first.
2-1 —	Reserved.
0 APAREN	Parallel mode enabled for AHB triggered Command (both read and write) . 0b - Flash will be accessed in Individual mode. 1b - Flash will be accessed in Parallel mode.

10.2.6.2.6 Interrupt Enable Register (INTEN)

10.2.6.2.6.1 Offset

Register	Offset
INTEN	10h

10.2.6.2.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved		SEQTIMEOUTEN	AHBBUSTIMEOUTEN	SCKSTOPBYWREN	SCKSTOPBYRDEN	DATALEARNFAILEN	IPTXWEEN	IPRXWAEN	AHBCMDERREN	IPCMDERREN	AHBCMDGEEN	IPCMDGEEN	IPCMDDONEEN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.2.6.2.6.3 Fields

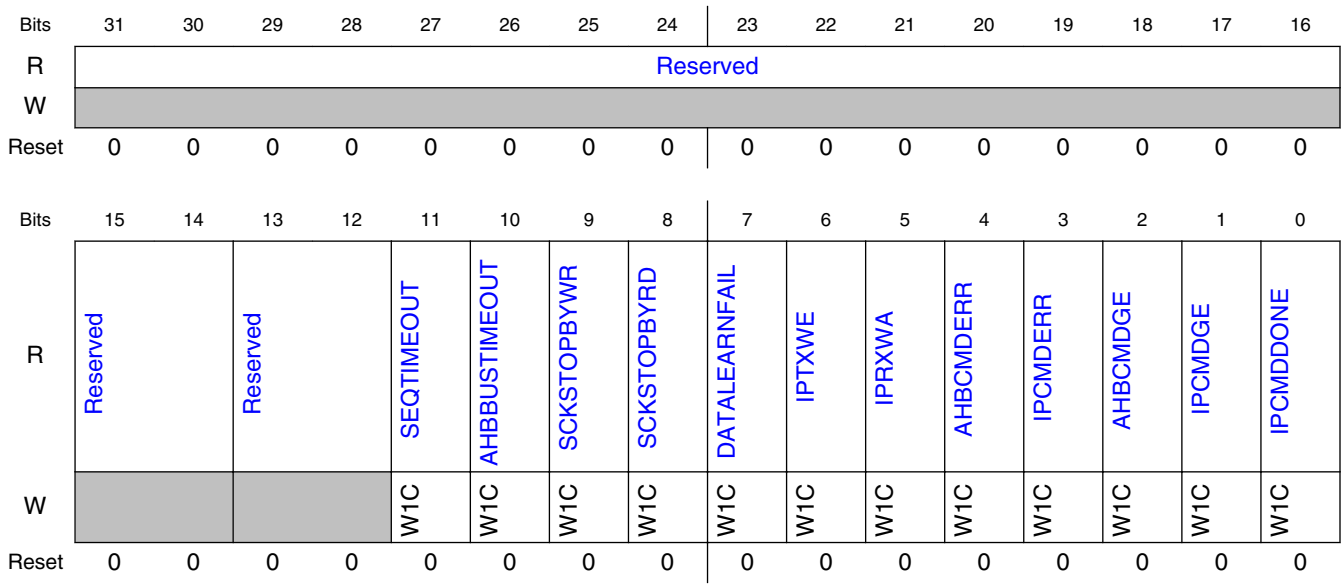
Field	Function
31-16 —	Reserved.
15-14 —	Reserved.
13-12 —	Reserved.
11 SEQTIMEOUTEN	Sequence execution timeout interrupt enable.Refer Interrupts chapter for more details.
10 AHBBUSTIMEOUTEN	AHB Bus timeout interrupt.Refer Interrupts chapter for more details.
9 SCKSTOPBYWREN	SCLK is stopped during command sequence because Async TX FIFO empty interrupt enable.
8 SCKSTOPBYRDEN	SCLK is stopped during command sequence because Async RX FIFO full interrupt enable.
7 DATALEARNFAILEN	Data Learning failed interrupt enable.
6 IPTXWEEN	IP TX FIFO WaterMark empty interrupt enable. IP TX FIFO has no less empty space than WaterMark level interrupt enable.
5 IPRXWAEN	IP RX FIFO WaterMark available interrupt enable. IP RX FIFO has no less valid data than WaterMark level interrupt enable.
4 AHBCMDERREN	AHB triggered Command Sequences Error Detected interrupt enable.
3 IPCMDERREN	IP triggered Command Sequences Error Detected interrupt enable.
2 AHBCMDGEEN	AHB triggered Command Sequences Grant Timeout interrupt enable.
1 IPCMDGEEN	IP triggered Command Sequences Grant Timeout interrupt enable.
0 IPCMDDONEEN	IP triggered Command Sequences Execution finished interrupt enable.

10.2.6.2.7 Interrupt Register (INTR)

10.2.6.2.7.1 Offset

Register	Offset
INTR	14h

10.2.6.2.7.2 Diagram



10.2.6.2.7.3 Fields

Field	Function
31-16 —	Reserved.
15-14 —	Reserved.
13-12 —	Reserved.
11 SEQTIMEOUT	Sequence execution timeout interrupt.
10 AHBBUSTIMEOUT	AHB Bus timeout interrupt.Refer Interrupts chapter for more details.
9 SCKSTOPBYWR	SCLK is stopped during command sequence because Async TX FIFO empty interrupt.
8	SCLK is stopped during command sequence because Async RX FIFO full interrupt.

Table continues on the next page...

Field	Function
SCKSTOPBYRD	
7 DATALEARNFAIL	Data Learning failed interrupt.
6 IPTXWE	IP TX FIFO watermark empty interrupt. IP TX FIFO has no less empty space than WaterMark level interrupt.
5 IPRXWA	IP RX FIFO watermark available interrupt. IP RX FIFO has no less valid data than WaterMark level interrupt.
4 AHBCMDERR	AHB triggered Command Sequences Error Detected interrupt. When an error detected for AHB command, this command will be ignored and not executed at all.
3 IPCMDERR	IP triggered Command Sequences Error Detected interrupt. When an error detected for IP command, this command will be ignored and not executed at all.
2 AHBCMDGE	AHB triggered Command Sequences Grant Timeout interrupt.
1 IPCMDGE	IP triggered Command Sequences Grant Timeout interrupt.
0 IPCMDDONE	IP triggered Command Sequences Execution finished interrupt. This interrupt is also generated when there is IPCMDGE or IPCMDERR interrupt generated.

10.2.6.2.8 LUT Key Register (LUTKEY)

10.2.6.2.8.1 Offset

Register	Offset
LUTKEY	18h

10.2.6.2.8.2 Function

The LUT Key Register contains the key to lock and unlock LUT. Refer to Look Up Table for details.

10.2.6.2.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	KEY															
Reset	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	KEY															
Reset	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0

10.2.6.2.8.4 Fields

Field	Function
31-0 KEY	The Key to lock or unlock LUT. The key is 0x5AF05AF0. Read value is always 0x5AF05AF0.

10.2.6.2.9 LUT Control Register (LUTCR)

10.2.6.2.9.1 Offset

Register	Offset
LUTCR	1Ch

10.2.6.2.9.2 Function

The LUT control register is used along with LUTKEY register to lock or unlock LUT. This register has to be written immediately after writing 0x5AF05AF0 to LUTKEY register for the lock or unlock operation to be successful. Refer Look Up Table for details on locking/unlocking LUT. Setting both the LOCK and UNLOCK bits as "00" or "11" is not allowed.

10.2.6.2.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														UNLOCK	LOCK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

10.2.6.2.9.4 Fields

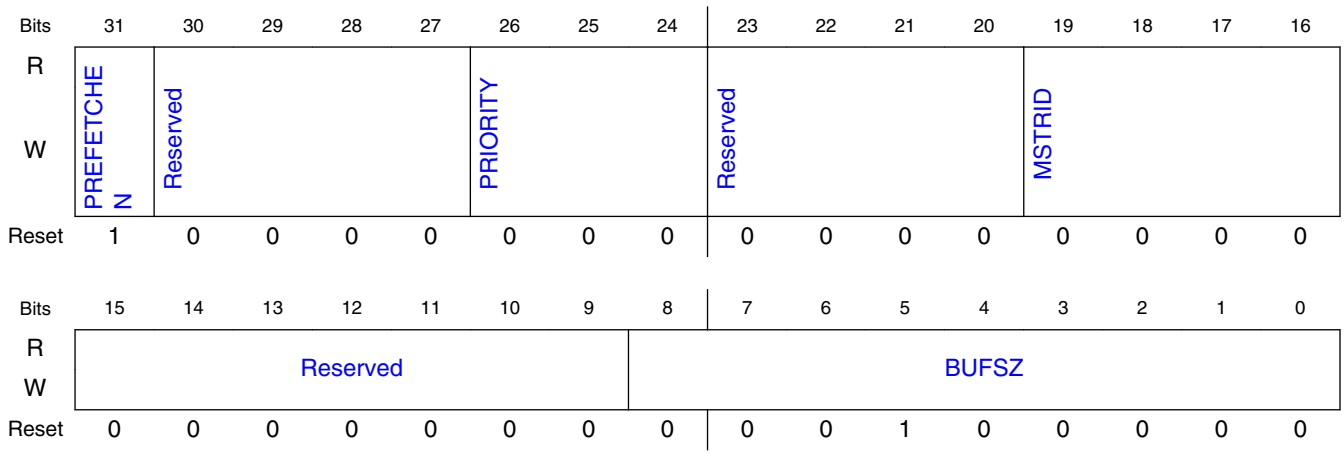
Field	Function
31-2 —	Reserved.
1 UNLOCK	Unlock LUT
0 LOCK	Lock LUT

10.2.6.2.10 AHB RX Buffer 0 Control Register 0 (AHBRXBUF0CR0)

10.2.6.2.10.1 Offset

Register	Offset
AHBRXBUF0CR0	20h

10.2.6.2.10.2 Diagram



10.2.6.2.10.3 Fields

Field	Function
31 PREFETCHEN	AHB Read Prefetch Enable for current AHB RX Buffer corresponding Master. The prefetch feature is disabled when AHBCR[PREFETCHEN] is set 0. This field allows prefetch disable/enable separately for each master.
30-27 —	Reserved.
26-24 PRIORITY	This priority for AHB Master Read which this AHB RX Buffer is assigned. Please refer to Command Abort and Suspend for more details.
23-20 —	Reserved.
19-16 MSTRID	This AHB RX Buffer is assigned according to AHB Master with ID (MSTR_ID). Please refer to AHB RX Buffer Management for AHB RX Buffer allocation.
15-9 —	Reserved.
8-0 BUFSZ	AHB RX Buffer Size in 64 bits. Please refer to AHB RX Buffer Management for more details.

10.2.6.2.11 AHB RX Buffer 1 Control Register 0 (AHBRXBUF1CR0)

10.2.6.2.11.1 Offset

Register	Offset
AHBRXBUF1CR0	24h

10.2.6.2.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PREFETCHEN	Reserved				PRIORITY			Reserved				MSTRID			
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								BUFSZ							
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

10.2.6.2.11.3 Fields

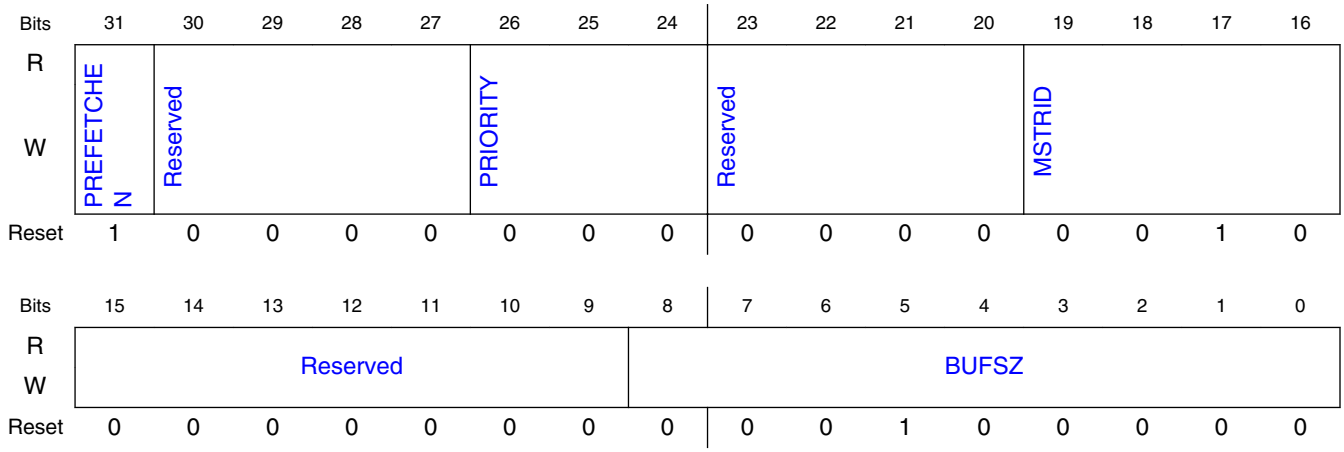
Field	Function
31 PREFETCHEN	AHB Read Prefetch Enable for current AHB RX Buffer corresponding Master. The prefetch feature is disabled when AHBCR[PREFETCHEN] is set 0. This field allows prefetch disable/enable separately for each master.
30-27 —	Reserved.
26-24 PRIORITY	This priority for AHB Master Read which this AHB RX Buffer is assigned. Please refer to Command Abort and Suspend for more details.
23-20 —	Reserved.
19-16 MSTRID	This AHB RX Buffer is assigned according to AHB Master with ID (MSTR_ID). Please refer to AHB RX Buffer Management for AHB RX Buffer allocation.
15-9 —	Reserved.
8-0 BUFSZ	AHB RX Buffer Size in 64 bits. Please refer to AHB RX Buffer Management for more details.

10.2.6.2.12 AHB RX Buffer 2 Control Register 0 (AHBRXBUF2CR0)

10.2.6.2.12.1 Offset

Register	Offset
AHBRXBUF2CR0	28h

10.2.6.2.12.2 Diagram



10.2.6.2.12.3 Fields

Field	Function
31 PREFETCHEN	AHB Read Prefetch Enable for current AHB RX Buffer corresponding Master. The prefetch feature is disabled when AHBCR[PREFETCHEN] is set 0. This field allows prefetch disable/enable separately for each master.
30-27 —	Reserved.
26-24 PRIORITY	This priority for AHB Master Read which this AHB RX Buffer is assigned. Please refer to Command Abort and Suspend for more details.
23-20 —	Reserved.
19-16 MSTRID	This AHB RX Buffer is assigned according to AHB Master with ID (MSTR_ID). Please refer to AHB RX Buffer Management for AHB RX Buffer allocation.
15-9 —	Reserved.
8-0 BUFSZ	AHB RX Buffer Size in 64 bits. Please refer to AHB RX Buffer Management for more details.

10.2.6.2.13 AHB RX Buffer 3 Control Register 0 (AHBRXBUF3CR0)

10.2.6.2.13.1 Offset

Register	Offset
AHBRXBUF3CR0	2Ch

10.2.6.2.13.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PREFETCHEN	Reserved				PRIORITY			Reserved				MSTRID			
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								BUFSZ							
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

10.2.6.2.13.3 Fields

Field	Function
31 PREFETCHEN	AHB Read Prefetch Enable for current AHB RX Buffer corresponding Master. The prefetch feature is disabled when AHBCR[PREFETCHEN] is set 0. This field allows prefetch disable/enable separately for each master.
30-27 —	Reserved.
26-24 PRIORITY	This priority for AHB Master Read which this AHB RX Buffer is assigned. Please refer to Command Abort and Suspend for more details.
23-20 —	Reserved.
19-16 MSTRID	This AHB RX Buffer is assigned according to AHB Master with ID (MSTR_ID). Please refer to AHB RX Buffer Management for AHB RX Buffer allocation.
15-9 —	Reserved.
8-0 BUFSZ	AHB RX Buffer Size in 64 bits. Please refer to AHB RX Buffer Management for more details.

10.2.6.2.14 AHB RX Buffer 4 Control Register 0 (AHBRXBUF4CR0)

10.2.6.2.14.1 Offset

Register	Offset
AHBRXBUF4CR0	30h

10.2.6.2.14.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PREFETCHEN	Reserved				PRIORITY			Reserved				MSTRID			
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								BUFSZ							
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

10.2.6.2.14.3 Fields

Field	Function
31 PREFETCHEN	AHB Read Prefetch Enable for current AHB RX Buffer corresponding Master. The prefetch feature is disabled when AHBCR[PREFETCHEN] is set 0. This field allows prefetch disable/enable separately for each master.
30-27 —	Reserved.
26-24 PRIORITY	This priority for AHB Master Read which this AHB RX Buffer is assigned. Please refer to Command Abort and Suspend for more details.
23-20 —	Reserved.
19-16 MSTRID	This AHB RX Buffer is assigned according to AHB Master with ID (MSTR_ID). Please refer to AHB RX Buffer Management for AHB RX Buffer allocation.
15-9 —	Reserved.
8-0 BUFSZ	AHB RX Buffer Size in 64 bits. Please refer to AHB RX Buffer Management for more details.

10.2.6.2.15 AHB RX Buffer 5 Control Register 0 (AHBRXBUF5CR0)

10.2.6.2.15.1 Offset

Register	Offset
AHBRXBUF5CR0	34h

10.2.6.2.15.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PREFETCHEN	Reserved				PRIORITY			Reserved				MSTRID			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								BUFSZ							
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

10.2.6.2.15.3 Fields

Field	Function
31 PREFETCHEN	AHB Read Prefetch Enable for current AHB RX Buffer corresponding Master. The prefetch feature is disabled when AHBCR[PREFETCHEN] is set 0. This field allows prefetch disable/enable separately for each master.
30-27 —	Reserved.
26-24 PRIORITY	This priority for AHB Master Read which this AHB RX Buffer is assigned. Please refer to Command Abort and Suspend for more details.
23-20 —	Reserved.
19-16 MSTRID	This AHB RX Buffer is assigned according to AHB Master with ID (MSTR_ID). Please refer to AHB RX Buffer Management for AHB RX Buffer allocation.
15-9 —	Reserved.
8-0 BUFSZ	AHB RX Buffer Size in 64 bits. Please refer to AHB RX Buffer Management for more details.

10.2.6.2.16 AHB RX Buffer 6 Control Register 0 (AHBRXBUF6CR0)

10.2.6.2.16.1 Offset

Register	Offset
AHBRXBUF6CR0	38h

10.2.6.2.16.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PREFETCHEN	Reserved				PRIORITY			Reserved				MSTRID			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								BUFSZ							
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

10.2.6.2.16.3 Fields

Field	Function
31 PREFETCHEN	AHB Read Prefetch Enable for current AHB RX Buffer corresponding Master. The prefetch feature is disabled when AHBCR[PREFETCHEN] is set 0. This field allows prefetch disable/enable separately for each master.
30-27 —	Reserved.
26-24 PRIORITY	This priority for AHB Master Read which this AHB RX Buffer is assigned. Please refer to Command Abort and Suspend for more details.
23-20 —	Reserved.
19-16 MSTRID	This AHB RX Buffer is assigned according to AHB Master with ID (MSTR_ID). Please refer to AHB RX Buffer Management for AHB RX Buffer allocation.
15-9 —	Reserved.

Table continues on the next page...

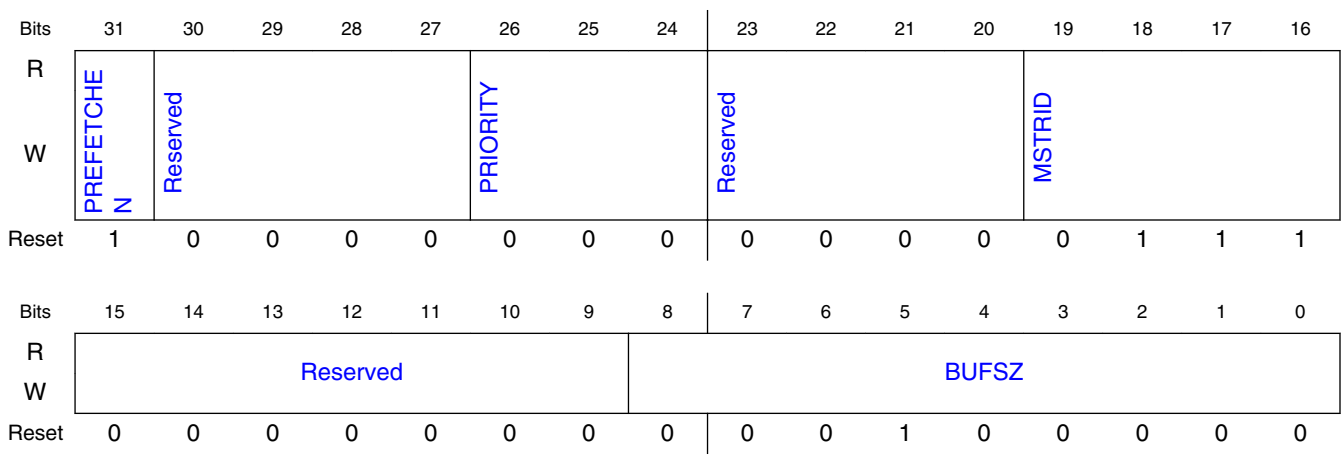
Field	Function
8-0 BUFSZ	AHB RX Buffer Size in 64 bits. Please refer to AHB RX Buffer Management for more details.

10.2.6.2.17 AHB RX Buffer 7 Control Register 0 (AHBRXBUF7CR0)

10.2.6.2.17.1 Offset

Register	Offset
AHBRXBUF7CR0	3Ch

10.2.6.2.17.2 Diagram



10.2.6.2.17.3 Fields

Field	Function
31 PREFETCHEN	AHB Read Prefetch Enable for current AHB RX Buffer corresponding Master. The prefetch feature is disabled when AHBCR[PREFETCHEN] is set 0. This field allows prefetch disable/enable separately for each master.
30-27 —	Reserved.
26-24 PRIORITY	This priority for AHB Master Read which this AHB RX Buffer is assigned. Please refer to Command Abort and Suspend for more details.
23-20 —	Reserved.

Table continues on the next page...

Field	Function
19-16 MSTRID	This AHB RX Buffer is assigned according to AHB Master with ID (MSTR_ID). Please refer to AHB RX Buffer Management for AHB RX Buffer allocation.
15-9 —	Reserved.
8-0 BUFSZ	AHB RX Buffer Size in 64 bits. Please refer to AHB RX Buffer Management for more details.

10.2.6.2.18 Flash Control Register 0 (FLSHA1CR0 - FLSHB2CR0)

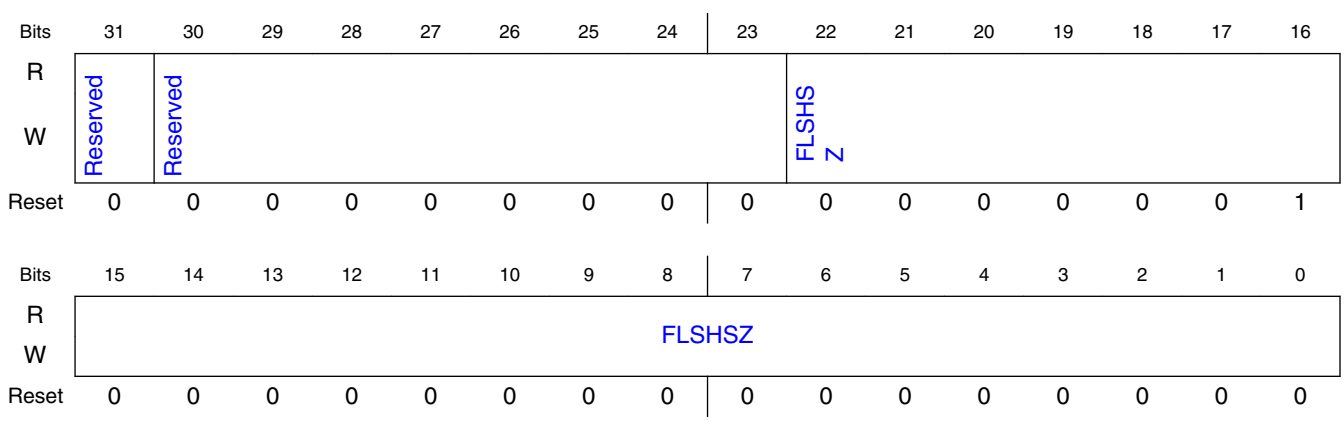
10.2.6.2.18.1 Offset

Register	Offset
FLSHA1CR0	60h
FLSHA2CR0	64h
FLSHB1CR0	68h
FLSHB2CR0	6Ch

10.2.6.2.18.2 Function

The Flash control register 0 contains Flash size setting. FlexSPI determines which device is accessed with this register setting (Chip Selection).

10.2.6.2.18.3 Diagram



10.2.6.2.18.4 Fields

Field	Function
31 —	Reserved.
30-23 —	Reserved.
22-0 FLSHSZ	Flash Size in KByte. The max flash size supported for each device is 0 GB. When FLSHSZ setting value is greater than 0x400000, the device flash size would be taken as 0 GB. The max total flash size supported (for all 4 devices) is also 0 GB. If the total flash size is larger than 0 GB, only 0 GB address space is accessible.

10.2.6.2.19 Flash Control Register 1 (FLSHA1CR1 - FLSHB2CR1)

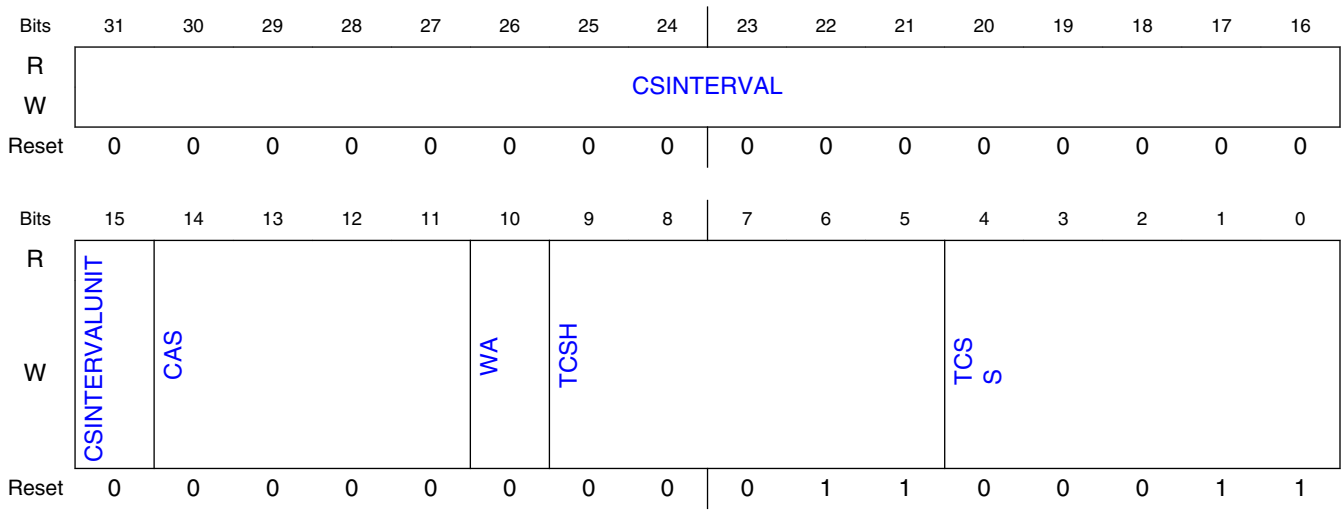
10.2.6.2.19.1 Offset

Register	Offset
FLSHA1CR1	70h
FLSHA2CR1	74h
FLSHB1CR1	78h
FLSHB2CR1	7Ch

10.2.6.2.19.2 Function

The Flash control register 1 contains the setting for FlexSPI to meet Flash device specific timings and Flash internal address space.

10.2.6.2.19.3 Diagram



10.2.6.2.19.4 Fields

Field	Function
31-16 CSINTERVAL	This field is used to set the minimum interval between flash device Chip selection deassertion and flash device Chip selection assertion. If external flash has a limitation on the interval between command sequences, this field should be set accordingly. If there is no limitation, set this field with value 0x0. When CSINTERVALUNIT is 0x0, the chip selection invalid interval is: CSINTERVAL * 1 serial clock cycle; When CSINTERVALUNIT is 0x1, the chip selection invalid interval is: CSINTERVAL * 256 serial clock cycle. **NOTE:** The chip selection interval is 2 cycle at least even if CSINTERVAL is less than 2.
15 CSINTERVALUNIT	CS interval unit 0b - The CS interval unit is 1 serial clock cycle 1b - The CS interval unit is 256 serial clock cycle
14-11 CAS	Column Address Size. When external flash has separate address field for row address and column address, this field should be set to flash column address bit width. FlexSPI will automatically split flash mapped address to Row address and Column address according to CAS field and WA field setting. This bit should be set to 0x0 when external Flash don't support column address. FlexSPI will transmit all flash address bits as Row address. For flash address mapping, please refer to Flash memory map for more detail.
10 WA	Word Addressable. This bit should be set when external Flash is word addressable. If Flash is word addressable, it should be access in terms of 16 bits. At this time, FlexSPI will not transmit Flash address bit 0 to external Flash. For flash address mapping, please refer to Flash memory map for more detail.
9-5 TCSH	Serial Flash CS Hold time. This field is used to meet flash TCSH timing requirement. Serial flash CS Hold time promised by FlexSPI is: TCSH in serial root clock cycles (for both SDR and DDR mode). Please refer to FlexSPI Input Timing for more detail.
4-0 TCSS	Serial Flash CS setup time.

Field	Function
	This field is used to meet flash TCSS timing requirement. Serial flash CS Setup time promised by FlexSPI is: (TCSS + 1/2) serial root clock cycles (for both SDR and DDR mode). Please refer to FlexSPI Input Timing for more detail.

10.2.6.2.20 Flash Control Register 2 (FLSHA1CR2 - FLSHB2CR2)

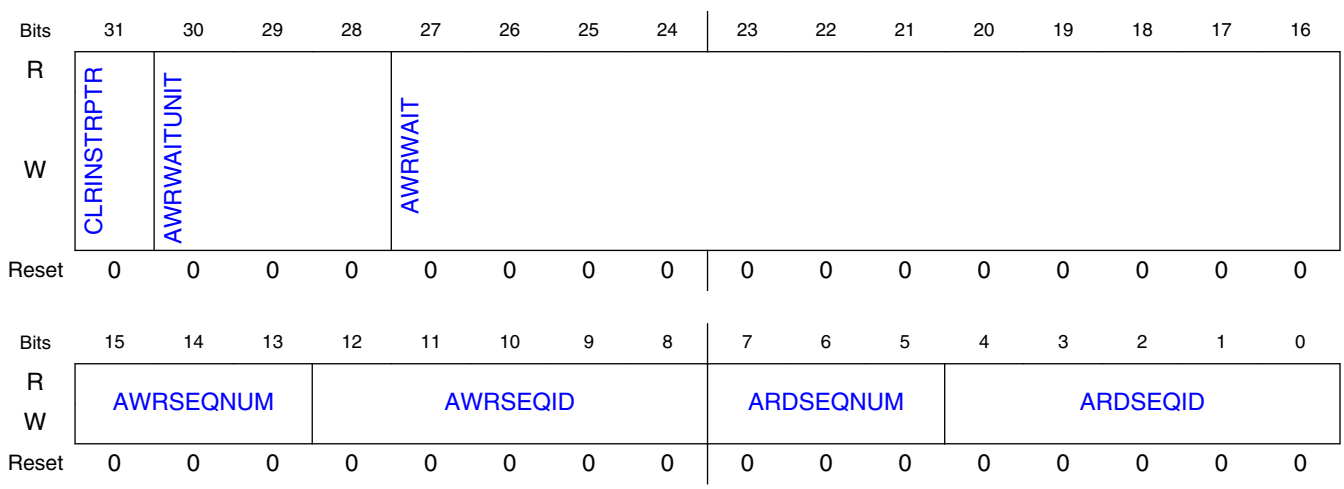
10.2.6.2.20.1 Offset

Register	Offset
FLSHA1CR2	80h
FLSHA2CR2	84h
FLSHB1CR2	88h
FLSHB2CR2	8Ch

10.2.6.2.20.2 Function

Flash Control Register 2 contains setting field for AHB Bus access configuration. If the 4 external device are in different types, AHB read/write command may use different command sequences and AHB bus ready wait time may be also different.

10.2.6.2.20.3 Diagram



10.2.6.2.20.4 Fields

Field	Function
31 CLRINSTRPTR	Clear the instruction pointer which is internally saved pointer by JMP_ON_CS. Refer Programmable Sequence Engine for details. This field is used for AHB Read access to external Flash supporting XIP (Execute-In-Place) mode.
30-28 AWRWAITUNIT	AWRWAIT unit 000b - The AWRWAIT unit is 2 ahb clock cycle 001b - The AWRWAIT unit is 8 ahb clock cycle 010b - The AWRWAIT unit is 32 ahb clock cycle 011b - The AWRWAIT unit is 128 ahb clock cycle 100b - The AWRWAIT unit is 512 ahb clock cycle 101b - The AWRWAIT unit is 2048 ahb clock cycle 110b - The AWRWAIT unit is 8192 ahb clock cycle 111b - The AWRWAIT unit is 32768 ahb clock cycle
27-16 AWRWAIT	For certain devices (such as FPGA), it need some time to write data into internal memory after the command sequences finished on FlexSPI interface. If another Read command sequence comes before previous programming finished internally, the read data may be wrong. This field is used to hold AHB Bus ready for AHB write access to wait the programming finished in external device. Then there will be no AHB read command triggered before the programming finished in external device. The Wait cycle between AHB triggered command sequences finished on FlexSPI interface and AHB return Bus ready: AWRWAIT * AWRWAITUNIT
15-13 AWRSEQNUM	Sequence Number for AHB Write triggered Command. For certain flash devices (E.g. HyperFlash/HyperRam/Serial NAND flash), a Flash programming access is done by several command sequences. AHB write Command will trigger (AWRSEQNUM+1) command sequences to external flash every time. FlexSPI executes the sequences in LUT incrementally. **NOTE:** • Software should make sure the last sequence index never exceed LUT sequence numbers: AWRSEQID+AWRSEQNUM < 32 • Software need to make sure field AWRSEQNUM and LUT is configured correctly according to external device spec. FlexSPI don't check the sequence and just execute them one by one.
12-8 AWRSEQID	Sequence Index for AHB Write triggered Command.
7-5 ARDSEQNUM	Sequence Number for AHB Read triggered Command in LUT. For certain flash devices (E.g. HyperFlash/HyperRam/Serial NAND flash), a Flash reading access is done by several command sequences. AHB read Command will trigger (ARDSEQNUM+1) command sequences to external flash every time. FlexSPI executes the sequences in LUT incrementally. **NOTE:** • Software should make sure the last sequence index never exceed LUT sequence numbers: ARDSEQID+ARDSEQNUM <= 32 • Software need to make sure field ARDSEQNUM and LUT is configured correctly according to external device spec. FlexSPI don't check the sequence and just execute them one by one.
4-0 ARDSEQID	Sequence Index for AHB Read triggered Command in LUT.

10.2.6.2.21 Flash Control Register 4 (FLSHCR4)

10.2.6.2.21.1 Offset

Register	Offset
FLSHCR4	94h

10.2.6.2.21.2 Function

The flash control register 4 provide the configuration for all external devices.

10.2.6.2.21.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										Reserved	Reserved	WMENB	WMENA	Reserved	WMOPT1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.2.6.2.21.4 Fields

Field	Function
31-6 —	Reserved.
5 —	Reserved.
4 —	Reserved.
3 WMENB	Write mask enable bit for flash device on port B. When write mask function is needed for memory device on port B, this bit must be set. 0b - Write mask is disabled, DQS(RWDS) pin will be un-driven when writing to external device. 1b - Write mask is enabled, DQS(RWDS) pin will be driven by FlexSPI as write mask output when writing to external device.
2 WMENA	Write mask enable bit for flash device on port A. When write mask function is needed for memory device on port A, this bit must be set. 0b - Write mask is disabled, DQS(RWDS) pin will be un-driven when writing to external device. 1b - Write mask is enabled, DQS(RWDS) pin will be driven by FlexSPI as write mask output when writing to external device.
1	Reserved.

Table continues on the next page...

Field	Function
—	
0 WMOPT1	Write mask option bit 1. This option bit could be used to remove AHB write burst start address alignment limitation. 0b - DQS pin will be used as Write Mask when writing to external device. There is no limitation on AHB write burst start address alignment when flash is accessed in individual mode. 1b - DQS pin will not be used as Write Mask when writing to external device. There is limitation on AHB write burst start address alignment when flash is accessed in individual mode.

10.2.6.2.22 IP Control Register 0 (IPCR0)

10.2.6.2.22.1 Offset

Register	Offset
IPCR0	A0h

10.2.6.2.22.2 Function

The IP control registers provide all the configuration required for IP commands. This register provides the flash device's start address, instead of SoC address, to be accessed for IP command. FlexSPI will determine the chip select automatically according to this start address.

NOTE

- It's not allowed to issue IP command crossing Flash device boundaries. Otherwise there will be IPCMDERR interrupt generated.
- This register should be set before IP command triggered.
- This register setting should not be changed while an IP command is in progress.

10.2.6.2.22.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SFAR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SFAR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.2.6.2.22.4 Fields

Field	Function
31-0 SFAR	Serial Flash Address for IP command.

10.2.6.2.23 IP Control Register 1 (IPCR1)

10.2.6.2.23.1 Offset

Register	Offset
IPCR1	A4h

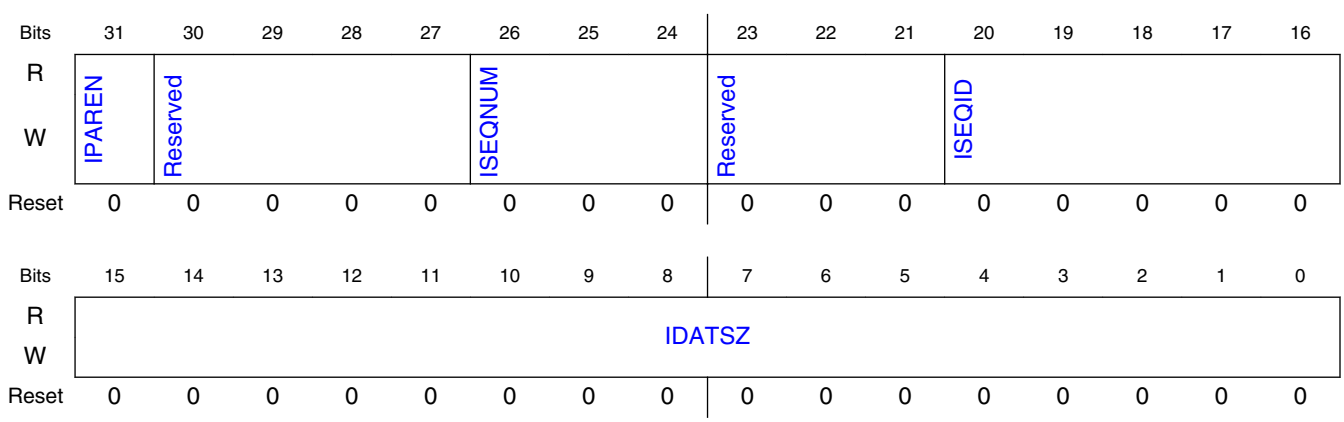
10.2.6.2.23.2 Function

The IP control registers provide all the configuration required for IP command. This register provides the flash read/program data size, sequence index in LUT, sequence number and individual/parallel mode setting for IP command.

NOTE

- This register should be before IP command triggered.
- This register setting should not be changed before IP command finished.

10.2.6.2.23.3 Diagram



10.2.6.2.23.4 Fields

Field	Function
31 IPAREN	Parallel mode Enabled for IP command. 0b - Flash will be accessed in Individual mode. 1b - Flash will be accessed in Parallel mode.
30-27 —	Reserved.
26-24 ISEQNUM	Sequence Number for IP command: ISEQNUM+1.
23-21 —	Reserved.
20-16 ISEQID	Sequence Index in LUT for IP command.
15-0 IDATSZ	Flash Read/Program Data Size (in Bytes) for IP command.

10.2.6.2.24 IP Command Register (IPCMD)

10.2.6.2.24.1 Offset

Register	Offset
IPCMD	B0h

10.2.6.2.24.2 Function

This register is used to trigger a IP command to access external flash device. IP command will be executed on FlexSPI interface after granted by arbitrator.

10.2.6.2.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															TRG
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.2.6.2.24.4 Fields

Field	Function
31-1 —	Reserved.
0 TRG	Setting this bit will trigger an IP Command. **NOTE:** • It's not allowed to trigger another IP command before previous IP command is finished on FlexSPI interface. Software need to poll register bit INTR_IP_CMD_DONE or wait for this interrupt in order to wait for IP command finished.

10.2.6.2.25 Data Learn Pattern Register (DLPR)

10.2.6.2.25.1 Offset

Register	Offset
DLPR	B4h

10.2.6.2.25.2 Function

This register provides the pattern to be used during Data Learning in FlexSPI.

10.2.6.2.25.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DLP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DLP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.2.6.2.25.4 Fields

Field	Function
31-0 DLP	Data Learning Pattern. The data learning pattern bit number is determined by operand in LEARN_SDR/LEARN_DDR Instruction Code. Data learning pattern bit number will be never more than 32 bits. If the operand in Instruction code

Field	Function
	is more than 32, 32 bits pattern will be used. If data learning pattern bit number is less than 32 bits, the lower pattern bits will be used. For more details, refer to Data Learning Feature.

10.2.6.2.26 IP RX FIFO Control Register (IPRXFCR)

10.2.6.2.26.1 Offset

Register	Offset
IPRXFCR	B8h

10.2.6.2.26.2 Function

This register provides the configuration fields for IP RX FIFO management.

10.2.6.2.26.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXWMRK						RXDMAEN	CLRIPRXF
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.2.6.2.26.4 Fields

Field	Function
31-8 —	Reserved.
7-2 RXWMRK	Watermark level is (RXWMRK+1)*64 Bits. Interrupt register bit IPRXWA is set when filling level in IP RX FIFO is no less than Watermark level by FlexSPI. There will be a DMA request when the filling level is no less than Watermark level and DMA read is enabled (register bit RXDMAEN is set). There will be an IPRXWA (IP RX FIFO watermark available) interrupt generated when the filling level is no less than Watermark level and IPRXWA interrupt is enabled (register bit INTEN_IPRXWA is set).

Table continues on the next page...

Field	Function
	NOTE: After write-1-clear to INTR[IPRXWA], read address should be rolled back to start address (memory mapped).If IP RX FIFO is read by IP bus, the read address to IP RX FIFO should roll back to RFDR0;If IP RX FIFO is read by AHB bus, the read address to IP RX FIFO should roll back to ARDF_BASE.
1 RXDMAEN	IP RX FIFO reading by DMA enabled. 0b - IP RX FIFO would be read by processor. 1b - IP RX FIFO would be read by DMA.
0 CLRIPRXF	Clear all valid data entries in IP RX FIFO. The read/write pointers in IP RX FIFO will be reset.

10.2.6.2.27 IP TX FIFO Control Register (IPTXFCR)

10.2.6.2.27.1 Offset

Register	Offset
IPTXFCR	BCh

10.2.6.2.27.2 Function

This register provides the configuration fields for IP TX FIFO management.

10.2.6.2.27.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							TXWMRK							TXDMAEN	CLRIPTXF
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.2.6.2.27.4 Fields

Field	Function
31-9	Reserved.

Table continues on the next page...

Field	Function
—	
8-2 TXWMRK	Watermark level is (TXWMRK+1)*64 Bits. Interrupt register bit IPTXWE is set when empty level in IP TX FIFO is no less than Watermark level by FlexSPI. There will be a DMA request when empty level is no less than Watermark level and DMA filling is enable (register bit TXDMAEN is set). There will be an IPTXWE (IP TX FIFO Watermark Empty) interrupt generated when empty level is no less than Watermark level and IPTXWE interrupt is enable (register bit INTEN_IPTXWE is set). **NOTE:** • The watermark level should be no more than the write window. • The watermark level should be no more than IP TX FIFO size. • The write address to IP RX FIFO should roll back to the start address of write window after pushing IP TX FIFO by writing-one-clear to INTR[IPTXWE].
1 TXDMAEN	IP TX FIFO filling by DMA enabled. 0b - IP TX FIFO would be filled by processor. 1b - IP TX FIFO would be filled by DMA.
0 CLRIPTXF	Clear all valid data entries in IP TX FIFO. The read/write pointers in IP TX FIFO will be reset.

10.2.6.2.28 DLL Control Register 0 (DLLACR - DLLBCR)

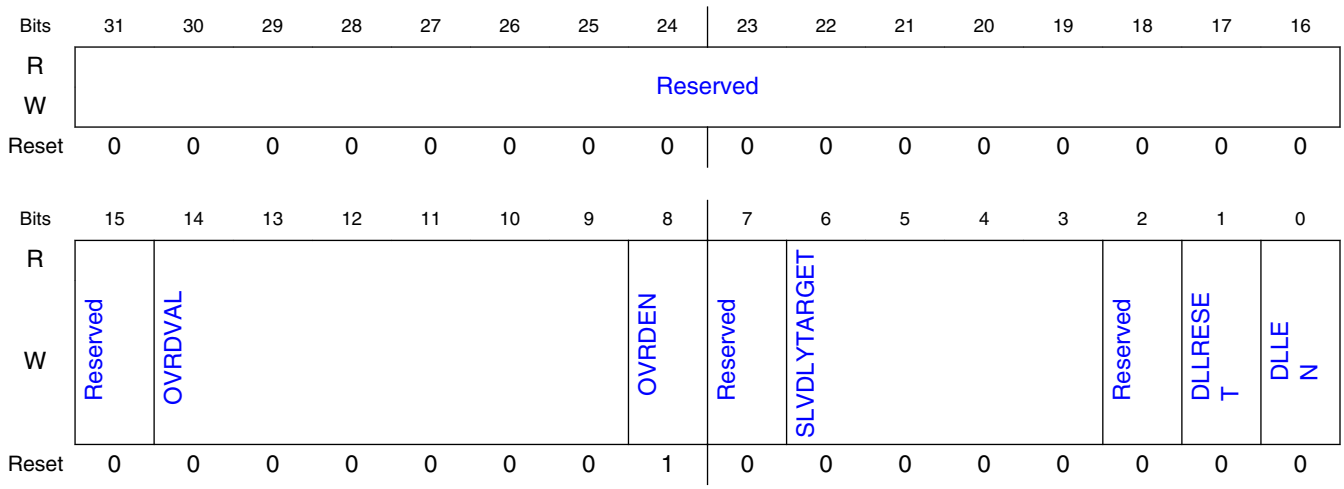
10.2.6.2.28.1 Offset

Register	Offset
DLLACR	C0h
DLLBCR	C4h

10.2.6.2.28.2 Function

This register provides the configuration fields for Flash A/B sample clock DLL.

10.2.6.2.28.3 Diagram



10.2.6.2.28.4 Fields

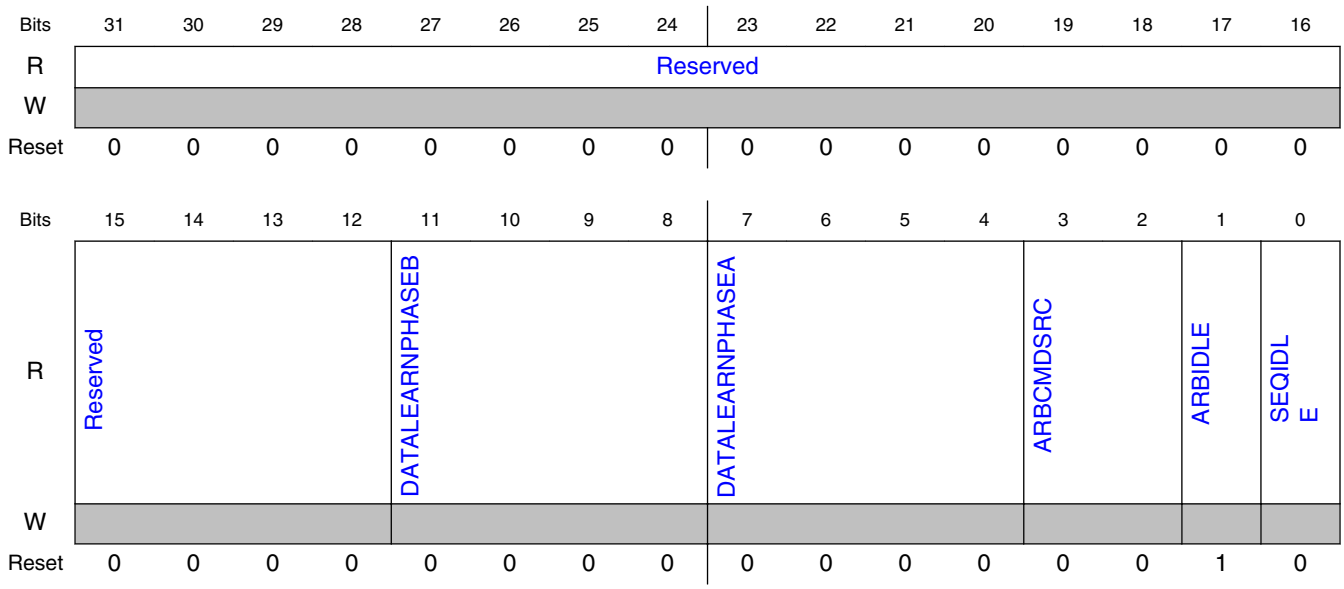
Field	Function
31-15 —	Reserved.
14-9 OVRDVAL	Slave clock delay line delay cell number selection override value. When OVRDEN is set 0x1, the delay cell number in DLL is OVRDVAL+1.
8 OVRDEN	Slave clock delay line delay cell number selection override enable.
7 —	Reserved.
6-3 SLVDLYTARGET	The delay target for slave delay line is: ((SLVDLYTARGET+1) * 1/32 * clock cycle of reference clock (serial root clock).
2 —	Reserved.
1 DLLRESET	Software could force a reset on DLL by setting this field to 0x1. This will cause the DLL to lose lock and re-calibrate to detect an ref_clock half period phase shift. The reset action is edge triggered, so software need to clear this bit after set this bit (no delay limitation).
0 DLLEN	DLL calibration enable. When this bit is cleared, DLL calibration is disabled and the delay cell number in slave delay line is always 1. Please note that SLV delay line is overridden when OVRDEN bit is set and this bit field setting is ignored.

10.2.6.2.29 Status Register 0 (STS0)

10.2.6.2.29.1 Offset

Register	Offset
STS0	E0h

10.2.6.2.29.2 Diagram



10.2.6.2.29.3 Fields

Field	Function
31-12 —	Reserved.
11-8 DATALEARNPHASEB	Indicate the sampling clock phase selection on Port B after Data Learning. This is similar as DATALEARNPHASEA field. The sampling clock phase is chosen separately for Port A and Port B.
7-4 DATALEARNPHASEA	Indicate the sampling clock phase selection on Port A after Data Learning. There are 16 clock phases for sampling clock which is generated by delay cell line. When data learning feature is not enabled, the default clock phase 0 will be used to sample Flash read data. When data learning feature is enabled and LEARN_SDR/LEARN_DDR instruction executed correctly on FlexSPI Interface, FlexSPI will determine the correct clock phase to sample Flash read data. Refer Data Learning Feature for more details.
3-2 ARBCMDSRC	This status field indicates the trigger source of current command sequence granted by arbitrator. This field value is meaningless when ARB_CTL is not busy (STS0[ARBIDLE]=0x1). 00b - Triggered by AHB read command (triggered by AHB read). 01b - Triggered by AHB write command (triggered by AHB Write). 10b - Triggered by IP command (triggered by setting register bit IPCMD.TRG). 11b - Triggered by suspended command (resumed).

Table continues on the next page...

Field	Function
1 ARBIDLE	This status bit indicates the state machine in ARB_CTL is busy and there is command sequence granted by arbitrator and not finished yet on FlexSPI interface. When ARB_CTL state (ARBIDLE=0x1) is idle, there will be no transaction on FlexSPI interface also (SEQIDLE=0x1). So this bit should be polled to wait for FlexSPI controller become idle instead of SEQIDLE.
0 SEQIDLE	This status bit indicates the state machine in SEQ_CTL is idle and there is command sequence executing on FlexSPI interface.

10.2.6.2.30 Status Register 1 (STS1)

10.2.6.2.30.1 Offset

Register	Offset
STS1	E4h

10.2.6.2.30.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				IPCMDERRCODE				Reserved			IPCMDERRID				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				AHBCMDERRCODE				Reserved			AHBCMDERRID				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.2.6.2.30.3 Fields

Field	Function
31-28 —	Reserved.

Table continues on the next page...

Field	Function
27-24 IPCMDERRCODE	Indicates the Error Code when IP command Error detected. This field will be cleared when INTR[IPCMDERR] is write-1-clear(w1c). 0000b - No error. 0010b - IP command with JMP_ON_CS instruction used in the sequence. 0011b - There is unknown instruction opcode in the sequence. 0100b - Instruction DUMMY_SDR/DUMMY_RWDS_SDR used in DDR sequence. 0101b - Instruction DUMMY_DDR/DUMMY_RWDS_DDR used in SDR sequence. 0110b - Flash access start address exceed the whole flash address range (A1/A2/B1/B2). 1110b - Sequence execution timeout. 1111b - Flash boundary crossed.
23-21 —	Reserved.
20-16 IPCMDERRID	Indicates the sequence Index when IP command error detected. This field will be cleared when INTR[IPCMDERR] is write-1-clear(w1c).
15-12 —	Reserved.
11-8 AHBCMDERRCODE	Indicates the Error Code when AHB command Error detected. This field will be cleared when INTR[AHBCMDERR] is write-1-clear(w1c). 0000b - No error. 0010b - AHB Write command with JMP_ON_CS instruction used in the sequence. 0011b - There is unknown instruction opcode in the sequence. 0100b - Instruction DUMMY_SDR/DUMMY_RWDS_SDR used in DDR sequence. 0101b - Instruction DUMMY_DDR/DUMMY_RWDS_DDR used in SDR sequence. 1110b - Sequence execution timeout.
7-5 —	Reserved.
4-0 AHBCMDERRID	Indicates the sequence index when an AHB command error is detected. This field will be cleared when INTR[AHBCMDERR] is write-1-clear(w1c).

10.2.6.2.31 Status Register 2 (STS2)

10.2.6.2.31.1 Offset

Register	Offset
STS2	E8h

10.2.6.2.31.2 Function

This register indicates the status of Flash A and B sample clock DLLs.

10.2.6.2.31.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		BREFSEL						BSLVSEL						BREFLOCK	BSLVLOCK
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		AREFSEL						ASLVSEL						AREFLOCK	ASLVLOCK
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

10.2.6.2.31.4 Fields

Field	Function
31-30 —	Reserved.
29-24 BREFSEL	Flash B sample clock reference delay line delay cell number selection.
23-18 BSLVSEL	Flash B sample clock slave delay line delay cell number selection.
17 BREFLOCK	Flash B sample clock reference delay line locked.
16 BSLVLOCK	Flash B sample clock slave delay line locked.
15-14 —	Reserved.
13-8 AREFSEL	Flash A sample clock reference delay line delay cell number selection.
7-2 ASLVSEL	Flash A sample clock slave delay line delay cell number selection .
1 AREFLOCK	Flash A sample clock reference delay line locked.
0 ASLVLOCK	Flash A sample clock slave delay line locked.

10.2.6.2.32 AHB Suspend Status Register (AHBSPNDSTS)

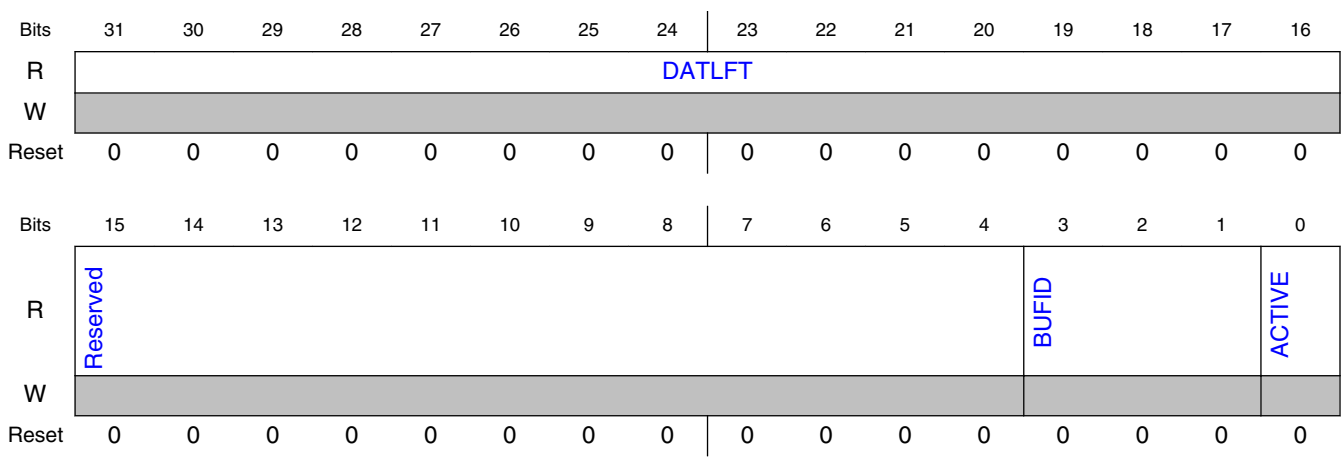
10.2.6.2.32.1 Offset

Register	Offset
AHBSPNDSTS	ECh

10.2.6.2.32.2 Function

Indicates the status of Suspended AHB Read Prefetch command sequence. When there is IP/AHB command triggered and arbitrator is processing an AHB Read sequence (prefetching more data not for current AHB burst), the prefetch sequence will be suspended and may be resumed when there is no transaction on FlexSPI any more. FlexSPI saves only one AHB prefetch sequence. When a new prefetch sequence is suspended with an active sequence suspended already, previous suspended sequence will be removed and never resumed. Refer Command Abort and Suspend for more details.

10.2.6.2.32.3 Diagram



10.2.6.2.32.4 Fields

Field	Function
31-16 DATLFT	Left Data size for suspended command sequence (in byte).
15-4 —	Reserved.

Table continues on the next page...

Field	Function
3-1 BUFID	AHB RX BUF ID for suspended command sequence.
0 ACTIVE	Indicates if an AHB read prefetch command sequence has been suspended.

10.2.6.2.33 IP RX FIFO Status Register (IPRXFSTS)

10.2.6.2.33.1 Offset

Register	Offset
IPRXFSTS	F0h

10.2.6.2.33.2 Function

This status register indicates the status of IP RX FIFO.

10.2.6.2.33.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RDCNTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								FILL							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.2.6.2.33.4 Fields

Field	Function
31-16 RDCNTR	Total Read Data Counter: RDCNTR * 64 Bits.
15-8 —	Reserved.
7-0 FILL	Fill level of IP RX FIFO. Valid Data entries in IP RX FIFO is: FILL * 64 Bits.

10.2.6.2.34 IP TX FIFO Status Register (IPTXFSTS)

10.2.6.2.34.1 Offset

Register	Offset
IPTXFSTS	F4h

10.2.6.2.34.2 Function

This status register indicates the status of IP TX FIFO.

10.2.6.2.34.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WRCNTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								FILL							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.2.6.2.34.4 Fields

Field	Function
31-16 WRCNTR	Total Write Data Counter: WRCNTR * 64 Bits.
15-8 —	Reserved.
7-0 FILL	Fill level of IP TX FIFO. Valid Data entries in IP TX FIFO is: FILL * 64 Bits.

10.2.6.2.35 IP RX FIFO Data Register a (RFDR0 - RFDR31)

10.2.6.2.35.1 Offset

For a = 0 to 31:

Register	Offset
RFDRa	100h + (a × 4h)

10.2.6.2.35.2 Function

These registers provide read access to IP RX FIFO by IPS bus. The read value is unknown for read access to invalid entries in IP RX FIFO.

10.2.6.2.35.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.2.6.2.35.4 Fields

Field	Function
31-0 RXDATA	RX Data

10.2.6.2.36 IP TX FIFO Data Register a (TFDR0 - TFDR31)

10.2.6.2.36.1 Offset

For a = 0 to 31:

Register	Offset
TFDRa	180h + (a × 4h)

10.2.6.2.36.2 Function

These registers provide write access to IP TX FIFO by IPS bus.

10.2.6.2.36.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	TXDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	TXDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.2.6.2.36.4 Fields

Field	Function
31-0 TXDATA	TX Data

10.2.6.2.37 LUT a (LUT0 - LUT127)

10.2.6.2.37.1 Offset

For a = 0 to 127:

Register	Offset
LUTa	200h + (a × 4h)

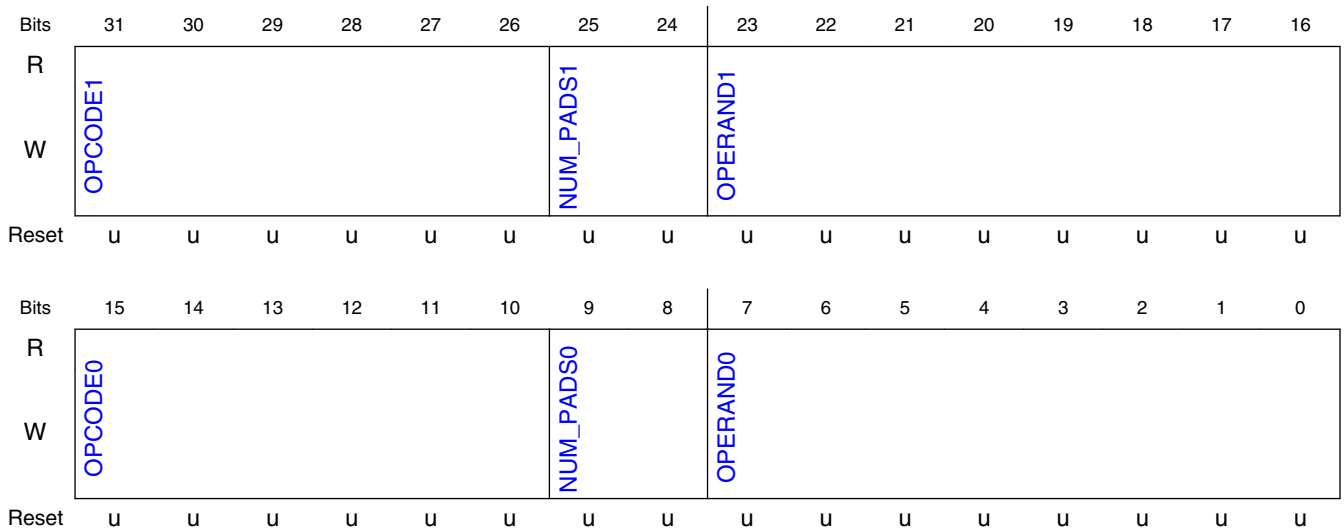
10.2.6.2.37.2 Function

The LUT is a look-up-table for command sequences. Software should set the sequence index before triggering an IP command or AHB command. FlexSPI will fetch the command sequence from LUT when IP/AHB command triggered. There are 32 command sequences in LUT. Refer Look Up Table for more details.

NOTE

LUT is implemented as memory, so the reset value is unknown.

10.2.6.2.37.3 Diagram



10.2.6.2.37.4 Fields

Field	Function
31-26 OPCODE1	OPCODE1
25-24 NUM_PADS1	NUM_PADS1
23-16 OPERAND1	OPERAND1
15-10 OPCODE0	OPCODE
9-8 NUM_PADS0	NUM_PADS0
7-0 OPERAND0	OPERAND0

10.2.7 AHB Memory Map definition

This section describes FlexSPI module AHB memory map in detail.

10.2.7.1 AHB Memory Map for Serial Flash memory access

Following address range is mapped for AHB read/write access to serial flash memory: 0x0 - 0x10000000.

AHB Bus feature supported for Serial Flash memory reading:
- Cachable and Non-Cachable access
- Prefetch Enable/Disable
- Burst size: 8/16/32/64 bits
- All burst type: SINGLE/INCR/WRAP4/INCR4/WRAP8/INCR8/WRAP16/INCR16

AHB Bus feature for Serial Flash memory writing:
- Bufferable and Non-Bufferable access
- Burst size: 8/16/32/64 bits
- All burst type: SINGLE/INCR/WRAP4/INCR4/WRAP8/INCR8/WRAP16/INCR16

Refer Flash access by AHB Command for more details about AHB access to Serial Flash memory.

10.2.7.2 AHB Memory Map for IP RX FIFO read access

Following address range is mapped for AHB read access to IP RX FIFO: 0x34000000 - 0x34000200.

AHB Bus feature supported for IP RX FIFO reading:
- Burst size: 8/16/32/64 bits
- All burst type: SINGLE/INCR/WRAP4/INCR4/WRAP8/INCR8/WRAP16/INCR16

Refer Reading Data from IP RX FIFO for more details about IP RX FIFO reading.

10.2.7.3 AHB Memory Map for IP TX FIFO write access

Following address range is mapped for AHB write access to IP TX FIFO: 0x33008000 - 0x33008400.

AHB Bus feature supported for IP TX FIFO writing:
- Burst size: 8/16/32/64 bits
- All burst type: SINGLE/INCR/WRAP4/INCR4/WRAP8/INCR8/WRAP16/INCR16

Refer Filling Data to IP TX FIFO for more details about IP TX FIFO filling.

10.3 Ultra Secured Digital Host Controller (uSDHC)

10.3.1 Introduction

The uSDHC provides the interface between the host system and the SD/SDIO/MMC cards, as depicted in Figure 10-53. It acts as a bridge, passing host bus transactions to the SD/SDIO/MMC cards by sending commands and performing data accesses to/from the cards. It handles the SD/SDIO/MMC protocols at the transmission level. The following are brief descriptions of the cards supported by uSDHC:

The Multi Media Card (MMC) is a universal low-cost data storage and communication media designed to cover a wide array of applications including mobile video and gaming. Previous MMC cards were based on a 7-pin serial bus with a single data pin, while the new high speed MMC communication is based on an advanced 11-pin serial bus designed to operate in the low-voltage range.

The Secure Digital Card (SD) is an evolution of the old MMC technology. It is specifically designed to meet the security, capacity, performance, and environmental requirements inherent in newly emerging audio and video consumer electronic devices. The physical form factor, pin assignment, and data transfer protocol are forward compatible with the old MMC (with some additions).

Under the SD protocol, it can be categorized into memory card, I/O card, and combo card, which have both memory and I/O functions. The memory card invokes a copyright protection mechanism that complies with the security of the SDMI standard. The I/O card, which is also known as SDIO card, provides high-speed data I/O with low-power consumption for mobile electronic devices. For the sake of simplicity, the next figure does not show cards with reduced size or mini cards.

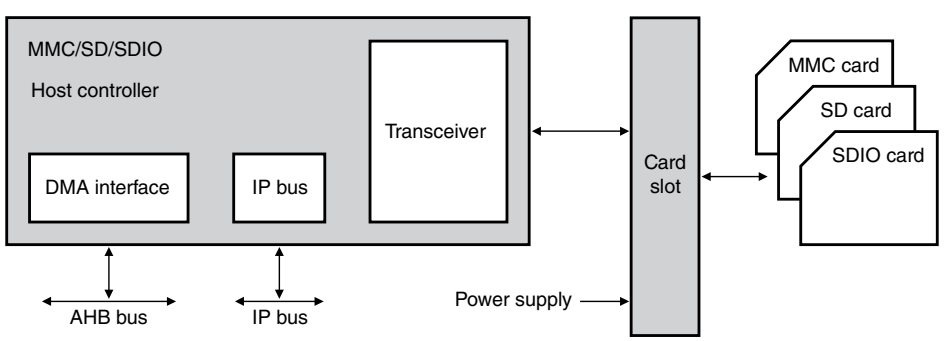


Figure 10-53. System connection of uSDHC

The following figure illustrates the block diagram of uSDHC.

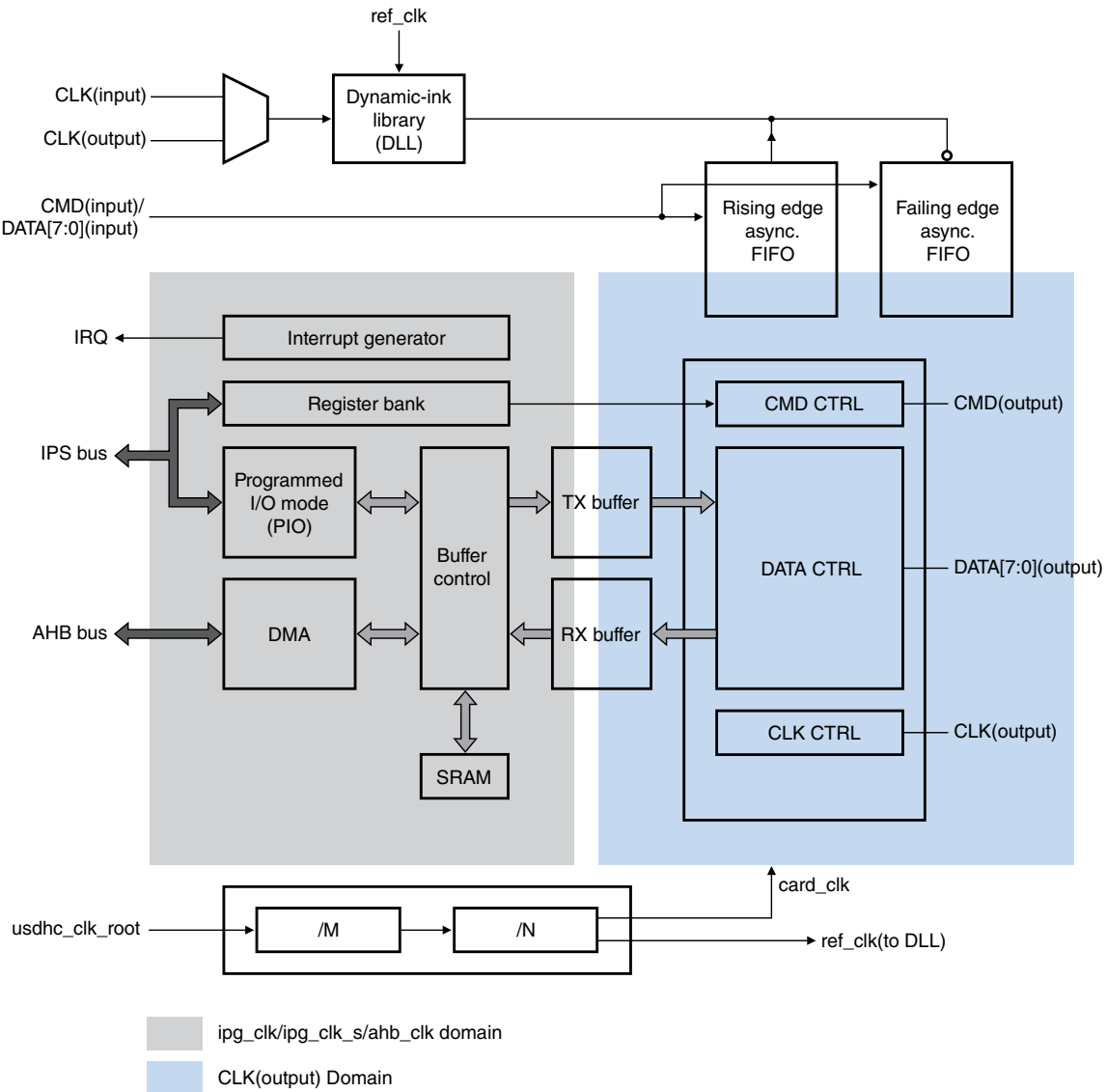


Figure 10-54. uSDHC block diagram

10.3.1.1 Features

The features of the uSDHC module include the following:

- Conforms to the SD Host Controller Standard Specification version 2.0/3.0
- Compatible with the MMC System Specification version 4.2/4.3/4.4/4.41/5.0/5.1
- Compatible with the SD Memory Card Specification version 3.0 and supports the Extended Capacity SD Memory Card
- Compatible with the SDIO Card Specification version 2.0/3.0
- Designed to work with SD Memory, miniSD Memory, SDIO, miniSDIO, SD Combo, MMC, MMC plus, and MMC RS cards
- Card bus clock frequency up to 208 MHz

- Supports 1-bit/4-bit SD and SDIO modes, and 1-bit/4-bit/8-bit MMC modes
  - Up to 832 Mbps of data transfer for SDIO cards using four parallel data lines in the Single Data Rate (SDR) mode
  - Up to 400 Mbps of data transfer for SDIO card using four parallel data lines in the Dual Data Rate (DDR) mode
  - Up to 832 Mbps of data transfer for SDXC cards using four parallel data lines in the Single Data Rate (SDR) mode
  - Up to 400 Mbps of data transfer for SDXC card using four parallel data lines in the Dual Data Rate (DDR) mode
  - Up to 416 Mbps of data transfer for MMC cards using eight parallel data lines in the Single Data Rate (SDR) mode
  - Up to 3200 Mbps of data transfer for MMC cards using eight parallel data lines in the Dual Data Rate (DDR) mode
- Supports a single block/multi-block read and write
- Supports block sizes of 1 ~ 4096 bytes
- Supports the write protection switch for write operations
- Supports both synchronous and asynchronous abort
- Supports pause during the data transfer at block gap
- Supports SDIO Read Wait and Suspend Resume operations
- Supports Auto CMD12 for multi-block transfer
- Host can initiate non-data transfer command while data transfer is in progress
- Allows cards to interrupt the host in 1-bit and 4-bit SDIO modes; also supports interrupt period
- Embodies two fully configurable 128x32-bit FIFO for read/write data
- Supports internaland external DMA capabilities
- Support voltage selection by configuring vendor-specific register bit
- Supports Advanced DMA to perform linked memory access
- support Command queue mechanism

10.3.1.2 Modes and operations

10.3.1.2.1 Data transfer modes

The uSDHC module can select the following modes for data transfer:

- SD 1-bit
- SD 4-bit
- MMC 1-bit
- MMC 4-bit
- MMC 8-bit
- Identification mode (up to 400 kHz)

- MMC full-speed mode (up to 26 MHz)
- MMC high-speed mode (up to 52 MHz)
- MMC HS400 mode (200 MHz both edges)
- MMC DDR mode (52 MHz both edges)
- SD/SDIO full-speed mode (up to 25 MHz)
- SD/SDIO high-speed mode (up to 50 MHz)
- SD/SDIO UHS-I mode(up to 208 MHz in SDR mode, up to 50 MHz in the DDR mode)

10.3.2 External signals

The following table describes the external signals of uSDHC:

Table 10-33. uSDHC external signals

Signal	Description	Direction
CLK	Clock for MMC/SD/SDIO card	O
CMD	CMD line connect to card	I/O
DATA7	DATA7 line in the 8-bit mode — Not used in other modes	I/O
DATA6	DATA6 line in the 8-bit mode — Not used in other modes	I/O
DATA5	DATA5 line in the 8-bit mode — Not used in other modes	I/O
DATA4	DATA4 line in the 8-bit mode — Not used in other modes	I/O
DATA3	DATA3 line in the 4/8-bit mode or configured as card detection pin. The bit may be configured as card detection pin in the 1-bit mode.	I/O
DATA2	DATA2 line or Read Wait in the 4-bit mode Read Wait in 1-bit mode	I/O
DATA1	DATA1 line in the 4/8-bit mode Also, used to detect interrupt in 1/4-bit mode	I/O
DATA0	DATA0 line in all the modes Also, used to detect busy state	I/O
CD_B	Card detection pin If not used (for the embedded memory), tie low to indicate that there is a card attached.	I
WP	Card write protect detect If not used (for the embedded memory), tie low to indicate that it is not write protected.	I

Table continues on the next page...

Table 10-33. uSDHC external signals (continued)

Signal	Description	Direction
LCTL	LED control used to drive an external LED active high, fully controlled by the driver Optional output	O
RESET_B	Card hardware reset signal, active low	O
VSELECT	IO power voltage selection signal	O
STROBE	Input clock for eMMC HS400 mode	I

10.3.2.1 Signals overview

uSDHC has 15 associated I/O signals.

- The CLK is an internally generated clock used to drive the MMC, SD, and SDIO cards.
- The CMD I/O is used to send commands and receive responses to and from the card. Eight data lines (DATA7~DATA0) are used to perform data transfers between the uSDHC module and the card.
- The CD_B and WP are card detection and write protection signals directly routed from the socket. These two signals are active low (0). A low on CD_B means that a card is inserted, and a high on WP means that the write protect switch is active.
- LCTL is an output signal used to drive an external LED to indicate that the SD interface is busy.
- RESET_B is an output signal used to reset the MMC card.
- VSELECT is an output signal used to change the voltage of the external power supplier.
- STROBE is input clock signal for eMMC HS400 mode.

CD_B, WP, LCTL, RESET_B, and VSELECT are all optional for system implementation. If uSDHC needs to support a 4-bit data transfer, DATA7~DATA4 can also be optional and tied to high. If the uSDHC does not support the HS400 mode, STROBE can also be optional and tied to low.

10.3.3 Functional description

The following sections provide a brief functional description of the major system blocks, including the data buffer, DMA AHB interface, register bank as well as IP Bus interface, dual-port memory wrapper, data/command controller, clock and reset manager, and clock generator.

10.3.3.1 Data buffer

The uSDHC module uses one configurable data buffer to transfer data between the system bus (IP bus or advanced high-performance bus (AHB) bus) and the SD card in an optimized manner, maximizing throughput between the two clock domains (IP peripheral clock and the master clock).

The buffer is used as a temporary storage for transferring data between the host system and the card. The watermark levels for read and write are both configurable and can range between 1 to 128 words. The burst lengths for read and write are also configurable and can range between 1 to 31 words. The next figure provides the uSDHC buffer scheme.

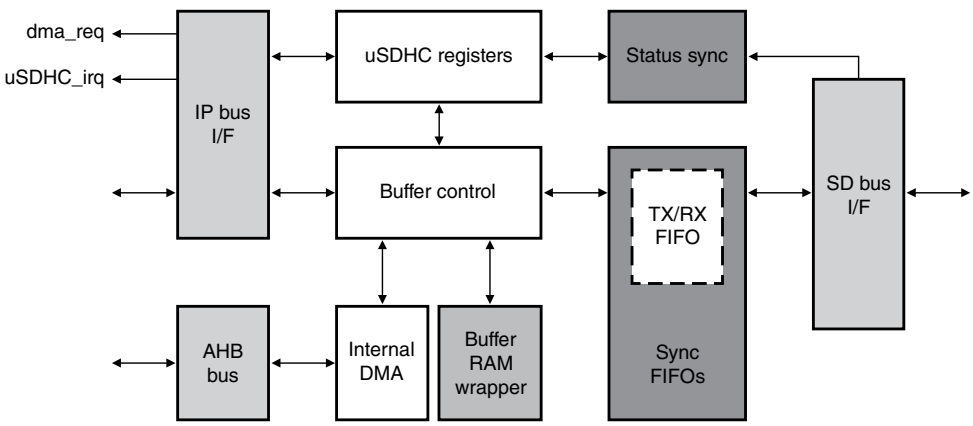


Figure 10-55. uSDHC buffer scheme

Here are 3 the two transfer modes to access the data buffer:

- CPU polling mode:
  - For a host-read operation, when the number of words received in the buffer meets or exceeds the RD_WML watermark value, by polling the BRR bit, the host driver can read the Buffer Data Port register to fetch the amount of words set in the RD_WML register from the buffer. The write operation is similar. For more information on the process of writing operation, see Write operation sequence.
- External DMA mode:
  - For a read operation, when there are more words received in the buffer than the amount set in the RD_WML register, a DMA request is sent out to inform the external DMA to fetch the data. The request will be immediately de-asserted when there is an access on the Buffer Data Port register. If the number of words in the buffer after the current burst meets or exceeds RD_WML value, the DMA request is asserted again. For instance, if there are twice as many words in the

buffer as there are in the RD_WML value, there are two successive DMA requests with only one cycle of de-assertion between. The write operation is similar. Note the accesses CPU polling mode and external DMA mode both use the IP bus, and if the external DMA is enabled, in both modes an external DMA request is sent when the buffer is ready.

- Internal DMA mode (includes simple and advanced DMA accesses):
  - The internal DMA access, either by simple or advanced DMA, is over the AHB bus. For internal DMA access mode, the external DMA request will never be sent out.

For a read operation, when there are more words in the buffer than the amount set in the RD_WML register, the internal DMA starts fetching data over the AHB bus. Except for INCR4 and INCR8, the burst type is always the INCR mode and the burst length depends on the shortest of the following factors:

- Burst length configured in the burst length field of the Watermark Level register
- Watermark level boundary
- Block size boundary
- Data boundary configured in the current descriptor (if the ADMA is active)
- 1 KB address boundary defined in the AHB protocol

The Write operation functions in a similar manner—sequential and contiguous access is necessary to ensure that the pointer address value is correct. Random or skipped access is not possible. The byte order, by reset, is little endian mode. The actual byte order is swapped inside the buffer, according to the endian mode configured by software (see the following figures). For a host write operation, the byte order is swapped after the data is fetched from the buffer and ready to send to the SD Bus. For a host read operation, the byte order is swapped before the data is stored in the buffer.

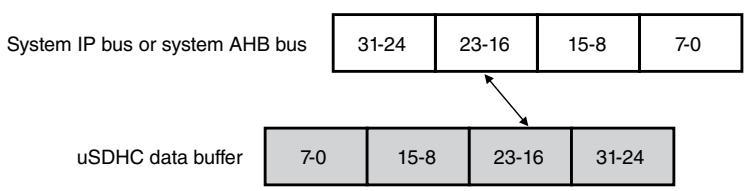


Figure 10-56. Data swap between system bus and uSDHC data buffer in the byte little endian mode

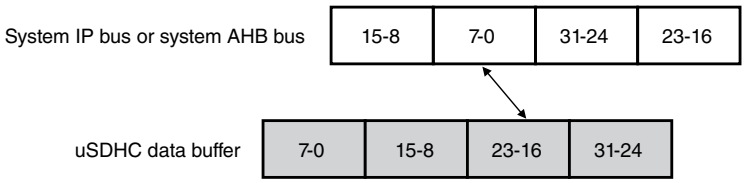


Figure 10-57. Data swap between system bus and uSDHC data buffer in half word big endian mode

10.3.3.1.1 Write operation sequence

There are 3 ways to write data into the buffer when the user transfers data to the card:

- External DMA through the uSDHC DMA request signal
- Processor core polling through the BWR bit in the Interrupt Status register (interrupt or polling)
- Internal DMA

When the internal DMA is not used, the DMAEN bit in the Transfer Type register is not set when the command is sent, uSDHC asserts a DMA request when the amount of buffer space exceeds the value set in the WR_WML register and is ready for receiving new data. At the same time, uSDHC sets the BWR bit. The buffer write ready interrupt is generated if it is enabled by software.

When internal DMA is used, uSDHC does not inform the system before all the required number of bytes are transferred (if no error is encountered). When an error occurs during the data transfer, uSDHC aborts the data transfer and abandons the current block. The host driver should read the contents of the DMA System Address register to obtain the starting address of the abandoned data block. If the current data transfer is in multi-block mode, uSDHC does not automatically send CMD12, even though the AC12EN bit in the Transfer Type register is set. The host driver sends CMD12 in this scenario and restarts the write operation from that address. It is recommended that a software reset for Data be applied before the transfer is restarted.

The uSDHC module does not start data transmission until the number of words set in the WR_WML register can be held in the buffer. If the buffer is empty and the host system does not write data in time, uSDHC stops the CLK to avoid the data buffer underrun situation.

10.3.3.1.2 Read operation sequence

There are 3 ways to read data from the buffer when the user transfers data to the card:

- External DMA through uSDHC DMA request signal

- Processor core polling through the BRR bit in the Interrupt Status register (interrupt or polling)
- Internal DMA

When internal DMA is not used (DMAEN bit in Transfer Type register is not set when the command is sent), uSDHC asserts a DMA request when the amount of data exceeds the value set in the RD_WML register, which is available and ready for system fetching data. At the same time, uSDHC sets the BRR bit. The buffer read ready interrupt is generated if it is enabled by the software.

When internal DMA is used, uSDHC does not inform the system before all the required number of bytes are transferred (if no error is encountered). When an error occurs during the data transfer, uSDHC aborts the data transfer and abandons the current block. The host driver should read the content of the DMA System Address register to get the starting address of the abandoned data block. If the current data transfer is in multi-block mode, uSDHC does not automatically send CMD12, even though the AC12EN bit in the Transfer Type register is set. The host driver sends CMD12 in this scenario and restarts the read operation from that address. It is recommended that a software reset for data be applied before the transfer is restarted.

For any write transfer mode, uSDHC does not start data transmission until the number of words set in the RD_WML register are in buffer. If the buffer is full and the host system does not read data in time, uSDHC stops the CLK to avoid the data buffer overrun situation.

10.3.3.1.3 Data buffer and block size

The user needs to know the buffer size for the buffer operation during a data transfer to utilize it in the most optimized way. In the uSDHC module, the only data buffer can hold up to 128 words (32-bit) and the watermark levels for write and read can be configured accordingly.

For both read and write, the watermark level can range between 1 to 128 words. For both read and write, the burst length can range between from 1 to 31 words. The host driver may configure the value according to the system situation and requirement.

During a multi-block data transfer, the block length can be set to any value between 1 and 4096 bytes, satisfying the requirements of the external card. The only restriction is from the external card, which can be limited in size or support of a partial block access (which is not the integer times of 512 bytes).

As uSDHC treats each block individually, for block sizes which are not multiples of four (not word-aligned), stuffed bytes are required at the end of each block. For example, if the block size is 7 bytes and there are 12 blocks to write, the system side must write twice

for each block. For each block, the ending byte is abandoned by uSDHC because it only sends 7 bytes to the card and picks data from the following system write, resulting in 24 beats of write access in total.

10.3.3.1.4 Dividing large data transfer

This SDIO command CMD53 definition limits the maximum data size of data transfers according to the following formula:

Max data size = Block size x Block count

The length of a multiple block transfer needs to be in block size units. If the total data length cannot be divided evenly into a multiple of the block size, then there are two ways to transfer the data. These two ways depend on the function and the card design. Option 1 is for the host driver to split the transaction. The remainder of the block size data is then transferred by using a single block command at the end. Option 2 is to add dummy data in the last block to fill the block size. For option 2, the card must manage the removal of the dummy data.

See the figure below for an example showing the division of large data transfers, assuming a kind of WLAN SDIO card that only supports a block size of up to 64 bytes. Although uSDHC supports a block size of up to 4096 bytes, the SDIO can only accept a block size of less than 64 bytes, so the data must be divided (see the example below).

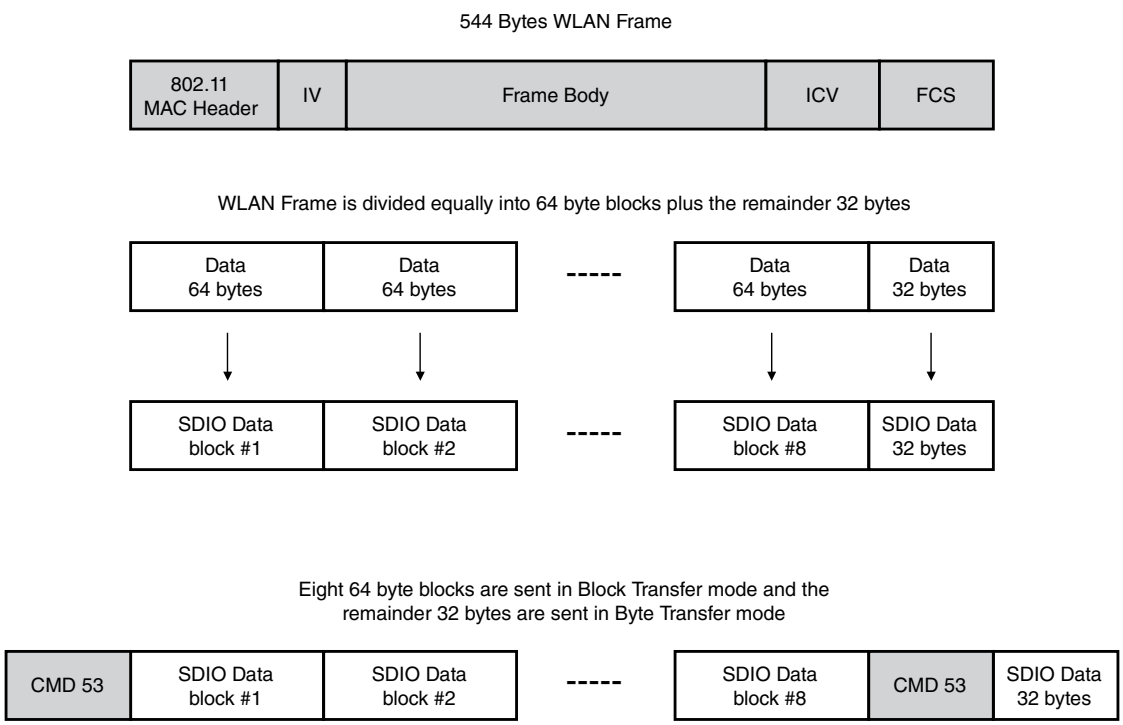


Figure 10-58. Example for dividing large data transfers

10.3.3.1.5 External DMA Request

When the internal DMA is not in use and external DMA is enabled, the Data Buffer will generate a DMA request to the system. During a write operation, when the number of WR_WML words can be held in the buffer free space, the signal uSDHC_dreq_b is asserted to 0, informing the Host System of a DMA write.

The BWR bit in the Interrupt Status register is also set, as long as the BWRSEN bit in the Interrupt Status Enable register is set. The DMA request is de-asserted after several accesses to the Data Port register are made while the buffer's free space can't meet the watermark condition (free space> write watermark level).

On read operation, when the number of RD_WML words are already in the buffer, the signal uSDHC_dreq_b is asserted to 0, informing the Host System for a DMA read. The BRR bit in the Interrupt Status register is also set, as long as the BRRSEN bit in the Interrupt Status Enable register is set. The DMA request is de-asserted after several accesses to the Data Port register are made while the buffer's data can't meet the watermark condition (the number of data in buffer > read watermark level).

If the DMA burst length can't change during a data transfer for an external DMA transfer, the watermark level (read or write) must be a divisor of the block size. If it is not, transferring the block may cause buffer under-run (read operation) or over-run (write operation). For example, if the block size is 512 bytes, the watermark level of read (or write) must be a power of two between 1 and 128. For processor core polling access there is no such issue, as the last access in the block transfer can be controlled by software. The watermark level can be any value, even larger than the block size (but no greater than 128 words) because the actual number of bytes transferred by the software can be controlled and does not exceed the block size in each transfer.

The uSDHC also supports non-word aligned block size, as long as the card supports that block size. In this case, the watermark level should be set as the number of words. For example, if the block size is 31 bytes, the watermark level can be set to any number of words. For this case, the BLKSIZE bits of the Block Attribute register will be set as 1fh. For the CPU polling access, the burst length can be 1 to 128 words, without restriction. This is because the software will transfer 8 words, and the uSDHC will also set the BWR or BRR bits when the remaining data does not violate data buffer. See DMA burst length for more details about the dynamic watermark level of the data buffer.

For the above example, even though 8 words are transferred via the Data Port register, the uSDHC will transfer only 31 bytes over the SD Bus, as required by the BLKSIZE bits. In this data transfer, with non-word aligned block size, the endian mode should be set cautiously or invalid data will be transferred to and from the card.

10.3.3.2 DMA AHB interface

The internal DMA implements a DMA engine and the AHB master. When the internal DMA is enabled, the uSDHC_dreq_b is not asserted during the transfer, but the BWR and BRR bits are set if the BWRSEN and BRRSEN bits have been set in the Interrupt Status Enable register.

See the figure below for an illustration of the DMA AHB interface block.

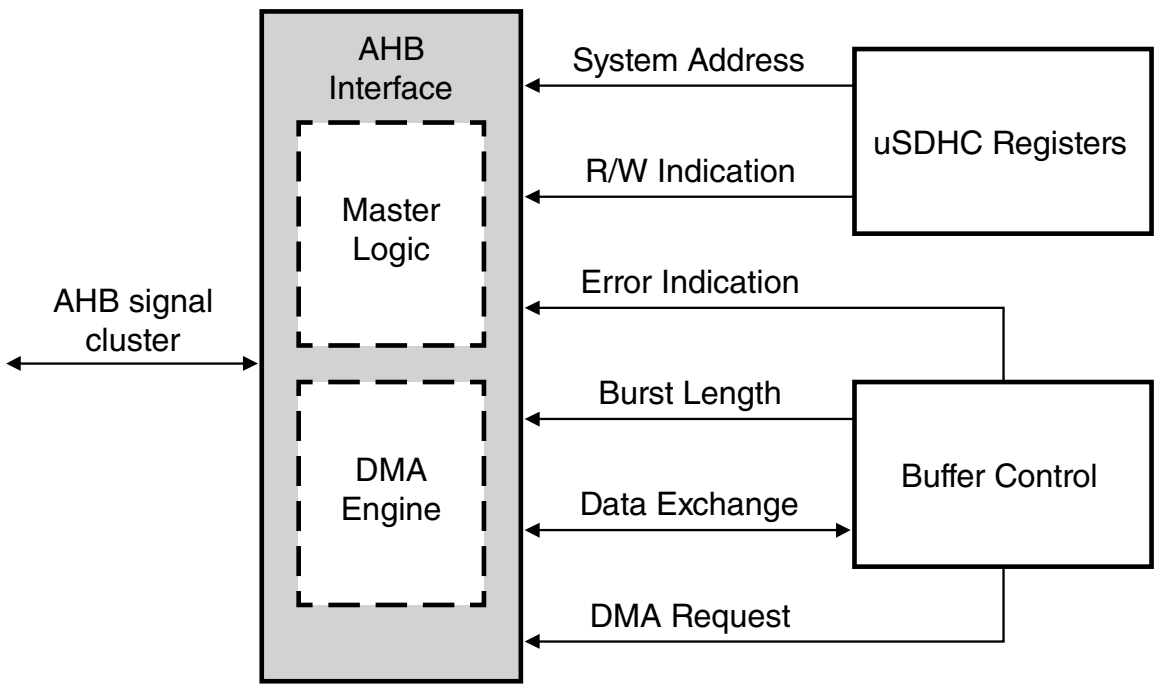


Figure 10-59. DMA AHB interface block

10.3.3.2.1 Internal DMA request

If the watermark level requirement is met in data transfer or if the last data of current block is ready in the data buffer, and the Internal DMA is enabled, the data buffer block sends a DMA request to AHB interface.Meanwhile, the external DMA request signal (uSDHC_dreq_b) is disabled.

The delay in response from the internal DMA engine depends on the system AHB bus loading and the priority assigned to uSDHC. The DMA engine does not respond to the request during its burst transfer, but is ready to serve as soon as the burst is over. The data buffer de-asserts the request if the data buffer space (for write) or bytes in data buffer is

smaller than the watermark level. Upon access to the buffer by internal DMA, the data buffer updates its internal buffer pointer, and when the watermark level is satisfied or the last data of the current block is ready in the data buffer, another DMA request is sent.

The data transfer is in the block unit, and the subsequent watermark level is always set as the remaining number of words. For instance, for a multi block data read with each block size of 31 bytes, and the burst length set to six words. After the first burst transfer, if there are more than two words in the buffer, which might contain some data of the next block), another DMA request is sent. This is because the remaining number of words to send for the current block is (31 - 6 * 4) / 4 = 2. The uSDHC module reads two words out of the buffer, with seven valid bytes and one stuffed byte.

10.3.3.2.2 DMA burst length

Just like a CPU polling access, the DMA burst length for the internal DMA engine can range between 1 to 16 words. The actual burst length for the DMA depends on the lesser of the configured burst length or the remaining words of the current block. See the example in Internal DMA request. After six words are read, the burst length is two words, then the next burst length is six words. This is because the next block starts, which is 31 bytes, more than six words. The host driver may take this variable burst length into account. It is also acceptable to configure the burst length as the divisor of the block size, so that each time the burst length is the same.

10.3.3.2.3 AHB master interface

It is possible that the internal AHB DMA engine could fail during the data transfer. Upon detection of an AHB bus error during DMA transfer, the DMA engine stops the transfer and goes to the idle state. At that point, the internal data buffer stops receiving incoming data and sending out data. The DMAE bit in the Interrupt Status register is generated to host CPU to report a bus error condition.

After the DMAE interrupt is received, the software sends a CMD12 to abort the current transfer and read the DS_ADDR bits of the DMA System Address register to get the starting address of the corrupted block. After the DMA error is fixed, the software should apply a data reset and restart the transfer from this address to recover the corrupted block. DMA operation resumes when the interrupt is serviced by the software.

10.3.3.2.4 ADMA engine

In the SD host controller standard, a new DMA transfer algorithm called the Advanced DMA (ADMA) is defined. For simple DMA, after the page boundary is reached, a DMA interrupt is generated and the new system address is programmed by the host driver.

The ADMA defines the programmable descriptor table in the system memory. The host driver can calculate the system address at the page boundary and program the descriptor table before executing ADMA. It reduces the frequency of interrupts to the host system. Therefore, higher speed DMA transfers could be realized because the host MCU intervention is not be needed during long DMA-based data transfers.

There are two types of ADMA in host controller: ADMA1 and ADMA2. ADMA1 can support data transfer of 4KB aligned data in system memory, and ADMA2 improves the restriction so that the data of any location and any size can be transferred in system memory. Their formats of Descriptor Table are different.

ADMA can recognize all kinds of descriptors defined in SD host controller Standard, and if the "End" flag is detected in the descriptor, ADMA stops after this descriptor is processed.

10.3.3.2.4.1 ADMA concept and descriptor format

ADMA1 includes the following descriptors:

- Valid/invalid descriptor
- Nop descriptor
- Set data length descriptor
- Set data address descriptor
- Link descriptor
- Interrupt flag and end flag in descriptor

ADMA2 includes the following descriptors:

- Valid/invalid descriptor
- Nop descriptor
- Rsv descriptor
- Set data length and address descriptor
- Link descriptor
- Interrupt flag and end flag in descriptor

ADMA starts read/write operation after it reaches the tran state, using the data length and data address analyzed from most recent descriptor(s).

For ADMA1, the valid data length descriptor is the last set type descriptor before the tran type descriptor. Every tran type triggers a transfer, and the transfer data length is extracted from the most recent set type descriptor. If there is no set type descriptor after the previous Trans descriptor, the data length is the value for previous transfer, or 0 if no set descriptor is ever met.

For ADMA2, the tran type descriptor contains both data length and transfer data address, so only a tran type descriptor can start a data transfer.

See the figure below for the format of the descriptor table for ADMA1.

Address/ Page Field		Address/ Page Field		Attribute Field					
31	12	11	6	5	4	3	2	1	0
Address or Data Length		000000		Act 2	Act 1	0	Int	End	Valid

Act 2	Act1	Symbol	Comment	31- 28	27- 12
0	0	Nop	No Operation	Don't Care	
0	1	Set	Set Data Length	0000	Data Length
1	0	Tran	Transfer Data	Data Address	
1	1	Link	Link Descriptor	Descriptor Address	

Valid	Valid = 1 indicates this line of descriptor is effective. If Valid = 0 generate ADMA Error Interrupt and stop ADMA.
End	End = 1 indicates current descriptor is the ending one.
Int	Int = 1 generates DMA Interrupt when this descriptor is processed.

Figure 10-60. Format of the ADMA1 descriptor table

See the figure below for the concept and access method of the descriptor table for ADMA1.

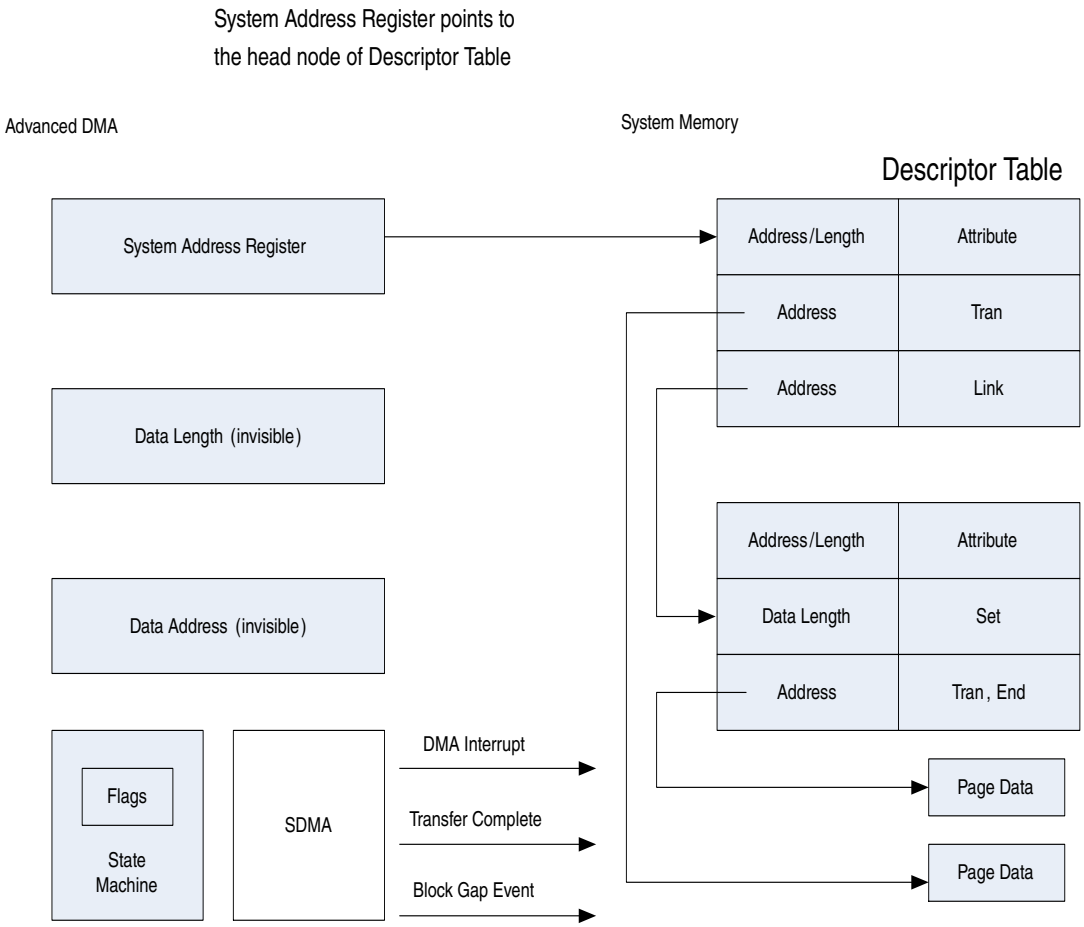


Figure 10-61. Concept and access method of the ADMA1 descriptor table

The figure below explains the ADMA2 format. ADMA2 deals with the lower 32-bit first, and then the higher 32-bit. If the 'Valid' flag of descriptor is 0, it ignores the higher 32-bit. The Address field should be set to word aligned (lower 2-bit is always set to 0). Data length is in byte unit.

Address Field		Length		Reserved		Attribute Field					
63	32	31	16	15	06	05	04	03	02	01	00
32-bit Address		16-bit length		0000000000		Act 2	Act 1	0	Int	End	Valid

Act 2	Act1	Symbol	Comment	Operation
0	0	Nop	No Operation	Don't Care
0	1	Rsv	Reserved	Same as Nop. Read this line and go to next one
1	0	Tran	Transfer Data	Transfer data with address and length set in this descriptor line
1	1	Link	Link Descriptor	Link to another descriptor

Valid	Valid = 1 indicates this line of descriptor is effective. If Valid = 0 generate ADMA Error Interrupt and stop ADMA.
End	End = 1 indicates current descriptor is the ending one.
Int	Int = 1 generates DMA Interrupt when this descriptor is processed.

Figure 10-62. Format of the ADMA2 descriptor table

See the figure below for the concept and access method of the descriptor table for ADMA2.

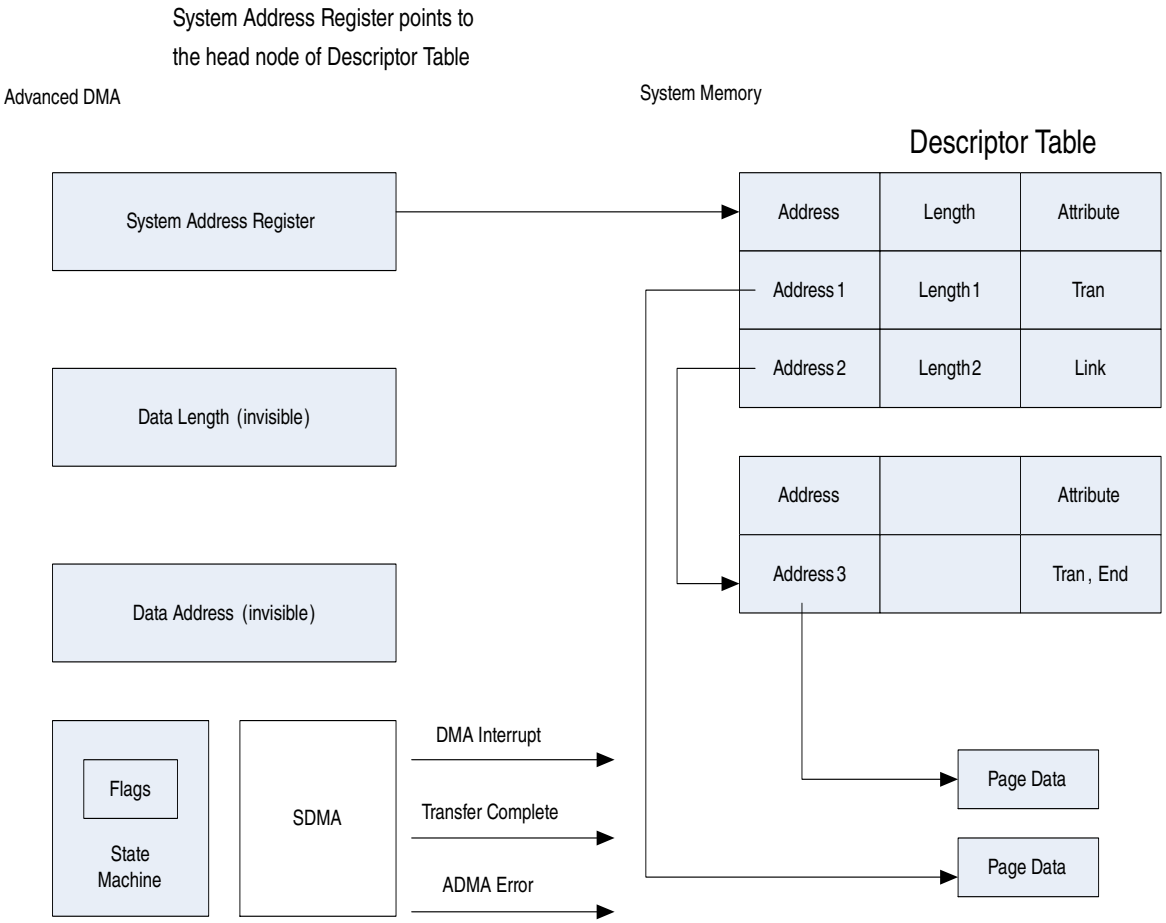


Figure 10-63. Concept and access method of the ADMA2 descriptor table

10.3.3.2.4.2 ADMA interrupt

If the interrupt flag descriptor is set, ADMA generates an interrupt according to various types of descriptors.

For ADMA1:

- Set type of descriptor: interrupt is generated when data length is set.
- Tran type descriptor: interrupt is generated when this transfer is complete.
- Link type of descriptor: interrupt is generated when new descriptor address is set.
- Nop type of descriptor: interrupt is generated just after this descriptor is fetched.

For ADMA2:

- Tran type of descriptor: interrupt is generated when this transfer is complete.
- Link type of descriptor: interrupt is generated when new descriptor address is set.
- Nop/Rsv type of descriptor: interrupt is generated just after this descriptor is fetched.

10.3.3.2.4.3 ADMA error

The ADMA stops whenever an error is encountered. These errors include:

- Fetching descriptor error
- AHB response error
- Data length mismatch error

An ADMA descriptor error is generated when it fails to detect a "valid" flag in the descriptor. If an ADMA descriptor error occurs, the interrupt is not generated even if the "Interrupt" flag of this descriptor is set.

When BLKCNTEN bit is set, data length set in buffer must be equal to the whole data length set in descriptor nodes, otherwise data length mismatch error is generated.

When BLKCNTEN bit is not set, then the whole data length set in descriptor is a multiple of block lengths; otherwise, when data set in the descriptor nodes are not performed at block boundaries, then data mismatch errors occur.

10.3.3.3 Register bank with IP bus interface

Register accesses through the IP bus interface are on the register bank.

See the figure below for the block diagram.

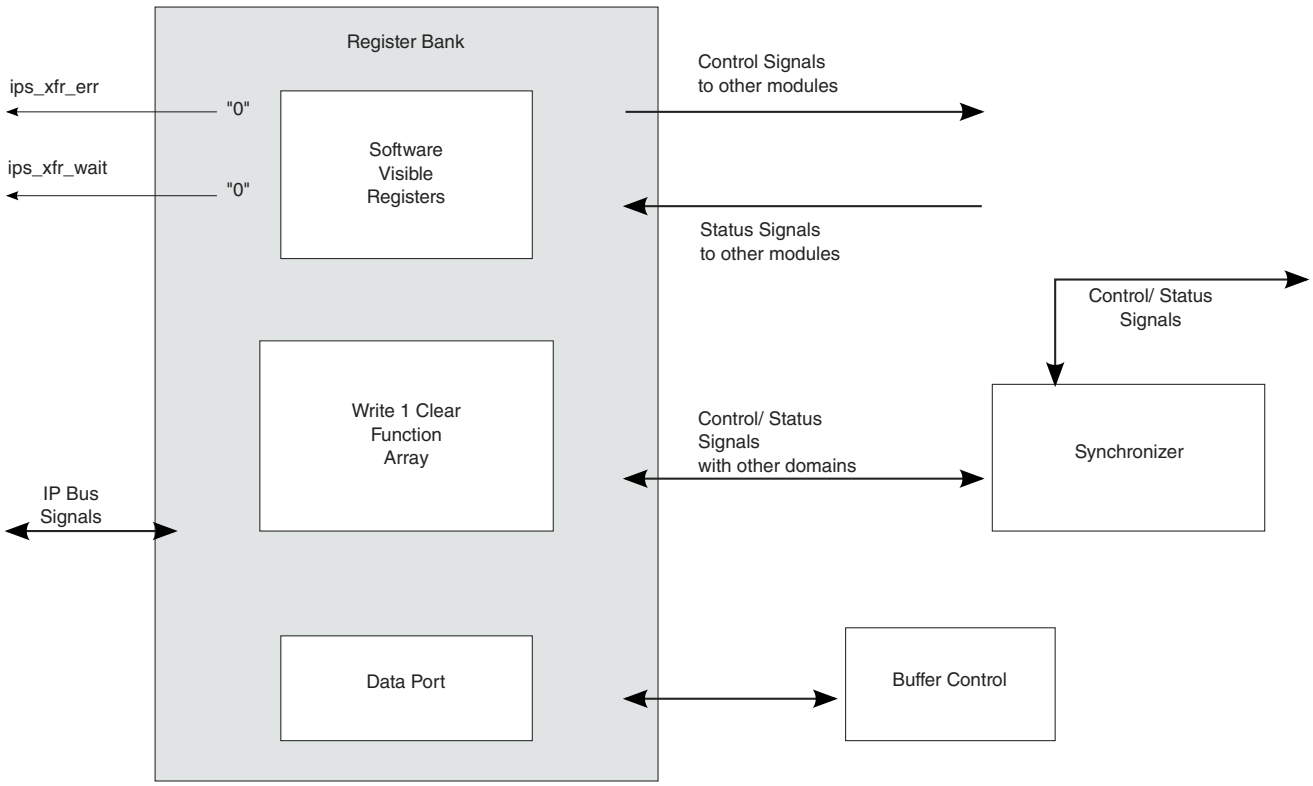


Figure 10-64. Register bank diagram

Only a 32-bit access is allowed, and no partial read/write is supported; therefore, all accesses are word aligned.

10.3.3.3.1 SD protocol unit

The SD protocol unit deals with all SD protocol affairs.

The SD protocol unit performs the following functions:

- Acts as the bridge between the internal buffer and the SD bus
- Sends the command data as well as its argument serially
- Stores the serial response bit stream into corresponding registers
- Detects the bus state on the CMD/DATA lines
- Monitors the interrupt from the SDIO card
- Asserts the read wait signal
- Gates off the SD clock when buffer is announcing danger status
- Detects the write protect state

The SD protocol unit consists of four submodules:

- SD control misc
- Command control

- Data control
- Clock control

10.3.3.3.2 SD control misc

In the SD control misc unit, the card detection (including the CD_B and DATA3 used for card detection), write protection, and card interrupt are implemented.

This module monitors the signal level on all the eight data lines and the command lines. It directly routes the level values into the register bank.

The module also detects the Write Protect (WP) line. If WP is active, writes to the register bank are ignored.

This module also drives the LCTL output signal when the LCTL bit is set by the driver.

10.3.3.3.3 SD clock control

If the internal data buffer is near full (for read) or near empty (for write), the SD clock must be gated off to avoid buffer over/under-run. This module asserts the gate of the output SD clock to shut the clock off. After the buffer has space (for read) or has data (for write), the clock gate of this module opens, and the SD clock is active again.

10.3.3.3.4 Command control

The Command Control module deals with the transactions on the CMD line.

See the figure below for an illustration of the structure for the Command CRC shift register.

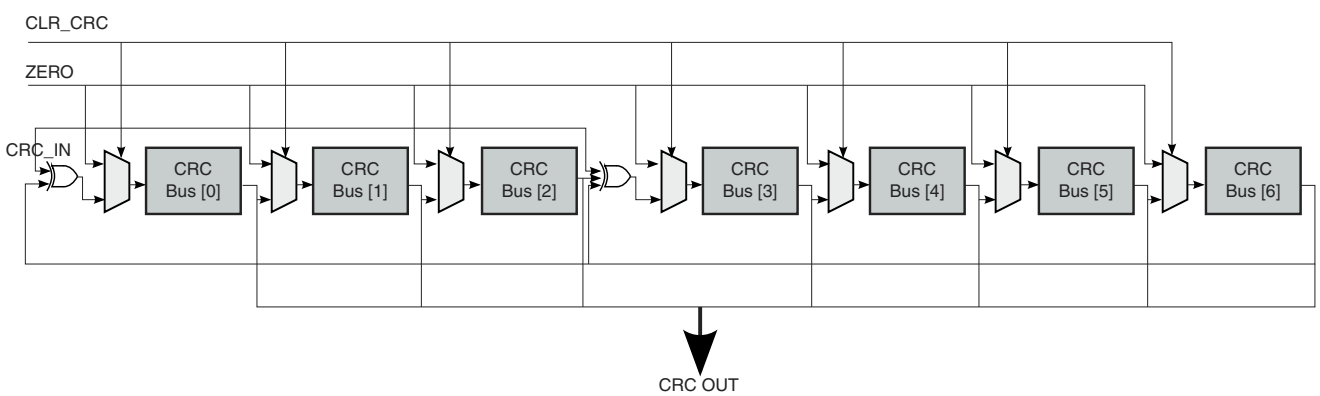


Figure 10-65. Command CRC shift register

The CRC polynomials for the CMD are as follows:

Generator polynomial: $G(x) = x^7 + x^3 + 1$
$M(x) = (\text{first bit}) * x^n + (\text{second bit}) * x^{n-1} + ... + (\text{last bit}) * x^0$
$CRC[6:0] = \text{Remainder } [(M(x) * x^7) / G(x)]$

10.3.3.3.5 Data control

The data agent deals with the transactions on the eight data lines. Moreover, this module also detects the busy state on the DATA0 line, and generates the read wait state by the request from the transceiver.

The CRC polynomials for the data are as follows:

Generator polynomial: $G(x) = x^{16} + x^{12} + x^5 + 1$

$M(x) = (\text{first bit}) * x^n + (\text{second bit}) * x^{n-1} + ... + (\text{last bit}) * x^0$

$CRC[15:0] = \text{Remainder } [(M(x) * x^{16}) / G(x)]$

10.3.3.4 Clock and reset manager

This module controls all four kinds reset signals within uSDHC:

- Hardware reset
- Software reset for all logic
- Software reset for the data logic
- Software reset for the command logic

All these signals are fed into this module and stable signals are generated inside the module to reset all other modules. The module also gates off all the inside signals.

10.3.3.5 Clock generator

The clock generator generates the card CLK by peripheral source clock in two stages. The clock divisor can be configured through register SYS_CTRL [SDCLKFS] is for prescaler configuration while the [DVS] is for divisor configuration. Details can be found in the register function description. See the following figure for the structure of the divider. The term "Base" represents the frequency of peripheral source clock.

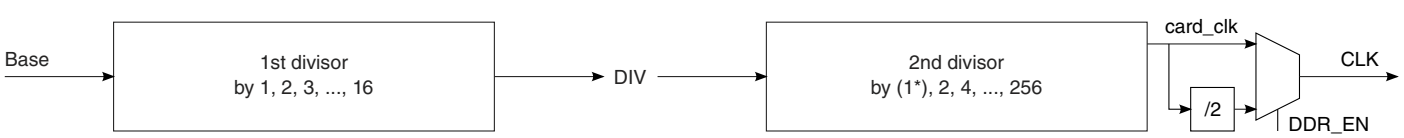


Figure 10-66. Two stages of the clock divider

The first stage outputs an intermediate clock (DIV) that can be Base, Base/2, Base/3, ..., or Base/16.

The second stage is a prescaler and outputs the actual internal working clock (card_clk). This clock is the driving clock for all the sub modules of the SD protocol unit, and helps in syncing FIFOs (see Figure 10-55) to synchronize with the data rate from the internal data buffer. The frequency of the clock output from this stage can be DIV, DIV/2, DIV/4,..., or DIV/256. Therefore, the highest frequency of the card_clk is base, and the next highest is Base/2, while the lowest frequency is Base/4096. If the duty cycle of the base clock is 50%, the duty cycle of card_clk is also 50%, even when the compound divisor is an odd value.

NOTE

CLK is different for the SDR and DDR modes.

- In the SDR mode, CLK is equal to the internal working clock (card_clk).

- In the DDR mode, CLK is equal to card_clk/2.

10.3.3.6 SDIO card interrupt

Information on interrupts in 1-bit mode, interrupts in 4-bit mode, and card interrupt handling are detailed in the sections below.

10.3.3.6.1 Interrupts in 1-bit mode

In this case, the DATA1 pin provides the interrupt function. An interrupt is asserted by pulling the DATA1 low from the SDIO card, until the interrupt service is finished to clear the interrupt.

10.3.3.6.2 Interrupt in 4-bit mode

As the interrupt and data line 1 share pin 8 in a 4-bit mode, an interrupt is only sent by the card and recognized by the host during a specific time. This is known as the interrupt period. The uSDHC module only provides sample the level on pin 8 during the interrupt period. At all other times, the host ignores the level on pin 8 and treats it as the data signal. The definition of the interrupt period is different for operations with single block and multiple block data transfers.

In the case of normal single data block transmissions, the interrupt period becomes active two clock cycles after the completion of a data packet. This interrupt period lasts until after the card receives the end bit of the next command that has a data block transfer associated with it.

For multiple block data transfers in a 4-bit mode, there is only a limited period of time that the interrupt period can be active because of the limited period of data line availability between the multiple blocks of data. This requires stricter definition of the interrupt period. For this case, the interrupt period is limited to two clock cycles. This begins two clocks after the end bit of the previous data block. During this 2-clock cycle interrupt period, if an interrupt is pending, the DATA1 line holds low for one clock cycle with the last clock cycle pulling DATA1 high. On completion of the interrupt period, the card releases the DATA1 line into the high Z state. The uSDHC module provides sample of the DATA1 during the interrupt period when the IABG bit in the Protocol Control register is set.

Refer to SDIO Card Specification v1.10 for further information about the SDIO card interrupt.

10.3.3.6.3 Card interrupt handling

When the CINTIEN bit in the Interrupt Signal Enable Register is set to 0, uSDHC clears the interrupt request to the host system. The host driver should clear this bit before servicing the SDIO interrupt and should set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts.

The SDIO Card Interrupt Status can be cleared by writing 1 to this bit. But as the interrupt source from the SDIO card does not clear, this bit is set again. To clear this bit, it is required to reset the interrupt source from the external card followed by writing 1 to this bit. In a 1-bit mode, uSDHC detects the SDIO interrupt with or without the SD clock (to support wakeup). In a 4-bit mode, the interrupt signal is sampled during the interrupt period, so there are some sample delays between the interrupt signal from the SDIO card and the interrupt to the Host System Interrupt Controller. When the SDIO status is set and the host driver needs to service this interrupt, the SDIO bit in the Interrupt Control Register of SDIO card is cleared. This is required to clear the SDIO interrupt status latched in uSDHC and to stop driving the interrupt signal to the System Interrupt Controller. The host driver must issue a CMD52 to clear the card interrupt. After completion of the card interrupt service, the SDIO Interrupt Status Enable bit is set to 1 and uSDHC starts sampling the interrupt signal again.

See the figure below for an illustration of the SDIO card interrupt scheme and for the sequences of software and hardware events that take place during a card interrupt handling procedure.

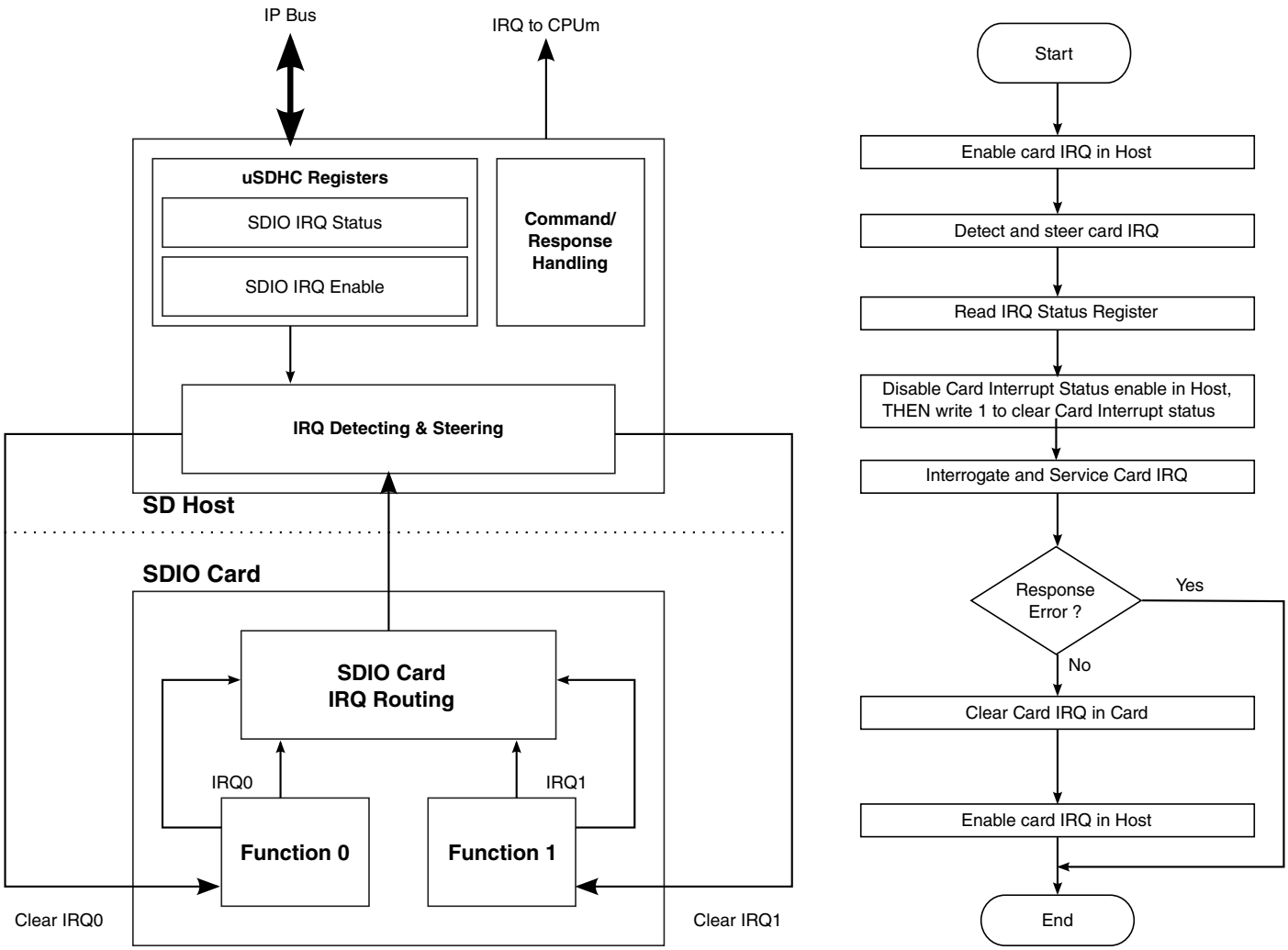


Figure 10-67. Card interrupt scheme, card interrupt detection, and handling procedure

10.3.3.7 Card insertion and removal detection

The uSDHC module uses either the DATA3 pin or the CD_B pin to detect card insertion or removal. When there is no card on the MMC/SD bus, the DATA3 is pulled to a low voltage level by default.

When any card is inserted to or removed from the socket, uSDHC detects the logic value changes on the DATA3 pin and generates an interrupt. When the DATA3 pin is not used for card detection (for example, it is implemented in GPIO), the CD_B pin must be connected for card detection. Whether DATA3 is configured for card detection or not, the CD_B pin is always a reference for card detection. Whether the DATA3 pin or the CD_B pin is used to detect card insertion, uSDHC sends an interrupt (if enabled) to inform the Host system that a card is inserted.

10.3.3.8 Power management and wakeup events

When there is no operation between uSDHC and the card through the SD bus, the user can completely disable the peripheral clock and base clock in the chip-level clock control module to save power. When the user needs to use uSDHC to communicate with the card, it can enable the clock and start the operation.

In some circumstances, when the clocks to uSDHC are disabled, for instance, when the system is in low-power mode, there are some events for which the user needs to enable the clock and handle the event. These events are called wakeup interrupts. The uSDHC module can generate these interrupts even when there are no clocks enabled. The three interrupts that can be used as wakeup events are these:

- Card removal interrupt
- Card insertion interrupt
- Interrupt from SDIO card

The uSDHC module offers a power management feature. By clearing the clock enabled bits in the System Control register, the clocks are gated in the low position to uSDHC. For maximum power saving, the user can disable all the clocks to uSDHC when there is no operation in progress.

These three wakeup events (or wakeup interrupts) can also be used to wakeup the system from low-power modes.

NOTE

To make the interrupt a wakeup event, when all the clocks to uSDHC are disabled or when the entire system is in low power mode, the corresponding wakeup enabled bit needs to be set. Refer to Protocol Control (PROT_CTRL) for more information on the uSDHC Protocol Control register.

10.3.3.8.1 Setting wakeup events

For uSDHC to respond to a wakeup event, the software must set the respective wakeup enable bit before the CPU enters the sleep mode.

Before the software disables the host clock, it should ensure that all the following conditions have been met:

- No read or write transfer is active
- Data and command lines are not active
- No interrupts are pending
- Internal data buffer is empty

10.3.3.9 MMC fast boot

The Embedded MultiMediaCard (eMMC4.3) specification adds a fast boot feature that requires hardware support. There are two types of fast boot modes in the eMMC4.3 specification: boot operation and alternative boot operation in the eMMC4.3 specification. Each type also has with-acknowledge and without-acknowledge modes.

In the boot operation mode, the master (MultiMediaCard host) can read boot data from the slave (MMC device) by keeping CMD line low after power-on, or sending CMD0 with argument + 0xFFFFFFFA (optional for slave), before issuing CMD1.

NOTE

For the eMMC4.3 card setting, please see the eMMC4.3 specification.

10.3.3.9.1 Boot operation

If the CMD line is held low for 74 clock cycles and more after power-up before the first command is issued, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within one second after the CMD line goes low, the slave starts to send the first boot data to the master on the DATA line(s). The master must keep the CMD line low to read all of the boot data.

NOTE

For the purposes of this documentation, fast boot is called "normal fast boot mode".

If boot acknowledge is enabled, the slave has to send acknowledge pattern '010' to the master within 50ms after the CMD line goes low. If boot acknowledge is disabled, the slave does not send out acknowledge pattern '010'.

The master can terminate the boot mode with the CMD line high.

The boot operation is terminated when all the contents of the enabled boot data are sent to the master. After the boot operation is executed, the slave gets ready for the CMD1 operation and the master needs to start a normal MMC initialization sequence by sending CMD1.

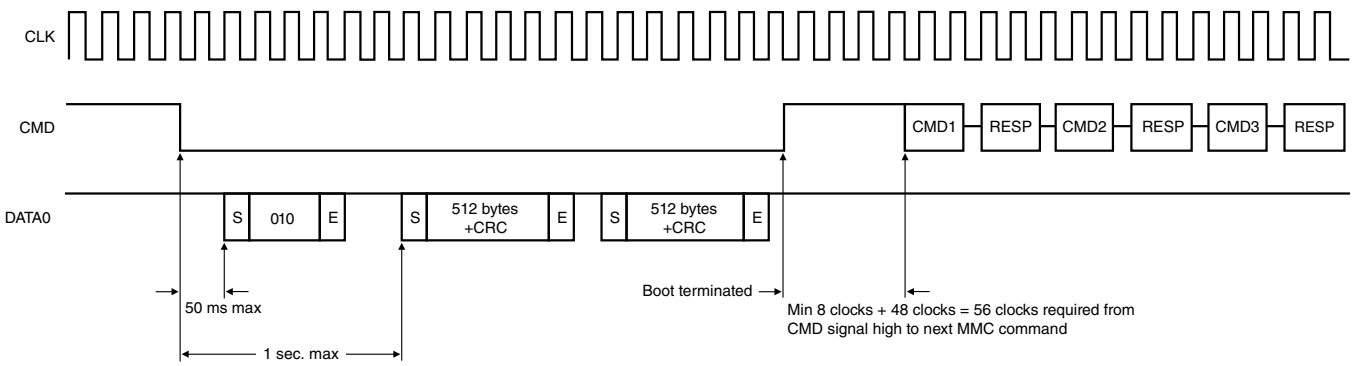


Figure 10-68. MultiMediaCard state diagram (normal boot mode)

10.3.3.9.2 Alternative boot operation

This boot function is optional for the device. If bit 0 in the extended CSD byte[228] is set to '1', the device supports the alternative boot operation.

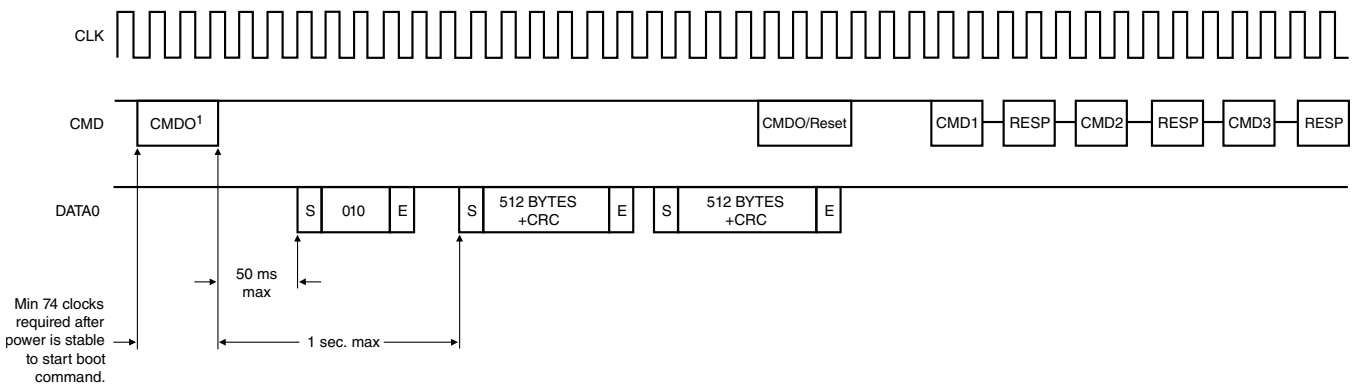
After power-up, if the host issues CMD0 with the argument of 0xFFFFFFFA after 74 clock cycles, before CMD1 is issued or the CMD line goes low, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within one second after CMD0 with the argument of 0xFFFFFFFA is issued, the slave starts to send the first boot data to the master on the DATA line(s).

If boot acknowledge is enabled, the slave has to send the acknowledge pattern '010' to the master within 50ms after the CMD0 with the argument of 0xFFFFFFFA is received. If boot acknowledge is disabled, the slave does not send out acknowledge pattern '010'.

The master can terminate the boot mode by issuing CMD0 (Reset).

Boot operation is terminated when all the contents of the enabled boot data are sent to the master. After the boot operation is executed, the slave gets ready for the CMD1 operation and the master needs to start a normal MMC initialization sequence by sending CMD1.



NOTE 1: CMD0 with argument 0xFFFFFFFA

Figure 10-69. MultiMediaCard state diagram (alternative boot mode)

10.3.4 Initialization/application of uSDHC

All communication between the system and cards are controlled by the host. The host sends commands of two types: broadcast and addressed (point-to-point).

Broadcast commands are intended for all cards, such as GO_IDLE_STATE, SEND_OP_COND, and ALL_SEND_CID. In the Broadcast mode, all cards are in the open-drain mode to avoid bus contention. Refer to Commands for MMC/SD/SDIO for the commands of bc and bcr categories.

After the broadcast command CMD3 is issued, the cards enter the standby mode. Addressed type commands are used from this point. In this mode, the CMD/DATA I/O pads turns to the push-pull mode to have the driving capability for maximum frequency operation. Refer to Commands for MMC/SD/SDIO for the commands of ac and adtc categories.

10.3.4.1 Command send & response receive basic operation

Assuming that the data type WORD is an unsigned 32-bit integer, the flow indicated below presents a guideline for sending a command to the card(s):

```
send_command(cmd_index, cmd_arg, other requirements)
{
WORD wCmd; // 32-bit integer to make up the data to write into Transfer Type register, it is
recommended to implement in a bit-field manner
wCmd = (<cmd_index> & 0x3f) >> 24; // set the first 8 bits as '00'+<cmd_index>
set CMDTYP, DPSEL, CICEN, CCCEN, RSTTYP, DTDSEL accorind to the command index;
if (internal DMA is used) wCmd |= 0x1;
if (multi-block transfer) {
          set MSBSEL bit;
          if (finite block number) {
                    set BCEN bit;
                    if (auto12 command is to use) set AC12EN bit;
          }
```

```
}
write_reg(CMDARG, <cmd_arg>); // configure the command argument
write_reg(XFERTYP, wCmd); // set Transfer Type register as wCmd value to issue the command
}
wait_for_response(cmd_index)
{
while (CC bit in IRQ Status register is not set); // wait until Command Complete bit is set
read IRQ Status register and check if any error bits about Command are set
if (any error bits are set) report error;
write 1 to clear CC bit and all Command Error bits;
}
```

For the sake of simplicity, the function wait_for_response is implemented here by means of polling. For an effective and formal way, the response is usually checked after the Command Complete Interrupt is received. When doing this, make sure that the corresponding interrupt status bits are enabled.

For some scenarios, the response time-out is expected. For instance, after all cards respond to CMD3 and move to the Standby state no response to the Host when CMD2 is sent. The host driver deals with "fake" errors like this with caution.

10.3.4.2 Card identification mode

When a card is inserted to the socket or the card is reset by the host, the host needs to validate the operation voltage range, identify the cards, request the cards to publish the Relative Card Address (RCA) or to set the RCA for MMC cards.

10.3.4.2.1 Card detect

See the figure below for a flow diagram showing the detection of MMC, SD, and SDIO cards using uSDHC.

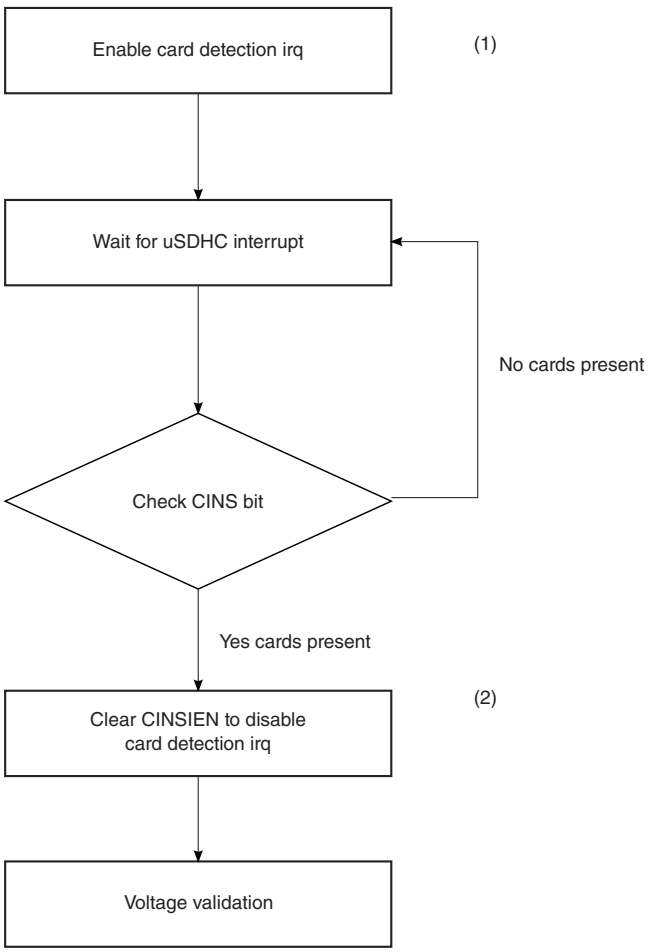


Figure 10-70. Flow diagram for card detection

Here is the card detect sequence:

- Set the CINSIEN bit to enable card detection interrupt.
- When an interrupt from uSDHC is received, check the CINS bit in the Interrupt Status register to see if it was caused by card insertion.
- Clear the CINSIEN bit to disable the card detection interrupt and ignore all card insertion interrupts afterwards.

10.3.4.2.2 Reset

The host consists of three types of resets:

- Hardware reset (Card and Host) that is driven by Power On Reset (POR).

- Software reset (Host only) is initiated by the write operation on the RSTD, RSTC, or RSTA bits of the System Control register to reset the data part, command part, or all parts of the host controller, respectively.
- Card reset (Card only): The command, "Go_Idle_State" (CMD0), is the software reset command for all types of MMC cards and SD memory cards. This command sets each card into the idle state regardless of the current card state. For an SD I/O card, CMD52 is used to write an I/O reset in CCCR. The cards are initialized with a default relative card address (RCA=0x0000) and with a default driver stage register setting (lowest speed, highest driving current capability).

After the card is reset, the host needs to validate the voltage range of the card. See the figure below for the software flow to reset both uSDHC and the card.

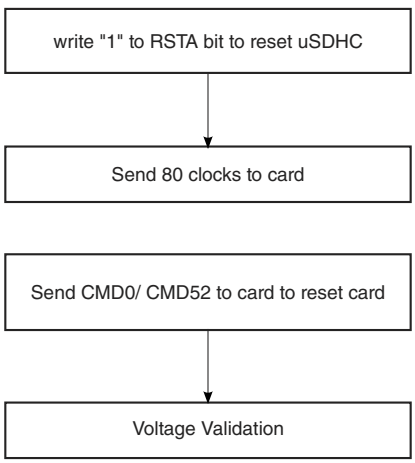


Figure 10-71. Flow chart for resetting uSDHC and SD I/O card

```
software_reset()
{
set_bit(SYSCTRL, RSTA); // software reset the Host
set DTOCV and SDCLKFS bit fields to get the CLK of frequency around 400kHz
configure IO pad to set the power voltage of external card to around 3.0V
poll bits CIHB and CDIHB bits of PRSSTAT to wait both bits are cleared
set_bit(SYSCTRL, INTIA); // send 80 clock ticks for card to power up
send_command(CMD_GO_IDLE_STATE, <other parameters>); // reset the card with CMD0
or send_command(CMD_IO_RW_DIRECT, <other parameters>);
}
```

10.3.4.2.3 Voltage validation

All cards should be able to establish communication with the host using any operation voltage in the maximum allowed voltage range specified in the card specification. However, the supported minimum and maximum values for VDD are defined in the Operation Conditions Register (OCR) and may not cover the whole range.

Cards that store the CID and CSD data in the preload memory are only able to communicate this information under data transfer VDD conditions. This means that if the host and card have non-common VDD ranges, the card is neither able to complete the identification cycle nor able to send CSD data.

Therefore, special commands Send_Op_Cont (CMD1 for MMC), SD_Send_Op_Cont (ACMD41 for SD Memory), and IO_Send_Op_Cont (CMD5 for SD I/O) are used. The voltage validation procedure is designed to provide a mechanism to identify and reject cards that do not match the VDD range(s) desired by the host. This is accomplished when the host sends the desired VDD voltage window as the operand of this command. Cards that cannot perform the data transfer in the specified range must discard themselves from further bus operations and go into the Inactive state. By omitting the voltage range in the command, the host can query each card and determine the common voltage range before sending out-of-range cards into the Inactive state. This query should be used if the host is able to select a common voltage range or if a notification is sent to the system when a non-usable card in the stack is detected.

The following steps show how to perform voltage validation when a card is inserted:

```
voltage_validation(voltage_range_arguement)
{
label the card as UNKNOWN;
send_command(IO_SEND_OP_COND, 0x0, <other parameters are omitted>); // CMD5, check SDIO
operation voltage, command argument is zero
if (RESP_TIMEOUT != wait_for_response(IO_SEND_OP_COND)) { // SDIO command is accepted
          if (0 < number of IO functions) {
                    label the card as SDIO;
                    IORDY = 0;
                    while (!(IORDY in IO OCR response)) { // set voltage range for each IO
function
                              send_command(IO_SEND_OP_COND, <voltage range>, <other
parameter>);
                              wait_for_response(IO_SEND_OP_COND);
                    } // end of while ...
          } // end of if (0 < ...
          if (memory part is present inside SDIO card) Label the card as SDCombo; // this is
an
SD-Combo card
} // end of if (RESP_TIMEOUT ...
if (the card is labelled as SDIO card) return; // card type is identified and voltage range
is
set, so exit the function;
send_command(APP_CMD, 0x0, <other parameters are omitted>); // CMD55, Application specific
CMD
prefix
if (no error calling wait_for_response(APP_CMD, <...>) { // CMD55 is accepted
          send_command(SD_APP_OP_COND, <voltage range>, <...>); // ACMD41, to set voltage
range
for memory part or SD card
          wait_for_response(SD_APP_OP_COND); // voltage range is set
          if (card type is UNKNOWN) label the card as SD;
          return; //
} // end of if (no error ...
else if (errors other than time-out occur) { // command/response pair is corrupted
          deal with it by program specific manner;
} // of else if (response time-out
else { // CMD55 is refuse, it must be MMC card  if (card is already labelled as SDCombo)
{ //
change label
```

```
                    re-label the card as SDIO;
                    ignore the error or report it;
                    return; // card is identified as SDIO card
          } // of if (card is ...
          send_command(SEND_OP_COND, <voltage range>, <...>);
          if (RESP_TIMEOUT == wait_for_response(SEND_OP_COND)) { // CMD1 is not accepted,
either
                    label the card as UNKNOWN;
                    return;
          } // of if (RESP_TIMEOUT ...
} // of else
}
```

10.3.4.2.4 Card registry

Card registry for the MMC and SD/SDIO/SD combo cards are different. For the SD card, the identification process starts at a clock rate lower than 400 kHz and the power voltage higher than 2.7 V (as defined by the card spec). Currently, the CMD line output drives are push-pull drivers instead of open-drain. After the bus is activated, the host requests the card to send their valid operation conditions.

The response to ACMD41 is the operation condition register of the card. The same command is sent to all the new cards in the system. Incompatible cards are put into the Inactive state. The host then issues the command, All_Send_CID (CMD2), to each card to get its unique card identification (CID) number. Cards that are currently unidentified (in the Ready state), send their CID number as the response. After the CID is sent by the card, the card goes into the Identification state.

The host then issues Send_Relative_Addr (CMD3), requesting the card to publish a new relative card address (RCA) that is shorter than the CID. This RCA is used to address the card for future data transfer operations. After the RCA is received, the card changes its state to the Standby state. At this point, if the host wants the card to have an alternative RCA number, it may ask the card to publish a new number by sending another Send_Relative_Addr command to the card. The last published RCA is the actual RCA of the card.

The host repeats the identification process with CMD2 and CMD3 for each card in the system until the last CMD2 gets no response from any of the cards in the system.

For MMC operation, the host starts the card identification process in the open-drain mode with the identification clock rate lower than 400 kHz and the power voltage higher than 2.7 V. The open drain driver stages on the CMD line to allow parallel card operation during card identification. After the bus is activated, the host requests the cards to send their valid operation conditions (CMD1). The response to CMD1 is the "wired OR" operation on the condition restrictions of all cards in the system. Incompatible cards are sent into the Inactive state. The host then issues the broadcast command All_Send_CID (CMD2), asking all cards for their unique card identification (CID) number. All unidentified cards (the cards in the Ready state) simultaneously start sending their CID

numbers serially, while bit-wise monitoring their outgoing bit stream. Those cards, whose outgoing CID bits do not match the corresponding bits on the command line in any one of the bit periods, stop sending their CID immediately and must wait for the next identification cycle. As the CID is unique for each card, only one card can be successfully sent its full CID to the host. This card then goes into the Identification state. Thereafter, the host issues Set_Relative_Addr (CMD3) to assign a relative card address (RCA) to the card. After the RCA is received, the state of the card changes to standby, and the card does not react in further identification cycles. Also, its output driver switches from open-drain to push-pull. The host repeats the process, mainly CMD2 and CMD3, until the host receives a time-out condition to recognize the completion of the identification process.

The following steps show how to perform an operation using MMC cards:

```
card_registry()
{
do { // decide RCA for each card until response time-out
          if(card is labelled as SDCombo or SDIO) { // for SDIO card like device
                    send_command(SET_RELATIVE_ADDR, 0x00, <...>); // ask SDIO card to
publish its
RCA
                    retrieve RCA from response;
          } // end if (card is labelled as SDCombo ...
          else if (card is labelled as SD) { // for SD card
                    send_command(ALL_SEND_CID, <...>);
                    if (RESP_TIMEOUT == wait_for_response(ALL_SEND_CID)) break;
                    send_command(SET_RELATIVE_ADDR, <...>);
                    retrieve RCA from response;
          } // else if (card is labelled as SD ...
          else if (card is labelled as MMC) {
                    send_command(ALL_SEND_CID, <...>);
                    rca = 0x1; // arbitrarily set RCA, 1 here for example
                    send_command(SET_RELATIVE_ADDR, 0x1 << 16, <...>); // send RCA at upper
16
bits
          } // end of else if (card is labelled as MMC ...
} while (response is not time-out);
}
```

10.3.4.3 Card access

Information about Block Write, Block Read, Suspense Resume, ADMA Usage, Transfer Error, and Card Interrupt are detailed in the sections below.

10.3.4.3.1 Block write

Information on Normal Write, DDR Write, and Write with Pause are detailed in the sections below.

10.3.4.3.1.1 Normal Write

During a block write (CMD24 - 27, CMD60, CMD61), one or more blocks of data are transferred from the host to the card with a CRC appended to the end of each block by the host. If the CRC fails, the card indicates that the failure on the DATA line. The transferred data is discarded and not written, and all further transmitted blocks (in multiple block write mode) are ignored.

If the host uses partial blocks whose accumulated length is not block aligned and block misalignment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set), the card detects the block misalignment error and aborts the programming before the beginning of the first misaligned block. The card sets the ADDRESS_ERROR error bit in the status register, and while ignoring all further data transfer, waits in the Receive-data-State for a stop command. The write operation is also aborted if the host tries to write over a write-protected area.

For MMC and SD cards, programming of the CID and CSD registers does not require a previous block length setting. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in ROM, then this unchangeable part must match the corresponding part of the receive buffer. If this match fails, then the card reports an error and does not change any register contents.

For all types of cards, some may require long and unpredictable periods of time to write a block of data. After receiving a block of data and completing the CRC check, the card begins writing and holds the DATA line low if its write buffer is full and unable to accept new data from a new WRITE_BLOCK command. The host may poll the status of the card with a SEND_STATUS command (CMD13) or other means for SDIO cards at any time, and the card responds with its status. The responded status indicates whether the card can accept new data or whether the write process is still in progress. The host may deselect the card by issuing a CMD7 (to select a different card) to place the card into the Standby state and release the DATA line without interrupting the write operation. When re-selecting the card, it reactivates the busy indication by pulling data to low if the programming is still in progress and the write buffer is unavailable.

The software flow to write to a card incorporates the internal DMA and the write operation is a multi-block write with the Auto CMD12 enabled. For the other two methods (by means of external DMA orCPU polling status) with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to write to a card is described below:

1. Check the card status, wait until the card is ready for data.
2. Set the card block length/size:

- For SD/MMC cards, use SET_BLOCKLEN (CMD16).
- For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT (CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).

3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred, or another error that occurred during the auto12 command sending and response receiving.

10.3.4.3.1.2 DDR write

uSDHC supports the dual data rate mode.

The software flow to write to a card in the DDR mode is described as below:

1. Check the card status and wait until the card is ready for data.

NOTE

For eMMC4.4 card, block length only can be set to 512byte.

2. Set the uSDHC number block register (NOB), where nob is 5, for instance.
3. Set the eMMC4.4 card to high-speed mode and use SWITCH(CMD6).
4. Set the eMMC4.4 card bus with 4-bit/8-bit DDR mode and use SWITCH(CMD6).
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The DDR_EN and AC12EN bits should be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred or another error that occurred during the auto12 command sending and response receiving.

10.3.4.3.1.3 Write with Pause

The write operation can be paused during the transfer. Instead of stopping the CLK at any time to pause all the operations, which is also inaccessible to the host driver, the driver can set the Stop At Block Gap Request (SABGREQ) bit in the Protocol Control register to pause the transfer between the data blocks. As there is no time-out condition in a write

operation during the data blocks, a write to all types of cards can be paused in this way, and if the DATA0 line is not required to de-assert to release the busy state, no suspend command is needed.

Similar to the flow described in Normal Write, the write with pause is shown with the same kind of write operation:

1. Check the card status and wait until the card is ready for data.
2. Set the card block length/size:
   - For SD/MMC, use SET_BLOCKLEN (CMD16).
   - For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Set the SABGREQ bit.
7. Wait for the Transfer Complete interrupt.
8. Clear the SABGREQ bit.
9. Check the status bit to see if a write CRC error occurred.
10. Set the CREQ bit to continue the write operation.
11. Wait for the Transfer Complete interrupt.
12. Check the status bit to see if a write CRC error occurred or some another error that occurred during the auto12 command sending and response receiving.

The number of blocks left during the data transfer is accessible by reading the contents of the BLKCNT field in the Block Attribute register. As the data transfer and the setting of the SABGREQ bit are concurrent, and the delay of register read and the register setting, the actual number of blocks left may not be exactly the value read earlier. The host driver reads the value of BLKCNT after the transfer is paused and the Transfer Complete interrupt is received.

It is also possible that the last block has begun when the Stop At Block Gap Request is sent to the buffer. In this case, the next block gap is the end of the transfer. These types of requests are ignored, the driver should treat these as a non-pause transfer, and deal with it as a common write operation.

When the write operation is paused, the data transfer inside the host system is not stopped, and the transfer is active until the data buffer is full. Because of this, it is recommended to avoid using the Suspend command for the SDIO card. This is because when such a command is sent, uSDHC interprets the system that switches to another

function on the SDIO card and flush the data buffer. uSDHC takes the Resume command as a normal command with data transfer, and it is left for the host driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the Transfer Type register are set as well as the AC12EN bit. However, the uSDHC module automatically sends a CMD12 to mark the end of the multi-block transfer.

10.3.4.3.2 Block read

Information about Normal read, DDR read, Read with Pause, and Delay Line (DLL) in Read Path are detailed in the sections below.

10.3.4.3.2.1 Normal read

For block reads, the basic unit of data transfer is a block whose maximum size is stored in areas defined by the corresponding card specification. A CRC is appended to the end of each block, ensuring data transfer integrity. The CMD17, CMD18, CMD53, CMD60, CMD61, and so on, can initiate a block read. After completing the transfer, the card returns to the Transfer state. For multi blocks read, data blocks are continuously transferred until a stop command is issued.

The software flow to read from a card incorporates the internal DMA and the read operation is a multi-block read with the Auto CMD12 enabled. For the other two methods (by means of external DMA orCPU polling status) with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to read from a card is described below:

1. Check the card status and wait until card is ready for data.
2. Set the card block length/size:
   - For SD/MMC, use SET_BLOCKLEN (CMD16).
   - For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer read ready interrupt, configure the DMA settings, and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.

7. Check the status bit to see if a read CRC error occurred, or another error occurred during the auto12 command sending and response receiving.

10.3.4.3.2.2 DDR read

The uSDHC module supports dual data rate mode.

The software flow to write to a card in the DDR mode is described below:

1. Check the card status ans wait until the card is ready for data.
2. For eMMC4.4 card, block length can be set to only 512 bytes.
3. Set the uSDHC number block register (NOB) where nob is 5, for instance.
4. Set the eMMC4.4 card to high-speed mode and use SWITCH (CMD6).
5. Set the eMMC4.4 card bus with 4-bit /8-bit DDR mode and use SWITCH(CMD6).
6. Disable the buffer write ready interrupt, configure the DMA settings, and enable the uSDHC DMA when sending the command with data transfer. The DDR_EN and AC12EN bits should be set.
7. Wait for the Transfer Complete interrupt.
8. Check the status bit to see if a write CRC error occurred, or another error that occurred during the auto12 command sending and response receiving.

10.3.4.3.2.3 Read with Pause

The read operation is not generally able to pause. Only the SDIO card (and SDCombo card working under I/O mode) supporting the Read Wait feature can pause during the read operation. If the SDIO card supports Read Wait (SRW bit in CCCR register is 1), the host driver can set the SABGREQ bit in the Protocol Control register to pause the transfer between the data blocks.

Before setting the SABGREQ bit, ensure that the RWCTL bit in the Protocol Control register is set, otherwise uSDHC does not assert the Read Wait signal during the block gap and data corruption occurs. It is recommended to set the RWCTL bit after the Read Wait capability of the SDIO card is recognized.

Similar to the flow described in Normal read, the read with pause is shown with the same kind of read operation:

1. Check the SRW bit in the CCR register on the SDIO card to confirm that the card supports the Read Wait mode.
2. Set the RWCTL bit.
3. Check the card status and wait until the card is ready for data.
4. Set the card block length/size:

- For SD/MMC, use SET_BLOCKLEN (CMD16).
- For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).

5. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
6. Set the uSDHC number block register (NOB), where nob is 5, for instance.
7. Disable the buffer read ready interrupt, configure the DMA setting, and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
8. Set the SABGREQ bit.
9. Wait for the Transfer Complete interrupt.
10. Clear the SABGREQ bit.
11. Check the status bit to see if read CRC error occurred.
12. Set the CREQ bit to continue the read operation.
13. Wait for the Transfer Complete interrupt.
14. Check the status bit to see if a read CRC error occurred, or another error occurred during the auto12 command sending and response receiving.

Similar to the Write operation, it is possible to meet the ending block of the transfer when paused. In this case, uSDHC ignores the Stop At Block Gap Request and treats it as a command read operation.

Unlike the write operation, there is no remaining data inside the buffer when the transfer is paused. All data received before the pause is transferred to the host system. No matter if the Suspend command is sent or not, the internal data buffer is not flushed.

If the Suspend command is sent and the transfer is later resumed by means of a Resume command, uSDHC takes the command as a normal one accompanied with data transfer. It is left for the host driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the Transfer Type register are set, as well as the AC12EN bit. However, the uSDHC automatically sends CMD12 to mark the end of multi-block transfer.

10.3.4.3.2.4 Delay Line (DLL) in Read Path

The DLL is newly added to assist in sampling read data. The DLL provides the ability to programmatically select a quantized delay (in fractions of the clock period) regardless of on-chip variations such as process, voltage, and temperature (PVT).

The reasons why DLL is needed for uSDHC are these:

- The path of read data traveling from card to host varies.
- In the SD/MMC DDR mode, the minimum input setup and hold time are both at 2.5 ns.

The data sampling window is so small that the delay of loopback clock needs to be accurate and consistent regardless of PVT. The DLL takes the divided card_clk as the reference clock and loopback clock as the input clock. It then generates a delayed version of the input clock according to the programmed target delay.

The DLL can be disabled or bypassed, and it can also be manually set for a fixed delay in the override mode. The override value set is the number of delay cells. In the override mode, there is no need to set the DLL_enable. Another DLL mode is target value mode. In this mode, the DLL automatically adjusts the number of delay cells according to the target value set by the user and PVT changes. Be aware that the target value is in units of 1/32 of the clock reference period. If the card_clk is 100Mhz, then the reference clock period is 10ns; setting target value of 16 means 5ns =(16/32)*10ns. The software can disable automatic update by the setting dll_gate_update bit.

As the user may change the frequency of card_clk from time to time by changing SDCLKFS[7:0]/DVS[3:0], the software must adjust the delay value to ensure it works correctly when the reference clock (card_clk) is changed. The following is the correct flow, which should be ignored if DLL_CTRL_ENABLE is not set.

Step 1: Set the DLL_CTRL_RESET bit

Step 2: Configure the SDCLKFS[7:0] and DVS[3:0]

Step 3: Wait until SDSTB is asserted

Step 4: Clear the DLL_CTRL_RESET bit

Step 5: Wait until both the DLL_STS_SLV_LOCK and DLL_STS_REF_LOCK are asserted

Step 6: Set the DLL_CTRL_SLV_FORCE_UPD

Step 7: Clear the DLL_CTRL_SLV_FORCE_UPD

NOTE

The software should make sure that the DLL_CTRL_SLV_FORCE_UPD lasts for at least one card_clk. So, the software may need to add some delay between step 6 and step 7.

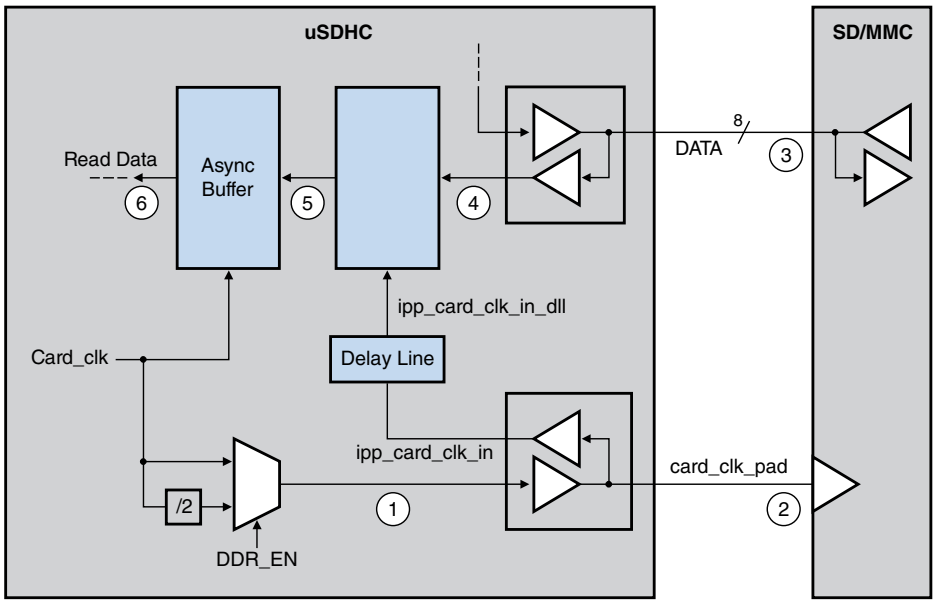


Figure 10-72. DLL in read path

10.3.4.3.3 Suspend Resume

The uSDHC module supports the Suspend Resume operations of SDIO cards, although slightly different than the suggested implementation of Suspend in the SDIO card specification.

10.3.4.3.3.1 Suspend

After setting the SABGREQ bit, the host driver may send a Suspend command to switch to another function of the SDIO card. The uSDHC module does not monitor the content of the response, so it does not know if the Suspend command succeeded or not. Accordingly, it does not de-assert Read Wait for read pause. To solve this problem, the host driver does not mark the Suspend command as "Suspend", (that is, setting the CMDTYP bits to 01). Instead, the driver sends this command as if it were a normal command, and only when the command succeeds, and the BS bit is set in the response, can the Driver send another command marked as "Suspend" to inform uSDHC that the current transfer is suspended. Here is the sequence for the Suspend operation:

1. Set the SABREQ bit to pause the current data transfer at block gap.
2. After the BGE bit is set, send the Suspend command to suspend the active function. The CMDTYP bit field must be 2'b00.
3. Check the BS bit of the CCCR in the response. If it is 1, repeat this step until the BS bit is cleared or abandon the suspend operation according to the Driver strategy.

4. Send another normal I/O command to the suspended function. The CMDTYP of this command must be 2'b01, so uSDHC can detect this special setting and be informed that the paused operation has successfully suspended. If the paused transfer is a read operation, uSDHC stops driving DATA2 and goes to the idle state.
5. Save the context registers in the system memory for later use, including the DMA System Address Register (for internal DMA operation), and the Block Attribute Register.
6. Begin operation for another function on the SDIO card.

10.3.4.3.3.2 Resume

To resume the data transfer, a Resume command is issued:

1. To resume the suspended function, restore the context register with the saved value in step #5 of the Suspend operation.
2. Send the Resume command: In the Transfer Type register, all the bit fields are set to the value as if this were another ordinary data transfer instead of a transfer resume (except the CMDTYP is set to 2'b10).
3. If the Resume command has responded, the data transfer is resumed.

10.3.4.3.4 ADMA usage

To use the ADMA in a data transfer, the host driver must prepare the correct descriptor chain prior to sending the read/write command.

The steps to prepare the correct descriptor chain are these:

1. Create a descriptor to set the data length that the current descriptor group is about to transfer. The data length should be even numbers of the block size.
2. Create another descriptor to transfer the data from the address setting in this descriptor. The data address must be at a page boundary (4KB address aligned).
3. If necessary, create a Link descriptor containing the address of the next descriptor. The descriptor group is created in steps 1 ~ 3.
4. Repeat steps 1 ~ 3 until all descriptors are created.
5. In the last descriptor, set the End flag to 1 and make sure the total length of all descriptors match the product of the block size and block number configured in the Block Attribute Register.
6. Set the ADMA System Address Register to the address of the first descriptor and set the DMAS field in the Protocol Control Register to 01 to select the ADMA.
7. Issue a write or read command with the DMAEN bit set to 1 in the Transfer Type Register.

Steps 1 ~ 5 are independent of step 6, so step 6 can finish before steps 1 ~ 5. Regarding the descriptor configuration, it is recommended not to use the Link descriptor as it requires extra system memory access.

10.3.4.3.5 Transfer error

Information about CRC, Internal DMA, Transfer ADMA, and Auto CMD12 errors are detailed in the sections below.

10.3.4.3.5.1 CRC error

It is possible at the end of a block transfer that a write CRC status error or read CRC error occurs. For this type of error, the latest block received is discarded. This is because the integrity of the data block is not guaranteed. It is recommended to discard the following data blocks and re-transfer the block from the corrupted one.

For a multi-block transfer, the host driver issues a CMD12 to abort the current process and start the transfer by a new data command. In this scenario, even when the AC12EN and BCEND bits are set, uSDHC does not automatically send a CMD12 because the last block is not transferred. On the other hand, if it is within the last block that the CRC error occurs, an Auto CMD12 is sent by uSDHC. In this case, the host driver re-sends or re-obtains the last block with a single block transfer.

10.3.4.3.5.2 Internal DMA error

During the data transfer with internal Simple DMA, if the DMA engine encounters an error on the AHB bus, the DMA operation is aborted, and the DMA error interrupt is sent to the host system. When acknowledged by such an interrupt, the host driver calculates the start address of data block in which the error occurs.

The start address can be calculated by either:

- Reading the DMA System Address register: The error occurs during the previous burst. Considering the block size, the previous burst length and the start address of the next burst transfer, it is straight forward to obtain the start address of the corrupted block.
- Reading the BLKCNT field of the Block Attribute register: Considering the number of blocks left, the total number to transfer, the start address of transfer, and the size of each block, the start address of corrupted block can be determined. When the BCEN bit is not set, the contents of the Block Attribute register do not change, so this method does not work.

When a DMA error occurs, it is recommended to abort the current transfer by means of a CMD12 (for multi block transfer), apply a reset for data, and restart the transfer from the corrupted block to recover from the error.

10.3.4.3.5.3 Transfer ADMA error

There are three kinds of possible ADMA errors: The AHB transfer, invalid descriptor, and data-length mismatch. Whenever these errors occur, the DMA transfer stops and the corresponding error status bit is set.

For acknowledging the status, the host driver should recover the error as shown below and re-transfer from the place of interruption.

- AHB transfer error: Such errors may occur during data transfer or descriptor fetch. For either scenario, it is recommended to retrieve the transfer context, reset for the data part and re-transfer the block that was corrupted, or to the next block if no block is corrupted.
- Invalid descriptor error: For such errors, it is recommended to retrieve the transfer context, reset for the data part and recreate the descriptor chain from the invalid descriptor and issue a new transfer. As the data to transfer now may be less than the previous setting, the data length configured in the new descriptor chain should match the new value.
- Data-length mismatch error: It is similar to recover from this error. The Host Driver polls relating registers to retrieve the transfer context, apply a reset for the data part, configure a new descriptor chain and make another transfer if there is data left. Like the previous scenario of the invalid descriptor error, the data length must match the new transfer.

10.3.4.3.5.4 Auto CMD12 error

After the last block of the multi-block transfer is sent or received, and the AC12EN bit is set when the data transfer is initiated by the data command, uSDHC automatically sends a CMD12 to the card to stop the transfer.

When errors with this command occur, it is recommended to the host driver to deal with the situations in the following manner:

- Auto CMD12 response time-out: It is not certain whether the command is accepted by the card or not. The host driver clears the auto CMD12 error status bits and re-send CMD12 until it is accepted by the card.

- Auto CMD12 response CRC error: As the card responds to CMD12, it aborts the transfer. The host driver may ignore the error and clear the error status bit.
- Auto CMD12 conflict error or not sent: The command is not sent; therefore, the host driver sends a CMD12 manually.

10.3.4.3.6 Card interrupt

The external cards can inform the host controller by means of some special signals. For the SDIO card, it can be the low-level on the DATA1 line during some special period. The uSDHC module only monitors the DATA1 line and supports the SDIO interrupt.

When the SDIO interrupt is captured by uSDHC, and the host system is informed by uSDHC asserting the uSDHC interrupt line, the interrupt service from the host driver is called.

As the interrupt source is controlled by the external card, the interrupt from the SDIO card must be serviced before the CINT bit is cleared by written. Refer to Card interrupt handling for the card interrupt handling flow.

10.3.4.4 Switch function

A switch command is issued by the host driver to enable new features added to the SD/MMC spec. The SD/MMC cards can transfer data at bus widths other than 1-bit. Different speed modes are also defined. To enable these features, a switch command is issued by the host driver.

For SDIO cards, the high-speed mode/DDR50/SDR50/SDR104 are enabled by writing the EHS bit in the CCCR register after the SHS bit is confirmed. For SD cards, the high-speed mode/DDR50/SDR50/SDR104 are queried and enabled by a CMD6 (with the mnemonic symbol as SWITCH_FUNC). For MMC cards, the high-speed mode/HS200/HS400 are queried by a CMD8 and enabled by a CMD6 (with the mnemonic symbol as SWITCH).

The SDR4-bit, SDR8-bit, DDR4-bit, and DDR8-bit width of MMC is also enabled by the SWITCH command, but with a different argument.

These new functions can also be disabled by a software reset. For SDIO cards, it can be done by setting the RES bit in the CCCR register. For other cards, it can be accomplished by issuing a CMD0. This method of restoring to the normal mode is not recommended because a complete identification process is needed before the card is ready for data transfer.

For the sake of simplicity, the following pseudocode examples do not show current capability check, which is recommended in the function switch process.

10.3.4.4.1 Query, enable, and disable SDIO high-speed mode

```
enable_sdio_high_speed_mode(void)
{
send CMD52 to query bit SHS at address 0x13;
if (SHS bit is '0') report the SDIO card does not support high speed mode and return;
send CMD52 to set bit EHS at address 0x13 and read after write to confirm EHS bit is set;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of around 50MHz;
(data transactions like normal peers)
}
disable_sdio_high_speed_mode(void)
{
send CMD52 to clear bit EHS at address 0x13 and read after write to confirm EHS bit is
cleared;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}
```

10.3.4.4.2 Query, enable, and disable SD high-speed mode/DDR50/SDR50/ SDR104

```
enable_sd_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0xFFFFFx and read 64 bytes of data accompanying the R1 response;
(high speed mode,x=1; SDR50,x=2; SDR104,x=3; DDR50,x=4;)
wait data transfer done bit is set;
check if the bit x of received 512 bits is set;
if (bit 401 is '0') report the SD card does not support high speed mode and return;
if (bit 402 is '0') report the SD card does not support SDR50 mode and return;
if (bit 403 is '0') report the SD card does not support SDR104 mode and return;
if (bit 404 is '0') report the SD card does not support DDR50 mode and return;
send CMD6, with argument 0x80FFFFFx and read 64 bytes of data accompanying the R1 response;
(high speed mode,x=1; SDR50,x=2; SDR104 x=3; DDR50 x=4;)
check if the bit field 379~376 is 0xF;
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of around 50MHz for high speed mode, 100MHz for SDR50, 200MHz for SDR104, 50MHz for
DDR50;
(data transactions like normal peers)
}
disable_sd_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0x80FFFFF0 and read 64 bytes of data accompanying the R1 response;
check if the bit field 379~376 is 0xF;
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}
```

10.3.4.4.3 Query, enable, and disable MMC high-speed mode

```
enable_mmc_high_speed_mode(void)
{
send CMD9 to get CSD value of MMC;
```

```
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the MMC does not support high speed mode and
return;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of MMC;
extract the value of CARD_TYPE field to check the 'high speed mode' in this MMC is 26MHz or
52MHz;
send CMD6 with argument 0x1B90100;
send CMD13 to wait card ready (busy line released);
send CMD8 to get EXT_CSD value of MMC;
check if HS_TIMING byte (byte number 185) is 1;
if (HS_TIMING is not 1) report MMC switching to high speed mode failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of around 26MHz or 52MHz according to the CARD_TYPE;
(data transactions like normal peers)
}
disable_mmc_high_speed_mode(void)
{
send CMD6 with argument 0x2B90100;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of MMC;
check if HS_TIMING byte (byte number 185) is 0;
if (HS_TIMING is not 0) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 20MHz;
(data transactions like normal peers)
}
```

10.3.4.4.4 Set MMC bus width

```
change_mmc_bus_width(void)
{
send CMD9 to get CSD value of MMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the MMC does not support multiple bit width and
return;
send CMD6 with argument 0x3B70x00; (8-bit(dual data rate), x=6; 4-bit(dual data rate), x=5;8-
bit, x=2; 4-bit, x=1; 1-bit, x=0)
send CMD13 to wait card ready (busy line released);
(data transactions like normal peers)
}
```

10.3.4.5 ADMA operation

Here are the codes for the ADMA1 and ADMA2 operations.

10.3.4.5.1 ADMA1 operation

```
Set_adma1_descriptor
{
if (to start data transfer) {
// Make sure the address is 4KB align.
Set 'Set' type descriptor;
{
Set Act bits to 01;
Set [31:12] bits data length (byte unit);
}
Set 'Tran' type descriptor;
{
Set Act bits to 10;
Set [31:12] bits address (4KB align);
}
```

```
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to 11;
Set [31:12] bits the next descriptor address (4KB aligned);
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set End bit to 1;
}
if (to generate interrupt for this descriptor) {
Set Int bit to 1;
}
Set Valid bit to 1;
}
```

10.3.4.5.2 ADMA2 operation

```
Set_adma2_descriptor
{
if (to start data transfer) {
// Make sure the address is a 32-bit boundary (lower 2-bit are always '00').
Set higher 32-bit of descriptor for this data transfer initial address;
Set [31:16] bits data length (byte unit);
Set Act bits to '10';
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to '11';
// Make sure the address is 32-bit boundary (lower 2-bit are always set to '00').
Set higher 32-bit of descriptor for the next descriptor address;
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set 'End' bit '1';
}
if (to generate interrupt for this descriptor) {
Set 'Int' bit '1';
}
Set the 'Valid' bit to '1';
}
```

10.3.4.6 Fast boot operation

10.3.4.6.1 Normal fast boot flow

Here are the steps of normal fast boot flow:

1. Software must configure the init_active bit (system control register bit 27) to make sure that 74 card clocks are finished.
2. Software must configure the MMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot), and bit 5 to 0 (normal fast boot), and bit 4 to select the ack mode. If the data is sent through the DMA mode, the software should configure bit 7 to enable the automatic stop at block gap feature, and configure bit 3-bit 0 to select the ack timeout value according to the SD CLK frequency.

3. Software then needs to configure the Block Attributes Register to set the block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes.
4. Software must configure the Protocol control register to set Data Transfer Width (DTW). If in the DDR fast boot mode, DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software needs to configure the Command Argument Register to set argument if needed (no need in normal fast boot).
6. Software must configure the Transfer Type Register to start the boot process. In normal boot mode, CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN and DMAEN retain the default value, where DPSEL bit is set to 1, DTDSEL is set to 1 and MSBSEL is set to 1.
7. DMAEN should be configured as 0 in the polling mode and if BCEN is configured as 1, it is recommended to configure the number of blocks in the Block Attributes Register to the maximum value. If in DDR fast boot mode, DDR_EN needs to be set to 1.
8. When step 6 is configured, the boot process begins. The software needs to poll the data buffer ready status to read the data from the buffer in time. If a boot timeout happens (ack times out or the first data read times out), an interrupt is triggered, and the software must configure MMC Boot Register to bit 6 to 0 to disable boot. This makes CMD high, then after at least 56 clocks, it is ready to begin a normal initialization process.
9. If there is no timeout, software needs to determine when the data read is finished and then configure MMC Boot Register bit 6 to 0 to disable boot. This render CMD line high and command completed asserted. After at least 56 clocks, it is ready to begin the normal initialization process.
10. Reset the host and then can begin the normal process.

10.3.4.6.2 Alternative fast boot flow

Here are the steps of alternative fast boot flow:

1. Software needs to configure init_active bit (system control register bit 27) to make sure 74 card clocks are finished.
2. Software needs to configure MMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot), and bit 5 to 1 (alternative boot), and bit 4 to select the ack mode or not. If data needs to be sent through the DMA mode, then configure bit 7 to enable the automatic stop at block gap feature. Software should also configure bit 3-bit 0 to select the ack timeout value according to the SD clock frequency.
3. Software then needs to configure the Block Attributes Register to set the block size and count. If in the DDR fast boot mode, the block size only can be configured to 512 bytes.

4. Software needs to configure the Protocol control register to set the data transfer width (DTW). If in the DDR fast boot mode, DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software needs to configure Command Argument Register to set argument to 0xFFFFFFFA.
6. Software needs to configure the Transfer Type Register to start the boot process by CMD0 with the 0xFFFFFFFA argument. In alternative boot, CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN, and DMAEN retain the default value. DPSEL bit is set to 1, DTDSEL is set to 1, and MSBSEL is set to 1. Note DMAEN should be configured as 0 in the polling mode, and if BCEN is configured as 1 in polling mode, it is recommended to configure the block count in the Block Attributes Register to the maximum value. If in the DDR fast boot mode, DDR_EN needs to be set to 1.
7. When step 6 is configured, the boot process begins. Software needs to poll the data buffer ready status to read the data from the buffer in time. If there is a boot timeout (ack data timeout in 50ms or data timeout in 1s), the host sends out the interrupt and the software needs to send CMD0 with reset and then configure the boot enable bit to 0 to stop this process.
8. If there is no time out, the software needs to decide when to stop the boot process, and send out the CMD0 with reset and then after the command is completed, configure the MMC Boot Register bit 6 to stop the process. After 8 clocks from the command completion, the slave (card) is ready for the identification step.
9. Reset the host and then begin the normal process.

10.3.4.6.3 Fast boot application case (in DMA mode)

In the boot application case, because the image destination and the image size are contained in the beginning of the image, it is necessary to switch DMA parameters on the fly during MMC fast boot.

In fast boot, the host can use Advanced DMA2 (ADMA2) with two destinations.

The detailed flow is described below:

1. The software needs to configure INIT_ACTIVE bit (system control register bit 27) to make sure that 74 card clocks are finished.
2. The software needs to configure the MMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot); and bit 5 to 0 (normal fast boot) or 1 (alternative boot); and bit 4 to select the ack mode. In DMA mode, configure bit 7 to 1 to enable the automatic stop at block gap feature. Also configure bits[31-16] to set the (BLK_CNT - VALUE1). Here VALUE1 is the value of the block count that needs to transfer the first time, so that the host stops at the block gap when the uSDHC controller gets VAULE1 blocks

from the device. Also, configure bits[3-0] to select the ack timeout value according to the SD clock frequency.

3. The software then needs to configure the Block Attributes Register to set block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes. In DMA mode, it is recommended to set the block count (BLK_CNT) to the max value (16'hffff).
4. The software needs to configure Protocol Control Register to set DTW (data transfer width). If in DDR fast boot mode, the DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software enable ADMA2 by configuring Protocol Control Register bits [9-8].
6. The software needs to set at least three pairs of ADMA2 descriptor in boot memory (that is, in IRAM, at least six words). The first pair descriptor defines the start address (that is, IRAM) and data length (that is,512byte*VALUE1) of the first part boot code. The software also needs to set the second pair descriptor, the second start address (any value that is writable), and data length is suggested to set 1~2word (record as VALUE2). Note that the second couple desc also transfers useful data even at lease 1 word, because our ADMA2 cannot support 0 data_length data transfer descriptor.
7. The software needs to configure Command Argument Register to set argument to 0xFFFFFFFA in alternative fast boot and do not need to be set in normal fast boot.
8. The software needs to configure Transfer Type Register to start the boot process. CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN, and DMAEN retain the default value. DPSEL bit is set to 1, DTDSEL is set to 1, and MSBSEL is set to 1. DMAEN is configured as 1 in the DMA mode. And, if BCEN is configured as 1, then configure blk no in Bock Attributes Register to the max value. And, if in the DDR fast boot mode, DDR_EN needs to be set to 1.
9. When step 8 is configured, boot process begins, the first VALUE1 block number data gets transferred. The software needs to poll the TC bit (bit1 in Interrupt Status Register) to determine first transfer is ended. Also, the software needs to polling the BGE bit (bit2 in Interrupt Status Register) to determine if the first transfer stops at the block gap.
10. When TC and BGE bits are set to 1, the software can analyze the first code of VALUE1 block, initializes the new memory device, if required, and sets the third pair of descriptors to define the start address and length of the remaining part of the boot code (VALUE3, the remain boot code block). Remember to set the last descriptor with END.
11. The software needs to configure the MMC Boot Register (offset 0xc4) again. Set bit 6 to 1 (enable boot); and bit 5 to 0 (normal fast boot), to 1 (alternative boot); and bit 4 to select the ack mode or not. In the DMA mode, configure bit 7 to 1 for enabling the automatically stop at block gap feature. Also, configure bit31-bit16 to set the (BLK_CNT - (VALUE1+1+VALUE3)), that host stops at block gap when the

uSDHC controller gets (VALUE1+1+VALUE3)) blocks from device totally include the blocks received in step 9. And need to configure bit 3-bit0 to select the ack timeout value according to the sd clk frequency. Note that the software does not need to configure the BLK_CNT again, because it is counted down automatically by the uSDHC controller.

12. The software needs to clear the TC and BGE bits and the software needs to clear SABGREQ (bit 16 in the Protocol control register) and set CREQ (bit17 in the Protocol control register) to 1 to resume the data transfer. Host transfers the VALUE2 and VALUE3 data to the destination that is set by descriptor.
13. The software needs to do poll BGE bit to determine if the fast boot is over.

Note:

1. When ADMA boot flow starts, for uSDHC, it is like a normal ADMA read operation. So, set ADMA2 descriptor as the normal ADMA2 transfer.
2. Need a few words length memory to keep descriptor.
3. For the 1~2-word data in second descriptor setting, it is a useful data, so the software needs to deal the data because of the application case.

10.3.5 Commands for MMC/SD/SDIO

A table containing the list of commands for the MMC/SD/SDIO cards is provided here.

Refer to the corresponding specifications for more details about the command information.

There are four kinds of commands defined to control the MultiMediaCard:

1. Broadcast commands (bc), no response
2. Broadcast commands with response (bcr), response from all cards simultaneously
3. Addressed (point-to-point) commands (ac), no data transfer on the DATA
4. Addressed (point-to-point) data transfer commands (adtc)

Response: A response is a token that is sent from the card to the host as an answer to a previously received command. A response is transferred serially on the CMD line.

Table 10-34. Commands for MMC/SD/SDIO cards

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	Resets all MMC and SD memory cards to idle state.

Table continues on the next page...

Table 10-34. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD1	bcr	[31:0] OCR without busy	R3	SEND_OP_COND	Asks all MMC and SD Memory cards in idle state to send their operation conditions register contents in the response on the CMD line.
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	Asks all cards to send their CID numbers on the CMD line.
CMD3[1]	ac	[31:6] RCA [15:0] stuff bits	R1 R6 (SDIO)	SET/ SEND_RELATIVE_ADDR	Assigns relative address to the card.
CMD4	bc	[31:0] DSR [15:0] stuff bits	-	SET_DSR	Programs the DSR of all cards.
CMD5	bc	[31:0] OCR without busy	R4	IO_SEND_OP_COND	Asks all SDIO cards in idle state to send them operation conditions register contents in the response on the CMD line.
CMD6[2]	adtc	[31] Mode 0: Check function 1: Switch function [30:8] Reserved for function groups 6 ~ 3 (All 0 or 0xFFFF) [7:4] Function group1 for command system [3:0] Function group2 for access mode	R1	SWITCH_FUNC	Checks switch ability (mode 0) and switch card function (mode 1). Refer to "SD Physical Specification V1.1" for more details.
CMD6[3]	ac	[31:26] Set to 0 [25:24] Access [23:16] Index [15:8] Value [7:3] Set to 0 [2:0] Cmd Set	R1b	SWITCH	Switches the mode of operation of the selected card or modifies the EXT_CSD registers. Refer to "The MultiMediaCard System Specification Version 4.0 Final draft 2" for more details.
CMD7	ac	[31:6] RCA [15:0] stuff bits	R1b	SELECT/ DESELECT_CARD	Toggles a card between the stand-by and transfer states or between the programming and disconnect states. In both cases, the card is selected by its own relative address and gets deselected by any other address. Address 0 deselects all.
CMD8	adtc	[31:0] stuff bits	R1	SEND_EXT_CSD	The card sends its EXT_CSD register as a block of data, with a block size of 512 bytes.
CMD9	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CSD	Addressed card sends its card-specific data (CSD) on the CMD line.

Table continues on the next page...

Table 10-34. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD10	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CID	Addressed card sends its card-identification (CID) on the CMD line.
CMD11	adtc	[31:0] data address	R1	READ_DAT_UNTIL_STOP	Reads data stream from the card, starting at the given address, until a STOP_TRANSMISSION follows.
CMD12	ac	[31:0] stuff bits	R1b	STOP_TRANSMISSION	Forces the card to stop transmission.
CMD13	ac	[31:6] RCA [15:0] stuff bits	R1	SEND_STATUS	Addressed card sends its status register.
CMD14	Reserved				
CMD15	ac	[31:6] RCA [15:0] stuff bits	-	GO_INACTIVE_STATE	Sets the card to inactive state in order to protect the card stack against communication breakdowns.
CMD16	ac	[31:0] block length	R1	SET_BLOCKLEN	Sets the block length (in bytes) for all following block commands (read and write). Default block length is specified in the CSD.
CMD17	adtc	[31:0] data address	R1	READ_SINGLE_BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command.
CMD18	adtc	[31:0] data address	R1	READ_MULTIPLE_BLOCK	Continuously transfers data blocks from card to host until interrupted by a stop command.
CMD19	adtc	[31:0] reserved bits(all 0)	R1	SEND_TUNING_BLOCK	64 bytes tuning pattern is sent for SDR50 and SDR104.
CMD20	adtc	[31:0] data address	R1	WRITE_DAT_UNTIL_STOP	Writes data stream from the host, starting at the given address, until a STOP_TRANSMISION follows.
CMD21	Reserved				
CMD22-23	Reserved				
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command.
CMD25	adtc	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	Continuously writes blocks of data until a STOP_TRANSMISSION follows.
CMD26	adtc	[31:0] stuff bits	R1	PROGRAM_CID	Programming of the card identification register. This command is issued only once per card. The card contains hardware to prevent this operation after the first programming. Normally this command is reserved for the manufacturer.
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	Programming of the programmable bits of the CSD.

Table continues on the next page...

Table 10-34. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD28	ac	[31:0] data address	R1b	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29	ac	[31:0] data address	R1b	CLR_WRITE_PROT	If the card provides write protection features, this command clears the write protection bit of the addressed group.
CMD30	adtc	[31:0] write protect data address	R1	SEND_WRITE_PROT	If the card provides write protection features, this command asks the card to send the status of the write protection bits.
CMD31	Reserved				
CMD32	ac	[31:0] data address	R1	TAG_SECTOR_START	Sets the address of the first sector of the erase group.
CMD33	ac	[31:0] data address	R1	TAG_SECTOR_END	Sets the address of the last sector in a continuous range within the selection of a single sector to be selected for erase.
CMD34	ac	[31:0] data address	R1	UNTAG_SECTOR	Removes one previously selected sector from the erase selection.
CMD35	ac	[31:0] data address	R1	TAG_ERASE_GROUP_START	Sets the address of the first erase group within a range to be selected for erase.
CMD36	ac	[31:0] data address	R1	TAG_ERASE_GROUP_END	Sets the address of the last erase group within a continuous range to be selected for erase.
CMD37	ac	[31:0] data address	R1	UNTAG_ERASE_GROUP	Removes one previously selected erase group from the erase selection.
CMD38	ac	[31:0] stuff bits	R1b	ERASE	Erase all previously selected sectors.
CMD39	ac	[31:0] RCA [15] register write flag [14:8] register address [7:0] register data	R4	FAST_IO	Used to write and read 8-bit (register) data fields. The command addresses a card, and a register, and provides the data for writing if the write flag is set. The R4 response contains data read from the address register. This command accesses application dependent registers which are not defined in the MMC standard.
CMD40	bcr	[31:0] stuff bits	R5	GO_IRQ_STATE	Sets the system into interrupt mode.
CMD41	Reserved				

Table continues on the next page...

Table 10-34. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CDM42	adtc	[31:0] stuff bits	R1b	LOCK_UNLOCK	Used to set/reset the password or lock/unlock the card. The size of the data block is set by the SET_BLOCK_LEN command.
CMD43~51	Reserved				
CMD52	ac	[31:0] stuff bits	R5	IO_RW_DIRECT	Access a single register within the total 128k of register space in any I/O function.
CMD53	ac	[31:0] stuff bits	R5	IO_RW_EXTENDED	Accesses a multiple I/O register with a single command. Allows the reading or writing of a large number of I/O registers.
CMD54	Reserved				
CMD55	ac	[31:16] RCA [15:0] stuff bits	R1	APP_CMD	Indicates to the card that the next command is an application specific command rather that a standard command.
CMD56	adtc	[31:1] stuff bits [0]: RD/WR	R1b	GEN_CMD	Used either to transfer a data block to the card or to get a data block from the card for general purpose / application specific commands. The size of the data block is set by the SET_BLOCK_LEN command.
CMD57-59	Reserved				
CMD60	adtc	[31] WR [30:24] stuff bits [23:16] address [15:8] stuff bits [7:0] byte count	R1b	RW_MULTIPLE_REGISTER	These registers are used to control the behavior of the device and to retrieve status information regarding the operation of the device. All Status and Control registers are WORD (32-bit) in size and are WORD aligned. CMD60 is used to read and write these registers.
CMD61	adtc	[31] WR [30:16] stuff bits [15:0] data unit count	R1b	RW_MULTIPLE_BLOCK	The host issues a RW_MULTIPLE_BLOCK (CMD61) to begin the data transfer.
CMD62-63	Reserved				
ACMD6[4]	ac	[31:2] stuff bits [1:0] bus width	R1	SET_BUS_WIDTH	Defines the data bus width ('00'=1bit or '10'=4bit bus) to be used for data transfer. The allowed data bus widths are given in SCR register.
ACMD13[4]	adtc	[31:0] stuff bits	R1	SD_STATUS	Send the SD Memory Card status.
ACMD22[4]	adtc	[31:0] stuff bits	R1	SEND_NUM_WR_SECTORS	Send the number of the written sectors (without errors). Responds with 32-bit plus the CRC data block.

Table continues on the next page...

Table 10-34. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
ACMD23[4]	ac	[31:23] stuff bits [22:0] Number of blocks	R1	SET_WR_BLK_ERASE_COUNT	Set the number of write blocks to be pre-erased before writing (to be used for fast Multiple Block WR command). "1"=default(one write block).
ACMD41[4]	bcr	[31:0] OCR	R3	SD_APP_OP_COND	Asks the accessed card to send its operating condition register (OCR) contents in the response on the CMD line.
ACMD42[4]	ac	[31:1] stuff bits [0] set_cd	R1	SET_CLR_CARD_DETECT	Connect(1)/Disconnect(0) the 50KOhm pull-up resistor on CD_B/DATA3 of the card.
ACMD51[4]	adtc	[31:0] stuff bits	R1	SEND_SCR	Reads the SD Configuration Register (SCR).

1. CMD3 differs for MMC and SD cards. For MMC cards, it is referred to as SET_RELATIVE_ADDR, with a response type of R1. For SD cards, it is referred to as SEND_RELATIVE_ADDR, with a response type of R6 (with RCA inside).
2. CMD6 differs completely between high speed MMC cards and high-speed SD cards. Command SWITCH_FUNC is for high-speed SD cards.
3. Command SWITCH is for high speed MMC cards. The Index field can contain any value from 0-255, but only values 0-191 are valid. If the Index value is in the 192-255 range the card does not perform any modification and the SWITCH_ERROR status bit in the EXT_CSD register is set. The Access Bits are shown in Table 2.
4. ACMDs is preceded with the APP_CMD command. Commands listed are used for SD only, other SD commands not listed are not supported on this module.

The access bits for the EXT_CSD access modes are shown below.

Table 10-35. EXT_CSD access modes

Bits	Access name	Operation
00	Command set	The command set is changed according to the Cmd Set field of the argument.
01	Set bits	The bits in the pointed byte are set, according to the bits set to 1 in the Value field.
10	Clear bits	The bits in the pointed byte are cleared, according to the bits set to 1 in the Value field.
11	Write byte	The Value field is written into the pointed byte.

10.3.6 Software restrictions

10.3.6.1 Initialization active

The driver cannot set INITA bit in System Control register when any of the command line or data lines are active, so the driver must ensure both CDIHB and CIHB bits are cleared.

10.3.6.2 Software polling procedure

For polling read or write, after the software begins a buffer read or write, it must access exactly the number of times as the values set in the Watermark Level Register; moreover, if the block size is not a multiple of the value in the Watermark Level Register (read and write respectively), the software must access exactly the remaining number of words at the end of each block.

For example, for a read operation, if the RD_WML is 4, indicating the watermark level is 16 bytes, block size is 40 bytes, and the block number is 2, then the access times for the burst sequence in the whole transfer process must be 4, 4, 2, 4, 4, 2.

10.3.6.3 Suspend operation

To suspend the data transfer, the software must inform uSDHC that the suspend command is successfully accepted. To achieve this, after the Suspend command is accepted by the SDIO card, software must send another normal command marked as suspend command (CMDTYP bits set as '01') to inform uSDHC that the transfer is suspended.

If software needs to resume the suspended transfer, it should read the value in BLKCNT register to save the remaining number of blocks before sending the normal command marked as suspend, otherwise on sending such a 'suspend' command, uSDHC treats the current transfer is aborted and change the BLKCNT register to its original value, instead of retaining the remaining number of blocks.

10.3.6.4 Data length setting

For either ADMA (ADMA1 or ADMA2) transfer, the data in the data buffer must be word aligned, so the data length set in the descriptor must be a multiple of 4.

10.3.6.5 (A)DMA address setting

To configure the ADMA1/ADMA2/DMA address register, when TC bit is set, the register always updates itself with the internal address value to support dynamic address synchronization, so the software must ensure that the TC bit is cleared prior to configuring the ADMA1/ADMA2/DMA address register.

10.3.6.6 Data port access

Data port does not support parallel access. For example, during an internal DMA access, it is not allowed to write any data to the data port by CPU; or during a CPU read operation, it is also prohibited to write any data to the data port, by either CPU or internal DMA. Otherwise the data is corrupted inside the uSDHC buffer.

10.3.6.7 Change clock frequency

The uSDHC module does not automatically gate off the card clock when the host driver changes the clock frequency. To prevent possible glitch on the card clock, clear the FRC_SDCLK_ON bit when changing the clock divisor value (SDCLKFS or DVS in System Control Register) or setting the RSTA bit.

Also, before changing the clock divisor value, the host driver should make sure that the SDSTB bit is high.

10.3.6.8 Multi-block read

For pre-defined multi-block read operation, that is, the number of blocks to read has been defined by previous CMD23 for MMC, or pre-defined number of blocks in CMD53 for SDIO/SDCombo, or whatever multi-block read without abort command at card side, an abort command, either automatic or manual CMD12/CMD52, is still required by uSDHC after the pre-defined number of blocks are done, to drive the internal state machine to idle mode.

In this case, the card may not respond to this extra abort command and uSDHC gets response timeout. It is recommended to manually send an abort command with RSPTYP[1:0] both bits cleared.

10.3.7 uSDHC memory map/register definition

10.3.7.1 uSDHC register descriptions

This section includes the module memory map and detailed descriptions of all registers.

See the table below for the register memory map of uSDHC. All these registers only support 32-bit accesses.

NOTE

The uSDHC registers are 32-bit wide and only support 32-bit access.

10.3.7.1.1 uSDHC memory map

uSDHC1 base address: 30B4_0000h

uSDHC2 base address: 30B5_0000h

uSDHC3 base address: 30B6_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	DMA System Address (DS_ADDR)	32	RW	0000_0000h
4h	Block Attributes (BLK_ATT)	32	RW	0001_0000h
8h	Command Argument (CMD_ARG)	32	RW	0000_0000h
Ch	Command Transfer Type (CMD_XFR_TYP)	32	RW	0000_0000h
10h	Command Response0 (CMD_RSP0)	32	RO	0000_0000h
14h	Command Response1 (CMD_RSP1)	32	RO	0000_0000h
18h	Command Response2 (CMD_RSP2)	32	RO	0000_0000h
1Ch	Command Response3 (CMD_RSP3)	32	RO	0000_0000h
20h	Data Buffer Access Port (DATA_BUFF_ACC_PORT)	32	RW	0000_0000h
24h	Present State (PRES_STATE)	32	RO	Table 10-82
28h	Protocol Control (PROT_CTRL)	32	RW	0880_0020h
2Ch	System Control (SYS_CTRL)	32	RW	0080_800Fh
30h	Interrupt Status (INT_STATUS)	32	W1C	0000_0000h
34h	Interrupt Status Enable (INT_STATUS_EN)	32	RW	0000_0000h
38h	Interrupt Signal Enable (INT_SIGNAL_EN)	32	RW	0000_0000h
3Ch	Auto CMD12 Error Status (AUTOCMD12_ERR_STATUS)	32	RW	0000_0000h
40h	Host Controller Capabilities (HOST_CTRL_CAP)	32	RW	07F3_B407h
44h	Watermark Level (WTMK_LVL)	32	RW	0810_0810h
48h	Mixer Control (MIX_CTRL)	32	RW	8000_0000h
50h	Force Event (FORCE_EVENT)	32	WORZ	0000_0000h
54h	ADMA Error Status (ADMA_ERR_STATUS)	32	RO	0000_0000h
58h	ADMA System Address (ADMA_SYS_ADDR)	32	RW	0000_0000h
60h	DLL (Delay Line) Control (DLL_CTRL)	32	RW	0000_0000h
64h	DLL Status (DLL_STATUS)	32	RO	0000_0200h
68h	CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS)	32	RW	0000_0000h
70h	Strobe DLL control (STROBE_DLL_CTRL)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
74h	Strobe DLL status (STROBE_DLL_STATUS)	32	RO	0000_0200h
C0h	Vendor Specific Register (VEND_SPEC)	32	RW	3000_7809h
C4h	MMC Boot (MMC_BOOT)	32	RW	0000_0000h
C8h	Vendor Specific 2 Register (VEND_SPEC2)	32	RW	0001_9006h
CCh	Tuning Control (TUNING_CTRL)	32	RW	0021_2800h
100h	Command Queue (CQE)	32	ROZ	0000_0510h

10.3.7.1.2 DMA System Address (DS_ADDR)

10.3.7.1.2.1 Offset

Register	Offset
DS_ADDR	0h

10.3.7.1.2.2 Function

This register contains the physical system memory address used for DMA transfers.

10.3.7.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DS_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DS_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.3.7.1.2.4 Fields

Field	Function
31-0 DS_ADDR	System address **DMA system address / argument 2**

Field	Function
	When ACMD23_ARGU2_EN is set to 0, SDMA uses this register as system address and supports only 32-bit addressing mode. Auto CMD23 cannot be used with SDMA. When ACMD23_ARGU2_EN is set to 1, SDMA uses ADMA System Address register (05Fh – 058h) instead of this register to support both 32-bit and 64-bit addressing. This register is used only for Argument2 and SDMA may use Auto CMD23. 1. **SDMA system address** Because the address must be word (4 bytes) aligned, the least 2 bits are reserved, always 0. When uSDHC stops a DMA transfer, this register points out the system address of the next contiguous data position. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operation during transfers may return an invalid value. The host driver initializes this register before starting a DMA transaction. After DMA has stopped, the system address of the next contiguous data position can be read from this register. This register is protected during a data transfer. When data lines are active, write to this register is ignored. The host driver waits until the DLA bit in the Present State register is cleared, before writing to this register. The uSDHC internal DMA does not support a virtual memory system. It only supports continuous physical memory access. And due to AHB burst limitations, if the burst must cross the 1 KB boundary, uSDHC automatically changes SEQ burst type to NSEQ. Because this register supports dynamic address reflecting, when TC bit is set, it automatically alters the value of internal address counter, so the software cannot change this register when TC bit is set. Such restriction is also listed in Software restrictions. 2. **Argument 2** This register is used with the Auto CMD23 to set a 32-bit block count value to the argument of the CMD23 while executing Auto CMD23. If Auto CMD23 is used with ADMA, the full 32-bit block count value can be used. If Auto CMD23 is used without ADMA, the available block count value is limited by the Block Count register. 65535 blocks is the maximum value in this case.

10.3.7.1.3 Block Attributes (BLK_ATT)

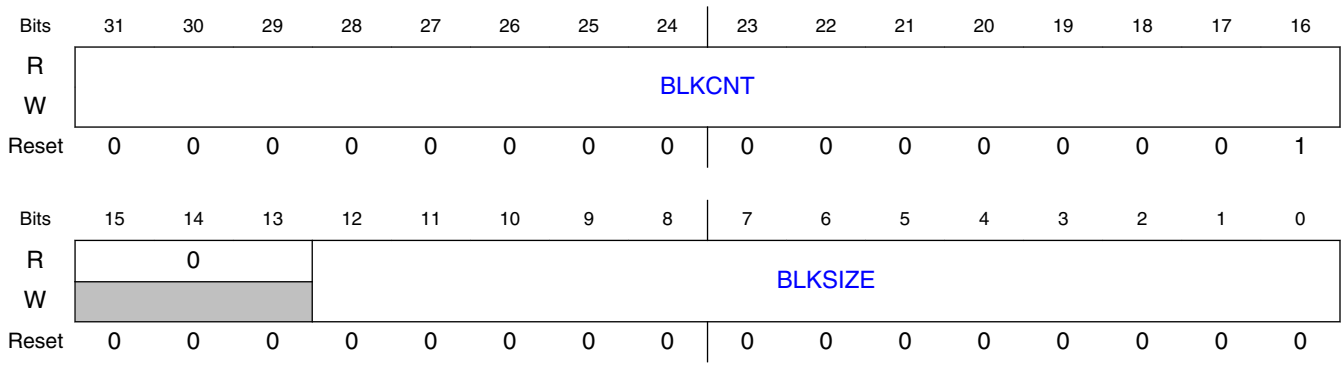
10.3.7.1.3.1 Offset

Register	Offset
BLK_ATT	4h

10.3.7.1.3.2 Function

This register is used to configure the number of data blocks and the number of bytes in each block.

10.3.7.1.3.3 Diagram



10.3.7.1.3.4 Fields

Field	Function
31-16 BLKCNT	Blocks count for current transfer This register is enabled when the Block Count Enable field in the Transfer Mode register is set to 1 and is valid only for multiple block transfers. For single block transfer, this register always reads as 1. The host driver sets this register to a value between 1 and the maximum block count. The uSDHC module decrements the block count after each block transfer and stops when the count reaches zero. Setting the block count to zero results in no data blocks being transferred. This register should be accessed only when no transaction is executing (that is, after transactions are stopped). During data transfer, read operations on this register may return an invalid value and write operations are ignored. When saving transfer content because of a Suspend command, the number of blocks yet to be transferred can be determined by reading this register. The reading of this register should be applied after transfer is paused by stop at block gap operation and before sending the command marked as suspend. This is because when the Suspend command is sent out, uSDHC treats the current transfer as aborted and change the BLKCNT register back to its original value instead of keeping the dynamical indicator of the remaining block count. When restoring transfer content prior to issuing a Resume command, the host driver restores the previously saved block count. **NOTE:** Although the BLKCNT field is 0 after reset, the read of reset value is 0x1. This is because when MSBSEL field is indicating a single block transfer, the read value of BLKCNT is always 1. 0000000000000000b - Stop count 0000000000000001b - 1 block 0000000000000010b - 2 blocks 1111111111111111b - 65535 blocks
15-13 —	Reserved
12-0 BLKSIZE	Transfer block size This register specifies the block size for block data transfers. Values ranging from 1 byte up to the maximum buffer size can be set. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operations during transfers may return an invalid value, and write operations are ignored. 0000000000000b - No data transfer 0000000000001b - 1 byte

Field	Function
	0000000000010b - 2 bytes 0000000000011b - 3 bytes 0000000000100b - 4 bytes 0000111111111b - 511 bytes 0001000000000b - 512 bytes 0100000000000b - 2048 bytes 1000000000000b - 4096 bytes

10.3.7.1.4 Command Argument (CMD_ARG)

10.3.7.1.4.1 Offset

Register	Offset
CMD_ARG	8h

10.3.7.1.4.2 Function

This register contains the SD/MMC command argument.

10.3.7.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CMDARG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CMDARG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.3.7.1.4.4 Fields

Field	Function
31-0 CMDARG	Command argument The SD/MMC command argument is specified as bits 39-8 of the command format in the SD or MMC specification. This register is write protected when the Command Inhibit (CMD) field in the Present State register is set.

10.3.7.1.5 Command Transfer Type (CMD_XFR_TYP)

10.3.7.1.5.1 Offset

Register	Offset
CMD_XFR_TYP	Ch

10.3.7.1.5.2 Function

This register is used to control the operation of data transfers. The host driver sets this register before issuing a command followed by a data transfer or before issuing a Resume command. To prevent data loss, uSDHC prevents writing to the bits, which are involved in the data transfer of this register, when data transfer is active. These bits are DPSEL, MBSEL, DTDSEL, AC12EN, BCEN, and DMAEN.

The host driver checks the Command Inhibit DAT field (PRES_STATE[CDIHB]) and the Command Inhibit CMD field (PRES_STATE[CIHB]) in the Present State register before writing to this register. When the CDIHB field in the Present State register is set, any attempt to send a command with data by writing to this register is ignored; when the CIHB field is set, any write to this register is ignored.

On sending commands with data transfer involved, it is mandatory that the block size is non-zero. Block count must also be non-zero, or indicated as a single block transfer (bit 5 of this register is '0' when written), or block count is disabled (bit 1 of this register is '0' when written), otherwise uSDHC ignores the sending of this command and do nothing. For write command, with all above restrictions, it is also mandatory that the write protect switch is not active (PRES_STATE[WPSPL] field of Present State register is '1'); otherwise, uSDHC also ignores the command.

If the commands with data transfer do not receive the response in 64 clock cycles, that is, if response time-out happens, uSDHC treats the external device, does not accept the command, and aborts the data transfer. In this scenario, the driver should issue the command again to retry the transfer. It is also possible that for some reason the card responds to the command but uSDHC does not receive the response, and if it is internal DMA (either simple DMA or ADMA) read operation, the external system memory is over-written by the internal DMA with data sent back from the card.

The table below shows the summary of how register settings determine the type of data transfer.

Table 10-36. Transfer type register setting for various transfer types

Multi/single block select	Block count enable	Block count	Function
0	Do not care	Do not care	Single transfer
1	0	Do not care	Infinite transfer
1	1	Positive number	Multiple transfer
1	1	Zero	No data transfer

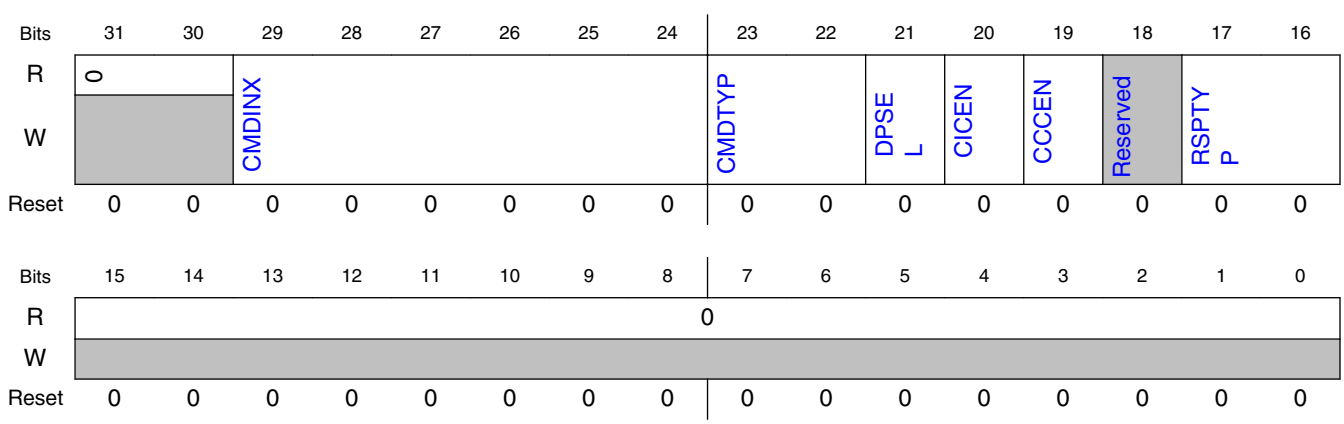
The table below shows the relationship between the Command Index Check Enable and the Command CRC Check Enable, regarding the Response Type bits as well as the name of the response type.

Table 10-37. Relationship between parameters and the name of the response type

Response type	Index check enable	CRC check enable	Name of response type
00	0	0	No response
01	0	1	R2
10	0	0	R3, R4
10	1	1	R1, R5, R6
11	1	1	R1b,R5b

- In the SDIO specification, response type notation for R5b is not defined. R5 includes R5b in the SDIO specification, but R5b is defined in this specification to specify that uSDHC checks the busy status after receiving a response. For example, usually CMD52 is used with R5, but the I/O abort command is used with R5b.
- The CRC fields for R3 and R4 are expected to be all 1 bits. The CRC check is disabled for these response types.

10.3.7.1.5.3 Diagram



10.3.7.1.5.4 Fields

Field	Function
31-30 —	Reserved
29-24 CMDINX	Command index These bits are set to the command number that is specified in bits 45-40 of the command-format in the SD Memory Card Physical Layer Specification and SDIO Card Specification.
23-22 CMDTYP	Command type There are three types of special commands: Suspend, Resume, and Abort. These bits are set to 00b for all other commands. • Suspend command: If the Suspend command succeeds, uSDHC assumes that the card bus has been released and that it is possible to issue the next command that uses the DATA line. Because uSDHC does not monitor the content of command response, it does not know if the Suspend command succeeded or not. It is the host driver's responsibility to check the status of the Suspend command and send another command marked as Suspend to inform uSDHC that a Suspend command was successfully issued. See Suspend Resume for more details. After the end bit of command is sent, uSDHC deasserts Read Wait for read transactions and stops checking busy for write transactions. In a 4-bit mode, the interrupt cycle starts. If the Suspend command fails, uSDHC maintains its current state, and the host driver restarts the transfer by setting the Continue Request field in the Protocol Control register. • Resume command: The host driver re-starts the data transfer by restoring the registers saved before sending the Suspend command and then sends the Resume command. The uSDHC module checks for a pending busy state before starting write transfers. • Abort command: If this command is set when executing a read transfer, uSDHC stops reads to the buffer. If this command is set when executing a write transfer, uSDHC stops driving the DATA line. After issuing the Abort command, the host driver should issue a software reset (Abort Transaction). 00b - Normal other commands 01b - Suspend CMD52 for writing bus suspend in CCCR 10b - Resume CMD52 for writing function select in CCCR 11b - Abort CMD12, CMD52 for writing I/O Abort in CCCR
21 DPSEL	Data present select This field is set to 1 to indicate that data is present and is transferred using the DATA line. It is set to 0 for the following: • Commands using only the CMD line (for example, CMD52) • Commands with no data transfer, but using the busy signal on DATA0 line (R1b or R5b (for example, CMD38)) **NOTE:** In resume command, this field is set, and other bits in this register is set the same as when the transfer was initially launched. When the write protect switch is on, (that is, the WPSPL field is active as '0'), any command with a write operation ignored. When this field is set, while the DTDSEL field is 0, writes to the register Transfer Type are ignored. 0b - No data present 1b - Data present
20 CICEN	Command index check enable If this field is set to 1, uSDHC checks the Index field in the response to see if it has the same value as the command index. If it is not, it is reported as a Command Index Error. If this field is set to 0, the Index field is not checked. 0b - Disable command index check 1b - Enables command index check

Table continues on the next page...

Field	Function
19 CCCEN	Command CRC check enable If this field is set to 1, uSDHC checks the CRC field in the response. If an error is detected, it is reported as a Command CRC Error. If this field is set to 0, the CRC field is not checked. The number of bits checked by the CRC field value changes according to the length of the response. See RSPTYP[1:0] and Command Transfer Type (CMD_XFR_TYP). 0b - Disables command CRC check 1b - Enables command CRC check
18 —	Reserved
17-16 RSPTYP	Response type select 00b - No response 01b - Response length 136 10b - Response length 48 11b - Response length 48, check busy after response
15-0 —	Reserved

10.3.7.1.6 Command Response0 (CMD_RSP0)

10.3.7.1.6.1 Offset

Register	Offset
CMD_RSP0	10h

10.3.7.1.6.2 Function

This register is used to store part 0 of the response bits from the card.

10.3.7.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.3.7.1.6.4 Fields

Field	Function
31-0 CMDRSP0	Command response 0 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this register for each response type.

10.3.7.1.7 Command Response1 (CMD_RSP1)

10.3.7.1.7.1 Offset

Register	Offset
CMD_RSP1	14h

10.3.7.1.7.2 Function

This register is used to store part 1 of the response bits from the card.

10.3.7.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.3.7.1.7.4 Fields

Field	Function
31-0 CMDRSP1	Command response 1 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this register for each response type.

10.3.7.1.8 Command Response2 (CMD_RSP2)

10.3.7.1.8.1 Offset

Register	Offset
CMD_RSP2	18h

10.3.7.1.8.2 Function

This register is used to store part 2 of the response bits from the card.

10.3.7.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.3.7.1.8.4 Fields

Field	Function
31-0 CMDRSP2	Command response 2 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this register for each response type.

10.3.7.1.9 Command Response3 (CMD_RSP3)

10.3.7.1.9.1 Offset

Register	Offset
CMD_RSP3	1Ch

10.3.7.1.9.2 Function

This register is used to store part 3 of the response bits from the card.

The table below describes the mapping of command responses from the SD bus to Command Response registers for each response type. In this table, R[ ] refers to a bit range within the response data as transmitted on the SD bus.

Table 10-38. Response bit definition for each response type

Response type	Meaning of response	Response field	Response register
R1,R1b (normal response)	Card status	R[39:8]	CMDRSP0
R1b (auto CMD12 response)	Card status for auto CMD12	R[39:8]	CMDRSP3
R2 (CID, CSD register)	CID/CSD register [127:8]	R[127:8]	{CMDRSP3[23:0], CMDRSP2, CMDRSP1, CMDRSP0}
R3 (OCR register)	OCR register for memory	R[39:8]	CMDRSP0
R4 (OCR register)	OCR register for I/O etc.	R[39:8]	CMDRSP0
R5, R5b	SDIO response	R[39:8]	CMDRSP0
R6 (publish RCA)	New published RCA[31:16] and card status[15:0]	R[39:9]	CMDRSP0

This table shows that most responses with a length of 48 (R[47:0]) have 32-bits of the response data (R[39:8]) stored in the CMDRSP0 register. Responses of type R1b (Auto CMD12 responses) have response data bits (R[39:8]) stored in the CMDRSP3 register. Responses with length 136 (R[135:0]) have 120-bits of the response data (R[127:8]) stored in the CMDRSP0, 1, 2, and 3 registers.

To be able to read the response status efficiently, uSDHC only stores part of the response data in the Command Response registers. This enables the host driver to efficiently read 32-bits of response data in one read cycle on a 32-bit bus system. Parts of the response, the Index field and the CRC, are checked by uSDHC (as specified by the Command Index Check Enable and the Command CRC Check Enable bits in the Transfer Type register) and generate an error interrupt if any error is detected. The bit range for the CRC check depends on the response length. If the response length is 48, uSDHC checks R[47:1], and if the response length is 136 the uSDHC checks R[119:1].

Because uSDHC may have a multiple block data transfer executing concurrently with a CMD_wo_DAT command, uSDHC stores the Auto CMD12 response in the CMDRSP3 register. The CMD_wo_DAT response is stored in CMDRSP0. This allows uSDHC to

avoid overwriting the Auto CMD12 response with the CMD_wo_DAT and vice versa. When uSDHC modifies part of the Command Response registers, as shown in the table above, it preserves the unmodified bits.

10.3.7.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.3.7.1.9.4 Fields

Field	Function
31-0 CMDRSP3	Command response 3 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this register for each response type.

10.3.7.1.10 Data Buffer Access Port (DATA_BUFF_ACC_PORT)

10.3.7.1.10.1 Offset

Register	Offset
DATA_BUFF_ACC_PORT	20h

10.3.7.1.10.2 Function

This is a 32-bit data port register used to access the internal buffer.

10.3.7.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATCONT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATCONT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.3.7.1.10.4 Fields

Field	Function
31-0 DATCONT	Data content The Buffer Data Port register is for 32-bit data access by the Arm platform or the external DMA. When the internal DMA is enabled, any write to this register is ignored, and any read from this register always yields 0s.

10.3.7.1.11 Present State (PRES_STATE)

10.3.7.1.11.1 Offset

Register	Offset
PRES_STATE	24h

10.3.7.1.11.2 Function

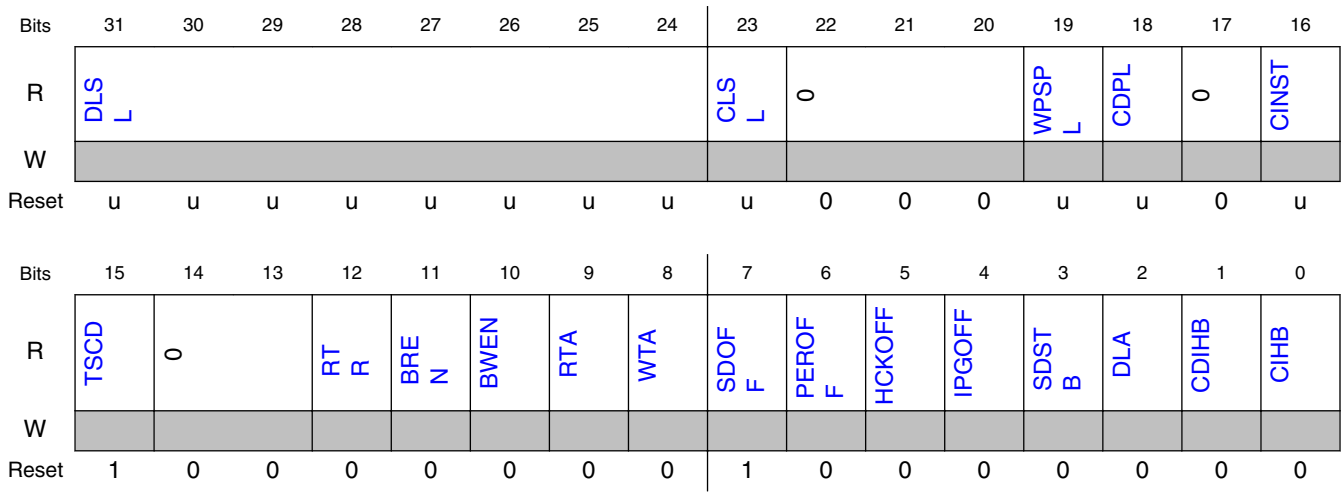
The host driver can get status of uSDHC from this 32-bit read only register.

The host driver can issue CMD0, CMD12, CMD13 (for memory) and CMD52 (for SDIO) when the DATA lines are busy during a data transfer. These commands can be issued when Command Inhibit (CMD) is set to zero. Other commands are issued when Command Inhibit (DATA) is set to zero. Possible changes to the SD Physical Specification may add other commands to this list in the future.

NOTE

The reset value of Present State register depends on board connectivity.

10.3.7.1.11.3 Diagram



10.3.7.1.11.4 Fields

Field	Function
31-24 DLSL	DATA[7:0] line signal level This status is used to check the DATA line level to recover from errors, and for debugging. This is especially useful in detecting the busy signal level from DATA0. The reset value is affected by the external pull-up / pull-down resistors. By default, the read value of this field field after reset is 8'b11110111, when DATA3 is pulled down and the other lines are pulled up. 00000000b - Data 0 line signal level 00000001b - Data 1 line signal level 00000010b - Data 2 line signal level 00000011b - Data 3 line signal level 00000100b - Data 4 line signal level 00000101b - Data 5 line signal level 00000110b - Data 6 line signal level 00000111b - Data 7 line signal level
23 CLSL	CMD line signal level This status is used to check the CMD line level to recover from errors, and for debugging. The reset value is affected by the external pull-up / pull-down resistor, by default, the read value of this field after reset is 1'b1, when the command line is pulled up.
22-20 —	Reserved
19 WPSPL	Write protect switch pin level The Write Protect switch is supported for memory and combo cards. This field reflects the inverted value of the WP pin of the card socket. A software reset does not affect this field. The reset value is affected by the external write protect switch. If the WP pin is not used, it should be tied low, so that the reset value of this field is high and write is enabled. 0b - Write protected (WP = 1) 1b - Write enabled (WP = 0)

Table continues on the next page...

Field	Function
18 CDPL	Card detect pin level This field reflects the inverse value of the CD_B pin for the card socket. Debouncing is not performed on this field. This field may be valid, but is not guaranteed, because of propagation delay. Use of this field is limited to testing because it must be debounced by software. A software reset does not affect this field. A write to the Force Event Register does not affect this field. The reset value is affected by the external card detection pin. This field shows the value on the CD_B pin (that is, when a card is inserted in the socket, it is 0 on the CD_B input, and consequently, the CDPL reads 1. 0b - No card present (CD_B = 1) 1b - Card present (CD_B = 0)
17 —	Reserved
16 CINST	Card inserted This field indicates whether a card has been inserted. The uSDHC module debounces this signal so that the host driver does not need to wait for it to stabilize. Changing from a 0 to 1 generates a Card Insertion interrupt in the Interrupt Status register. Changing from a 1 to 0 generates a Card Removal interrupt in the Interrupt Status register. A write to the Force Event Register does not affect this field. The Software Reset For All in the System Control register does not affect this field. A software reset does not affect this field. 0b - Power on reset or no card 1b - Card inserted
15 TSCD	Tape select change done This field indicates the delay setting is effective after write CLK_TUNE_CTRL_STATUS register. 0b - Delay cell select change is not finished. 1b - Delay cell select change is finished.
14-13 —	Reserved
12 RTR	Re-Tuning Request (only for SD3.0 SDR104 mode and EMMC HS200 mode) Host controller may request host driver to execute re-tuning sequence by setting this field when the data window is shifted by temperature drift and a tuned sampling point does not have a good margin to receive correct data. This field is cleared when a command is issued with setting Execute Tuning field in MIXER_CTRL register. Changing of this field from 0 to 1 generates Re-Tuning Event. See Interrupt status registers for more detail. This field isn't set to 1 if Sampling Clock Select in the MIXER_CTRL register is set to 0 (using fixed sampling clock). 0b - Fixed or well tuned sampling clock 1b - Sampling clock needs re-tuning
11 BREN	Buffer read enable This status field is used for non-DMA read transfers. The uSDHC module implements an internal buffer to transfer data efficiently. This read only flag indicates that valid data exists in the host side buffer. If this field is high, valid data greater than the watermark level exist in the buffer. A change of this field from 1 to 0 occurs when some reads from the buffer (read DATPORT (Base + 0x20)) are made and the buffer hasn't valid data greater than the watermark level. A change of this field from 0 to1 occurs when there is enough valid data ready in the buffer and the Buffer Read Ready interrupt has been generated and enabled. 0b - Read disable

Table continues on the next page...

Field	Function
	1b - Read enable
10 BWEN	Buffer write enable This status field is used for non-DMA write transfers. The uSDHC module implements an internal buffer to transfer data efficiently. This read only flag indicates if space is available for write data. If this field is 1, valid data greater than the watermark level can be written to the buffer. A change of this field from 1 to 0 occurs when some writes to the buffer (write DATPORT(Base + 0x20)) are made and the buffer hasn't valid space greater than the watermark level. A change of this field from 0 to 1 occurs when the buffer can hold valid data greater than the write watermark level and the Buffer Write Ready interrupt is generated and enabled. 0b - Write disable 1b - Write enable
9 RTA	Read transfer active This status field is used for detecting completion of a read transfer. This field is set for either of the following conditions: • After the end field of the read command • When writing a 1 to the Continue Request field in the Protocol Control register to restart a read transfer A transfer complete interrupt is generated when this field changes to 0. This field is cleared for either of the following conditions: • When the last data block as specified by block length is transferred to the System, that is, all data are read away from uSDHC internal buffer. • When all valid data blocks have been transferred from uSDHC internal buffer to the System and no current block transfers are being sent because of the Stop At Block Gap Request being set to 1. 0b - No valid data 1b - Transferring data
8 WTA	Write transfer active This status field indicates a write transfer is active. If this field is 0, it means no valid write data exists in uSDHC. This field is set in either of the following cases: • After the end field of the write command • When writing 1 to the Continue Request field in the Protocol Control register to restart a write transfer This field is cleared in either of the following cases: • After getting the CRC status of the last data block as specified by the transfer count (Single and Multiple) • After getting the CRC status of any block where data transmission is about to be stopped by a Stop At Block Gap Request During a write transaction, a Block Gap Event interrupt is generated when this field is changed to 0, as result of the Stop At Block Gap Request being set. This status is useful for the host driver in determining when to issue commands during Write Busy state. 0b - No valid data 1b - Transferring data
7 SDOFF	SD clock gated off internally This status field indicates that the SD clock is internally gated off, because of buffer over / under-run or read pause without read wait assertion, or the driver set FRC_SDCLK_ON field is 0 to stop the SD clock in idle status. This field is for the host driver to debug data transaction on the SD bus.

Table continues on the next page...

Field	Function
	0b - SD clock is active. 1b - SD clock is gated off.
6 PEROFF	IPG_PERCLK gated off internally This status field indicates that the IPG_PERCLK is internally gated off. This field is for the host driver to debug transaction on the SD bus. When IPG_CLK_SOFT_EN is cleared, IPG_PERCLK is gated off, otherwise IPG_PERCLK is always active. 0b - IPG_PERCLK is active. 1b - IPG_PERCLK is gated off.
5 HCKOFF	HCLK gated off internally This status field indicates that the HCLK is internally gated off. This field is for the host driver to debug during a data transfer. 0b - HCLK is active. 1b - HCLK is gated off.
4 IPGOFF	Peripheral clock gated off internally This status field indicates that the peripheral clock is internally gated off. This field is for the host driver to debug. 0b - Peripheral clock is active. 1b - Peripheral clock is gated off.
3 SDSTB	SD clock stable This status field indicates that the internal card clock is stable. This field is for the host driver to poll clock status when changing the clock frequency. It is recommended to clear FRC_SDCLK_ON field in System Control register to remove glitches on the card clock when the frequency is changing. Before changing clock divisor value (SDCLKFS or DVS), host driver should make sure the SDSTB field is high. 0b - Clock is changing frequency and not stable. 1b - Clock is stable.
2 DLA	Data line active This status field indicates whether one of the DATA lines on the SD bus is in use. In the case of read transactions: This status indicates if a read transfer is executing on the SD bus. Changes in this value from 1 to 0, between data blocks, generates a Block Gap Event interrupt in the Interrupt Status register. This field is set in either of the following cases: • After the end field of the read command • When writing a 1 to the Continue Request field in the Protocol Control register to restart a read transfer This field is cleared in either of the following cases: • When the end field of the last data block is sent from the SD bus to uSDHC. • When the Read Wait state is stopped by a Suspend command and the DATA2 line is released. The uSDHC module waits at the next block gap by driving Read Wait at the start of the interrupt cycle. If the Read Wait signal is already driven (data buffer cannot receive data), uSDHC can wait for a current block gap by continuing to drive the Read Wait signal. It is necessary to support Read Wait to use the suspend / resume function. This field remains 1 during Read Wait. In the case of write transactions: This status indicates that a write transfer is executing on the SD bus. Changes in this value from 1 to 0 generate a Transfer Complete interrupt in the Interrupt Status register.

Table continues on the next page...

Field	Function
	This field is set in either of the following cases: • After the end field of the write command • When writing to 1 to the Continue Request field in the Protocol Control register to continue a write transfer This field is cleared in either of the following cases: • When the SD card releases Write Busy of the last data block, uSDHC also detects if the output is not busy. If the SD card does not drive the busy signal after the CRC status is received, uSDHC assumes the card drive "Not Busy". • When the SD card releases write busy, prior to waiting for write transfer, and because of a Stop At Block Gap Request. In the case of command with busy pending: This status indicates that a busy state follows the command and the data line is in use. This field is cleared when the DATA0 line is released. 0b - DATA line inactive 1b - DATA line active
1 CDIHB	Command inhibit (DATA) This status field is generated if either the DAT Line Active or the Read Transfer Active is set to 1. If this field is 0, it indicates that uSDHC can issue the next SD / MMC Command. Commands with a busy signal belong to Command Inhibit (DATA) (for example. R1b, R5b type). Changing from 1 to 0 generates a Transfer Complete interrupt in the Interrupt Status register. **NOTE:** The SD host driver can save registers for a suspend transaction after this field has changed from 1 to 0. 0b - Can issue command that uses the DATA line 1b - Cannot issue command that uses the DATA line
0 CIHB	Command inhibit (CMD) If this status bit is 0, it indicates that the CMD line is not in use and uSDHC can issue a SD / MMC command using the CMD line. This field is set also immediately after the Transfer Type register is written. This field is cleared when the command response is received. Even if the Command Inhibit (DATA) is set to 1, commands using only the CMD line can be issued if this field is 0. Changing from 1 to 0 generates a Command Complete interrupt in the Interrupt Status register. If uSDHC cannot issue the command because of a command conflict error (see Command CRC Error) or because of a Command Not Issued By Auto CMD12 Error, this field remains 1 and the Command Complete is not set. The Status of issuing an auto CMD12 does not show on this field. 0b - Can issue command using only CMD line 1b - Cannot issue command

10.3.7.1.12 Protocol Control (PROT_CTRL)

10.3.7.1.12.1 Offset

Register	Offset
PROT_CTRL	28h

10.3.7.1.12.2 Function

There are three cases to restart the transfer after stop at the block gap. Which case is appropriate depends on whether uSDHC issues a Suspend command or the SD card accepts the Suspend command.

- If the host driver does not issue a Suspend command, the Continue request is used to restart the transfer.
- If the host driver issues a Suspend command and the SD card accepts it, a Resume command is used to restart the transfer.
- If the host driver issues a Suspend command and the SD card does not accept it, the Continue request is used to restart the transfer.

Any time stop at block gap request stops the data transfer, the host driver waits for a Transfer Complete (in the Interrupt Status register), before attempting to restart the transfer. When restarting the data transfer by Continue Request, the host driver clears the Stop At Block Gap Request before or simultaneously.

10.3.7.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	NON_EXACT_BLK_RD	BURST_LEN_EN			WECRM	WECINS	WECINT	Reserved			RD_DONE_NO_8CLK	IABG	RWCTL	CREQ	SABGREQ
W																
Reset	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						DMASEL		CDSS	CDTL	EMODE		D3CD	DTW		LCTL
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

10.3.7.1.12.4 Fields

Field	Function
31	Reserved
—	Always write as 0

Table continues on the next page...

Field	Function
30 NON_EXACT_BLK_RD	Non-exact block read Current block read is non-exact block read. It is only used for SDIO. 0b - The block read is exact block read. Host driver does not need to issue abort command to terminate this multi-block read. 1b - The block read is non-exact block read. Host driver needs to issue abort command to terminate this multi-block read.
29-27 BURST_LEN_EN	BURST length enable for INCR, INCR4 / INCR8 / INCR16, INCR4-WRAP / INCR8-WRAP / INCR16-WRAP This is used to enable / disable the burst length for the external AHB2AXI bridge. It is useful especially for INCR transfer because without burst length indicator, the AHB2AXI bridge does not know the burst length in advance. Without burst length indicator, AHB INCR transfers can only be converted to SINGLEs on the AXI side. 1xxb - Burst length is enabled for INCR4-WRAP / INCR8-WRAP / INCR16-WRAP. x1xb - Burst length is enabled for INCR4 / INCR8 / INCR16. xx1b - Burst length is enabled for INCR.
26 WECRM	Wakeup event enable on SD card removal This field enables a wakeup event, via a card removal, in the Interrupt Status register. FN_WUS (Wake Up Support) in CIS does not affect this field. When this field is set, the Card Removal Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Removal Status and uSDHC interrupt. 0b - Disables wakeup event enable on SD card removal 1b - Enables wakeup event enable on SD card removal
25 WECINS	Wakeup event enable on SD card insertion This field enables a wakeup event, via a card insertion, in the Interrupt Status register. FN_WUS (Wake Up Support) in CIS does not affect this field. When this field is set, the Card Insertion Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Insertion Status and uSDHC interrupt. 0b - Disable wakeup event enable on SD card insertion 1b - Enable wakeup event enable on SD card insertion
24 WECINT	Wakeup event enable on card interrupt This field enables a wakeup event, via a card interrupt, in the Interrupt Status register. This field can be set to 1 if FN_WUS (Wake Up Support) in CIS is set to 1. When this field is set, the Card Interrupt Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Interrupt Status and uSDHC interrupt. 0b - Disables wakeup event enable on card interrupt 1b - Enables wakeup event enable on card interrupt
23-21 —	Reserved Always write as 3'b100
20 RD_DONE_NO_8CLK	Read performed number 8 clock According to the SD/MMC spec, for read data transaction, 8 clocks are needed after the end field of the last data block. So, by default(RD_DONE_NO_8CLK=0), 8 clocks are active after the end field of the last read data transaction. However, these 8 clocks should not be active if user wants to use stop at block gap (include the auto stop at block gap in boot mode) feature for read and the RWCTL field (bit18) is not enabled. In this case, software should set RD_DONE_NO_8CLK to avoid these 8 clocks. Otherwise, the device might send extra data to uSDHC while uSDHC ignores these data. In a summary, this field should be set only if the use case needs to use stop at block gap feature while the device can't support the read wait feature.

Table continues on the next page...

Field	Function
19 IABG	Interrupt at block gap This field is valid only in 4-bit mode, of the SDIO card, and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer. Setting to 0 disables interrupt detection during a multiple block transfer. If the SDIO card cannot signal an interrupt during a multiple block transfer, this field should be set to 0 to avoid an inadvertent interrupt. When the host driver detects an SDIO card insertion, it sets this field according to the CCCR of the card. 0b - Disables interrupt at block gap 1b - Enables interrupt at block gap
18 RWCTL	Read wait control The read wait function is optional for SDIO cards. If the card supports read wait, set this field to enable use of the read wait protocol to stop read data using the DATA2 line. Otherwise, uSDHC has to stop the SD clock to hold read data, which restricts commands generation. When the host driver detects an SDIO card insertion, it sets this field according to the CCCR of the card. If the card does not support read wait, this field should never be set to 1; otherwise, DATA line conflicts might occur. If this field is set to 0, stop at block gap during read operation is also supported, but uSDHC stops the SD clock to pause reading operation. 0b - Disables read wait control and stop SD clock at block gap when SABGREQ field is set 1b - Enables read wait control and assert read wait without stopping SD clock at block gap when SABGREQ field is set
17 CREQ	Continue request This field is used to restart a transaction which was stopped using the stop at block gap request. When a suspend operation is not accepted by the card, it is also by setting this field to restart the paused transfer. To cancel stop at the block gap, set stop at block gap request to 0 and set this field to 1 to restart the transfer. The uSDHC module automatically clears this field, therefore it is not necessary for the host driver to set this field to 0. If both stop at block gap request and this field are 1, the continue request is ignored. 0b - No effect 1b - Restart
16 SABGREQ	Stop at block gap request This field is used to stop executing a transaction at the next block gap for both DMA and non-DMA transfers. Until the transfer complete is set to 1, indicating a transfer completion, the host driver leaves this field set to 1. Clearing both the stop at block gap request and continue request does not cause the transaction to restart. Read Wait is used to stop the read transaction at the block gap. The uSDHC module supports the stop at block gap request for write transfers, but for read transfers it requires that the SDIO card support read wait. Therefore, the host driver does not set this field during read transfers unless the SDIO card supports Read Wait and has set the read wait control to 1; otherwise, uSDHC stops the SD bus clock to pause the read operation during block gap. In the case of write transfers in which the host driver writes data to the Data Port register, the host driver sets this field after all block data is written. If this field is set to 1, the host driver does not write data to the Data Port register after a block is sent. Once this field is set, the host driver does not clear this field before the Transfer Complete field in Interrupt Status register is set, otherwise uSDHC's behavior is undefined. This field effects read transfer active, write transfer active, DATA Line Active and Command Inhibit (DATA) in the Present State register. 0b - Transfer 1b - Stop
15-10 —	Reserved
9-8 DMASEL	DMA select This field is valid while DMA (SDMA or ADMA) is enabled and selects the DMA operation.

Table continues on the next page...

Field	Function
	00b - No DMA or simple DMA is selected. 01b - ADMA1 is selected. 10b - ADMA2 is selected. 11b - Reserved
7 CDSS	Card detect signal selection This field selects the source for the card detection. 0b - Card detection level is selected (for normal purpose). 1b - Card detection test level is selected (for test purpose).
6 CDTL	Card detect test level This bit is enabled while the card detection signal selection is set to 1 and it indicates card insertion. 0b - Card detect test level is 0, no card inserted 1b - Card detect test level is 1, card inserted
5-4 EMODE	Endian mode The uSDHC module supports all three endian modes in data transfer. See Data buffer for more details. 00b - Big endian mode 01b - Half word big endian mode 10b - Little endian mode 11b - Reserved
3 D3CD	DATA3 as card detection pin If this field is set, DATA3 should be pulled down to act as a card detection pin. Be cautious when using this feature, because DATA3 is also a chip-select for the SPI mode. A pull-down on this pin and CMD0 might set the card into the SPI mode, which uSDHC does not support. 0b - DATA3 does not monitor card insertion 1b - DATA3 as card detection pin
2-1 DTW	Data transfer width This field selects the data width of the SD bus for a data transfer. The host driver sets it to match the data width of the card. Possible data transfer width is 1-bit, 4-bits or 8-bits. 00b - 1-bit mode 01b - 4-bit mode 10b - 8-bit mode 11b - Reserved
0 LCTL	LED control This field, fully controlled by the host driver, is used to caution the user not to remove the card while the card is being accessed. If the software is going to issue multiple SD commands, this field can be set during all these transactions. It is not necessary to change for each transaction. When the software issues multiple SD commands, setting the field once before the first command is sufficient: it is not necessary to reset the bit between commands. 0b - LED off 1b - LED on

10.3.7.1.13 System Control (SYS_CTRL)

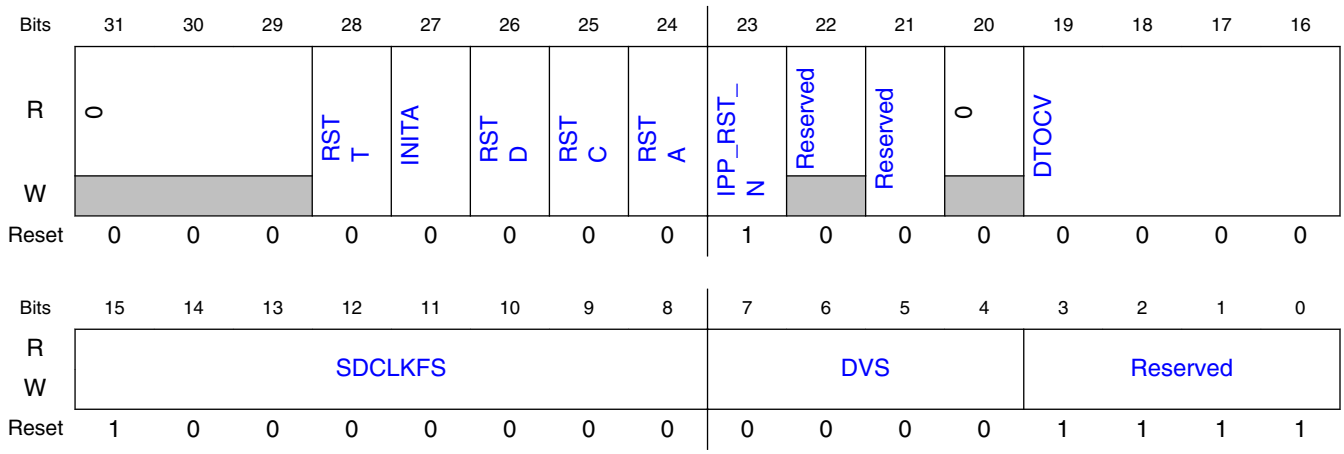
10.3.7.1.13.1 Offset

Register	Offset
SYS_CTRL	2Ch

10.3.7.1.13.2 Function

This register provides control of the system. See detail in the field description.

10.3.7.1.13.3 Diagram



10.3.7.1.13.4 Fields

Field	Function
31-29 —	Reserved
28 RSTT	Reset tuning When set this field to 1, it resets tuning circuit. After tuning circuits are reset, bit value is 0. Clearing execute_tuning field in AUTOCMD12_ERR_STATUS also sets this field to 1 to reset tuning circuit
27 INITA	Initialization active When this field is set, 80 SD-clocks are sent to the card. After the 80 clocks are sent, this field is self cleared. This field is very useful during the card power-up period when 74 SD-clocks are needed and the clock auto gating feature is enabled. Writing 1 to this bit when this field is already 1 has no effect. Writing 0 to this field at any time has no effect. When either of the CIHB and CDIHB fields in the Present State register are set, writing 1 to this field is ignored (that is, when command line or data lines are active, write to this field is not allowed). On the other-hand, when this field is set, that is, during initialization active period, it is allowed to issue command, and the command bit stream appears on the CMD pad after all 80 clock cycles are done. So, when this command ends, the driver can make sure the 80 clock cycles are sent out. This is very useful when the driver needs to send 80 cycles to the card and does not want to wait till this field is self cleared.
26	Software reset for data line

Table continues on the next page...

Field	Function
RSTD	Only part of the data circuit is reset. DMA circuit is also reset. After this field is set, the software waits for self-clear. The following registers and bits are cleared by this field: • Data Port register • Buffer is cleared and initialized • Present State register • Buffer read enable • Buffer write enable • Read transfer active • Write transfer active • DATA line active • Command Inhibit (DATA) Protocol Control register • Continue request • Stop At Block Gap Request Interrupt Status register • Buffer read ready • Buffer write ready • DMA interrupt • Block gap event • Transfer complete **NOTE:** When reset, the software must make sure there is no incomplete data transferring. If there is data transfer going on, the software needs to wait TC or DC INT_STATUS register is set. 0b - No reset 1b - Reset
25 RSTC	Software reset for CMD line Only part of the command circuit is reset. After this field is set, the software waits for self-clear. The following registers and bits are cleared by this field: • Present State Register Command Inhibit (CMD) • Interrupt Status register Command Complete 0b - No reset 1b - Reset
24 RSTA	Software reset for all This reset effects the entire host controller except for the card detection circuit. Register bits of type ROC, RW, RW1C, RWAC are cleared. During its initialization, the host driver is set this field to 1 to reset uSDHC. The uSDHC module resets this field to 0 when the capabilities registers are valid and the host driver can read them. Additional use of Software Reset For All does not affect the value of the capabilities registers. After this field is set, it is recommended that the host driver reset the external card and re-initialize it. After this field is set, the software should wait for self-clear. In tuning process, after every CMD19 is finished, this field is set to retest uSDHC. **NOTE:** When reset, the software must make sure there is no incomplete data transferring. If there is data transfer going on, the software needs to wait TC or DC INT_STATUS register is set. 0b - No reset 1b - Reset
23 IPP_RST_N	Hardware reset This register's value is output to card through pad directly to hardware reset pin of the card if the card supports this feature.
22 —	Reserved
21	Reserved

Table continues on the next page...

Field	Function
—	
20 —	Reserved
19-16 DTOCV	Data timeout counter value This value determines the interval by which DAT line timeouts are detected. See the Data Timeout Error field in the Interrupt Status register for information on factors that dictate time-out generation. Time-out clock frequency is generated by dividing the base clock SDCLK value by this value. The host driver can clear the Data Timeout Error Status Enable (in the Interrupt Status Enable register) to prevent inadvertent time-out events. 0000b - SDCLK x 2^{14} 0001b - SDCLK x 2^{15} 0010b - SDCLK x 2^{16} 0011b - SDCLK x 2^{17} 0100b - SDCLK x 2^{18} 0101b - SDCLK x 2^{19} 0110b - SDCLK x 2^{20} 0111b - SDCLK x 2^{21} 1000b - SDCLK x 2^{22} 1001b - SDCLK x 2^{23} 1010b - SDCLK x 2^{24} 1011b - SDCLK x 2^{25} 1100b - SDCLK x 2^{26} 1101b - SDCLK x 2^{27} 1110b - SDCLK x 2^{28} 1111b - SDCLK x 2^{29}
15-8 SDCLKFS	SDCLK frequency select This register is used to select the frequency of the SDCLK pin. The frequency is not programmed directly, rather this register holds the prescaler (this register) and divisor (next register) of the Base Clock Frequency register. In Single Data Rate mode (DDR_EN field of MIXERCTRL is '0') Only the following settings are allowed: 80h) Base clock divided by 256 40h) Base clock divided by 128 20h) Base clock divided by 64 10h) Base clock divided by 32 08h) Base clock divided by 16 04h) Base clock divided by 8 02h) Base clock divided by 4 01h) Base clock divided by 2 00h) Base clock divided by 1 While in Dual Data Rate mode (DDR_EN field of MIXERCTRL is '1') Only the following settings are allowed: 80h) Base clock divided by 512 40h) Base clock divided by 256 20h) Base clock divided by 128

Table continues on the next page...

Field	Function
	10h) Base clock divided by 64 08h) Base clock divided by 32 04h) Base clock divided by 16 02h) Base clock divided by 8 01h) Base clock divided by 4 00h) Base clock divided by 2 When the software changes the DDR_EN field, SDCLKFS might need to be changed also. In Single Data Rate mode, setting 00h bypasses the frequency prescaler of the SD clock. Multiple bits must not be set, or the behavior of this prescaler is undefined. The two default divider values can be calculated by the frequency of ipg_perclk and the following Divisor bits. The frequency of SDCLK is set by the following formula: Clock Frequency = (Base Clock) / (prescaler x divisor) For example, in Single Data Rate mode, if the Base Clock Frequency is 96 MHz, and the target frequency is 25 MHz, then choosing the prescaler value of 01h and divisor value of 1h yields 24 MHz, which is the nearest frequency less than or equal to the target. Similarly, to approach a clock value of 400 kHz, the prescaler value of 08h and divisor value of eh yields the exact clock value of 400 kHz. The reset value of this field is 80h, so if the input Base Clock (ipg_perclk) is about 96 MHz, the default SD clock after reset is 375 kHz. According to the SD Physical Specification Version 1.1 and the SDIO Card Specification Version 1.2, the maximum SD clock frequency is 50 MHz and is never exceed this limit. Before changing clock divisor value (SDCLKFS or DVS), host driver should make sure the SDSTB field is high. If setting SDCLKFS and DVS can generate the same clock frequency,(for example, in SDR mode, SDCLKFS = 01h is same as DVS = 01h.) SDCLKFS is highly recommended.
7-4 DVS	Divisor This register is used to provide a more exact divisor to generate the desired SD clock frequency. Note the divider can even support odd divisors without deterioration of duty cycle. Before changing clock divisor value (SDCLKFS or DVS), Host driver should make sure the SDSTB field is high. The settings are as follows: 0000b - Divide-by-1 0001b - Divide-by-2 1110b - Divide-by-15 1111b - Divide-by-16
3-0 —	Reserved Always write as 1.

10.3.7.1.14 Interrupt Status (INT_STATUS)

10.3.7.1.14.1 Offset

Register	Offset
INT_STATUS	30h

10.3.7.1.14.2 Function

An interrupt is generated when the normal interrupt signal enable is enabled and at least one of the status fields is set to 1. For all fields, writing 1 to a bit clears it; writing to 0 keeps the bit unchanged. More than one status can be cleared with a single register write. For card interrupt, before writing 1 to clear, it is required that the card stops asserting the interrupt, meaning that when the card driver services the interrupt condition; otherwise, the CINT field is asserted again.

The table below shows the relationship between the command timeout error and the command complete.

Table 10-39. uSDHC status for command timeout error/command complete bit combinations

Command CRC Error	Command timeout error	Meaning of the status
0	0	X
X	1	Response not received within 64 SDCLK cycles
1	0	Response received

The table below shows the relationship between the transfer complete and the data timeout error.

Table 10-40. uSDHC status for data timeout error/transfer complete bit combinations

Transfer complete	Data timeout error	Meaning of the status
0	0	X
0	1	Timeout occurred during transfer
1	X	Data transfer complete

The table below shows the relationship between the command CRC error and command timeout error.

Table 10-41. uSDHC status for command CRC error/command timeout error bit combinations

Command complete	Command timeout error	Meaning of the status
0	0	No error

Table continues on the next page...

Table 10-41. uSDHC status for command CRC error/command timeout error bit combinations (continued)

Command complete	Command timeout error	Meaning of the status
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

10.3.7.1.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			DMAE	0	TNE	0	AC12E	0	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE
W				W1C		W1C		W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CQI	TP	RTE	0			CINT	CRM	CINS	BRR	BWR	DINT	BGE	TC	CC
W		W1C	W1C	W1C				W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.3.7.1.14.4 Fields

Field	Function
31-29 —	Reserved
28 DMAE	DMA error Occurs when an Internal DMA transfer has failed. This field is set to 1, when some error occurs in the data transfer. This error can be caused by either simple DMA or ADMA, depending on which DMA is in use. The value in DMA System Address register is the next fetch address where the error occurs. Because any error corrupts the whole data block, the host driver restarts the transfer from the corrupted block boundary. The address of the block boundary can be calculated either from the current DS_ADDR value or from the remaining number of blocks and the block size. 0b - No error 1b - Error
27 —	Reserved
26 TNE	Tuning error: (only for SD3.0 SDR104 mode and EMMC HS200 mode)

Table continues on the next page...

Field	Function
	This field is set when an unrecoverable error is detected in a tuning circuit. By detecting Tuning Error, host driver needs to abort a command executing and perform tuning.
25 —	Reserved
24 AC12E	Auto CMD12 error Occurs when detecting that one of the fields in the Auto CMD12 Error Status register has changed from 0 to 1. This field is set to 1, not only when the errors in Auto CMD12 occur, but also, when the Auto CMD12 is not executed due to the previous command error. 0b - No error 1b - Error
23 —	Reserved
22 DEBE	Data end bit error Occurs either when detecting 0 at the end field position of read data that uses the DATA line, or at the end field position of the CRC. This field is not asserted in tuning process. 0b - No error 1b - Error
21 DCE	Data CRC error Occurs when detecting a CRC error when transferring read data that uses the DATA line, or when detecting the Write CRC status having a value other than 010. This field is not asserted in tuning process. 0b - No error 1b - Error
20 DTOE	Data timeout error Occurs when detecting one of following time-out conditions. • Busy time-out for R1b, R5b type • Busy time-out after Write CRC status • Read Data time-out. This field is not asserted in tuning process. 0b - No error 1b - Time out
19 CIE	Command index error Occurs if a command index error occurs in the command response. This field is not asserted in tuning process. 0b - No error 1b - Error
18 CEBE	Command end bit error Occurs when detecting that the end field of a command response is 0. This field is not asserted in tuning process. 0b - No error 1b - End bit error generated
17	Command CRC error

Table continues on the next page...

Field	Function
CCE	Command CRC Error is generated in two cases. • If a response is returned and the Command Timeout Error is set to 0 (indicating no time-out), this field is set when detecting a CRC error in the command response. • The uSDHC module detects a CMD line conflict by monitoring the CMD line when a command is issued. If uSDHC drives the CMD line to 1, but detects 0 on the CMD line at the next SDCLK edge, then uSDHC aborts the command (Stop driving CMD line) and set this bit to 1. The Command Timeout Error should also be set to 1 to distinguish CMD line conflict. This field is not asserted in tuning process. 0b - No error 1b - CRC error generated
16 CTOE	Command timeout error Occurs only if no response is returned within 64 SDCLK cycles from the end field of the command. If uSDHC detects a CMD line conflict, in which case a Command CRC Error is also be set (as shown in Interrupt Status (INT_STATUS)), this field is set without waiting for 64 SDCLK cycles. This is because the command is aborted by uSDHC. This field is not asserted in tuning process. 0b - No error 1b - Time out
15 —	Reserved
14 CQI	Command queuing interrupt This interrupt is asserted when at least one of the bits in CQIS register is set. This interrupt is cleared only by clearing the source interrupt in CQIS register.
13 TP	Tuning pass:(only for SD3.0 SDR104 mode and EMMC HS200 mode) Current CMD19 transfer is done successfully. That is, current sampling point is correct.
12 RTE	Re-tuning event: (only for SD3.0 SDR104 mode and EMMC HS200 mode) This status is set if Re-Tuning Request in the Present State register changes from 0 to 1. host controller requests host driver to perform re-tuning for next data transfer. Current data transfer (not large block count) can be completed without re-tuning. 0b - Re-tuning is not required. 1b - Re-tuning should be performed.
11-9 —	Reserved
8 CINT	Card interrupt This status field is set when an interrupt signal is detected from the external card. In 1-bit mode, uSDHC detects the Card Interrupt without the SD clock to support wakeup. In 4-bit mode, the card interrupt signal is sampled during the interrupt cycle, so the interrupt from card can only be sampled during interrupt cycle, introducing some delay between the interrupt signal from the SDIO card and the interrupt to the host system. Writing this field to 1 can clear this field, but as the interrupt source from the SDIO card does not clear, this field is set again. to clear this field, it is required to reset the interrupt source from the external card followed by a writing 1 to this field. When this status has been set, and the host driver needs to service this interrupt, the Card Interrupt Signal Enable in the Interrupt Signal Enable register should be 0 to stop driving the interrupt signal to the host system. After completion of the card interrupt service (It should reset the interrupt sources in the SDIO card and the interrupt signal might not be asserted), write 1 to clear this field, set the Card Interrupt Signal Enable to 1, and start sampling the interrupt signal again. 0b - No card interrupt

Table continues on the next page...

Field	Function
	1b - Generate card interrupt
7 CRM	Card removal This status field is set if the Card Inserted field in the Present State register changes from 1 to 0. When the host driver writes this field to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state might possibly be changed when the host driver clears this field and the interrupt event might not be generated. When this field is cleared, it is set again if no card is inserted. to leave it cleared, clear the Card Removal Status Enable field in Interrupt Status Enable register. 0b - Card state unstable or inserted 1b - Card removed
6 CINS	Card insertion This status field is set if the Card Inserted field in the Present State register changes from 0 to 1. When the host driver writes this field to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state might possibly be changed when the host driver clears this field and the interrupt event might not be generated. When this field is cleared, it is set again if a card is inserted. to leave it cleared, clear the Card Inserted Status Enable field in Interrupt Status Enable register. 0b - Card state unstable or removed 1b - Card inserted
5 BRR	Buffer read ready This status field is set if the Buffer Read Enable field, in the Present State register, changes from 0 to 1. See the Buffer Read Enable field in the Present State register for additional information. This field indicates that cmd19 is finished in tuning process. 0b - Not ready to read buffer 1b - Ready to read buffer
4 BWR	Buffer write ready This status field is set if the Buffer Write Enable field, in the Present State register, changes from 0 to 1. See the Buffer Write Enable field in the Present State register for additional information. 0b - Not ready to write buffer 1b - Ready to write buffer
3 DINT	DMA interrupt Occurs only when the internal DMA finishes the data transfer successfully. Whenever errors occur during data transfer, this field does not be set. Instead, the DMAE field is set. Either Simple DMA or ADMA finishes data transferring, this field is set. 0b - No DMA interrupt 1b - DMA interrupt is generated.
2 BGE	Block gap event If the Stop At Block Gap Request field in the Protocol Control register is set, this field is set when a read or write transaction is stopped at a block gap. If Stop At Block Gap Request is not set to 1, this field is not set to 1. In the case of a Read Transaction: This field is set at the falling edge of the DATA Line Active Status (When the transaction is stopped at SD bus timing). The Read Wait must be supported to use this function. In the case of Write Transaction: This field is set at the falling edge of Write Transfer Active Status (After getting CRC status at SD Bus timing). 0b - No block gap event 1b - Transaction stopped at block gap

Table continues on the next page...

Field	Function
1 TC	Transfer complete This field is set when a read or write transfer is completed. In the case of a Read Transaction: This field is set at the falling edge of the Read Transfer Active Status. There are two cases in which this interrupt is generated. The first is when a data transfer is completed as specified by the data length (after the last data has been read to the host system). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request field in the Protocol Control register (after valid data has been read to the host system). In the case of a Write Transaction: This field is set at the falling edge of the DATA Line Active Status. There are two cases in which this interrupt is generated. The first is when the last data is written to the SD card as specified by the data length and the busy signal is released. The second is when data transfers are stopped at the block gap, by setting the Stop At Block Gap Request field in the Protocol Control register, and the data transfers are completed. (after valid data is written to the SD card and the busy signal released). In the case of a command with busy, this field is set when busy is deasserted. This field is not asserted in tuning process. 0b - Transfer does not complete 1b - Transfer complete
0 CC	Command complete This field is set when you receive the end field of the command response (except auto CMD12). See the Command Inhibit (CMD) in the Present State register. This field is not asserted in tuning process. 0b - Command not complete 1b - Command complete

10.3.7.1.15 Interrupt Status Enable (INT_STATUS_EN)

10.3.7.1.15.1 Offset

Register	Offset
INT_STATUS_EN	34h

10.3.7.1.15.2 Function

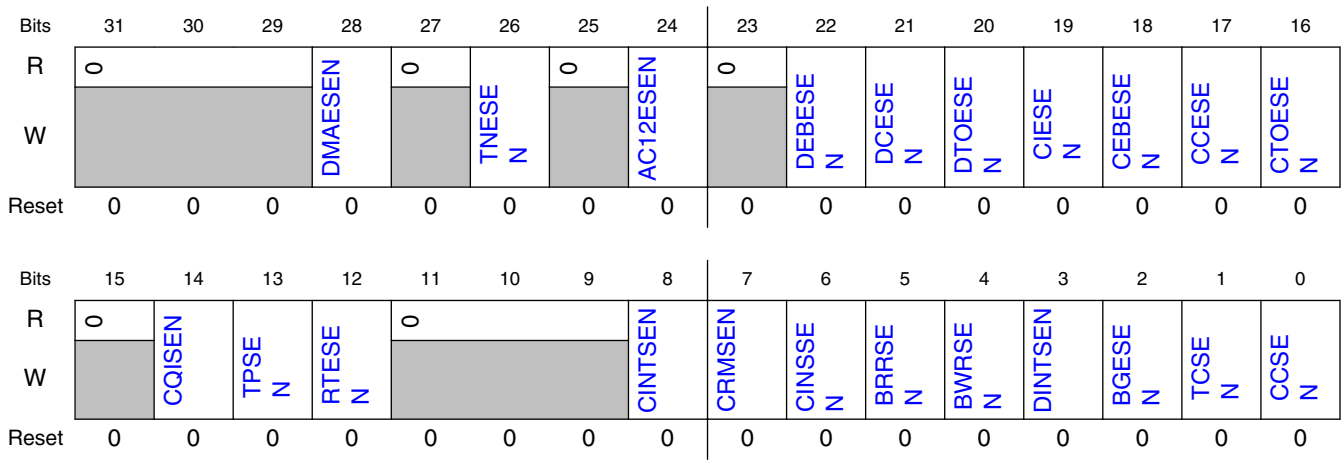
Setting the fields in this register to 1 enables the corresponding Interrupt Status to be set by the specified event. If any field is cleared, the corresponding Interrupt Status field is also cleared (that is, when the field in this register is cleared, the corresponding field in Interrupt Status register is always 0).

- Depending on IABG field setting, uSDHC might be programmed to sample the card interrupt signal during the interrupt period and hold its value in the flip-flop. There

are some delays on the Card Interrupt, asserted from the card, to the time the host system is informed.

- To detect a CMD line conflict, the host driver must set both Command Timeout Error Status Enable and Command CRC Error Status Enable to 1.

10.3.7.1.15.3 Diagram



10.3.7.1.15.4 Fields

Field	Function
31-29 —	Reserved
28 DMAESEN	DMA error status enable 0b - Masked 1b - Enabled
27 —	Reserved
26 TNESEN	Tuning error status enable 0b - Masked 1b - Enabled
25 —	Reserved
24 AC12ESEN	Auto CMD12 error status enable 0b - Masked 1b - Enabled
23 —	Reserved
22 DEBESEN	Data end bit error status enable 0b - Masked 1b - Enabled

Table continues on the next page...

Field	Function
21 DCESEN	Data CRC error status enable 0b - Masked 1b - Enabled
20 DTOESEN	Data timeout error status enable 0b - Masked 1b - Enabled
19 CIESEN	Command index error status enable 0b - Masked 1b - Enabled
18 CEBESEN	Command end bit error status enable 0b - Masked 1b - Enabled
17 CCESEN	Command CRC error status enable 0b - Masked 1b - Enabled
16 CTOESEN	Command timeout error status enable 0b - Masked 1b - Enabled
15 —	Reserved
14 CQISEN	Command queuing status enable 0b - Masked 1b - Enabled
13 TPSEN	Tuning pass status enable 0b - Masked 1b - Enabled
12 RTESEN	Re-tuning event status enable 0b - Masked 1b - Enabled
11-9 —	Reserved
8 CINTSEN	Card interrupt status enable If this field is set to 0, uSDHC clears the interrupt request to the system. The Card Interrupt detection is stopped when this field is cleared and restarted when this field is set to 1. The host driver should clear the Card Interrupt Status Enable before servicing the Card Interrupt and should set this field again after all interrupt requests from the card are cleared to prevent inadvertent interrupts. 0b - Masked 1b - Enabled
7 CRMSEN	Card removal status enable 0b - Masked 1b - Enabled
6 CINSSEN	Card insertion status enable 0b - Masked 1b - Enabled
5 BRRSEN	Buffer read ready status enable 0b - Masked 1b - Enabled
4	Buffer write ready status enable 0b - Masked

Table continues on the next page...

Field	Function
BWRSEN	1b - Enabled
3 DINTSEN	DMA interrupt status enable 0b - Masked 1b - Enabled
2 BGESEN	Block gap event status enable 0b - Masked 1b - Enabled
1 TCSEN	Transfer complete status enable 0b - Masked 1b - Enabled
0 CCSEN	Command complete status enable 0b - Masked 1b - Enabled

10.3.7.1.16 Interrupt Signal Enable (INT_SIGNAL_EN)

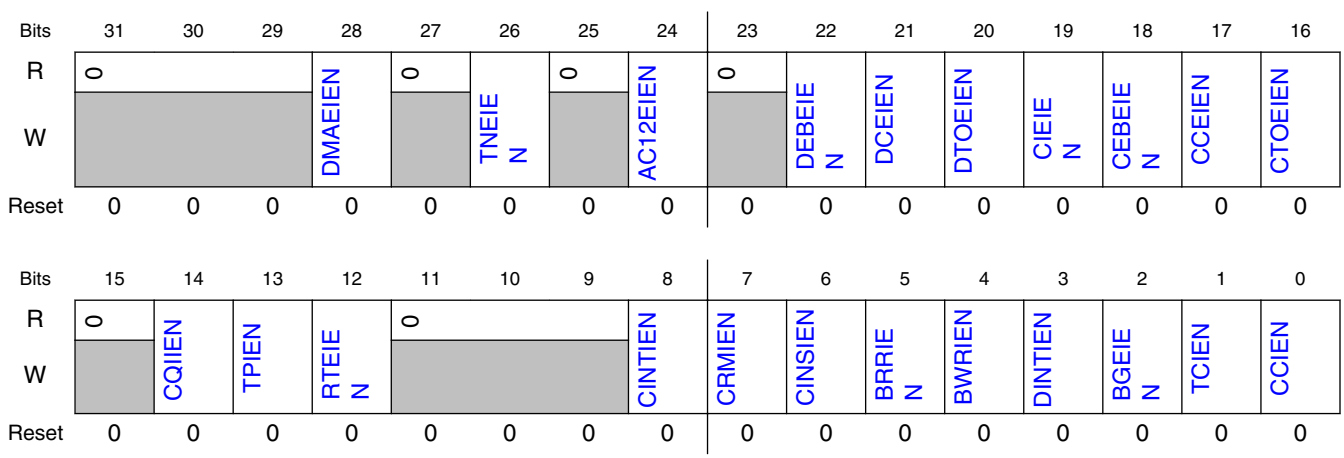
10.3.7.1.16.1 Offset

Register	Offset
INT_SIGNAL_EN	38h

10.3.7.1.16.2 Function

This register is used to select which interrupt status is indicated to the host system as the interrupt. These status fields all share the same interrupt line. Setting any of these fields to 1 enables interrupt generation. The corresponding Status register field generates an interrupt when the corresponding interrupt signal enable field is set.

10.3.7.1.16.3 Diagram



10.3.7.1.16.4 Fields

Field	Function
31-29 —	Reserved
28 DMAEIEN	DMA error interrupt enable 0b - Masked 1b - Enable
27 —	Reserved
26 TNEIEN	Tuning error interrupt enable 0b - Masked 1b - Enabled
25 —	Reserved
24 AC12EIEN	Auto CMD12 error interrupt enable 0b - Masked 1b - Enabled
23 —	Reserved
22 DEBEIEN	Data end bit error interrupt enable 0b - Masked 1b - Enabled
21 DCEIEN	Data CRC error interrupt enable 0b - Masked 1b - Enabled
20 DTOEIEN	Data timeout error interrupt enable 0b - Masked 1b - Enabled
19 CIEIEN	Command index error interrupt enable 0b - Masked 1b - Enabled
18 CEBEIEN	Command end bit error interrupt enable 0b - Masked 1b - Enabled
17 CCEIEN	Command CRC error interrupt enable 0b - Masked 1b - Enabled
16 CTOEIEN	Command timeout error interrupt enable 0b - Masked 1b - Enabled
15 —	Reserved
14 CQIIEN	Command queuing signal enable 0b - Masked 1b - Enabled

Table continues on the next page...

Field	Function
13 TPIEN	Tuning pass interrupt enable 0b - Masked 1b - Enabled
12 RTEIEN	Re-tuning event interrupt enable 0b - Masked 1b - Enabled
11-9 —	Reserved
8 CINTIEN	Card interrupt enable 0b - Masked 1b - Enabled
7 CRMIEN	Card removal interrupt enable 0b - Masked 1b - Enabled
6 CINSIEN	Card insertion interrupt enable 0b - Masked 1b - Enabled
5 BRRIEN	Buffer read ready interrupt enable 0b - Masked 1b - Enabled
4 BWRIEN	Buffer write ready interrupt enable 0b - Masked 1b - Enabled
3 DINTIEN	DMA interrupt enable 0b - Masked 1b - Enabled
2 BGEIEN	Block gap event interrupt enable 0b - Masked 1b - Enabled
1 TCIEN	Transfer complete interrupt enable 0b - Masked 1b - Enabled
0 CCIEN	Command complete interrupt enable 0b - Masked 1b - Enabled

10.3.7.1.17 Auto CMD12 Error Status (AUTOCMD12_ERR_STATUS)

10.3.7.1.17.1 Offset

Register	Offset
AUTOCMD12_ERR_STATUS	3Ch

10.3.7.1.17.2 Function

When the Auto CMD12 Error Status field in the Status register is set, the host driver checks this register to identify what kind of error the Auto CMD12 / CMD 23 indicated. Auto CMD23 errors are indicated in field 04-01. This register is valid only when the Auto CMD12 Error status field is set.

The table below shows the relationship between the Auto CMGD12 CRC Error and the Auto CMD12 Command Timeout Error.

Table 10-42. Relationship between command CRC error and command timeout error for auto CMD12

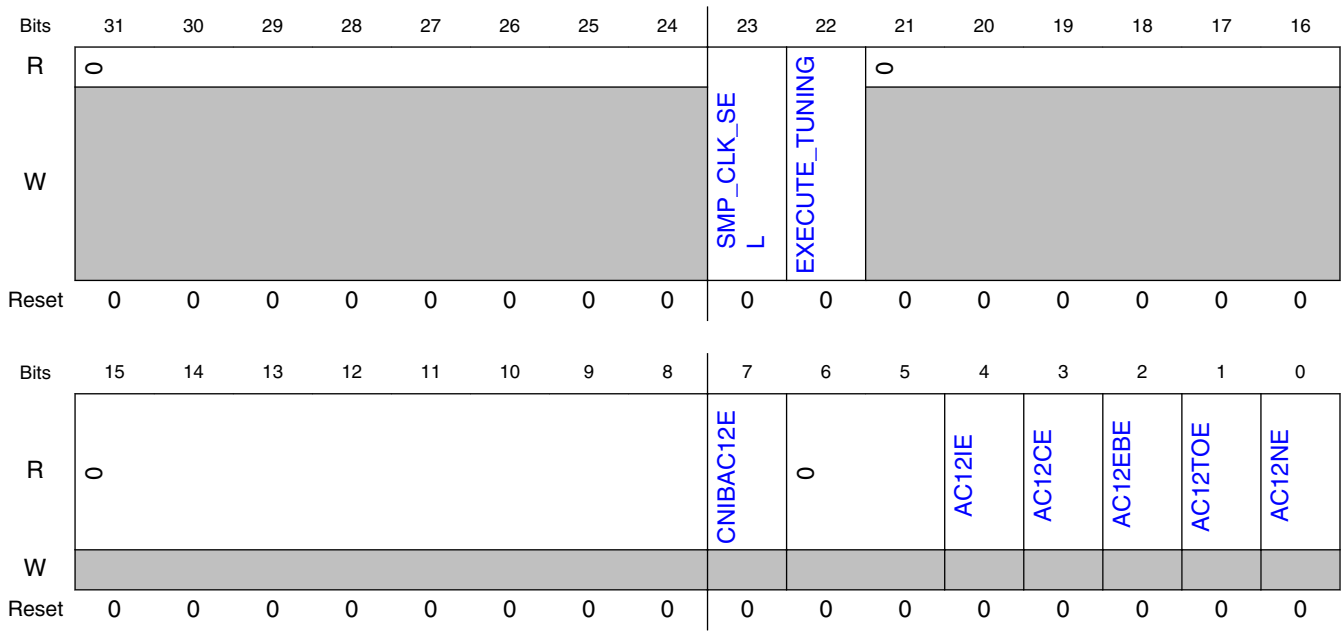
Auto CMD12 CRC error	Auto CMD12 timeout error	Type of error
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

Changes in Auto CMD12 Error Status register can be classified in three scenarios:

- When uSDHC is going to issue an Auto CMD12
  - Set field 0 to 1 if the Auto CMD12 cannot be issued due to an error in the previous command
  - Set field 0 to 0 if the Auto CMD12 is issued
- At the end field of an Auto CMD12 response
  - Check errors correspond to fields 1-4
  - Set fields 1-4 corresponding to detected errors
  - Clear fields 1-4 corresponding to detected errors
- Before reading the Auto CMD12 Error Status field 7
  - Set field 7 to 1 if there is a command that can't be issued
  - Clear field 7 if there is no command to issue

The timing for generating the Auto CMD12 Error and writing to the Command register are asynchronous. After that, field 7 is sampled when the driver is not writing to the Command register. So, it is suggested to read this register only when the AC12E field in Interrupt Status register is set. An Auto CMD12 Error Interrupt is generated when one of the error fields (0-4) is set to 1. The Command Not Issued By Auto CMD12 Error does not generate an interrupt.

10.3.7.1.17.3 Diagram



10.3.7.1.17.4 Fields

Field	Function
31-24 —	Reserved
23 SMP_CLK_SEL	Sample clock select When std_tuning_en field is set, this field is used to select sampling clock to receive CMD and DATA. Otherwise, this field is reserved. This field is set by ty tuning procedure and valid after the completion of tuning(When Execute Tuning is cleared). Setting 1 means that tuning is completed successfully and setting 0 means that tuning is failed. Writing 1 to this field is meaningless and ignored. A tuning circuit is reset by writing to 0. This field can be cleared with setting Execute Tuning. Once the tuning circuit is reset, it takes time to complete tuning sequence. Therefore, host driver should keep this field to 1 to perform re-tuning sequence to complete re-tuning sequence in a short time. Change of this field is not allowed while the Host controller us receiving response or a read data block. 0b - Fixed clock is used to sample data 1b - Tuned clock is used to sample data
22 EXECUTE_TUNING	Execute tuning When std_tuning_en field is set, this field is used to start tuning procedure. Otherwise, this field is reserved. This field is set to start tuning procedure and automatically cleared when runing procedure is completed. The result of tuning is indicated to sam_clk_sel field. Tuning procedure is aborted by writing 0.
21-8 —	Reserved
7 CNIBAC12E	Command not issued by Auto CMD12 error Setting this field to 1 means CMD_wo_DAT is not executed due to an Auto CMD12 error (D04-D01) in this register.

Table continues on the next page...

Field	Function
	0b - No error 1b - Not issued
6-5 —	Reserved
4 AC12IE	Auto CMD12 / 23 index error Occurs if the command index error occurs in response to a command. 0b - No error 1b - Error, the CMD index in response is not CMD12/23
3 AC12CE	Auto CMD12 / 23 CRC error Occurs when detecting a CRC error in the command response. 0b - No CRC error 1b - CRC error met in Auto CMD12/23 response
2 AC12EBE	Auto CMD12 / 23 end bit error Occurs when detecting that the end field of command response is 0 which should be 1. 0b - No error 1b - End bit error generated
1 AC12TOE	Auto CMD12 / 23 timeout error Occurs if no response is returned within 64 SDCLK cycles from the end field of the command. If this field is set to1, the other error status fields (2-4) have no meaning. 0b - No error 1b - Time out
0 AC12NE	Auto CMD12 not executed If memory multiple block data transfer is not started, due to a command error, this field is not set because it is not necessary to issue an Auto CMD12. Setting this field to 1 means uSDHC cannot issue the Auto CMD12 to stop a memory multiple block data transfer due to some error. If this field is set to 1, other error status fields (1-4) have no meaning. 0b - Executed 1b - Not executed

10.3.7.1.18 Host Controller Capabilities (HOST_CTRL_CAP)

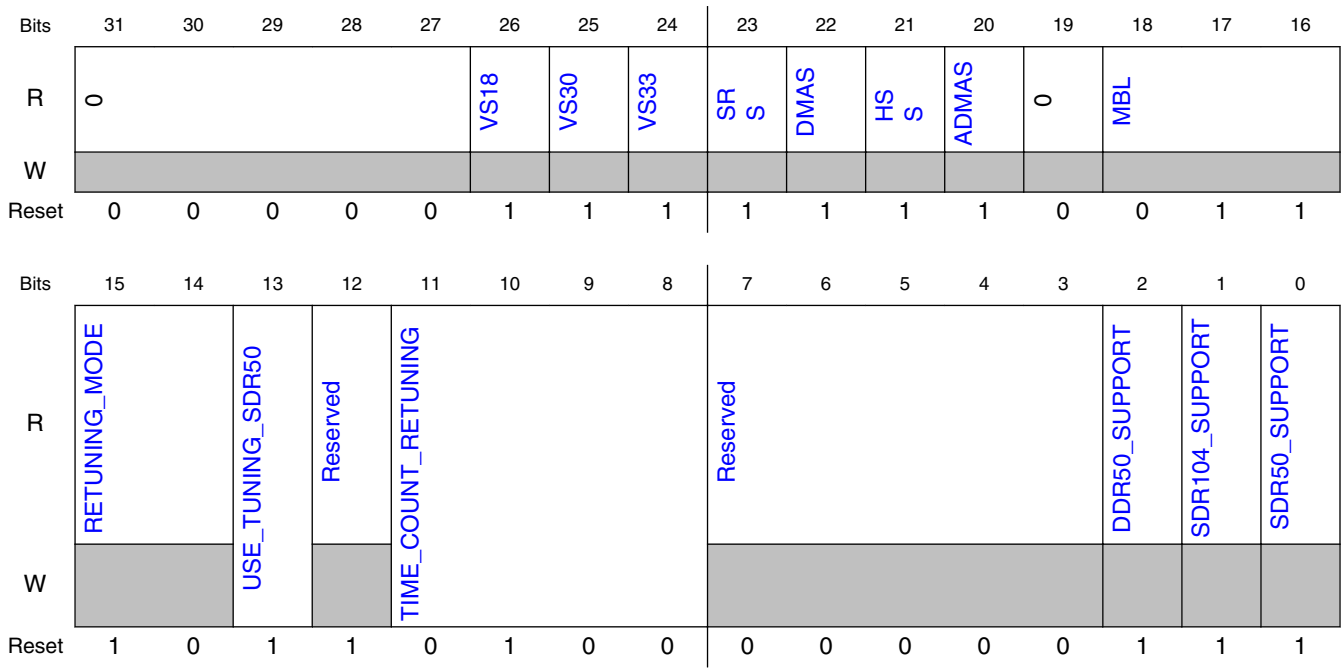
10.3.7.1.18.1 Offset

Register	Offset
HOST_CTRL_CAP	40h

10.3.7.1.18.2 Function

This register provides the host driver with information specific to uSDHC implementation. The value in this register is the power-on-reset value and does not change with a software reset.

10.3.7.1.18.3 Diagram



10.3.7.1.18.4 Fields

Field	Function
31-27 —	Reserved
26 VS18	Voltage support 1.8 V This field depends on the host system ability. 0b - 1.8 V not supported 1b - 1.8 V supported
25 VS30	Voltage support 3.0 V This field depends on the host system ability. 0b - 3.0 V not supported 1b - 3.0 V supported
24 VS33	Voltage support 3.3 V This field depends on the host system ability. 0b - 3.3 V not supported 1b - 3.3 V supported
23 SRS	Suspend / resume support This field indicates whether uSDHC supports Suspend / Resume functionality. If this field is 0, the Suspend and Resume mechanism, as well as the read wait, are not supported, and the host driver does not issue either Suspend or Resume commands. 0b - Not supported

Table continues on the next page...

Field	Function
	1b - Supported
22 DMAS	DMA support This field indicates whether uSDHC is capable of using the internal DMA to transfer data between system memory and the data buffer directly. 0b - DMA not supported 1b - DMA supported
21 HSS	High speed support This field indicates whether uSDHC supports High Speed mode and the host system can supply a SD clock frequency from 25 MHz to 50 MHz. 0b - High speed not supported 1b - High speed supported
20 ADMAS	ADMA support This field indicates whether uSDHC supports the ADMA feature. 0b - Advanced DMA not supported 1b - Advanced DMA supported
19 —	Reserved
18-16 MBL	Max block length This value indicates the maximum block size that the host driver can read and write to the buffer in uSDHC. The buffer transfers block size without wait cycles. 000b - 512 bytes 001b - 1024 bytes 010b - 2048 bytes 011b - 4096 bytes
15-14 RETUNING_MODE	Retuning Mode This field selects retuning method. 00b - Mode 1 01b - Mode 2 10b - Mode 3 11b - Reserved
13 USE_TUNING_SDR50	Use Tuning for SDR50 This field is set to 1. Host controller requires tuning to operate SDR50. 0b - SDR does not require tuning. 1b - SDR50 requires tuning.
12 —	Reserved
11-8 TIME_COUNT_RETUNING	Time counter for retuning This field indicates an initial value of the Retuning Timer for Re-Tuning Mode1 and 3.Setting to 0 disables Retuning Timer.
7-3 —	Reserved
2 DDR50_SUPPORT	DDR50 support This field indicates support of DDR50 mode.

Table continues on the next page...

Field	Function
1 SDR104_SUPPORT	SDR104 support This field indicates support of SDR104 mode.
0 SDR50_SUPPORT	SDR50 support This field indicates support of SDR50 mode.

10.3.7.1.19 Watermark Level (WTMK_LVL)

10.3.7.1.19.1 Offset

Register	Offset
WTMK_LVL	44h

10.3.7.1.19.2 Function

Both write and read watermark levels (FIFO threshold) are configurable. Their value can range from 1 to 128 words. Both write and read burst lengths are also Configurable. There value can range from 1 to 31 words.

10.3.7.1.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			WR_BRST_LEN					WR_WML							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			RD_BRST_LEN					RD_WML							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0

10.3.7.1.19.4 Fields

Field	Function
31-29 —	Reserved

Table continues on the next page...

Field	Function
28-24 WR_BRST_LEN	Write burst length due to system restriction, the actual burst length might not exceed 16 The number of words uSDHC writes in a single burst. The write burst length must be less than or equal to the write watermark level, and all bursts within a watermark level transfer is in back-to-back mode. On reset, this field is 8. Writing 0 to this field results in '01000' (that is, it is not able to clear this field).
23-16 WR_WML	Write watermark level The number of words used as the watermark level (FIFO threshold) in a DMA write operation. Also, the number of words as a sequence of write bursts in back-to-back mode. The maximum legal value for the write watermark level is 128.
15-13 —	Reserved
12-8 RD_BRST_LEN	Read burst length due to system restriction, the actual burst length might not exceed 16 The number of words uSDHC reads in a single burst. The read burst length must be less than or equal to the read watermark level, and all bursts within a watermark level transfer is in back-to-back mode. On reset, this field is 8. Writing 0 to this field results in '01000' (that is, it is not able to clear this field).
7-0 RD_WML	Read watermark level The number of words used as the watermark level (FIFO threshold) in a DMA read operation. Also, the number of words as a sequence of read bursts in back-to-back mode. The maximum legal value for the read water mark level is 128.

10.3.7.1.20 Mixer Control (MIX_CTRL)

10.3.7.1.20.1 Offset

Register	Offset
MIX_CTRL	48h

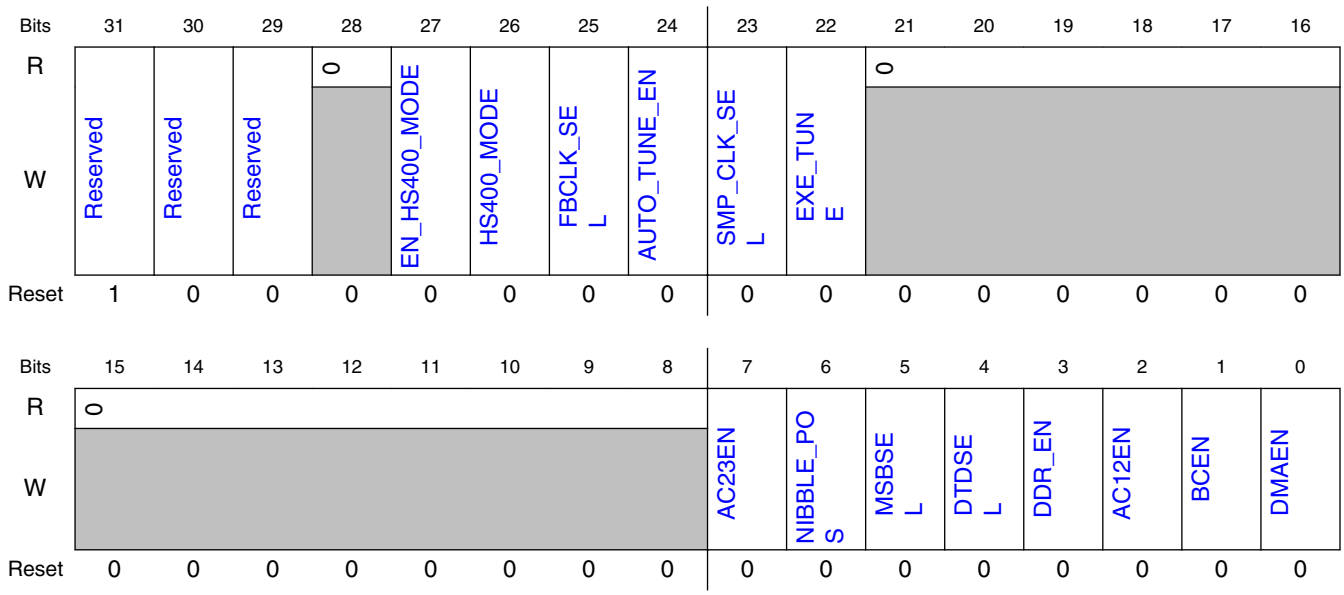
10.3.7.1.20.2 Function

This register is used to DMA and data transfer. To prevent data loss, The software should check if data transfer is active before writing this register. These fields are DPSEL, MBSEL, DTDSEL, AC12EN, BCEN, and DMAEN.

Table 10-43. Transfer type register setting for various transfer types

Multi/single block select	Block count enable	Block count	Function
0	Do not care	Do not care	Single transfer
1	0	Do not care	Infinite transfer
1	1	Positive number	Multiple transfer
1	1	Zero	No data transfer

10.3.7.1.20.3 Diagram



10.3.7.1.20.4 Fields

Field	Function
31 —	Reserved Always write as 1.
30 —	Reserved Always write as 0.
29 —	Reserved Always write as 0.
28 —	Reserved
27 EN_HS400_MODE	Enable enhance HS400 mode Enhance HS400 enable
26 HS400_MODE	Enable HS400 mode HS400 Enable
25 FBCLK_SEL	Feedback clock source selection (Only used for SD3.0, SDR104 mode and EMMC HS200 mode) 0b - Feedback clock comes from the loopback CLK 1b - Feedback clock comes from the ipp_card_clk_out
24 AUTO_TUNE_EN	Auto tuning enable (Only used for SD3.0, SDR104 mode and and EMMC HS200 mode) 0b - Disable auto tuning 1b - Enable auto tuning
23 SMP_CLK_SEL	Clock selection When STD_TUNING_EN is 0, this field is used to select Tuned clock or Fixed clock to sample data / cmd (Only used for SD3.0, SDR104 mode and EMMC HS200 mode)

Table continues on the next page...

Field	Function
	0b - Fixed clock is used to sample data / cmd 1b - Tuned clock is used to sample data / cmd
22 EXE_TUNE	Execute tuning: (Only used for SD3.0, SDR104 mode and EMMC HS200 mode) When STD_TUNING_EN is 0, this field is set to 1 to indicate the host driver is starting tuning procedure. Tuning procedure is aborted by writing 0. 0b - Not tuned or tuning completed 1b - Execute tuning
21-8 —	Reserved
7 AC23EN	Auto CMD23 enable When this field is set to 1, the host controller issues a CMD23 automatically before issuing a command specified in the Command Register.
6 NIBBLE_POS	Nibble position indication In DDR 4-bit mode nibble position indication. 0- the sequence is 'odd high nibble -> even high nibble -> odd low nibble -> even low nibble'; 1- the sequence is 'odd high nibble -> odd low nibble -> even high nibble -> even low nibble'.
5 MSBSEL	Multi / Single block select This field enables multiple block DATA line data transfers. For any other commands, this field can be set to 0. If this field is 0, it is not necessary to set the Block Count register. See Command Transfer Type (CMD_XFR_TYP). 0b - Single block 1b - Multiple blocks
4 DTDSEL	Data transfer direction select This field defines the direction of DATA line data transfers. The field is set to 1 by the host driver to transfer data from the SD card to uSDHC and is set to 0 for all other commands. 0b - Write (Host to card) 1b - Read (Card to host)
3 DDR_EN	Dual data rate mode selection
2 AC12EN	Auto CMD12 enable Multiple block transfers for memory require a CMD12 to stop the transaction. When this field is set to 1, uSDHC issues a CMD12 automatically when the last block transfer has completed. The host driver is not set this field to issue commands that do not require CMD12 to stop a multiple block data transfer. In particular, secure commands defined in File Security Specification (see reference list) do not require CMD12. In single block transfer, uSDHC ignores this field no matter it is set or not. 0b - Disable 1b - Enable
1 BCEN	Block count enable This field is used to enable the Block Count register, which is only relevant for multiple block transfers. When this field is 0, the internal counter for block is disabled, which is useful in executing an infinite transfer. 0b - Disable 1b - Enable
0 DMAEN	DMA enable

Field	Function
	This field enables DMA functionality. If this field is set to 1, a DMA operation begins when the host driver sets the DPSEL field of this register. Whether the simple DMA or the advanced DMA is active depends on the DMA Select field of the Protocol Control register. 0b - Disable 1b - Enable

10.3.7.1.21 Force Event (FORCE_EVENT)

10.3.7.1.21.1 Offset

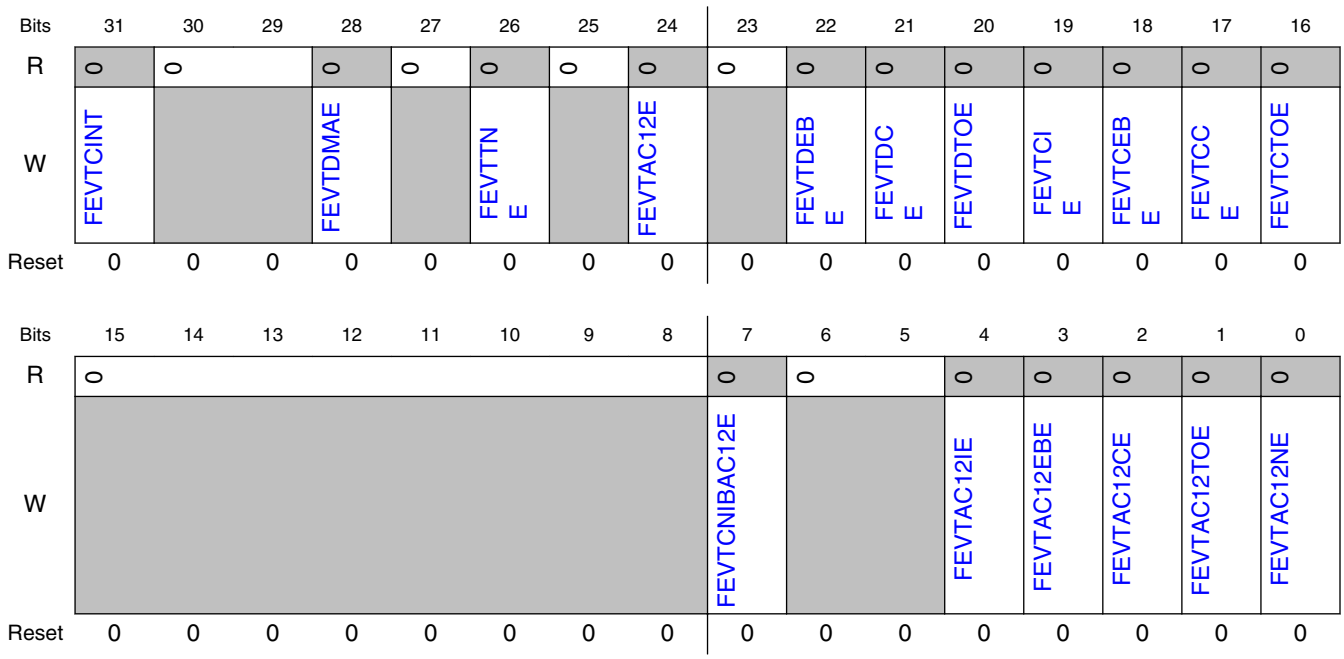
Register	Offset
FORCE_EVENT	50h

10.3.7.1.21.2 Function

The Force Event Register is not a physically implemented register. Rather, it is an address at which the Interrupt Status register can be written if the corresponding field of the Interrupt Status Enable Register is set. This register is a write only register and writing 0 to it has no effect. Writing 1 to this register actually sets the corresponding field of Interrupt Status register. A read from this register always results in 0's. to change the corresponding status fields in the Interrupt Status register, make sure to set IPGEN field in System Control Register so that peripheral clock is always active.

Forcing a card interrupt generates a short pulse on the DATA1 line, and the driver might treat this interrupt as a normal interrupt. The interrupt service routine might skip polling the card interrupt factor as the interrupt is self cleared.

10.3.7.1.21.3 Diagram



10.3.7.1.21.4 Fields

Field	Function
31 FEVTCINT	Force event card interrupt Writing 1 to this field generates a short low-level pulse on the internal DATA1 line, as if a self clearing interrupt was received from the external card. If enabled, the CINT field is set and the interrupt service routine might treat this interrupt as a normal interrupt from the external card.
30-29 —	Reserved
28 FEVTDMAE	Force event DMA error Forces the DMAE field of Interrupt Status register to be set.
27 —	Reserved
26 FEVTTNE	Force tuning error Forces the TNE field of Interrupt Status register to be set.
25 —	Reserved
24 FEVTAC12E	Force event Auto Command 12 error Forces the AC12E field of Interrupt Status register to be set.
23 —	Reserved
22	Force event data end bit error

Table continues on the next page...

Field	Function
FEVTDEBE	Forces the DEBE field of Interrupt Status register to be set.
21 FEVTDCE	Force event data CRC error Forces the DCE field of Interrupt Status register to be set.
20 FEVTDTOE	Force event data time out error Force the DTOE field of Interrupt Status register to be set.
19 FEVTCIE	Force event command index error Forces the CCE field of Interrupt Status register to be set.
18 FEVTCEBE	Force event command end bit error Forces the CEBE field of Interrupt Status register to be set.
17 FEVTCCE	Force event command CRC error Forces the CCE field of Interrupt Status register to be set.
16 FEVTCTOE	Force event command time out error Forces the CTOE field of Interrupt Status register to be set.
15-8 —	Reserved
7 FEVTCNIBAC12E	Force event command not executed by Auto Command 12 error Forces the CNIBAC12E field in the Auto Command12 Error Status register to be set.
6-5 —	Reserved
4 FEVTAC12IE	Force event Auto Command 12 index error Forces the AC12IE field in the Auto Command12 Error Status register to be set.
3 FEVTAC12EBE	Force event Auto Command 12 end bit error Forces the AC12EBE field in the Auto Command12 Error Status register to be set.
2 FEVTAC12CE	Force event auto command 12 CRC error Forces the AC12CE field in the Auto Command12 Error Status register to be set.
1 FEVTAC12TOE	Force event auto command 12 time out error Forces the AC12TOE field in the Auto Command12 Error Status register to be set.
0 FEVTAC12NE	Force event auto command 12 not executed Forces the AC12NE field in the Auto Command12 Error Status register to be set.

10.3.7.1.22 ADMA Error Status (ADMA_ERR_STATUS)

10.3.7.1.22.1 Offset

Register	Offset
ADMA_ERR_STATUS	54h

10.3.7.1.22.2 Function

When an ADMA Error Interrupt has occurred, the ADMA Error States field in this register holds the ADMA state and the ADMA System Address register holds the address around the error descriptor.

For recovering from this error, the host driver requires the ADMA state to identify the error descriptor address as follows:

- ST_STOP: Previous location set in the ADMA System Address register is the error descriptor address.
- ST_FDS: Current location set in the ADMA System Address register is the error descriptor address.
- ST_CADR: This state is never set because it only increments the descriptor pointer and does not generate an ADMA error.
- ST_TFR: Previous location set in the ADMA System Address register is the error descriptor address.

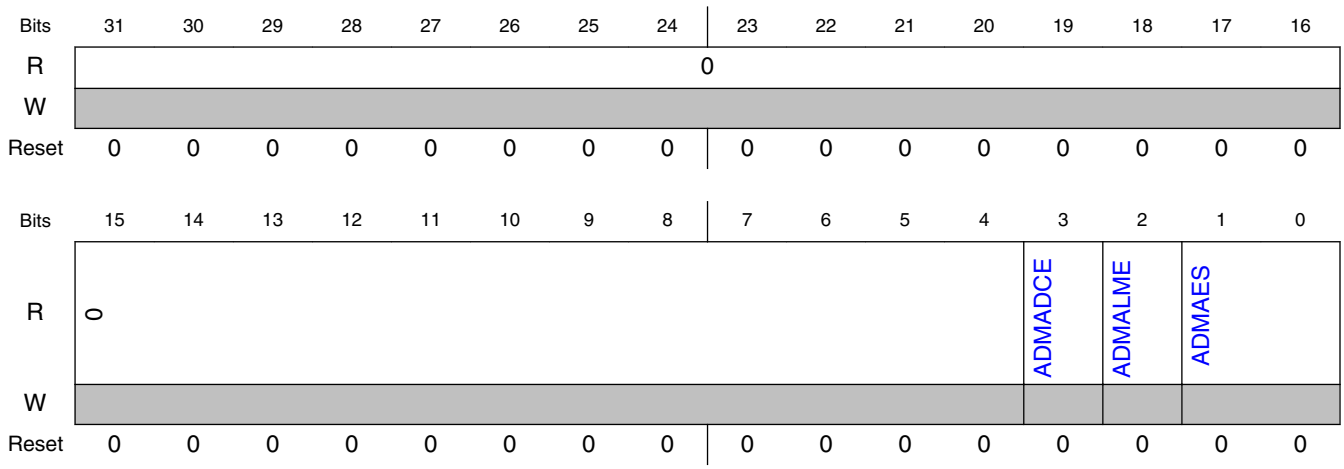
In case of a write operation, the host driver should use the ACMD22 to get the number of the written block, rather than using this information, because unwritten data might exist in the host controller.

The host controller generates the ADMA Error Interrupt when it detects invalid descriptor data (Valid=0) in the ST_FDS state. The host driver can distinguish this error by reading the Valid field of the error descriptor.

Table 10-44. ADMA error state coding

D01-D00	ADMA error state (when error has occurred)	Contents of ADMA System Address register
00	ST_STOP (Stop DMA)	Holds the address of the next executable descriptor command
01	ST_FDS (Fetch descriptor)	Holds the valid descriptor address
10	ST_CADR (Change address)	No ADMA error is generated
11	ST_TFR (Transfer data)	Holds the address of the next executable descriptor command

10.3.7.1.22.3 Diagram



10.3.7.1.22.4 Fields

Field	Function
31-4 —	Reserved
3 ADMADCE	ADMA descriptor error This error occurs when invalid descriptor fetched by ADMA. 0b - No error 1b - Error
2 ADMALME	ADMA length mismatch error This error occurs in the following two cases: • While the block count enable is being set, the total data length specified by the descriptor table is different from that specified by the block count and block length. • Total data length cannot be divided by the block length. 0b - No error 1b - Error
1-0 ADMAES	ADMA error state (when ADMA error is occurred) This field indicates the state of the ADMA when an error has occurred during an ADMA data transfer. See ADMA Error Status (ADMA_ERR_STATUS) for more details.

10.3.7.1.23 ADMA System Address (ADMA_SYS_ADDR)

10.3.7.1.23.1 Offset

Register	Offset
ADMA_SYS_ADDR	58h

10.3.7.1.23.2 Function

This register contains the physical system memory address used for ADMA transfers.

10.3.7.1.23.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ADS_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADS_ADDR														0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.3.7.1.23.4 Fields

Field	Function
31-2 ADS_ADDR	ADMA system address This register holds the word address of the executing command in the Descriptor table. At the start of ADMA, the host driver sets the start address of the Descriptor table. The ADMA engine increments this register address whenever fetching a Descriptor command. When the ADMA is stopped at the Block Gap, this register indicates the address of the next executable Descriptor command. When the ADMA Error Interrupt is generated, this register holds the valid Descriptor address depending on the ADMA state. The lower 2 bits of this register is tied to '0' so the ADMA address is always word aligned. Because this register supports dynamic address reflecting, when TC field is set, it automatically alters the value of internal address counter, so the software cannot change this register when TC field is set. Such restriction is also listed in Software restrictions.
1-0 —	Reserved

10.3.7.1.24 DLL (Delay Line) Control (DLL_CTRL)

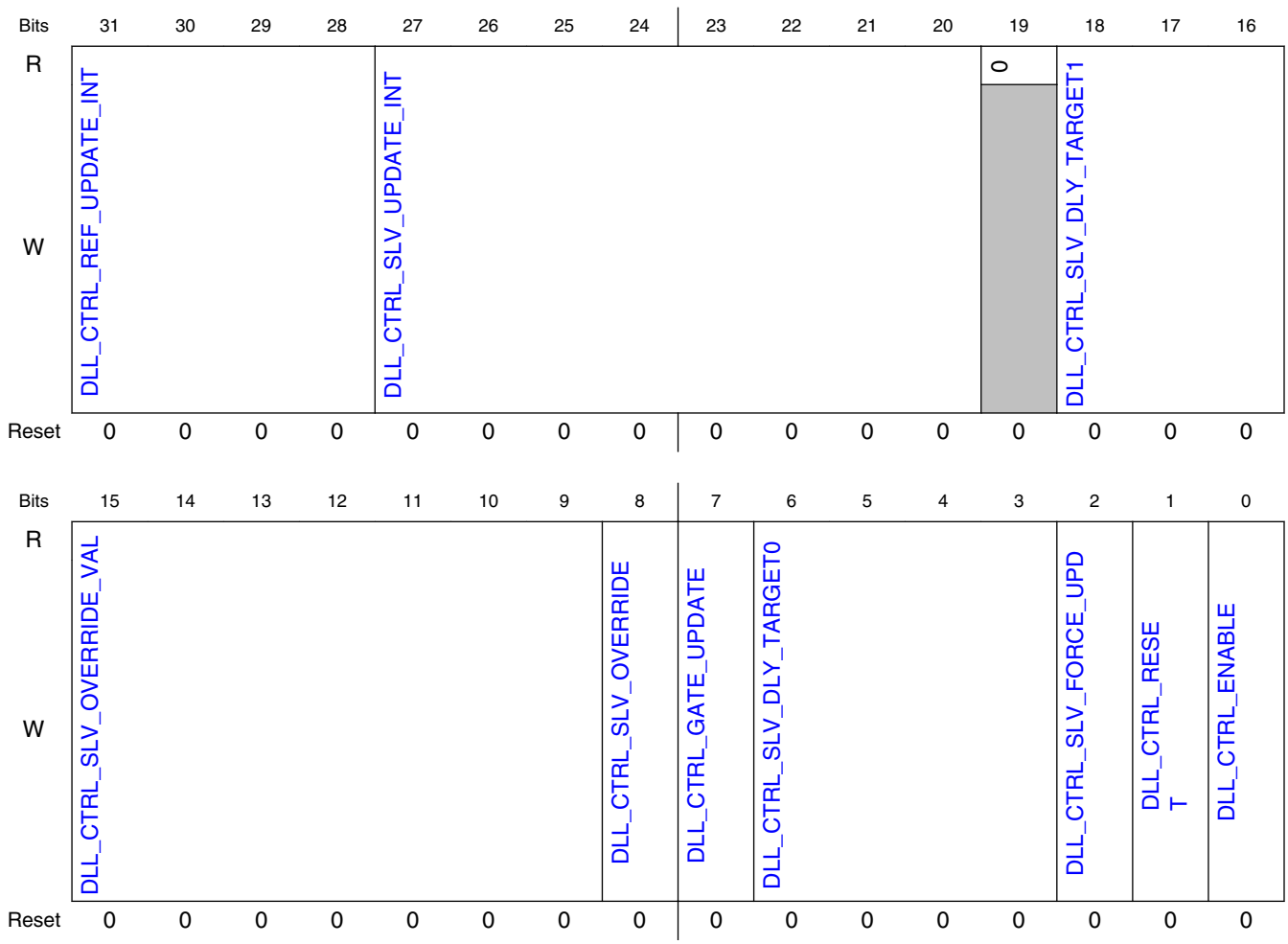
10.3.7.1.24.1 Offset

Register	Offset
DLL_CTRL	60h

10.3.7.1.24.2 Function

This register contains control fields for DLL.

10.3.7.1.24.3 Diagram



10.3.7.1.24.4 Fields

Field	Function
31-28 DLL_CTRL_REF_UPDATE_INT	DLL control loop update interval The interval cycle is (2 + REF_UPDATE_INT) * REF_CLOCK. By default, the DLL control loop updates every two REF_CLOCK cycles. It should be noted that increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that might effect the delay (such as voltage and temperature)
27-20	Slave delay line update interval

Table continues on the next page...

Field	Function
DLL_CTRL_SLV_UPDATE_INT	If default 0 is used, it means 256 cycles of REF_CLOCK. A value of 0x0f results in 15 cycles and so on. Note that software can always cause an update of the slave-delay line using the SLV_FORCE_UPDATE register. Note that the slave delay line also updates automatically when the reference DLL transitions to a locked state (from an un-locked state).
19 —	Reserved
18-16 DLL_CTRL_SLV_DLY_TARGET1	DLL slave delay target1 See DLL_CTRL_SLV_DLY_TARGET0 below.
15-9 DLL_CTRL_SLV_OVERRIDE_VAL	DLL slave override val When SLV_OVERRIDE = 1, this field is used to select 1 of 128 physical taps manually. A value of 0 selects tap 1, and a value of 0x7f selects tap 128.
8 DLL_CTRL_SLV_OVERRIDE	DLL slave override Set this field to 1 to Enable manual override for slave delay chain using SLV_OVERRIDE_VAL; to set 0 to disable manual override. This feature does not require the DLL to be enabled using the ENABLE field. In fact to reduce power, if SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE = 0
7 DLL_CTRL_GATE_UPDATE	DLL gate update Set this field to 1 to prevent the DLL from updating (because when clock_in exists, glitches might appear during DLL updates). This field might be used by software if such a condition occurs. Clear the bit to 0 to allow the DLL to update automatically.
6-3 DLL_CTRL_SLV_DLY_TARGET0	DLL slave delay target0 The delay target for uSDHC loopback read clock can be programmed in 1/16th increments of an ref_clock half-period. The delay is (({DLL_CTRL_SLV_DLY_TARGET1,DLL_CTRL_SLV_DLY_TARGET0} +1)* REF_CLOCK / 2) / 16 So the input read-clock can be delayed relative input data from (REF_CLOCK / 2) / 16 to REF_CLOCK * 4.
2 DLL_CTRL_SLV_FORCE_UPD	DLL slave delay line Setting this field to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line updates automatically based on the SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered). Be sure to use it when uSDHC is idle. This function might not work when uSDHC is working on data / cmd / response.
1 DLL_CTRL_RESET	DLL reset Setting this field to 1 force a reset on DLL. This causes the DLL to lose lock and re-calibrate to detect an REF_CLOCK half period phase shift. This signal is used by the DLL as edge-sensitive, so to create a subsequent reset, RESET must be taken low and then asserted again.
0 DLL_CTRL_ENABLE	DLL and delay chain Set this field to 1 to enable the DLL and delay chain; otherwise; set to 0 to bypasses DLL. Note that using the slave delay line override feature with SLV_OVERRIDE and SLV_OVERRIDE VAL, the DLL does not need to be enabled.

10.3.7.1.25 DLL Status (DLL_STATUS)

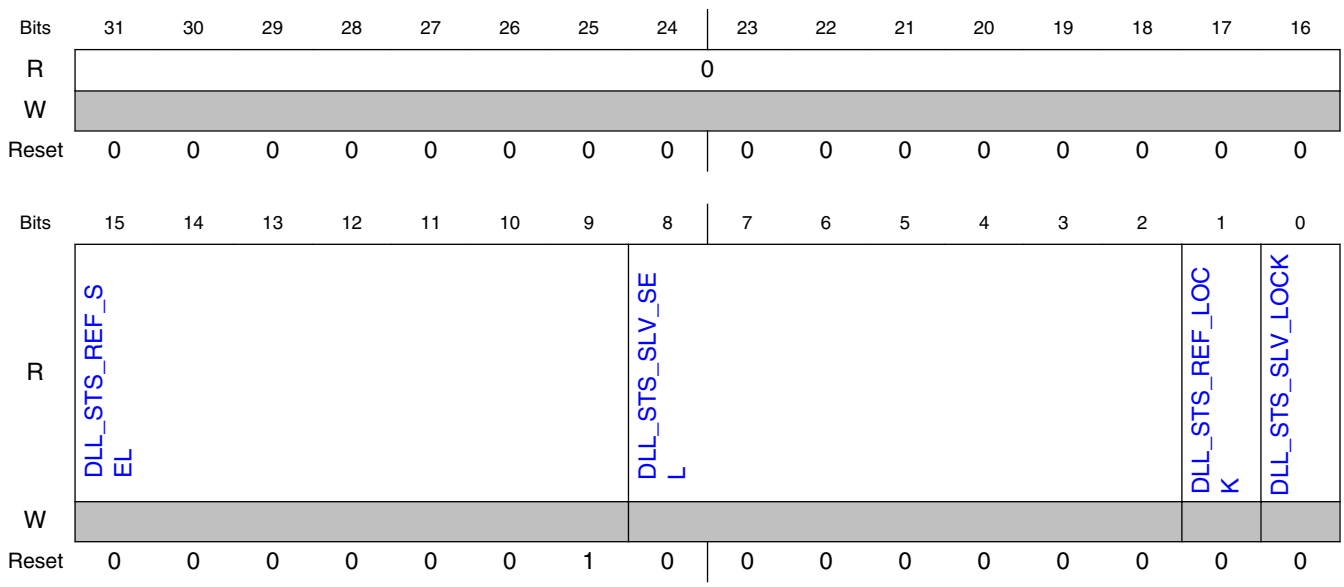
10.3.7.1.25.1 Offset

Register	Offset
DLL_STATUS	64h

10.3.7.1.25.2 Function

This register contains the DLL status information. All fields are read only and reads the same as the power-reset value.

10.3.7.1.25.3 Diagram



10.3.7.1.25.4 Fields

Field	Function
31-16 —	Reserved
15-9 DLL_STS_REF_SEL	Reference delay line select taps This is encoded by 7 fields for 127 taps.
8-2 DLL_STS_SLV_SEL	Slave delay line select status This is the instant value generated from reference chain. Because the reference chain can only be updated when REF_CLOCK is detected, this value should be the right value to be updated when the reference is locked.
1	Reference DLL lock status

Table continues on the next page...

Field	Function
DLL_STS_REF_LOCK	This signifies that the DLL has detected and locked to a half-phase ref_clock shift, allowing the slave delay-line to perform programmed clock delays
0	Slave delay-line lock status
DLL_STS_SLV_LOCK	This signifies that a valid calibration has been set to the slave-delay line and that the slave-delay line is implementing the programmed delay value

10.3.7.1.26 CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS)

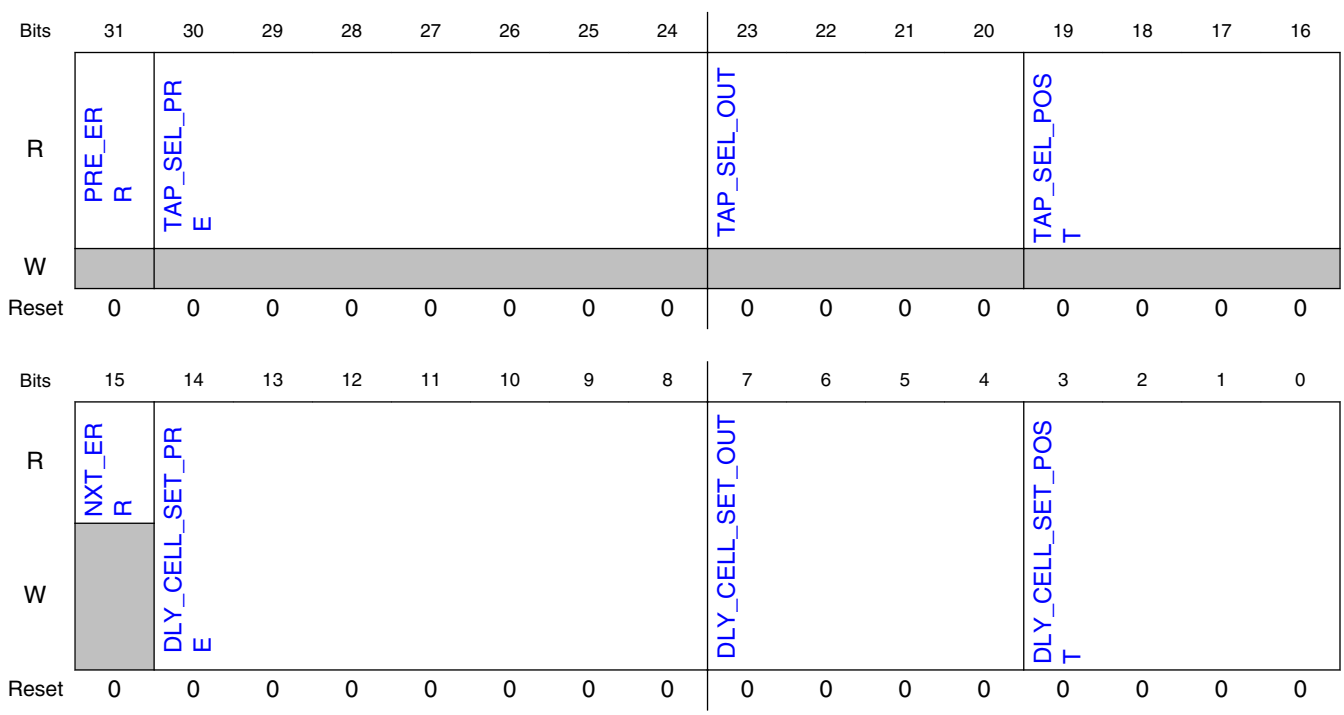
10.3.7.1.26.1 Offset

Register	Offset
CLK_TUNE_CTRL_STATUS	68h

10.3.7.1.26.2 Function

This register contains the Clock Tuning Control status information. All fields are read only and reads the same as the power-reset value. This register is added to support SD3.0 UHS-I SDR104 mode and EMMC HS200 mode.

10.3.7.1.26.3 Diagram



10.3.7.1.26.4 Fields

Field	Function
31 PRE_ERR	PRE error PRE error which means the number of delay cells added on the feedback clock is too small. It is valid only when SMP_CLK_SEL of Mix control register (bit23 of 0x48) is enabled.
30-24 TAP_SEL_PRE	TAP_SEL_PRE Reflects the number of delay cells added on the feedback clock between the feedback clock and CLK_PRE. When AUTO_TUNE_EN (bit24 of 0x48) is disabled, TAP_SEL_PRE is always equal to DLY_CELL_SET_PRE. When AUTO_TUNE_EN (bit24 of 0x48) is enabled, TAP_SEL_PRE is updated automatically according to the status of the auto tuning circuit to adjust the sample clock phase.
23-20 TAP_SEL_OUT	Delay cells added on the feedback clock between CLK_PRE and CLK_OUT Reflects the number of delay cells added on the feedback clock between CLK_PRE and CLK_OUT.
19-16 TAP_SEL_POST	Delay cells added on the feedback clock between CLK_OUT and CLK_POST Reflects the number of delay cells added on the feedback clock between CLK_OUT and CLK_POST.
15 NXT_ERR	NXT error NXT error which means the number of delay cells added on the feedback clock is too large. It's valid only when SMP_CLK_SEL of Mix control register (bit23 of 0x48) is enabled.
14-8 DLY_CELL_SET_PRE	delay cells on the feedback clock between the feedback clock and CLK_PRE Set the number of delay cells on the feedback clock between the feedback clock and CLK_PRE.
7-4 DLY_CELL_SET_OUT	Delay cells on the feedback clock between CLK_PRE and CLK_OUT Set the number of delay cells on the feedback clock between CLK_PRE and CLK_OUT.
3-0 DLY_CELL_SET_POST	Delay cells on the feedback clock between CLK_OUT and CLK_POST Set the number of delay cells on the feedback clock between CLK_OUT and CLK_POST.

10.3.7.1.27 Strobe DLL control (STROBE_DLL_CTRL)

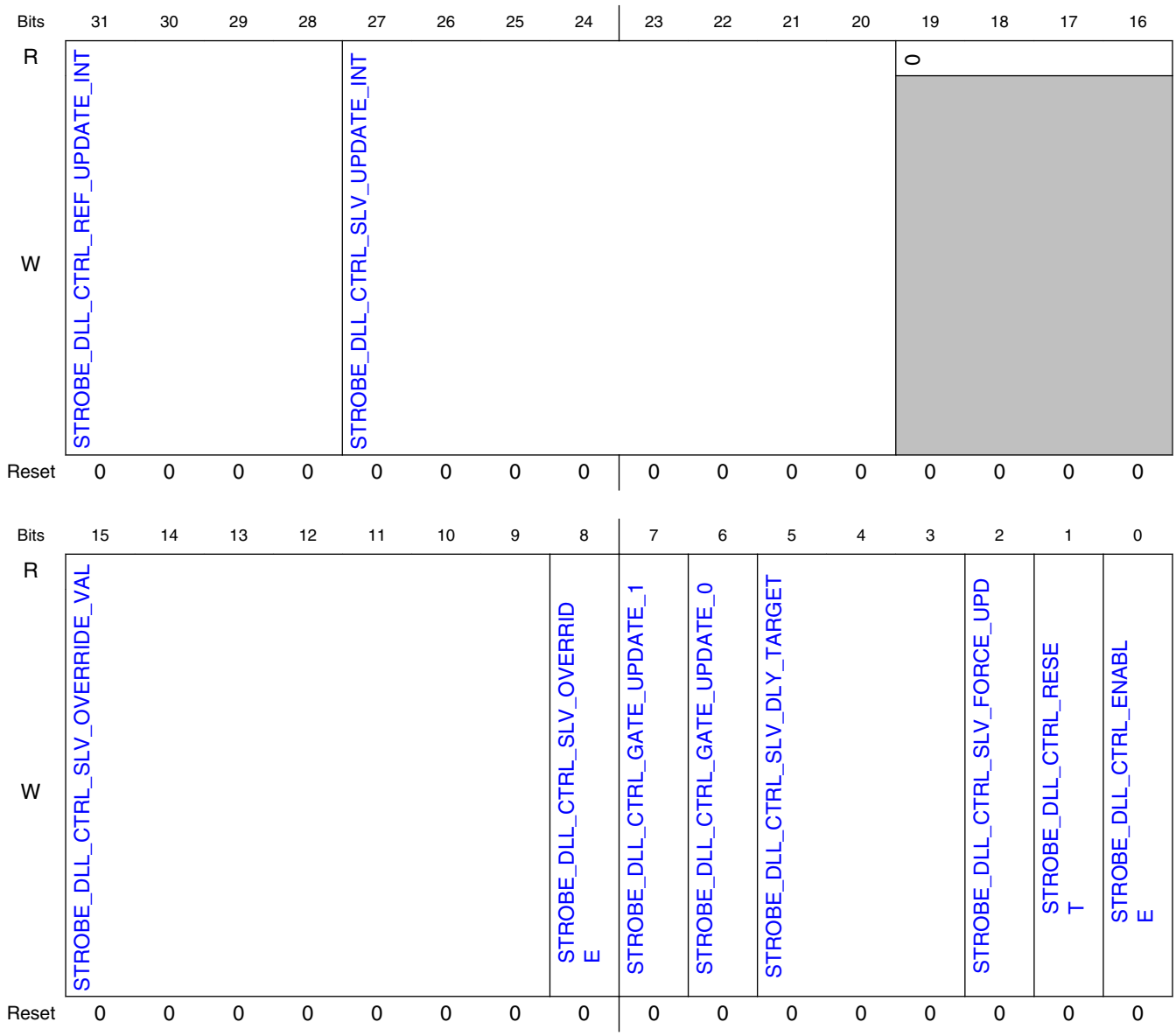
10.3.7.1.27.1 Offset

Register	Offset
STROBE_DLL_CTRL	70h

10.3.7.1.27.2 Function

This register contains the strobe DLL control information.

10.3.7.1.27.3 Diagram



10.3.7.1.27.4 Fields

Field	Function
31-28 STROBE_DLL_CTRL_REF_UPDATE_INT	Strobe DLL control reference update interval The interval cycle is (2 + STROBE_REF_UPDATE_INT) * STROBE_REF_CLOCK. By default, the DLL control loop updates every two STROBE_REF_CLOCK cycles.

Table continues on the next page...

Field	Function
	NOTE: Increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that might effect the delay (such as voltage and temperature)
27-20 STROBE_DLL_CTRL_SLV_UPDATE_INT	Strobe DLL control slave update interval Slave delay line update interval. If default 0 is used, it means 256 cycles of STROBE_REF_CLOCK. A value of 0x0F results in 15 cycles and so on. **NOTE:** software can always cause an update of the slave-delay line using the STROBE_SLV_FORCE_UPDATE register. The slave delay line also updates automatically when the reference DLL transitions to a locked state (from an un-locked state).
19-16 —	This field is reserved. This read-only field is reserved and always has the value 0.
15-9 STROBE_DLL_CTRL_SLV_OVERRIDE_VAL	Strobe DLL control slave Override value When STROBE_SLV_OVERRIDE = 1, this field is used to manually select one of 128 physical taps. A value of 0 selects tap 1, and a value of 0x7F selects tap 128.
8 STROBE_DLL_CTRL_SLV_OVERRIDE	Strobe DLL control slave override Set this field to 1 to Enable manual override for slave delay chain using STROBE_SLV_OVERRIDE_VAL; set this field to 0 to disable manual override. This feature does not require the DLL to be enabled using the ENABLE field. In fact to reduce power, if STROBE_SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE = 0.
7 STROBE_DLL_CTRL_GATE_UPDATE_1	Strobe DLL control gate update Set this field to 1 to prevent the DLL from updating (because when STROBE_CLOCK_IN exists, glitches might appear during DLL updates). This field can be used by software if such a condition occurs. Clear the field to 0 to allow the DLL to update automatically.
6 STROBE_DLL_CTRL_GATE_UPDATE_0	Strobe DLL control gate update Set this field to 1 to prevent the DLL from updating (because when STROBE_CLOCK_IN exists, glitches might appear during DLL updates). This field can be used by software if such a condition occurs. Clear the field to 0 to allow the DLL to update automatically.
5-3 STROBE_DLL_CTRL_SLV_DLY_TARGET	Strobe DLL Control Slave Delay Target The delay target for uSDHC loopback read clock can be programmed in 1/16th increments of an STROBE_REF_CLOCK half-period. The delay is ((STROBE_DLL_CTRL_SLV_DLY_TARGET + 1) * STROBE_REF_CLOCK / 2) / 16, So the input read-clock can be delayed relative input data from (STROBE_REF_CLOCK / 2) / 16 to (STROBE_REF_CLOCK *4)/16.
2 STROBE_DLL_CTRL_SLV_FORCE_UPD	Strobe DLL control slave force updated Setting this field to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line should automatically update the STROBE_SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if STROBE_SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered). Be sure to use it when uSDHC is idle. This function might not work when uSDHC is working on data / cmd / response.
1 STROBE_DLL_CTRL_RESET	Strobe DLL control reset Setting this field to 1 to force a reset on DLL. This causes the DLL to lose lock and re-calibrate to detect an REF_CLOCK half period phase shift. This signal is used by the DLL as edge-sensitive, to create a subsequent reset, RESET must be taken low and then asserted again.
0 STROBE_DLL_CTRL_ENABLE	Strobe DLL control enable Set this field to 1 to enable the DLL and delay chain; otherwise, set to 0 to bypasses DLL. **NOTE:** Using the slave delay line override feature with STROBE_SLV_OVERRIDE and STROBE_SLV_OVERRIDE VAL, the DLL does not need to be enabled.

10.3.7.1.28 Strobe DLL status (STROBE_DLL_STATUS)

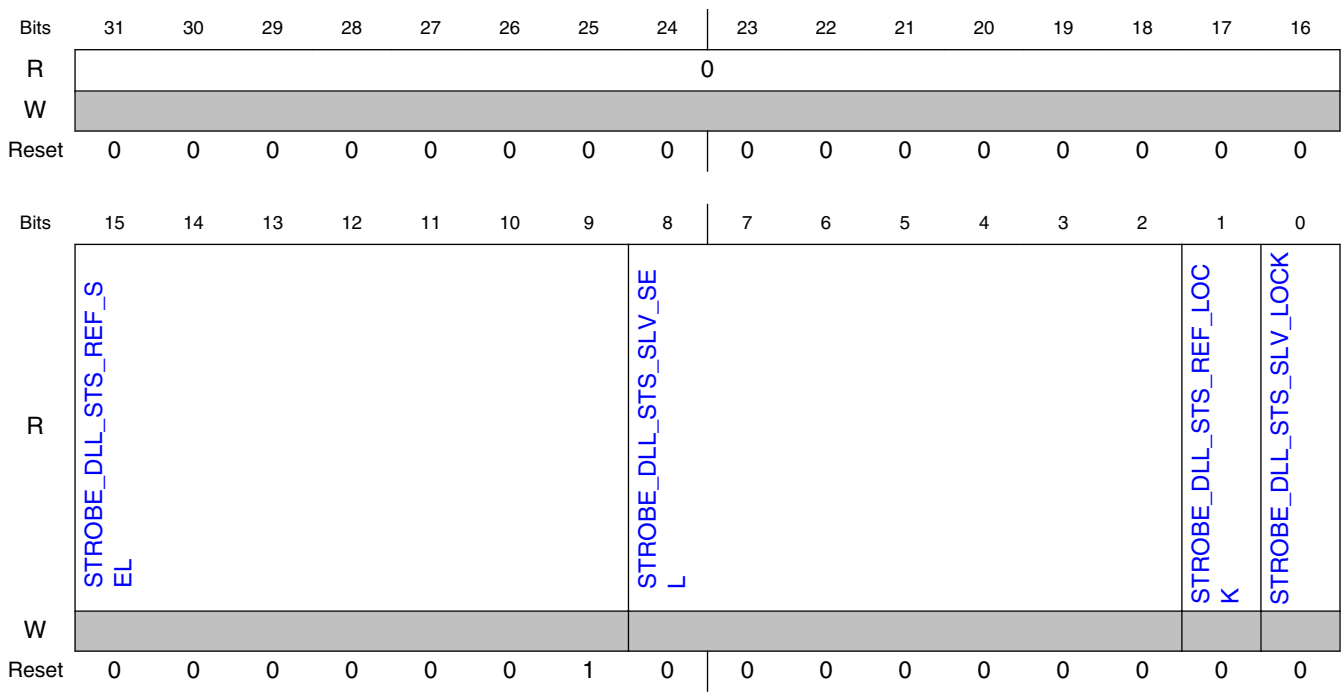
10.3.7.1.28.1 Offset

Register	Offset
STROBE_DLL_STATUS	74h

10.3.7.1.28.2 Function

This register contains the strobe DLL status information. All fields are read only and read the same as the power-reset value.

10.3.7.1.28.3 Diagram



10.3.7.1.28.4 Fields

Field	Function
31-16	This field is reserved.
—	This read-only field is reserved and always has the value 0.
15-9	Strobe DLL status reference select

Table continues on the next page...

Field	Function
STROBE_DLL_STS_REF_SEL	Reference delay line select taps. This is encoded by 7 fields for 127 taps.
8-2 STROBE_DLL_STS_SLV_SEL	Strobe DLL status slave select Slave delay line select status. This is the instant value generated from reference chain. Because the reference chain can only be updated when STROBE_REF_CLOCK is detected, this value can be updated with the right value when the reference is locked.
1 STROBE_DLL_STS_REF_LOCK	Strobe DLL status reference lock This signifies that the DLL has detected and locked to a half-phase REF_CLOCK shift, it allows the slave delay-line to perform programmed clock delays.
0 STROBE_DLL_STS_SLV_LOCK	Strobe DLL status slave lock Slave delay-line lock status. This signifies that a valid calibration has been set to the slave-delay line, and the slave-delay line is implementing the programmed delay value.

10.3.7.1.29 Vendor Specific Register (VEND_SPEC)

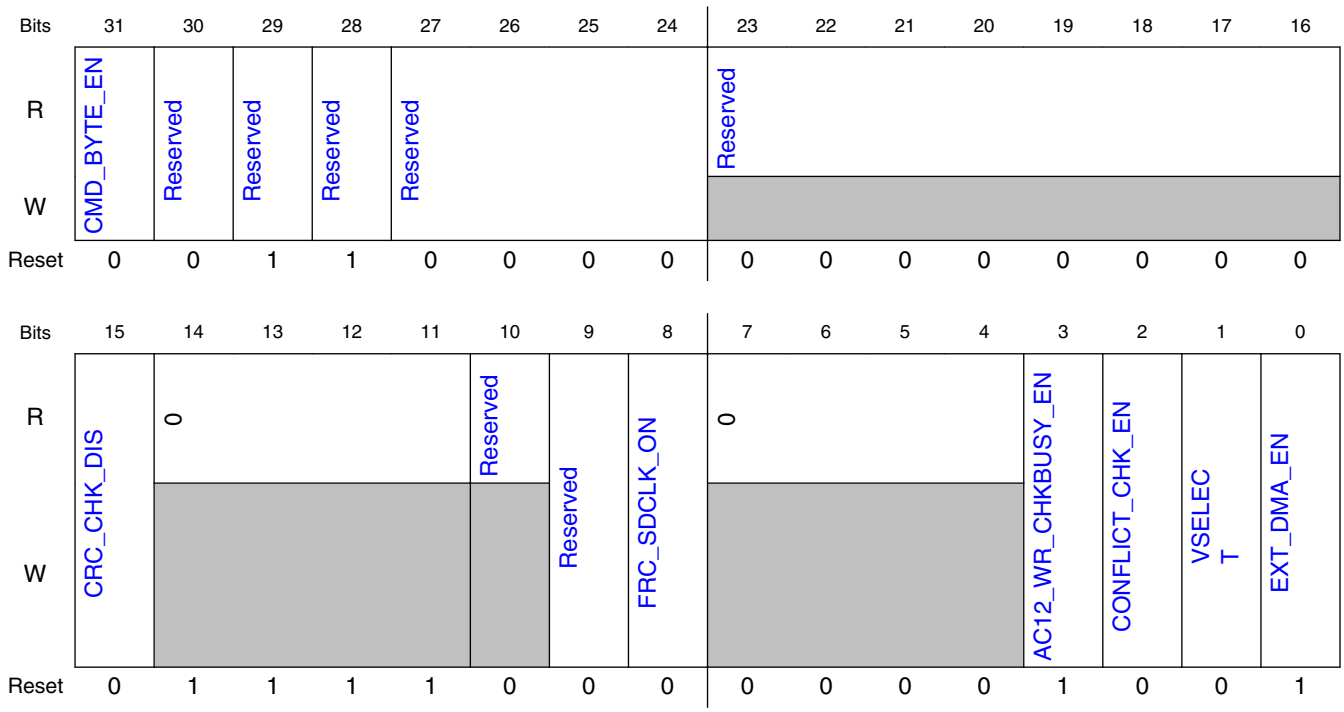
10.3.7.1.29.1 Offset

Register	Offset
VEND_SPEC	C0h

10.3.7.1.29.2 Function

This register contains the vendor specific control / status register.

10.3.7.1.29.3 Diagram



10.3.7.1.29.4 Fields

Field	Function
31 CMD_BYTE_EN	Byte access This bit controls the byte access. 0b - Disable 1b - Enable
30 —	Reserved Always write as 0.
29 —	Reserved Always write as 1
28 —	Reserved Always write as 0.
27-24 —	Reserved Always write as 4'b0000.
23-16 —	Reserved Always write as 8'h00.
15 CRC_CHK_DIS	CRC Check Disable 0b - Check CRC16 for every read data packet and check CRC fields for every write data packet 1b - Ignore CRC16 check for every read data packet and ignore CRC fields check for every write data packet
14-11	Reserved

Table continues on the next page...

Field	Function
—	Reserved
10 —	Reserved Always write as 0.
9 —	Reserved Always write as 0.
8 FRC_SDCLK_ON	Force CLK Force CLK output active 0b - CLK active or inactive is fully controlled by the hardware. 1b - Force CLK active
7-4 —	Reserved
3 AC12_WR_CHKBUSY_EN	Check busy enable Check busy enable after auto CMD12 for write data packet 0b - Do not check busy after auto CMD12 for write data packet 1b - Check busy after auto CMD12 for write data packet
2 CONFLICT_CHK_EN	Conflict check enable It is not implemented in uSDHC IP. 0b - Conflict check disable 1b - Conflict check enable
1 VSELECT	Voltage selection Change the value of output signal VSELECT, to control the voltage on pads for external card. There must be a control circuit out of uSDHC to change the voltage on pads. 0b - Change the voltage to high voltage range, around 3.0 V 1b - Change the voltage to low voltage range, around 1.8 V
0 EXT_DMA_EN	External DMA request enable Enable the request to external DMA. When the internal DMA (either Simple DMA or Advanced DMA) is not in use, and this field is set, uSDHC sends out DMA request when the internal buffer is ready. This field is particularly useful when transferring data by Arm platform polling mode, and it is not allowed to send out the external DMA request. By default, this field is set. 0b - In any scenario, uSDHC does not send out external DMA request. 1b - When internal DMA is not active, the external DMA request is sent out.

10.3.7.1.30 MMC Boot (MMC_BOOT)

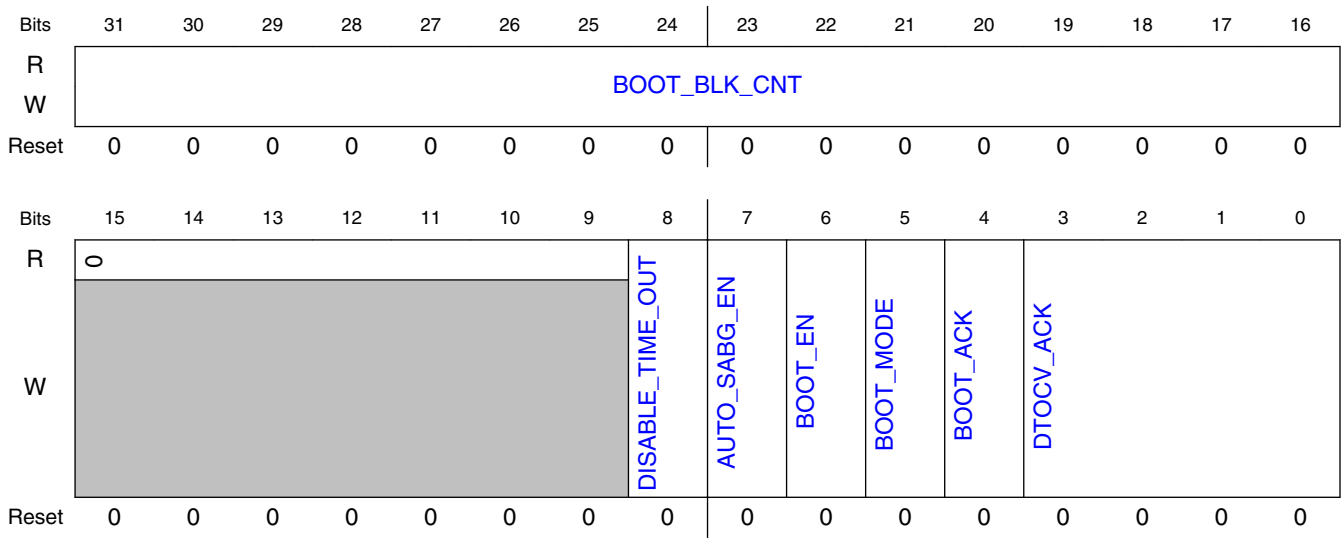
10.3.7.1.30.1 Offset

Register	Offset
MMC_BOOT	C4h

10.3.7.1.30.2 Function

This register contains the MMC Fast Boot control register.

10.3.7.1.30.3 Diagram



10.3.7.1.30.4 Fields

Field	Function
31-16 BOOT_BLK_CNT	Stop At Block Gap value of automatic mode The value defines the Stop At Block Gap value of automatic mode. When received, card block cnt is equal to (BLK_CNT - BOOT_BLK_CNT) and AUTO_SABG_EN is 1, then Stop At Block Gap. Here, BLK_CNT is defined in the Block Attributes Register, field 31 - 16 of 0x04.
15-9 —	Reserved
8 DISABLE_TIME_OUT	Time out **NOTE:** When this field is set, there is no timeout check no matter whether BOOT_EN is set or not. 0b - Enable time out 1b - Disable time out
7 AUTO_SABG_EN	Auto stop at block gap During boot, enable auto stop at block gap function. This function is triggered, and host stops at block gap when received card block cnt is equal to (BLK_CNT - BOOT_BLK_CNT).
6 BOOT_EN	Boot enable Boot mode enable 0b - Fast boot disable 1b - Fast boot enable
5 BOOT_MODE	Boot mode Boot mode select

Table continues on the next page...

Field	Function
	0b - Normal boot 1b - Alternative boot
4 BOOT_ACK	BOOT ACK Boot ACK mode select 0b - No ack 1b - Ack
3-0 DTOCV_ACK	DTOCV_ACK Boot ACK time out counter value 0000b - SDCLK x 2^32 0001b - SDCLK x 2^33 0010b - SDCLK x 2^18 0011b - SDCLK x 2^19 0100b - SDCLK x 2^20 0101b - SDCLK x 2^21 0110b - SDCLK x 2^22 0111b - SDCLK x 2^23 1110b - SDCLK x 2^30 1111b - SDCLK x 2^31

10.3.7.1.31 Vendor Specific 2 Register (VEND_SPEC2)

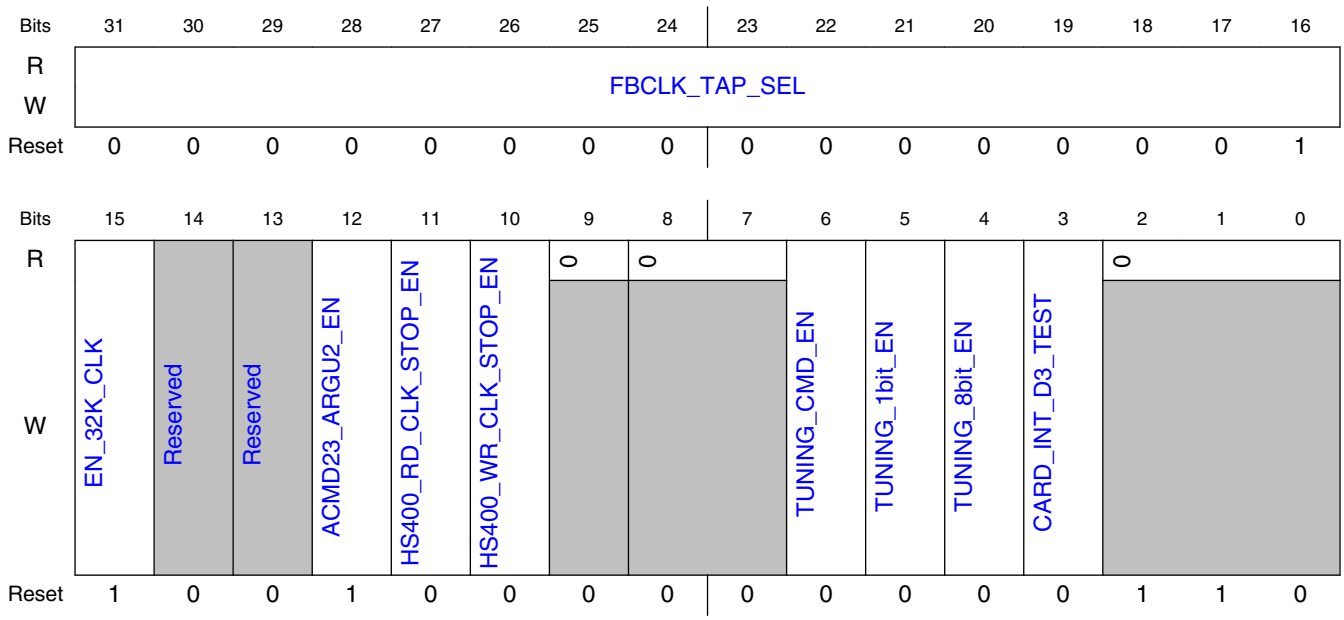
10.3.7.1.31.1 Offset

Register	Offset
VEND_SPEC2	C8h

10.3.7.1.31.2 Function

This register contains the vendor specific control 2 register.

10.3.7.1.31.3 Diagram



10.3.7.1.31.4 Fields

Field	Function
31-16 FBCLK_TAP_SEL	Enable extra delay on internal feedback clock
15 EN_32K_CLK	Enable 32khz clock for card detection Use 32khz clock for card detection
14 —	Reserved
13 —	Reserved
12 ACMD23_ARGU2_EN	Argument2 register enable for ACMD23 0b - Disable 1b - Argument2 register enable for ACMD23 sharing with SDMA system address register. Default is enabled.
11 HS400_RD_CLK_STOP_EN	HS400 read clock stop enable Only stop clock at read block gap.
10 HS400_WR_CLK_STOP_EN	HS400 write clock stop enable Only stop clock at write block gap.
9 —	Reserved

Table continues on the next page...

Field	Function
8-7 —	Reserved
6 TUNING_CMD_EN	Tuning command enable Enable the auto tuning circuit to check the CMD line. 0b - Auto tuning circuit does not check the CMD line. 1b - Auto tuning circuit checks the CMD line.
5 TUNING_1bit_EN	Tuning 1bit enable Enable the auto tuning circuit to check the DATA0 only. It is used with the TUNING_8bit_EN together.
4 TUNING_8bit_EN	Tuning 8bit enable Enable the auto tuning circuit to check the DATA[7:0]. It is used with the TUNING_1bit_EN together. 0b00 - Tuning circuit only checks the DATA[3:0] 0b01 - Tuning circuit only checks the DATA0 0b10 - Tuning circuit checks the whole DATA[7:0] 0b11 - Invalid **NOTE:** The format of these two fields are [TUNNING_8bit_EN:TUNNING_1bit_EN].
3 CARD_INT_D3_TEST	Card interrupt detection test This field only uses for debugging. 0b - Check the card interrupt only when DATA3 is high. 1b - Check the card interrupt by ignoring the status of DATA3.
2-0 —	Reserved

10.3.7.1.32 Tuning Control (TUNING_CTRL)

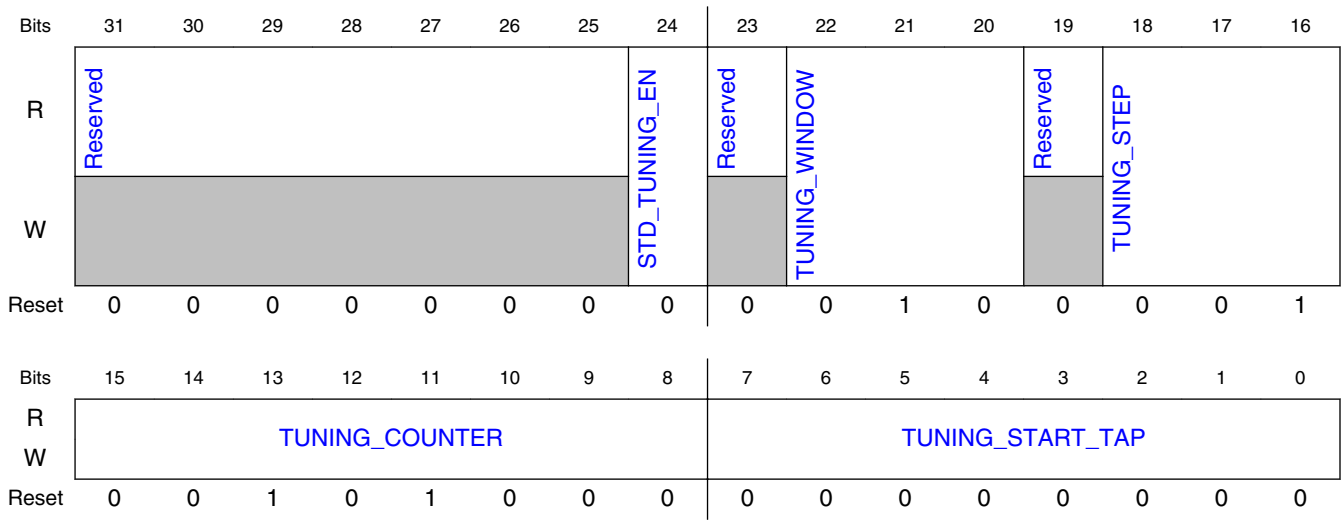
10.3.7.1.32.1 Offset

Register	Offset
TUNING_CTRL	CCh

10.3.7.1.32.2 Function

The register contains configuration of tuning circuit.

10.3.7.1.32.3 Diagram



10.3.7.1.32.4 Fields

Field	Function
31-25 —	Reserved
24 STD_TUNING_EN	Standard tuning circuit and procedure enable This field is used to enable standard tuning circuit and procedure.
23 —	Reserved
22-20 TUNING_WINDOW	Data window Select data window value for auto tuning
19 —	Reserved
18-16 TUNING_STEP	TUNING_STEP The increasing delay cell steps in tuning procedure.
15-8 TUNING_COUNTER	Tuning counter The MAX repeat CMD19 times in tuning procedure.
7-0 TUNING_START_TAP	Tuning start The start delay cell point when send first CMD19 in tuning procedure.

10.3.7.1.33 Command Queue (CQE)

10.3.7.1.33.1 Offset

Register	Offset
CQE	100h

10.3.7.1.33.2 Function

All the CQE registers are defined in JESD84-B51.pdf. See the B.4 section of JESD84-B51.pdf

10.3.7.1.33.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0

10.3.7.1.33.4 Fields

Field	Function
31-0 —	Reserved

Chapter 11
Connectivity

11.1 Universal Serial Bus Controller (USB)

11.1.1 Overview

The USB controller block provides high performance USB functionality that conforms to the *Universal Serial Bus Specification*, Rev. 2.0 (Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips; 2000), and the *On-The-Go and Embedded Host Supplement to the USB Revision* 2.0 Specification (Hewlett-Packard Company, Intel Corporation, LSI Corporation, Microsoft Corporation, Renesas Electronics Corporation, ST-Ericsson; 2012).

The following figure is a block diagram of USB.

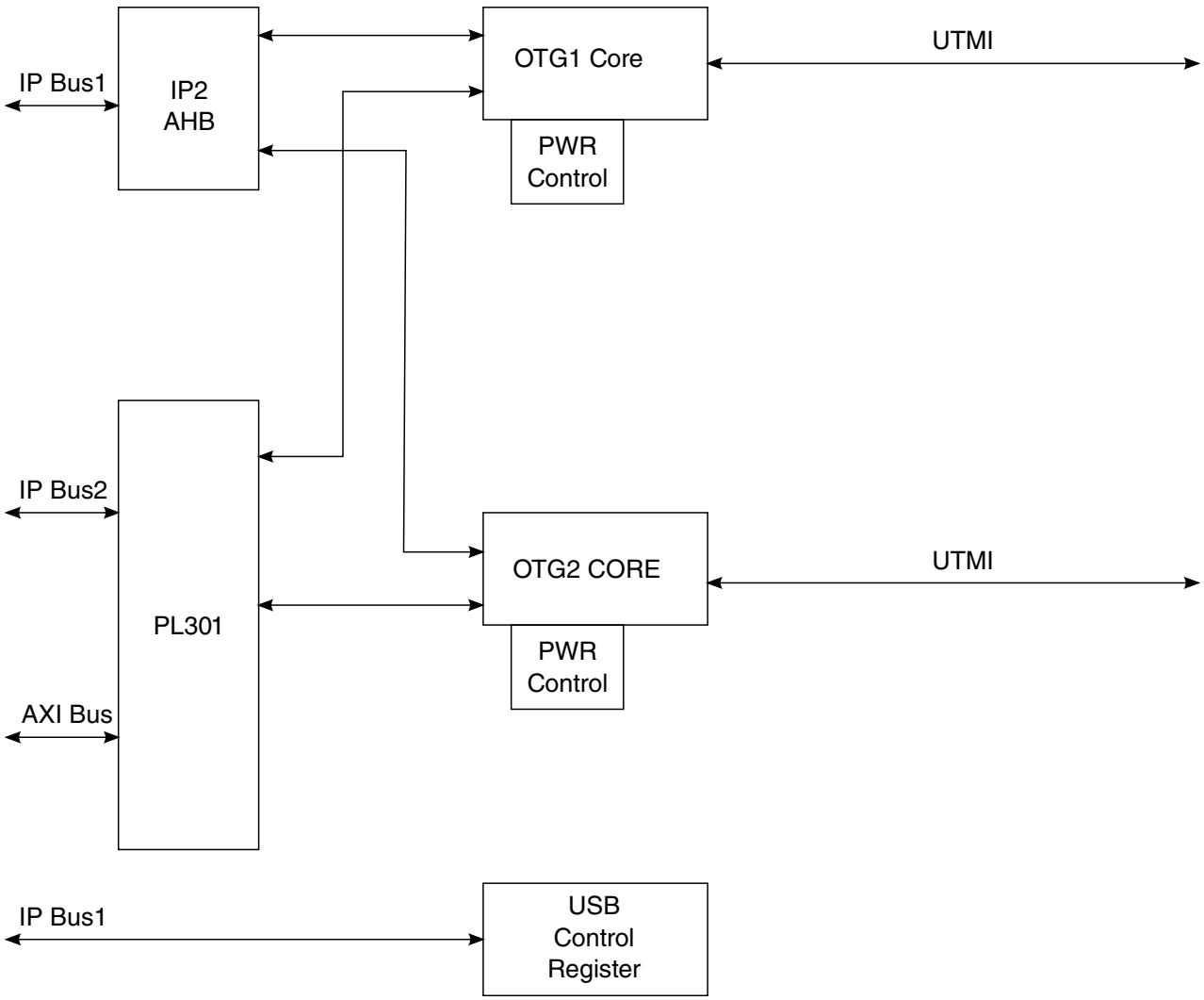


Figure 11-1. USB block diagram

11.1.1.1 Features

There aretwo USB 2.0 controller cores in this chip:

- Controller Core 0 is also named 'OTG1 Core'; its connected port is named 'OTG1 port'.
- Controller Core 1 is also named 'OTG2 Core'; its connected port is named 'OTG2 port'.

The following list provides features of each of the controller cores.

- USB 2.0 Controller Core 0
  - High-Speed/Full-Speed/Low-Speed OTG core
  - HS/FS/LS UTMI compliant interface connected to on-chip UTMI PHY
  - High Speed, Full Speed and Low Speed operation in Host mode (with UTMI transceiver)
  - High Speed, and Full Speed operation in Peripheral mode (with UTMI transceiver)
  - Hardware support for OTG signaling, Session Request Protocol (SRP), Host Negotiation Protocol (HNP), and Attach Detection Protocol (ADP). ADP support includes dedicated timer hardware and register interface.
  - Up to 8 bidirectional endpoints
- USB 2.0 Controller Core 1
  - High-Speed/Full-Speed/Low-Speed OTG core
  - HS/FS/LS UTMI compliant interface connected to on-chip UTMI PHY
  - High Speed, Full Speed and Low Speed operation in Host mode (with UTMI transceiver)
  - High Speed, and Full Speed operation in Peripheral mode (with UTMI transceiver)
  - Hardware support for OTG signaling, Session Request Protocol (SRP), Host Negotiation Protocol (HNP), and Attach Detection Protocol (ADP). ADP support includes basic register interface only.
  - Up to 8 bidirectional endpoints
  - Supports charger detection with register interface only
- Low-power mode with local and remote wake-up capability
- Embedded DMA controller in each core

11.1.1.2 Modes of Operation

The USB has two main modes of operation: normal mode and low power mode.

Each USB OTG controller core can operate in High Speed operation (480 Mbps), Full Speed operation (12 Mbps) and Low Speed operation (1.5 Mbps).

This chapter explains the operation modes.

11.1.1.2.1 Normal Mode

The OTG controller core can operate in Host mode and Device (Peripheral) mode. The Host-only controller core can operate in Host mode only.

Each USB controller core has its corresponding port, which can work in one or more interface modes.

NOTE

Each controller supports only the interface type listed below. Selecting a different interface type in the PORTSC.PTS field results in unpredictable behavior and may cause the system to hang.

- OTG1 port
  - This port supports on-chip UTMI transceiver only.
- OTG2 port
  - This port supports on-chip UTMI transceiver only.

11.1.1.2.2 Low-Power Mode

Each USB controller core has a low-power mode (Suspend mode) to save power consumption.

As described in the USB 2.0 specification, the device can go into the Suspend state after it sees a constant Idle state on the upstream facing port. The OTG controller core enters Suspend mode after 3 ms of inactivity on the port when it is in Device Operation mode. Host controllers, including the OTG controller in Host mode, do not suspend automatically but can be placed in Suspend mode by software.

Either the local Arm platform or the remote USB Host/Peripheral can initiate a wake-up sequence to resume USB communication. For details about Suspend/Resume, see USB Power Control.

11.1.2 Functional Description

These sections describe the functionality of the various building blocks of the USB.

11.1.2.1 USB 2.0 Controller Core 1

The USB 2.0 Controller 1 is an instantiation of an EHCI-compatible core which supports high-, full-, and low-speed operation.

In Host mode, this controller core supports high-, full-, and low-speed operation. In Device mode, it supports high- and full-speed operation.

11.1.2.1.1 Host Mode

The controller supports direct connection of a HS/FS/LS device with on-chip UTMI transceiver.

Although there is no separate Transaction Translator block in the system, the transaction translator function normally associated with a USB 2.0 high speed hub has been implemented within the DMA and protocol engine blocks to support connection to full and low speed devices.

11.1.2.1.2 Peripheral (Device) Mode

- Up to eight bidirectional endpoints
- High/full-speed operation
- Support of HNP, ADP, and SRP
- Remote wake-up capability

11.1.2.2 USB 2.0 Controller Core 1

USB 2.0 Controller Core 1 is an instantiation of EHCI-compatible core which supports High Speed / Full Speed / Low Speed operation .

11.1.2.3 USB Power Control

The USB controller supports suspend and wake-up functionality.

The power control block allows for placing the transceiver in USB low power mode when USB bus is IDLE, and supports local and remote wake-up to bring the transceiver out of USB low power mode when needed. Additionally, the power control block can wake-up the Arm platform from core sleep mode by generating an interrupt.

11.1.2.3.1 Entering Low Power Suspend Mode

In Host operation mode, low power suspend mode is entered as follows:

1. Clear the ASE and PSE bits in USB_USBCMD, and wait until the AS and PS bits in USB_USBSTS become "0".
2. Set the "SUSPEND" bit in USB_PORTSC1
3. Set the "PHCD" bit in USB_PORTSC1
4. Set all PWD bits in USBPHYx_PWD
5. Set CLKGATE in USBPHYx_CTRL

NOTE

Step 3,4,5 shall be done in atomic operation. That is, interrupt should be disabled during these three steps.

For device operation mode, low power suspend mode is entered as follows:

1. After Host drive is IDLE for 3ms, an SLI interrupt is issued (the "DCSUSPEND" or "SLI" bit in USB_USBSTS)
2. Set the "PHCD" bit on USB_PORTSC1
3. Set all PWD bits in USBPHYx_PWD
4. Set CLKGATE in USBPHYx_CTRL

NOTE

Step 2,3,4 shall be done in atomic operation. That is, interrupt should be disabled during these three steps.

11.1.2.3.2 Wake-Up Events

The power control block monitors the USB bus when the USB core is in the USB suspend state.

Depending on whether the core is on Host or Device mode, a number of wake-up conditions are monitored. Upon detection of a wake-up condition, an interrupt (asynchronous) will be generated to Arm platform if the related wake-up interrupt enable bit is set.

USB wake-up interrupt also re-activates the Arm platform clocks if they were stopped during the suspend.

11.1.2.3.2.1 Host Mode Events

The host controller wakes up on the following events:

- Remote Wake-up Request

A peripheral can request the host to reactivate the bus by driving wake-up signaling on the DM/DP lines. The power control block sends a wake-up request to the USB core when a J-K transition on DM/DP line is detected.

- Wake-Up On Overcurrent

If Wake-Up On Overcurrent is enabled (WKOC bit in the USB core register PORTSC1 is set '1'), the power control block sends a wake-up request to the USB core when an overcurrent event is detected.

- Wake-Up On Disconnect

If Wake-Up On Disconnect is enabled (WKDC bit in the USB core register PORTSC1 is set '1'), the power control block sends a wake-up request to the USB core when a disconnection event is detected (J-SE0/K-SE0 transition on DM/DP line).

- Wake-Up On Connect

If a Wake-Up On Connect is enabled (WKCN bit in the USB core register PORTSC1 is set '1'), the power control sub-block sends a wake-up request to the USB core when the connection event is detected (SE0-J/SE0-K transition on DM/DP line).

For a detailed description of register bits WKOC, WKDC, WKCN, please see Port Status & Control (USB_*n*PORTSC1).

11.1.2.4 Interrupts

11.1.2.4.1 USB Core Interrupts

Each USB core uses one dedicated vector in the Interrupt Table. The vector numbers associated with each of the cores can be found in the Interrupt section.

With the exception of the wake-up interrupts, all of the interrupt sources are controlled in the USB Cores. Refer to the Interrupt Enable Register (USB_*n*USBINTR) for details.

11.1.2.4.2 USB Wake-Up Interrupts

Each USB Core has an associated wake-up interrupt. The wake-up interrupts are generated outside of the USB controller cores, but using the same vector as the corresponding USB controller cores interrupt.

These interrupts are generated by the Power Control blocks which run on the 32 KHz standby clock. The wake-up interrupt is designed to work even when the USB and Arm platform clocks are disabled, such that a wake-up condition on the USB bus can re-activate the Arm platform clocks.

Because the wake-up interrupt is generated and cleared on a 32 KHz clock, this interrupt request responds very slowly to clear actions. For this reason, the software must disable the wake-up interrupt to clear the request flag. Disabling the interrupt masks the request instantaneously as this is clocked by the Arm platform clock. The software should wait for at least three 32 KHz clock cycles before re-enabling this interrupt to allow sufficient time for the request flag to clear. Because this interrupt is only used during low power modes of the USB, it is sufficient to enable the wake-up interrupt just prior to entering the USB suspend mode.

11.1.3 USB Operation Model

This section describes the detailed application knowledge for OTG1 and OTG2 ports.

11.1.3.1 Register Interface

Configuration, control and status registers are divided into three categories, identification, capability and operational registers.

NOTE

USB controller registers support only DWORD (32-bit) access.

- Identification registers are used to declare the slave interface presence along with the complete set of the hardware configuration parameters.
- Static, read only capability registers define the software limits, restrictions, and capabilities of the host/device controller.
- Operational registers are dynamic control or status registers that may be read only, read/write, or read/write-to-clear. The following sections define the use of these registers.

EHCI registers are listed alongside device registers to show the complementary nature of host and device control.

The following table describes the Interface register sets.

Table 11-1. Interface Register Sets

Offset	Register Set	Explanation
000h-07Ch	Identification Registers	Identification registers are used to declare the slave interface presence and include a table of the hardware configuration parameters.
100h-124h	Capability Registers	Capability registers specify the limits, restrictions, and capabilities of a host/device controller implementation. These values are used as parameters to the host/device controller driver.

Table continues on the next page...

Table 11-1. Interface Register Sets (continued)

080h-0FCh 140h-1FCh	Operational Registers	Operational registers are used by the system software to control and monitor the operational state of the host/device controller.

11.1.3.1.1 Configuration, Control and Status Register Set

The following table describes the Device/Host capability registers.

NOTE

Depending on implementation, "*x*" can have the following values: UOG1 , UOG2.

Table 11-2. Device/Host Capability Registers

Offset	Size (Bytes)	Mnemonic	Register Name	Device Mode	Host Mode
000h	4	USB_*x*_ID	Identification Register	O	O
004h	4	USB_*x*_HWGENERAL	General Hardware Parameters	O	O
008h	4	USB_*x*_HWHOST	Host Hardware Parameters	X	O
00Ch	4	USB_*x*_HWDEVICE	Device Hardware Parameters	O	X
010h	4	USB_*x*_HWTXBUF	TX Buffer Hardware Parameters	O	O
014h	4	USB_*x*_HWRXBUF	RX Buffer Hardware Parameters	O	O
018-07Fh		-	Reserved		
080h	4	USB_*x*_GPTIMER0LD	General Purpose Timer #0 Load Register	O	O
084h	4	USB_*x*_GPTIMER0CTRL	General Purpose Timer #0 Control Register	O	O
088h	4	USB_*x*_GPTIMER1LD	General Purpose Timer #1 Load Register	O	O
08Ch	4	USB_*x*_GPTIMER1CTRL	General Purpose Timer #1 Control Register	O	O
090h	4	USB_*x*_SBUSCFG	System Bus Interface Configuration Register	O	O
094-09Fh		-	Reserved		
100h	1	USB_*x*_CAPLENGTH	Capability Register Length	O	O
101h		-	Reserved		
102h	2	USB_*x*_HCIVERSION	Host Controller Interface Version Number	X	O
104h	4	USB_*x*_HCSPARAMS	Host Controller Structural Parameters	X	O
108h	4	USB_*x*_HCCPARAMS	Host Controller Capability Parameters	X	O
10C-11Fh		-	Reserved		
120h	2	USB_*x*_DCIVERSION	Device Controller Interface Version Number	O	X
122h	2	-	Reserved		
124h	4	USB_*x*_DCCPARAMS	Device Controller Capability Parameters	O	X
128-13Fh		-	Reserved		
140h	4	USB_*x*_USBCMD	USB Command Register	O	O

Table continues on the next page...

Table 11-2. Device/Host Capability Registers (continued)

Offset	Size (Bytes)	Mnemonic	Register Name	Device Mode	Host Mode
144h	4	USB_*x*_USBSTS	USB Status Register	O	O
148h	4	USB_*x*_USBINTR	USB Interrupt Enable Register	O	O
14Ch	4	USB_*x*_FRINDEX	USB Frame Index	O	O
150h	4	-	Reserved		
154h	4	USB_*x*_PERIODICLISTBASE	Frame List Base Address	X	O
		USB_*x*_DEVICEADDR	USB Device Address	O	X
158h	4	USB_*x*_ASYNCLISTADDR	Next Asynchronous List Address	X	O
	4	USB_*x*_ENDPOINTLISTADDR	Address at Endpoint list in memory	O	X
15Ch	4	-	Reserved		
160h	4	USB_*x*_BURSTSIZE	Programmable Burst Size	O	O
164h	4	USB_*x*_TXFILLTUNING	Host Transmit Pre-Buffer Packet Tuning	X	O
168h	4	-	Reserved		
16Ch	4	USB_*x*_IC_USB	IC_USB enable and voltage negotiation	O	O
170h	4	-	Reserved		
178h	4	USB_*x*_ENDPTNAK	Endpoint NAK register	O	X
17Ch	4	USB_*x*_ENDPTNAKEN	Endpoint NAK Enable register	O	X
180h	4	USB_*x*_CONFIGFLAG	Configured Flag Register	X	O
184h	4	USB_*x*_PORTSC1	Port Status/Control Register 1	O	O
188-1A3h		-	Reserved		
1A4h	4	USB_*x*_OTGSC	On-The-Go Status/Control Register (OTG only)	O	O
1A8h	4	USB_*x*_USBMODE	USB Controller Operating Mode	O	O
1ACh	4	USB_*x*_ENDPTSETUPSTAT	Endpoint Setup Status	O	X
1B0h	4	USB_*x*_ENDPTPRIME	Endpoint Initialization	O	X
1B4h	4	USB_*x*_ENDPTFLUSH	Endpoint De-Initialization	O	X
1B8h	4	USB_*x*_ENDPTSTATUS	Endpoint Status	O	X
1BCh	4	USB_*x*_ENDPTCOMPLETE	Endpoint Complete	O	X
1C0 1C4 ... 1DCh	64	USB_*x*_ENDPTCTRL0 USB_*x*_ENDPTCTRL1 .... USB_*x*_ENDPTCTRL7	Endpoint Control Register 0-7	O	X

NOTE

"O" means the register is available in host/device operation mode;

"X" means the register is reserved in host/device operation mode

11.1.3.1.2 Identification Registers

Identification registers are used to declare the slave interface presence and include a table of the hardware configuration parameters.

11.1.3.1.3 OTG Operations

11.1.3.1.3.1 Register Bits

In the previous section, the Register interface has behaviors described for device mode and behaviors described for host mode. However, for OTG operations it is necessary to perform tasks independent of the controller mode.

NOTE

The only way to transit the controller mode out of host or device mode is with the controller reset bit. Therefore, it is also necessary for the OTG tasks to be performed independent of a controller reset as well as independent of the controller mode.

The following figure shows the controller mode.

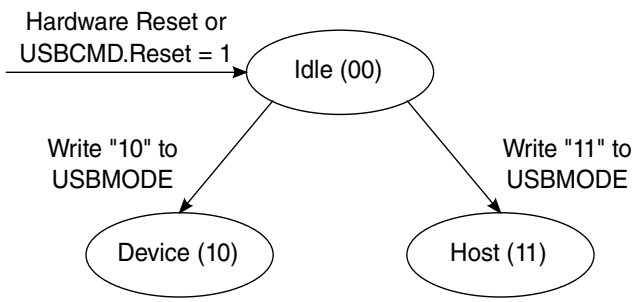


Figure 11-2. Controller Mode

To this end, listed below are the register bits that are used for OTG operations, which are independent of the controller mode and are also not affected by a write to the reset bit in the USBCMD register:

All Identification Registers

All Device/Host Capability Registers

OTGSC: All bits

PORTSC1:

Physical Interface Select

Physical Interface Serial Select

Physical Interface Data Width

Physical Interface Low Power

Physical Interface Wake Signals

Port Indicators

Port Power

11.1.3.2 Host Data Structures

This section defines the interface data structures used to communicate control, status, and data between HCD (software) and the Enhanced Host Controller (hardware).

The data structure definitions in this chapter support a 32-bit memory buffer address space. The interface consists of a Periodic Schedule, Periodic Frame List, Asynchronous Schedule, Isochronous Transaction Descriptors, Split-transaction Isochronous Transfer Descriptors, Queue Heads, and Queue Element Transfer Descriptors.

The periodic frame list is the root of all periodic (isochronous and interrupt transfer type) transfers for the host controller. The asynchronous list is the root for all the bulk and control transfers. Isochronous data streams are managed using Isochronous Transaction Descriptors. Isochronous split-transaction data streams are managed with Split-transaction Isochronous Transfer Descriptors. All Interrupt, Control, and Bulk data streams are managed via queue heads and Queue Element Transfer Descriptors. These data structures are optimized to reduce the total memory footprint of the schedule and to reduce (on average) the number of memory accesses needed to execute a USB transaction.

Note that software must ensure that no interface data structure reachable by the EHCI host controller spans a 4 K-page boundary.

The data structures defined in this section are (from the host controller's perspective) a mix of read-only and read/writeable fields. The host controller must preserve the read-only fields on all data structure writes.

11.1.3.2.1 Periodic Frame List

This schedule is for all periodic transfers (isochronous and interrupt). The periodic schedule is referenced from the operational registers space using the USB_PERIODICLISTBASE address register and the USB_FRINDEX register.

The periodic schedule is based on an array of pointers called the Periodic Frame List.

The USB_PERIODICLISTBASE address register is combined with the USB_FRINDEX register to produce a memory pointer into the frame list. The Periodic Frame List implements a sliding window of work over time.

The following figure shows the organization of periodic schedule.

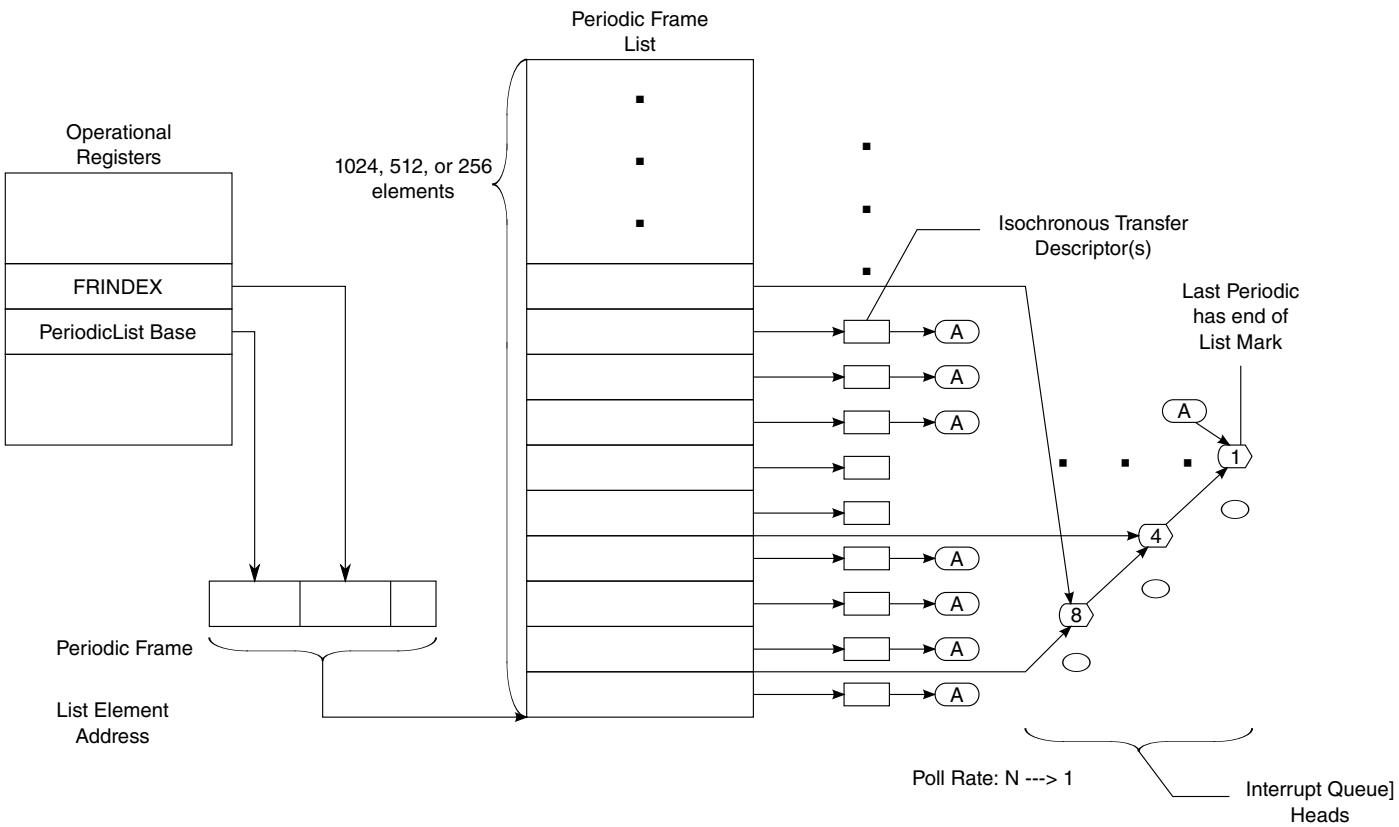


Figure 11-3. Periodic Schedule Organization

Split transaction Interrupt, Bulk and Control are also managed using queue heads and queue element transfer descriptors.

The periodic frame list is a 4 K-page aligned array of Frame List Link pointers. The length of the frame list may be programmable. The programmability of the periodic frame list is exported to system software via the USB_HCCPARAMS register. If non-programmable, the length is 1024 elements. If programmable, the length can be selected by system software as one of 256, 512, or 1024 elements. An implementation must

support all three sizes. Programming the size (that is, the number of elements) is accomplished by system software writing the appropriate value into Frame List Size field in the USB_USBCMD register.

Frame List Link pointers direct the host controller to the first work item in the frame's periodic schedule for the current micro-frame. The link pointers are aligned on DWord boundaries within the Frame List.

The table below illustrates the format of the Frame list element pointer.

Table 11-3. Format of Frame List Element Pointer

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Frame List Link Pointer																											0		Typ			03-00H

Frame List Link pointers always reference memory objects that are 32-byte aligned. The referenced object may be an isochronous transfer descriptor for high-speed devices, a split-transaction isochronous transfer descriptor (for full-speed isochronous endpoints), or a queue head (used to support high-, full- and low-speed interrupt). System software should not place non-periodic schedule items into the periodic schedule. The least significant bits in a frame list pointer are used to key the host controller as to the type of object the pointer is referencing.

The least significant bit is the T-Bit (bit 0). When this bit is set to a one, the host controller never uses the value of the frame list pointer as a physical memory pointer. The Typ field is used to indicate the exact type of data structure being referenced by this pointer. The value encodings are.

Table 11-4. Typ Field Value Definitions

Value	Meaning
00b	Isochronous Transfer Descriptor
01b	Queue Head
10b	Split Transaction Isochronous Transfer Descriptor.
11b	Frame Span Traversal Node.

11.1.3.2.2 Asynchronous List Queue Head Pointer

The Asynchronous Transfer List (based at the USB_ASYNCLISTADDR register) is where all of the control and bulk transfers are managed.

Host controllers use this list only when it reaches the end of the periodic list, the periodic list is disabled, or the periodic list is empty.

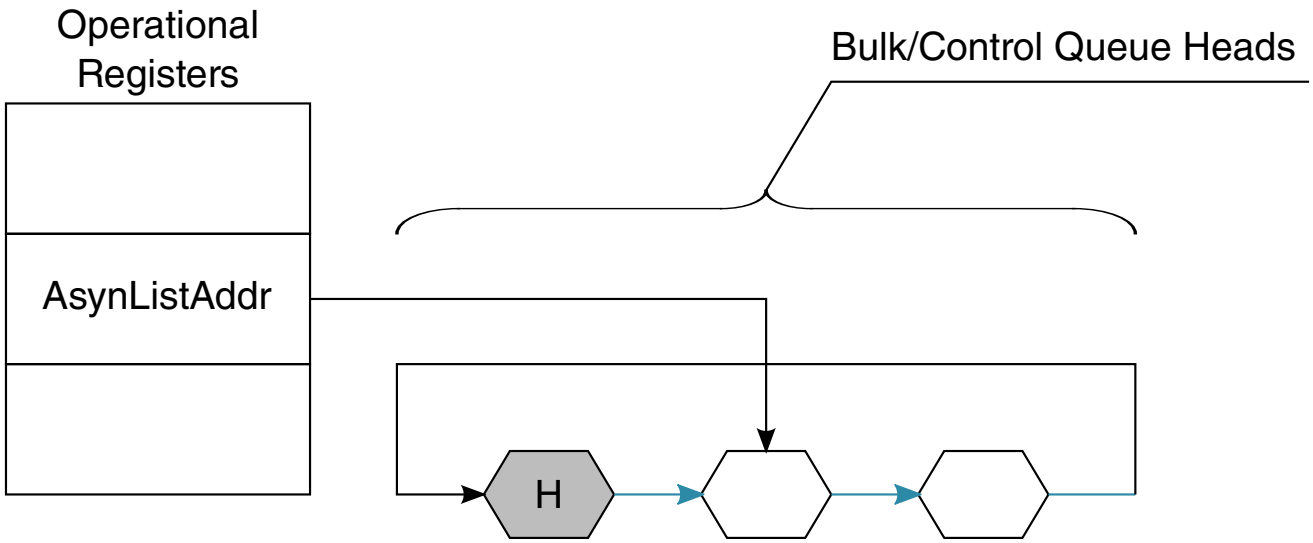


Figure 11-4. Asynchronous Schedule Organization

The Asynchronous list is a simple circular list of queue heads. The USB_ASYNCLISTADDR register is simply a pointer to the next queue head. This implements a pure round-robin service for all queue heads linked into the asynchronous list.

11.1.3.2.3 Isochronous (High-Speed) Transfer Descriptor (iTD)

The format of an isochronous transfer descriptor is shown in the table below.

This structure is used only for high-speed isochronous endpoints. All other transfer types should use queue structures. Isochronous TDs must be aligned on a 32-byte boundary.

Table 11-5. Isochronous Transaction Descriptor (iTD)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Next Link Pointer																											0		Typ		T	03-00 H
Status				Transaction 0 Length												IOC	PG*			Transaction 0 Offset*												07-04 H
Status				Transaction 1 Length												IOC	PG*			Transaction 1 Offset*												0B-08H
Status				Transaction 2 Length												IOC	PG*			Transaction 2 Offset*												0F-0CH
Status				Transaction 3 Length												IOC	PG*			Transaction 3 Offset*												13-10 H

Table continues on the next page...

Table 11-5. Isochronous Transaction Descriptor (iTD) (continued)

Status	Transaction 4 Length	IOC	PG*	Transaction 4 Offset*					17-14H
Status	Transaction 5 Length	IOC	PG*	Transaction 5 Offset*					1B-18H
Status	Transaction 6 Length	IOC	PG*	Transaction 6 Offset*					1F-1CH
Status	Transaction 7 Length	IOC	PG*	Transaction 7 Offset*					23-20H
Buffer Pointer (Page 0)				EndPt	R	Device Address			27-24H
Buffer Pointer (Page 1)				I/O	Maximum Packet Size				2B-28H
Buffer Pointer (Page 2)				-				Mult	2F-2CH
Buffer Pointer (Page 3)				-					33-30H
Buffer Pointer (Page 4)				-					37-34H
Buffer Pointer (Page 5)				-					3B-38H
Buffer Pointer (Page 6)				-					3F-3CH

Host Controller Read/Write

Host Controller Read Only

These fields may be modified by the host controller if the I/O field indicates an OUT.

11.1.3.2.3.1 Next Link Pointer

The first DWord of an iTD is a pointer to the next schedule data structure.

The following table describes the Next Schedule Element pointer field.

Table 11-6. Next Schedule Element Pointer

Bit	Description
31-5 Link Pointer (LP)	These bits correspond to memory address signals [31:5], respectively. This field points to another Isochronous Transaction Descriptor (iTD/siTD) or Queue Head (QH).

Table continues on the next page...

Table 11-6. Next Schedule Element Pointer (continued)

4-3 Reserved	These bits are reserved and their value has no effect on operation. Software should initialize this field to zero.
2-1 QH/(s)iTD Select (Typ)	This field indicates to the Host Controller whether the item referenced is an iTD, siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor 11b FSTN (frame span traversal node)
0 Terminate (T)	1= Link Pointer field is not valid. 0= Link Pointer field is valid.

11.1.3.2.3.2 iTD Transaction Status and Control List

DWords 1 through 8 are eight slots of transaction control and status.

Each transaction description includes:

- Status results field
- Transaction length (bytes to send for OUT transactions and bytes received for IN transactions).
- Buffer offset. The PG and Transaction X Offset fields are used with the buffer pointer list to construct the starting buffer address for the transaction.

The host controller uses the information in each transaction description plus the endpoint information contained in the first three DWords of the Buffer Page Pointer list, to execute a transaction on the USB.

The following table describes iTD Transaction Status and Control fields.

Table 11-7. iTD Transaction Status and Control

Bit	Description	
31-28 Status	This field records the status of the transaction executed by the host controller for this slot. This field is a bit vector with the following encoding:	
	Bit	Definition
	31	Active. Set to one by software to enable the execution of an isochronous transaction by the Host Controller. When the transaction associated with this descriptor is completed, the Host Controller sets this bit to zero indicating that a transaction for this element should not be executed when it is next encountered in the schedule.
	30	Data Buffer Error. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). If an overrun condition occurs, no action is necessary.
	29	Babble Detected. Set to one by the Host Controller during status update when" babble" is detected during the transaction generated by this descriptor.

Table continues on the next page...

Table 11-7. iTD Transaction Status and Control (continued)

Bit	Description	
	28	Transaction Error (XactErr). Set to one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). This bit may only be set for isochronous IN transactions.
27-16 Transaction X Length	For an OUT, this field is the number of data bytes the host controller sends during the transaction. The host controller is not required to update this field to reflect the actual number of bytes transferred during the transfer. For an IN, the initial value of the endpoint to deliver. During the status update, the host controller writes back the field is the number of bytes the host expects the number of bytes successfully received. The value in this register is the actual byte count (0‡zero length data, 1‡one byte, 2‡two bytes, etc.). The maximum value this field may contain is 0xC00 (3072).	
15 Interrupt On Complete (IOC)	If this bit is set to one, it specifies that when this transaction completes, the Host Controller should issue an interrupt at the next interrupt threshold.	
14-12 Page Select (PG)	These bits are set by software to indicate which of the buffer page pointers the offset field in this slot should be concatenated to produce the starting memory address for this transaction. The valid range of values for this field is 0 to 6.	
11-0 Transaction X Offset	This field is a value that is an offset, expressed in bytes, from the beginning of a buffer. This field is concatenated onto the buffer page pointer indicated in the adjacent *PG* field to produce the starting buffer address for this transaction.	

11.1.3.2.3.3 iTD Buffer Page Pointer List (Plus)

DWords 9-15 of an isochronous transaction descriptor are nominally page pointers (4 K aligned) to the data buffer for this transfer descriptor. This data structure requires the associated data buffer to be contiguous (relative to virtual memory), but allows the physical memory pages to be non-contiguous.

Seven page pointers are provided to support the expression of eight isochronous transfers. The seven pointers allow for 3 (transactions) * 1024 (maximum packet size) * 8 (transaction records) (24576 bytes) to be moved with this data structure, regardless of the alignment offset of the first page.

Because each pointer is a 4 K aligned page pointer, the least significant 12 bits in several of the page pointers are used for other purposes.

The tables below illustrate the field descriptions.

Table 11-8. iTD Buffer Pointer Page 0 (Plus)

Bit	Description
31-12 Buffer Pointer (Page 0)	This is a 4 K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11-8	This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.

Table continues on the next page...

Table 11-8. iTD Buffer Pointer Page 0 (Plus) (continued)

Bit	Description
Endpoint Number (Endpt)	
7 Reserved	Bit reserved for future use and should be initialized by software to zero.
6-0 Device Address	This field selects the specific device serving as the data source or sink.

Table 11-9. iTD Buffer Pointer Page 1 (Plus)

Bit	Description
31-12 Buffer Pointer (Page 1)	This is a 4K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11 Direction (I/O)	0 = OUT; 1 = IN. This field encodes whether the high-speed transaction should use an IN or OUT PID.
10-0 Maximum Packet Size	This directly corresponds to the maximum packet size of the associated endpoint *(wMaxPacketSize)*. This field is used for high-bandwidth endpoints where more than one transaction is issued per transaction description (per micro-frame). This field is used with the *Multi* field to support high-bandwidth pipes. This field is also used for all IN transfers to detect packet babble. Software should not set a value larger than 1024 (400h). Any value larger yields undefined results.

Table 11-10. iTD Buffer Pointer Page 2 (Plus)

Bit	Description
31-12 Buffer Pointer	This is a 4K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11-2 Reserved	This bit reserved for future use and should be set to zero.
1-0 Multi	This field is used to indicate to the host controller the number of transactions that should be executed per transaction description (per micro-frame). The valid values are: Value Meaning 00b Reserved. A zero in this field yields undefined results. 01b One transaction to be issued for this endpoint per micro- frame. 10b Two transactions to be issued for this endpoint per micro- frame. 11b Three transactions to be issued for this endpoint per micro- frame.

Table 11-11. iTD Buffer Pointer Page 3-6

Bit	Description
31-12 Buffer Pointer	This is a 4 K aligned pointer to physical memory. Corresponds to memory address bits [31:12].

Table continues on the next page...

Table 11-11. iTD Buffer Pointer Page 3-6 (continued)

Bit	Description
11-0 Reserved	These bits reserved for future use and should be set to zero.

11.1.3.2.4 Split Transaction Isochronous Transfer Descriptor (siTD)

All Full-speed isochronous transfers through the internal transaction translator are managed using the siTD data structure. This data structure satisfies the operational requirements for managing the split transaction protocol.

The following table shows the Split Transaction Isochronous Transfer Descriptor (siTD).

Table 11-12. Split Transaction Isochronous Transfer Descriptor

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Next Link Pointer																											0		Typ		T	03-00
I/O	Port Number							-	Hub Addr							Reserved				EndPt				-	Device Address							07-04[1]
Reserved																µFrame C-mask								µFrame S-mask								0B-08[1]
ioc	P	Reserved				Total Bytes to Transfer										µFrame C-prog-mask								Status								0F-0C[2]
Buffer Pointer (Page 0)																				Current Offset												13-10[2]
Buffer Pointer (Page 1)																				Reserved							TP		T-count			17-14[2]
Back Pointer																											0				T	1B-18

1. 04-0B: Static Endpoint State
2. 0C-13: Transfer results

Host Controller Read/Write

Host Controller Read Only

11.1.3.2.4.1 Next Link Pointer

DWord0 of a siTD is a pointer to the next schedule data structure.

The following table describes the Next Link Pointer fields.

Table 11-13. Next Link Pointer

Bit	Description
31-5	Next Link Pointer (LP). This field contains the address of the next data object to be processed in the periodic list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved. These bits must be written as zeros.
2-1	QH/(s)iTD Select (Typ). This field indicates to the Host Controller whether the item referenced is an iTD/siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor 11b FSTN (frame span traversal node)
0	Terminate (T). 1 = Link Pointer field is not valid. 0 = Link Pointer is valid.

11.1.3.2.4.2 siTD Endpoint Capabilities/Characteristics

DWords 1 and 2 specify static information about the full-speed endpoint, the addressing of the parent Companion Controller, and micro-frame scheduling control.

The tables below describe the Endpoint and transaction translator characteristics and micro-frame schedule control fields.

Table 11-14. Endpoint and Transaction Translator Characteristics

Bit	Description
31	Direction (I/O).0 = OUT; 1 = IN. This field encodes whether the full-speed transaction should be an IN or OUT.
30-24	Port Number. This field is the port number of the recipient Transaction Translator.
23	Reserved. Bit reserved and should be set to zero.
22-16	Hub Address. This field holds the device address of the Companion Controllers' hub.
15-12	Reserved. Field reserved and should be set to zero.
11-8	Endpoint Number (Endpt). This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.
7	Reserved. Bit is reserved for future use. It should be set to zero.
6-0	Device Address. This field selects the specific device serving as the data source or sink.

Table 11-15. Micro-frame Schedule Control

Bit	Description
31-16	Reserved. This field reserved for future use. It should be set to zero.

Table continues on the next page...

Table 11-15. Micro-frame Schedule Control (continued)

Bit	Description
15-8	Split Completion Mask (mFrame C-Mask). This field (along with the *Active* and *SplitX- state* fields in the *Status* byte) is used to determine during which micro-frames the host controller should execute complete-split transactions. When the criteria for using this field is met, an all zeros value has undefined behavior. The host controller uses the value of the three low-order bits of the FRINDEX register to index into this bit field. If the FRINDEX register value indexes to a position where the m*Frame C-Mask* field is a one, then this siTD is a candidate for transaction execution. There may be more than one bit in this mask set.
7-0	Split Start Mask (mFrame S-mask). This field (along with the *Active* and *SplitX-state* fields in the *Status* byte) is used to determine during which micro-frames the host controller should execute start-split transactions. The host controller uses the value of the three low-order bits of the FRINDEX register to index into this bit field. If the FRINDEX register value indexes to a position where the m*Frame S-mask* field is a one, then this siTD is a candidate for transaction execution. An all zeros value in this field, in combination with existing periodic frame list has undefined results.

11.1.3.2.4.3 siTD Transfer State

DWords 3-6 are used to manage the state of the transfer.

The following table describes siTD transfer state fields.

Table 11-16. siTD Transfer Status and Control

Bit	Description
31	Interrupt On Complete (ioc). 0 = Do not interrupt when transaction is complete. 1 = Do interrupt when transaction is complete. When the host controller determines that the split transaction has completed it asserts a hardware interrupt at the next interrupt threshold.
30	Page Select (P). Used to indicate which data page pointer should be concatenated with the *CurrentOffset* field to construct a data buffer pointer (0 selects *Page 0* pointer and 1 selects *Page 1*). The host controller is not required to write this field back when the siTD is retired (*Active* bit transitioned from a one to a zero).
29-26	Reserved. This field reserved for future use and should be set to zero.
25-16	Total Bytes To Transfer. This field is initialized by software to the total number of bytes expected in this transfer. Maximum value is 1023 (3FFh)
15-8	µFrame Complete-split Progress Mask (C-prog-Mask). This field is used by the host controller to record which split-completes has been executed.
7-0: Status—This field records the status of the transaction executed by the host controller for this slot. It is a bit vector with the encoding shown in the following rows.	
7	Active. Set to one by software to enable the execution of an isochronous split transaction by the Host Controller.
6	ERR. Set to a one by the Host Controller when an ERR response is received from the Companion Controller.
5	Data Buffer Error. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). In the case of an under run, the Host Controller transmits an incorrect CRC (thus invalidating the data at the endpoint). If an overrun condition occurs, no action is necessary.
4	Babble Detected. Set to a one by the Host Controller during status update when" babble" is detected during the transaction generated by this descriptor.

Table continues on the next page...

Table 11-16. siTD Transfer Status and Control (continued)

Bit	Description
3	Transaction Error (XactErr). Set to a one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). This bit is set only for IN transactions.
2	Missed Micro-Frame. The host controller detected that a host-induced hold- off caused the host controller to miss a required complete-split transaction.
1	Split Transaction State (SplitXstate). The bit encodings are: Value Meaning 00b Do Start Split. This value directs the host controller to issue a Start split transaction to the endpoint when a match is encountered in the S-mask. 01b Do Complete Split. This value directs the host controller to issue a Complete split transaction to the endpoint when a match is encountered in the C-mask.
0	Reserved. Bit reserved for future use and should be set to zero.

11.1.3.2.4.4 siTD Buffer Pointer List (plus)

DWords 4 and 5 are the data buffer page pointers for the transfer. This structure supports one physical page cross. The most significant 20 bits of each DWord in this section are the 4 K (page) aligned buffer pointers.

The least significant 12 bits of each DWord are used as additional transfer state. The following table describes the siTD buffer pointer fields.

Table 11-17. Buffer Page Pointer List (plus)

Bit	Description
31-12	Buffer Pointer List. Bits [31:12] of DWords 4 and 5 are 4 K page aligned physical memory addresses. These bits correspond to physical address bits [31:12] respectively. The lower 12 bits in each pointer are defined and used as specified below. The field *P* (see siTD Transfer State) specifies the *current* active pointer.
Bits 11-0 (Page 0)	Current Offset—The 12 least significant bits of the Page 0 pointer are the current byte offset for the current page pointer (as selected with the page indicator bit (*P* field). The host controller is not required to write this field back when the siTD is retired (*Active* bit transitioned from a one to a zero).
Bits 11-0 (Page 1)—The least significant bits of the Page 1 pointer are split into three subfields as shown in the following rows.	
11-5 (Page 1)	Reserved
4-3 (Page 1)	Transaction position (TP). This field is used with T-count to determine whether to send *all*, *first*, *middle*, or *last* with each outbound transaction payload. System software must initialize this field with the appropriate starting value. The host controller must correctly manage this state during the lifetime of the transfer. The bit encodings are: Value Meaning

Table continues on the next page...

Table 11-17. Buffer Page Pointer List (plus) (continued)

Bit	Description
	00b All. The entire full-speed transaction data payload is in this transaction (that is, less than or equal to 188 bytes). 01b Begin. This is the first data payload for a full- speed that is greater than 188 bytes. 10B Mid. This is the *middle* payload for a full-speed OUT transaction that is larger than 188 bytes. 11b End. This is the *last* payload for a full-speed OUT transaction that was larger than 188 bytes.
2-0 (Page 1)	Transaction count (T-Count). Software initializes this field with the number of OUT start-splits this transfer requires. Any value larger than 6 is undefined.

11.1.3.2.4.5 siTD Back Link Pointer

DWord 6 of a siTD is simply another schedule link pointer. This pointer is always zero, or references a siTD, and it cannot reference any other schedule data structure.

The following table describes the siTD back link pointer fields.

Table 11-18. siTD Back Link Pointer

Bit	Description
31-5	siTD Back Pointer. This field is a physical memory pointer to a siTD.
4-1	Reserved. This field is reserved for future use. It should be set to zero.
0	Terminate (T). 1 = siTD Back Pointer field is not valid. 0 = siTD Back Pointer field is valid.

11.1.3.2.5 Queue element transfer descriptor (qTD)

This data structure is only used with a queue head. It describes one or more USB transactions to transfer up to 20480 (5*4096) bytes.

The structure contains two structure pointers used for queue advancement, a DWord of transfer state, and a five-element array of data buffer pointers.

It is 32 bytes and must be physically contiguous.

The buffer associated with this transfer must be virtually contiguous. The buffer may start on any byte boundary; however, for optimal utilization of on-chip busses it is recommended to align the buffers on a 32-byte boundary. A separate buffer pointer list element must be used for each physical page in the buffer, regardless of whether the buffer is physically contiguous.

Host controller updates (host controller writes) to stand-alone qTDs only occur during transfer retirement. References in the following bit field definitions of updates to the qTD are to the qTD portion of a queue head.

The following table shows the queue element transfer descriptor data structure.

Table 11-19. Queue element transfer descriptor data structure

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Next qTD Pointer																											0				T	03-00
Alternate Next qTD Pointer																											0				T	07-04
dt	Total Bytes to Transfer															ioc	C_Page			Cerr		PID Code		Status								0B-08[1]
Buffer Pointer (page 0)																				Current Offset												0F-0C[1]
Buffer Pointer (page 1)																				Reserved												13-10
Buffer Pointer (page 2)																				Reserved												17-14
Buffer Pointer (page 3)																				Reserved												1B-18
Buffer Pointer (page 4)																				Reserved												1F-1C

1. 08-0F: Transfer Results

Host Controller Read/Write

Host Controller Read Only

Queue Element Transfer Descriptors must be aligned on 32-byte boundaries.

11.1.3.2.5.1 Next qTD Pointer

The first DWord of an element transfer descriptor is a pointer to another transfer element descriptor.

The following table describes Next qTD pointer fields.

Table 11-20. qTD Next Element Transfer Pointer (DWord 0)

Bit	Description
31-5	Next Transfer Element Pointer. This field contains the physical memory address of the next qTD to be processed. The field corresponds to memory address signals[31:5], respectively.
4-1	Reserved
0	Terminate (T). 1= pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Host Controller that there are no more valid entries in the queue.

11.1.3.2.5.2 Alternate Next qTD Pointer

The second DWord of a queue element transfer descriptor is used to support hardware-only advance of the data stream to the next transfer descriptor on short packet. To be more explicit the host controller always uses this pointer when the current qTD is retired due to short packet.

The following table describes the TD Alternate Next Element Transfer Pointer field descriptions.

Table 11-21. TD Alternate Next Element Transfer Pointer (DWord 1)

Bit	Description
31-5	Alternate Next Transfer Element Pointer. This field contains the physical memory address of the next qTD to be processed in the event that the current qTD execution encounters a short packet (for an IN transaction). The field corresponds to memory address signals [31:5], respectively.
4-1	Reserved
0	Terminate (T). 1= pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Host Controller that there are no more valid entries in the queue.

11.1.3.2.5.3 qTD Token

The third DWord of a queue element transfer descriptor contains most of the information the host controller requires to execute a USB transaction (the remaining endpoint-addressing information is specified in the queue head).

NOTE

The field descriptions forward reference fields defined in the queue head. Where necessary, these forward references are preceded with a QH notation.

The following table describes the TD Token fields.

Table 11-22. TD Token (DWord 2)

Bit	Description
31 Data Toggle	This is the data toggle sequence bit. The use of this bit depends on the setting of the *Data Toggle Control* bit in the queue head.
30-16 Total Bytes to Transfer	This field specifies the total number of bytes to be moved with this transfer descriptor. This field is decremented by the number of bytes actually moved during the transaction, only on the successful completion of the transaction. The maximum value software may store in this field is 5 * 4K (5000H). This is the maximum number of bytes 5 page pointers can access. If the value of this field is zero when the host controller fetches this transfer descriptor (and the active bit is set), the host controller executes a zero-length transaction and retires the transfer descriptor. It is not a requirement for OUT transfers that *Total Bytes To Transfer* be an even multiple of QHD.Maximum Packet Length. If software builds such a transfer descriptor for an OUT transfer, the last transaction is always less than QHD.Maximum Packet Length. Although it is possible to create a transfer up to 20K this assumes the 1st offset into the first page is 0. When the offset cannot be predetermined, crossing past the 5th page can be guaranteed by limiting the total bytes to 16K**. Therefore, the maximum recommended transfer is 16 K(4000H).
15 Interrupt On Complete (IOC)	If this bit is set to a one, it specifies that when this qTD is completed, the Host Controller should issue an interrupt at the next interrupt threshold.
14-12 Current Page (C_Page)	This field is used as an index into the qTD buffer pointer list. Valid values are in the range 0H to 4H. The host controller is not required to write this field back when the qTD is retired.
11-10 Error Counter (CERR)	This field is a 2-bit down counter that keeps track of the number of consecutive Errors detected while executing this qTD. If this field is programmed with a non-zero value during set-up, the Host Controller decrements the count and writes it back to the qTD if the transaction fails. If the counter counts from one to zero, the Host Controller marks the qTD inactive, sets the *Halted* bit to a one, and error status bit for the error that caused *CERR* to decrement to zero. An interrupt is generated if the *USB Error Interrupt Enable* bit in the USBINTR register is set to a one. If HCD programs this field to zero during set-up, the Host Controller does not count errors for this qTD and there is no limit on the retries of this qTD. Note that write-backs of intermediate execution state are to the queue head overlay area, not the qTD. Transaction Error - Decrement Data Buffer Error - No Decrement[3] Stalled - No Decrement[1] Babble Detected - No Decrement[1] No Error - No Decrement[2]

Error	Decrement Counter
1	Detection of Babble or Stall automatically halts the queue head. Thus, count is not decremented
2	If the EPS field indicates a HS device or the queue head is in the Asynchronous Schedule (and *PIDCode* indicates an IN or OUT) and a bus transaction completes and the host controller does not detect a transaction error, then the host controller should reset *CERR* to extend the total number of errors for this transaction. For example, *CERR* should be reset with maximum value (3) on each successful completion of a transaction. The host controller must never reset this field if the value at the start of the transaction is 00b.

Table continues on the next page...

Table 11-22. TD Token (DWord 2) (continued)

Bit	Description	
		See Split Transaction Interrupt for CERR adjustment rules when the EPS field indicates a FS or LS device and the queue head is in the Periodic Schedule. See Asynchronous - Do Complete Split for CERR adjustment rules when the EPS field indicates a FS or LS device, the queue head is in the Asynchronous schedule and the *PIDCode* indicates a SETUP.
	3	Data buffer errors are host problems. They don't count against the device's retries.
	NOTE: Software must not program CERR to a value of zero when the EPS field is programmed with a value indicating a Full- or Low-speed device. This combination could result in undefined behavior.	
9-8 PID Code	This field is an encoding of the token, which should be used for transactions associated with this transfer descriptor. Encodings are:	
	00b	OUT Token generates token (E1H)
	01b	IN Token generates token (69H)
	10b	SETUP Token generates token (2DH) (undefined if endpoint is an interrupt, the queue head is non-zero) transfer type, for example, µ*Frame S-mask* field in.
	11b	Reserved
7-0 Status	This field is used by the Host Controller to communicate individual command execution states back to HCD. This field contains the status of the last transaction performed on this qTD. The bit encodings are:	
	Bit	Status Field Description
	7	Active. Set to one by software to enable the execution of transactions by the Host Controller.
	6	Halted. Set to one by the Host Controller during status updates to indicate that a serious error has occurred at the device/endpoint addressed by this qTD. This can be caused by babble, the error counter counting down to zero, or reception of the STALL handshake from the device during a transaction. Any time that a transaction results in the Halted bit being set to a one, the Active bit is also set to zero.
	5	Data Buffer Error. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). If an overrun condition occurs, the Host Controller forces a timeout condition on the USB, invalidating the transaction at the source. If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.
	4	Babble Detected. Set to a one by the Host Controller during status update when" babble" is detected during the transaction. In addition to setting this bit, the Host Controller also sets the *Halted* bit to a one. Because "babble" is considered a fatal error for the transfer, setting the Halted bit to a one insures that no more transactions occur because of this descriptor.
	3	Transaction Error (XactErr). Set to a one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.

Table continues on the next page...

Table 11-22. TD Token (DWord 2) (continued)

Bit	Description	
	2	Missed Micro-Frame. This bit is ignored unless the *QH.EPS* field indicates a full- or low-speed endpoint and the queue head is in the periodic list. This bit is set when the host controller detected that a host-induced hold-off caused the host controller to miss a required complete-split transaction. If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.
	1	Split Transaction State (SplitXstate). This bit is ignored by the host controller unless the *QH.EPS* field indicates a full- or low-speed endpoint. When a Full- or Low-speed device, the host controller uses this bit to track the state of the split-transaction. The functional requirements of the host controller for managing this state bit and the split transaction protocol depends on whether the endpoint is in the periodic or asynchronous schedule. The bit encodings are: Value Meaning 0b Do Start Split. This value directs the host controller to issue a Start split transaction to the endpoint. 1b Do Complete Split. This value directs the host controller to issue a Complete split transaction to the endpoint.
	0	Ping State (P)/ERR. If the *QH.EPS* field indicates a High-speed device and the *PID_Code* indicates an OUT endpoint, then this is the state bit for the Ping protocol. The bit encodings are: Value Meaning 0b Do OUT. This value directs the host controller to issue an OUT PID to the endpoint. 1b Do Ping. This value directs the host controller to issue a PING PID to the endpoint. If the *QH.EPS* field does not indicate a High-speed device, then this field is used as an error indicator bit. It is set to a one by the host controller whenever a periodic split-transaction receives an ERR handshake.

11.1.3.2.5.4 qTD Buffer Page Pointer List

The last five DWords of a queue element transfer descriptor is an array of physical memory address pointers. These pointers reference the individual pages of a data buffer.

System software initializes Current Offset field to the starting offset into the current page, where current page is selected through the value in the C_Page field.

The following table describes the qTD Buffer Pointer(s) (DWords 3-7) fields.

Table 11-23. qTD Buffer Pointer(s) (DWords 3-7)

Bit	Description
31-12	Buffer Pointer List. Each element in the list is a 4 K page aligned physical memory address. The lower 12 bits in each pointer are reserved (except for the first one), as each memory pointer must reference the start of a 4 K page. The field C_Page specifies the current active pointer. When the transfer element descriptor is fetched, the starting buffer address is selected using C_Page (similar to an array index to

Table continues on the next page...

Table 11-23. qTD Buffer Pointer(s) (DWords 3-7) (continued)

	select an array element). If a transaction spans a 4K buffer boundary, the host controller must detect the page-span boundary in the data stream, increment C_Page and advance to the next buffer pointer in the list, and conclude the transaction through the new buffer pointer.
11-0	Current Offset (Reserved). This field is reserved in all pointers except the first one (for example Page 0). The host controller should ignore all reserved bits. For the page 0 current offset interpretation, this field is the byte offset into the current page (as selected by C_Page). The host controller is not required to write this field back when the qTD is retired. Software should ensure the Reserved fields are initialized to zero.

11.1.3.2.6 Queue Head

The table located in this section shows the Queue Head structure layout.

The following table shows the queue head structure layout.

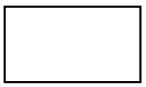
Table 11-24. Queue Head Structure Layout

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Queue Head Horizontal Link Pointer																											0		Typ		T	03-00
RL				C	Maximum Packet Length											H	dtc	EP		EndPt				I	Device Address							07-04[1]
Mult		Port Number[2]						Hub Addr[2]								μFrame C-mask[2]								μFrame S-mask								0B-08[1]
Current qTD Pointer																											0					0F-0C
Next qTD Pointer																											0				T	13-10[3]
Alternate Next qTD pointer																											NakCnt				T	17-14[4]
dt	Total Bytes to Transfer															ioc	C_Page			Cerr		PID Code		Status								1B-18
Buffer Pointer (Page 0)																				Current Offset												1F-1C
Buffer Pointer (Page 1)																				Reserved				C-prog-mask[2]								23-20
Buffer Pointer (Page 2)																				S-bytes[2]										FrameTag[2]		27-24[4]
Buffer Pointer (Page 3)																				Reserved												2B-28
Buffer Pointer (Page 4)																				Reserved												2F-2C[3]

1. 04-0B: Static endpoint state.
2. These fields are used exclusively to support split transactions to USB 2.0 hubs

3. 10-2F: Transfer overlay.
4. 14-27: Transfer results.

Host Controller Read/Write



Host Controller Read Only

11.1.3.2.6.1 Queue Head Horizontal Link Pointer

The first DWord of a Queue Head contains a link pointer to the next data object to be processed after any required processing in this queue has been completed, as well as the control bits defined below.

This pointer may reference a queue head or one of the isochronous transfer descriptors. It must not reference a queue element transfer descriptor.

The following table describes the Queue head DWord 0 fields.

Table 11-25. Queue Head DWord 0

Bit	Description
31-5	Queue Head Horizontal Link Pointer (QHLP). This field contains the address of the next data object to be processed in the horizontal list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	QH/(s)iTD Select (Typ). This field indicates to the hardware whether the item referenced by the link pointer is an iTD, siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor) 11b FSTN (frame span traversal node)
0	Terminate (T). 1=Last QH (pointer is invalid). 0=Pointer is valid. If the queue head is in the context of the periodic list, a one bit in this field indicates to the host controller that this is the end of the periodic list. This bit is ignored by the host controller when the queue head is in the Asynchronous schedule. Software must ensure that queue heads reachable by the host controller always have valid horizontal link pointers.

11.1.3.2.6.2 Queue Head Endpoint Capabilities/Characteristics

The second and third DWords of a Queue Head specifies static information about the endpoint. This information does not change over the lifetime of the endpoint.

There are three types of information in this region:

- Endpoint Characteristics. These are the USB endpoint characteristics including addressing, maximum packet size, and endpoint speed.
- Endpoint Capabilities. These are adjustable parameters of the endpoint. They effect how the endpoint data stream is managed by the host controller.
- Split Transaction Characteristics. This data structure is used to manage full- and low-speed data streams for bulk, control, and interrupt via split transactions to USB2.0 Hub Transaction Translator. There are additional fields used for addressing the hub and scheduling the protocol transactions (for periodic).

The host controller must not modify the bits in this region.

The following table describes the Endpoint characteristics: Queue head DWord 1 fields.

Table 11-26. Endpoint Characteristics: Queue Head DWord 1

Bit	Description	
31-28	Nak Count Reload (RL). This field contains a value, which is used by the host controller to reload the Nak Counter field.	
27	Control Endpoint Flag (C). If the *QH.EPS* field indicates the endpoint is not a high-speed device, and the endpoint is a control endpoint, then software must set this bit to a one. Otherwise, it should always set this bit to zero.	
26-16	Maximum Packet Length. This directly corresponds to the maximum packet size of the associated endpoint (*wMaxPacketSize*). The maximum value this field may contain is 0x400 (1024).	
15	Head of Reclamation List Flag (H). This bit is set by System Software to mark a queue head as being the head of the reclamation list.	
14	Data Toggle Control (DTC). This bit specifies where the host controller should get the initial data toggle on an overlay transition. 0b Ignore DT bit from incoming qTD. Host controller preserves DT bit in the queue head. 1b Initial data toggle comes from incoming qTD DT bit. Host controller replaces DT bit in the queue head from the DT bit in the qTD.	
13-12	Endpoint Speed (EPS). This is the speed of the associated endpoint. Bit combinations are:	
	Value	Meaning
	00b	Full-Speed (12 Mbits/sec)
	01b	Low-Speed (1.5 Mbits/sec)
	10b	High-Speed (480 Mbits/sec)
	11b	Reserved
	This field must not be modified by the host controller.	
11-8	Endpoint Number (Endpt). This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.	
7	Inactivate on Next Transaction (I). This bit is used by system software to request that the host controller set the Active bit to zero. See Rebalancing the periodic schedule , for full operational details. This field is only valid when the queue head is in the Periodic Schedule and the *EPS* field indicates a Full or Low-speed endpoint. Setting this bit to one when the queue head is in the Asynchronous Schedule or the *EPS* field indicates a high-speed device yields undefined results.	
6-0	Device Address. This field selects the specific device serving as the data source or sink.	

The table below describes the Endpoint capabilities: Queue head DWord 2 field descriptions.

Table 11-27. Endpoint Capabilities: Queue Head DWord 2

Bit	Description
31-30	High-Bandwidth Pipe Multiplier (Mult). This field is a multiplier used to key the host controller as the number of successive packets the host controller may submit to the endpoint in the current execution. The host controller makes the simplifying assumption that software properly initializes this field (regardless of location of queue head in the schedules or other run time parameters). The valid values are: Value Meaning 00b Reserved. A zero in this field yields undefined results. 01b One transaction to be issued for this endpoint per micro-frame. 10b Two transactions to be issued for this endpoint per micro-frame. 11b Three transactions to be issued for this endpoint per micro-frame.
29-23	Port Number. This field is ignored by the host controller unless the *EPS* field indicates a full- or low-speed device. The value is the port number identifier on the USB 2.0 Hub (for hub at device address *Hub Addr* below), below which the full- or low-speed device associated with this endpoint is attached. This information is used in the split-transaction protocol.
22-16	Hub Addr. This field is ignored by the host controller unless the *EPS* field indicates a full-or low-speed device. The value is the USB device address of the USB 2.0 Hub below which the full- or low-speed device associated with this endpoint is attached. This field is used in the split-transaction protocol.
15-8	Split Completion Mask (µFrame C-Mask). This field is ignored by the host controller unless the *EPS* field indicates this device is a low- or full-speed device and this queue head is in the periodic list. This field (along with the *Active* and *SplitX-state* fields) is used to determine during which micro-frames the host controller should execute a complete-split transaction. When the criteria for using this field are met, a zero value in this field has undefined behavior. This field is used by the host controller to match against the three low-order bits of the FRINDEX register. If the FRINDEX register bits decode to a position where the µ*Frame C- Mask* field is a one, then this queue head is a candidate for transaction execution. There may be more than one bit in this mask set.
7-0	Interrupt Schedule Mask (µFrame S-mask). This field is used for all endpoint speeds. Software should set this field to a zero when the queue head is on the asynchronous schedule. A non-zero value in this field indicates an interrupt endpoint. The host controller uses the value of the three low-order bits of the FRINDEX register as an index into a bit position in this bit vector. If the µ*Frame S-mask* field has a one at the indexed bit position then this queue head is a candidate for transaction execution. If the *EPS* field indicates the endpoint is a high-speed endpoint, then the transaction executed is determined by the *PID_Code* field contained in the execution area. This field is also used to support split transaction types: Interrupt (IN/OUT). This condition is true when this field is non-zero and the *EPS* field indicates this is either a full- or low-speed device. A zero value in this field, in combination with existing in the periodic frame list has undefined results.

11.1.3.2.6.3 Transfer Overlay-Queue Head

The nine DWords in this area represent a transaction working space for the host controller. The general operational model is that the host controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then it follows the Queue Head Horizontal Link Pointer to the next queue head. The host controller will never follow the Next Transfer Queue Element or Alternate Queue Element pointers unless it is actively attempting to advance the queue. For the

duration of the transfer, the host controller keeps the incremental status of the transfer in the overlay area. When the transfer is complete, the results are written back to the original queue element.

The DWord3 of a Queue Head contains a pointer to the source qTD currently associated with the overlay. The host controller uses this pointer to write back the overlay area into the source qTD after the transfer is complete.

The following table describes the current qTD link pointer field descriptions.

Table 11-28. Current qTD Link Pointer

Bit	Description
31-5	Current Element Transaction Descriptor Link Pointer. This field contains the address Of the current transaction being processed in this queue and corresponds to memory address signals [31:5], respectively.
4-0	Reserved (R). These bits are ignored by the host controller when using the value as an address to write data. The actual value may vary depending on the usage.

The DWords 4-11 of a queue head are the transaction overlay area. This area has the same base structure as a Queue Element Transfer Descriptor. The queue head utilizes the reserved fields of the page pointers to implement tracking the state of split transactions.

This area is characterized as an overlay because when the queue is advanced to the next queue element, the source queue element is merged onto this area. This area serves as execution cache for the transfer.

The table below describes the Host-controller rules for bits in overlay.

Table 11-29. Host-Controller Rules for Bits in Overlay (DWords 5, 6, 8 and 9)

DWord	Bit	Description
5	4-1	Nak Counter (NakCnt)µRW. This field is a counter the host controller decrements whenever a transaction for the endpoint associated with this queue head results in a Nak or Nyet response. This counter is reloaded from *RL* before a transaction is executed during the first pass of the reclamation list (relative to an Asynchronous List Restart condition). It is also loaded from RL during an overlay.
6	31	Data Toggle. The *Data Toggle Control* controls whether the host controller preserves this bit when an overlay operation is performed.
6	15	Interrupt On Complete (IOC). The IOC control bit is always inherited from the source qTD when the overlay operation is performed.
6	11-10	Error Counter (C_ERR). This two-bit field is copied from the qTD during the overlay and written back during queue advancement.
6	0	Ping State (P)/ERR. If the *EPS* field indicates a high-speed endpoint, then this field should be preserved during the overlay operation.
8	7-0	Split-transaction Complete-split Progress (C-prog-mask). This field is initialized to zero during any overlay. This field is used to track the progress of an interrupt split-transaction.
9	4-0	Split-transaction Frame Tag (Frame Tag). This field is initialized to zero during any overlay. This field is used to track the progress of an interrupt split-transaction.

Table continues on the next page...

Table 11-29. Host-Controller Rules for Bits in Overlay (DWords 5, 6, 8 and 9) (continued)

9	11-5	S-bytes. Software must ensure that the *S-bytes* field in a *qTD* is zero before activating the *qTD*. This field is used to keep track of the number of bytes sent or received during an IN or OUT split transaction.

11.1.3.2.7 Periodic Frame Span Traversal Node (FSTN)

This data structure is to be used only for managing Full- and Low-speed transactions that span a Host-frame boundary.

See Host Controller Operational Model for FSTNs for full operational details. Software must not use an FSTN in the Asynchronous Schedule. An FSTN in the Asynchronous schedule results in undefined behavior. Software must not use the FSTN feature with a host controller whose USB_HCIVERSION register indicates a revision implementation below 0096h. FSTNs are not defined for implementations before 0.96 and their use yields undefined results.

Table 11-30. Frame Span Traversal Node Structure Layout

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Normal Path Link Pointer																											0		Typ		T	03-00
Back Path Link Pointer																											0		Typ[1]		T	07-04

1. Must be set to indicate a queue head

Host Controller Read/Write

Host Controller Read Only

11.1.3.2.7.1 FSTN Normal Path Pointer

The first DWord of an FSTN contains a link pointer to the next schedule object. This object can be of any valid periodic schedule data type.

The following table describes the FSTN normal path pointer fields.

Table 11-31. FSTN Normal Path Pointer Field Descriptions

Bit	Description
31-5	Normal Path Link Pointer (NPLP). This field contains the address of the next data object to be processed in the periodic list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved

Table continues on the next page...

Table 11-31. FSTN Normal Path Pointer Field Descriptions (continued)

2-1	QH/(s)iTD/FSTN Select (Typ). This field indicates to the Host Controller whether the item referenced is a iTD/siTD, a QH or an FSTN. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor) 11b FSTN (Frame Span Traversal Node)
0	Terminate (T). 1 = Link Pointer field is not valid. 0 = Link Pointer is valid.

11.1.3.2.7.2 FSTN Back Path Link Pointer

The second DWord of an FTSN node contains a link pointer to a queue head.

If the T-bit in this pointer is zero, then this FSTN is a Save-Place indicator. Its Typ field must be set by software to indicate the target data structure is a queue head. If the T-bit in this pointer is set to one, then this FSTN is the Restore indicator. When the T-bit is one, the host controller ignores the Typ field.

The following table describes the FSTN back path link pointer fields.

Table 11-32. FSTN Back Path Link Pointer Field Descriptions

Bit	Description
31-5	Back Path Link Pointer (BPLP). This field contains the address of a Queue Head. This field corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	Typ. Software must ensure this field is set to indicate the target data structure is a Queue Head. Any other value in this field yields undefined results.
0	Terminate (T). 1=Link Pointer field is not valid (that is the host controller must not use bits [31:5] as a valid memory address). This value also indicates that this FSTN is a Restore indicator. 0=Link Pointer is valid (that is the host controller may use bits [31:5] (in combination with the CTRLDSSEGMENT register if applicable) as a valid memory address). This value also indicates that this FSTN is a Save-Place indicator.

11.1.3.3 Host Operational Model

The general operational model is for the enhanced interface host controller hardware and enhanced interface host controller driver (generally referred to as system software).

Each significant operational feature of the EHCI host controller is discussed in a separate section. Each section presents the operational model requirements for the host controller hardware. Where appropriate, recommended system software operational models for features are also presented.

11.1.3.3.1 Host Controller Initialization

After initial power-on or HCReset (hardware or through HCReset bit in the USB_USBCMD register), all of the operational registers are at their default values. After a hardware reset, only the operational registers not contained in the Auxiliary power well are at their default values.

The following table describes the default values of operational registers.

.

Table 11-33. Default Values of Operational Register Space

Operational Register	Default Value (after Reset)
USB_USBCMD	00080000h (00080B00h, if *Asynchronous Schedule Park Capability is one*)
USB_USBSTS	00001000h
USB_USBINTR	00000000h
USB_FRINDEX	00000000h
USB_CTRLDSSEGMENT	00000000h
USB_PERIODICLISTBASE	Undefined
USB_ASYNCLISTADDR	Undefined
USB_CONFIGFLAG	00000000h
USB_PORTSC1	00002000h (w/*PPC* set to one); 00003000h (w/*PPC* set to zero)

To initialize the host controller, software should perform the following steps:

- Write the appropriate value to the USB_USBINTR register to enable the appropriate interrupts.
- Write the base address of the Periodic Frame List to the USB_PERIODICLIST BASE register. If no work items are in the periodic schedule, all elements of the Periodic Frame List should have their T-Bits set to one.
- Write the USB_USBCMD register to set the desired interrupt threshold, frame list size (if applicable) and turn the host controller ON through setting the Run/Stop bit.

At this point, the host controller is up and running and the port registers begin reporting device connects, and so on. System software can enumerate a port through the reset process (where the port is in the enabled state). At this point, the port is active with SOFs occurring down the enabled ports, but the schedules have not enabled. To communicate

with devices through the asynchronous schedule, system software must write the USB_ASYNCLISTADDR register with the address of a control or bulk queue head. Software must then enable the asynchronous schedule by writing one to the Asynchronous Schedule Enable bit in the USB_USBCMD register. To communicate with devices through the periodic schedule, system software must enable the periodic schedule by writing one to the Periodic Schedule Enable bit in the USB_USBCMD register.

NOTE

The schedules can be turned on before the first port is reset (and enabled).

When the USB_USBCMD register is written, system software must ensure the appropriate bits are preserved, depending on the intended operation.

11.1.3.3.2 Port Routing and Control

The EHCI specification defines that a USB 2.0 Host controller is comprised of one high-speed host controller, which implements the EHCI programming interface and 0 to N USB 1.1 companion host controllers.

Companion host controllers (cHCs) may be implementations of either Universal or Open host controller specifications. This configuration is used to deliver the required full USB 2.0-defined port capability; for example, Low-, Full-, and High-speed capability for every port.

NOTE

The USB controllers on do not require nor support companion controllers to support Full and Low Speed device. Full and Low Speed devices are supported within the USB controller by emulating the functionality of a high-speed HUB. Therefore, no port routing is present in the controller. Please refer to Embedded Transaction Translator Function for details.

The following figure illustrates a simple block diagram of the port routing logic and its relationship to the high-speed and companion host controllers within a USB 2.0 host controller.

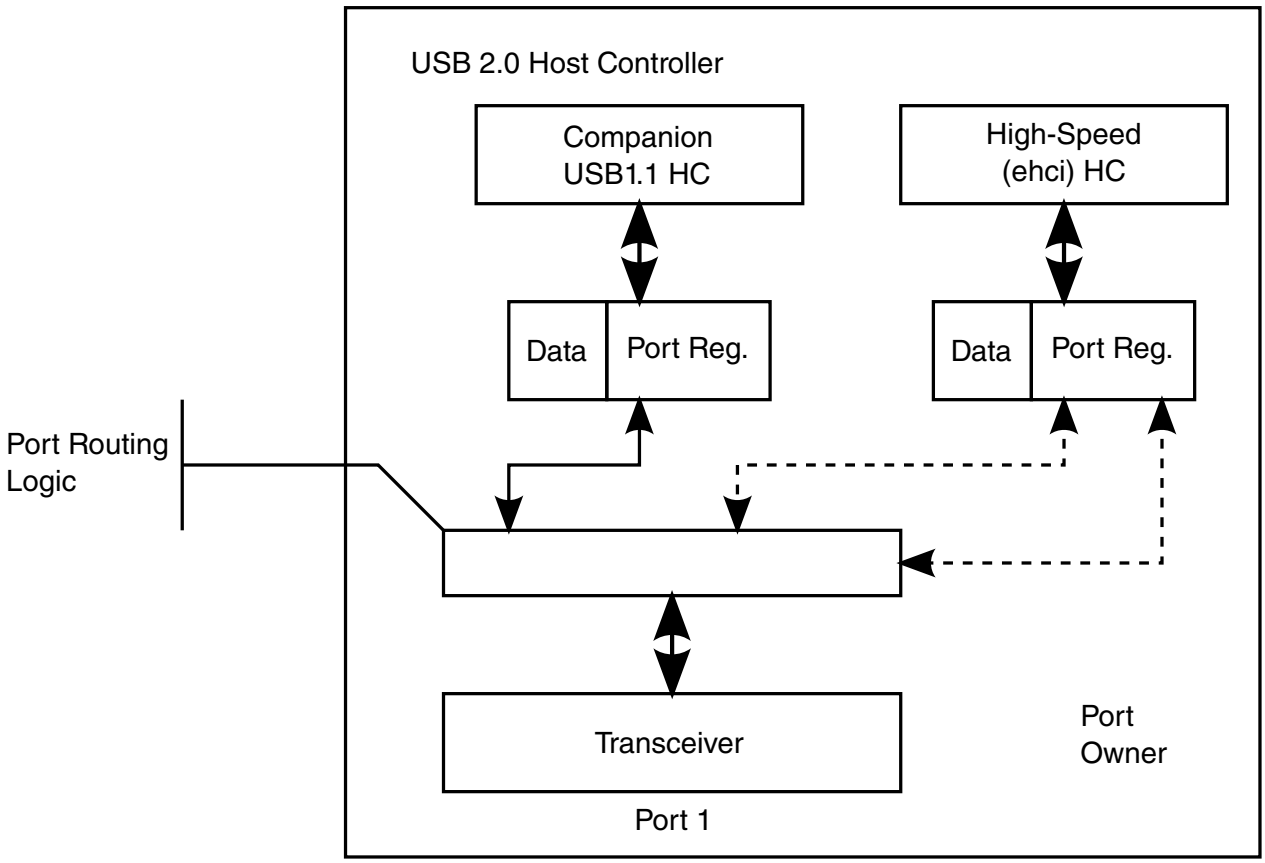


Figure 11-5. Example USB 2.0 Host Controller Port Routing Block Diagram

There exists one transceiver per physical port and each host controller block has its own port status and control registers. The EHCI controller has port status and control registers for every port. Each companion host controller has only the port control and status registers it is required to operate. Either the EHCI host controller or one companion host controller controls each transceiver. Routing logic lies between the transceiver, the port status and control registers.[1]

The port routing logic is controlled from signals originating in the EHCI host controller. The EHCI host controller has a global routing policy control field and per-port ownership control fields. The Configured Flag (CF) bit is the global routing policy control. At power-on or reset, the default routing policy is to the companion controllers (if they exist). If the system does not include a driver for the EHCI host controller and the host controller includes Companion Controllers, then the ports still work in Full- and Low-speed mode (assuming the system includes a driver for the companion controllers). In general, when the EHCI owns the ports, the companion host controllers' port registers do not see a connect indication from the transceiver. Similarly, when a companion host controller owns a port, the EHCI controller's port registers do not see a connect indication

1. The routing logic should not be implemented in the 480 MHz clock domain of the transceiver.

from the transceiver. The details on the rules for the port routing logic are described in the following sections. The USB 2.0 host controller must be implemented as a multi-function PCI device if the implementation includes companion controllers. The companion host controllers' function numbers must be less than the EHCI host controller function number. The EHCI host controller must be a larger function number with respect to the companion host controllers associated with this EHCI host controller. If a PCI device implementation contains only an EHCI controller (that is no companion controllers or other PCI functions), then the EHCI host controller must be function zero, in accordance with the PCI Specification. The N_CC field in the Structural Parameter register (HCSPARAMS) indicates whether the controller implementation includes companion host controllers. When N_CC has a non-zero value there exists companion host controllers. If N_CC has a value of zero, then the host controller implementation does not include companion host controllers. If the host controller root ports are exposed to attachment of full- or low-speed devices, the ports always fails the high-speed chirp during reset and the ports are not enabled. System software can notify the user of the illegal condition. This type of implementation requires a USB 2.0 hub be connected to a root port to provide full and low-speed device connectivity.

System software uses information in the host controller capability registers to determine how the ports are routed to the companion host controllers. See Host Controller Structural Parameters (USB_*n*HCSPARAMS)

11.1.3.3.2.1 Port Routing Control through EHCI Configured (CF) Bit

Each port in the USB 2.0 host controller are routed either to a single companion host controller or to the EHCI host controller.

The port routing logic is controlled by two mechanisms in the EHCI HC: a host controller global flag and per-port control. The Configured Flag (CF) bit, is used to globally set the policy of the routing logic. Each port register has a Port Owner control bit which allows the EHCI Driver to explicitly control the routing of individual ports. Whenever the CF bit transitions from zero to one (this transition is only available under program control) the port routing unconditionally routes all of the port registers to the EHCI HC (all Port Owner bits go to zero). While the CF-bit is one, the EHCI Driver controls individual ports' routing through the Port Owner control bit. Likewise, whenever the CF bit transitions from one to zero (as a result of Aux power application, HCRESET, or software writing zero to CF-bit), the port routing unconditionally routes all of the port registers to the appropriate companion HC. The default value for the EHCI HC's CF bit (after Aux power application or HCRESET) is zero.

The *view* of the port depends on the current owner. A Universal or Open companion host controller will see port register bits consistent with the appropriate specification. Port bit definitions that are required for EHCI host controllers are not visible to companion host controllers.

The following table summarizes the default routing for all the ports, based on the value of the EHCI HC's CF bit.

Table 11-34. Default Port Routing Depending on EHCI HC CF Bit

HS CF Bit	Default Port Ownership	Explanation
0B	Companion HCs	The companion host controllers own the ports and only Full- and Low-speed devices are supported in the system. The exact port assignments are implementation dependent. The ports behave only as Full- and Low-speed ports in this configuration
1B	EHCI HC	The EHCI host controller has default ownership over all of the ports. The routing logic inhibits device connect events from reaching the companion HCs' port status and control registers when the port owner is the EHCI HC.The EHCI HC has access to the additional port status and control bits defined in this specification (see). The EHCI HC can temporarily release control of the port to a companion HC by setting the *PortOwner* bit in the PORTSC1 register to one.

11.1.3.3.2.2 Port Routing Control through PortOwner and Disconnect Event

Manipulating the port routing through the CF-bit is an extreme process and not intended to be used during normal operation.

The normal mode of port ownership transferal is on the granularity of individual ports using the Port Owner bit in the EHCI HC's USB_PORTSC1 register (for hand-offs from EHCI to companion host controllers). Individual port ownership is returned to the EHCI controller when the port registers a device disconnect. When the disconnect is detected, the port routing logic immediately returns the port ownership to the EHCI controller. The companion host controller port register detects the device disconnect and operates normally.

Under normal operating conditions (assuming all HC drivers loaded and operational and the EHCI *CF-bit* is set to one), the typical port enumeration sequence proceeds as illustrated below:

- Initial condition is that EHCI is port owner. A device is connected causing the port to detect a connect, set the port connect change bit and issue a port-change interrupt (if enabled).
- EHCI Driver identifies the port with the new connect change bit asserted and sends a change report to the hub driver. Hub driver issues a GetPortStatus() request and

identifies the connect change. It then issues a request to clear the connect change, followed by a request to reset and enable the port.

- When the EHCI Driver receives the request to reset and enable the port, it first checks the value reported by the LineStatus bits in the USB_PORTSC1 register. If they indicate the attached device is a full-speed device (for example, D+ is asserted), then the EHCI Driver sets the PortReset control bit to one (and sets the PortEnable bit to zero) which begins the reset-process. Software times the duration of the reset, then terminates reset signaling by writing zero to the port reset bit. The reset process is actually complete when software reads zero in the PortReset bit. The EHCI Driver checks the PortOwner bit in the USB_PORTSC1 register. If set to one, the connected device is a high-speed device and EHCI Driver (root hub emulator) issues a change report to the hub driver and the hub driver continues to enumerate the attached device.
- At the time the EHCI Driver receives the port reset and enable request the LineStatus bits might indicate a low-speed device. Additionally, when the port reset process is complete, the PortEnable field may indicate that a full-speed device is attached. In either case the EHCI driver sets the PortOwner bit in the USB_PORTSC1 register to one to release port ownership to a companion host controller.
- When the EHCI Driver sets PortOwner bit to one, the port routing logic makes the connection state of the transceiver available to the companion host controller port register and removes the connection state from the EHCI HC port. The EHCI USB_PORTSC1 register observes and reports a disconnect event through the disconnect change bit. The EHCI Driver detects the connection status change (either by polling or by port change interrupt) and then sends a change report to the hub driver. When the hub driver requests that port-state, the EHCI Driver responds with a reset complete change set to one, a connect change set to one and a connect status set to zero. This information is derived directly from the EHCI port register. This allows the hub driver to assume the device was disconnected during reset. It acknowledges the change bits and wait for the next change event. While the EHCI controller does not own the port, it simply remains in a state where the port reports no device connected. The device-connect evaluation circuitry of the companion HC activates and detects the device, the companion Driver detects the connection and enumerates the port.

When a port is routed to a companion HC, it remains under the control of the companion HC until the device is disconnected from the root port (ignoring for now the scenario where EHCI's CF-bit transitions from1b to 0b). When a disconnect occurs, the disconnect event is detected by both the companion HC port control and the EHCI port ownership control. On the event, the port ownership is returned immediately to the EHCI controller. The companion HC stack detects the disconnect and acknowledges as it would in an ordinary standalone implementation. Subsequent connects is detected by the EHCI port register and the process repeats.

11.1.3.3.2.3 Example Port Routing State Machine

The following figure illustrates an example of how the port ownership should be managed. The following sections describe the entry conditions to each state.

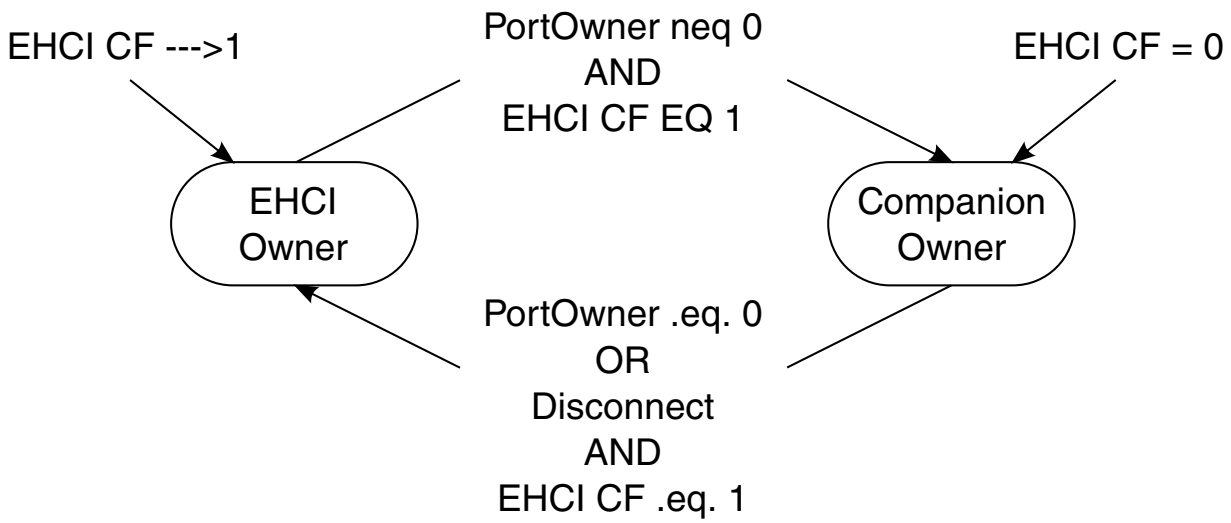


Figure 11-6. Port Owner Handoff State Machine

11.1.3.3.2.3.1 EHCI HC Owner

Entry to this state occurs when one of the following events occur:

- When the EHCI HC's Configure Flag (CF) bit in the USB_CONFIGFLAG register transitions from zero to one. This signals the fact that the system has a host controller driver for the EHCI HC and that all ports in the USB 2.0 host controller must default route to the EHCI controller.
- When the port is owned by a companion HC and the device is disconnected from the port. The EHCI port routing control logic is notified of the disconnect, and returns port routing to the EHCI controller. The connection state of the companion HC goes immediately to the disconnected state (with appropriate side effect to connect change, enable and enable change). The companion HC driver acknowledges the disconnect by setting the connect status change bit to zero. This allows the companion HC's driver to interact with the port completely through the disconnect process.
- When system software writes zero to the PortOwner bit in the USB_PORTSC1 register. This allows software to take ownership of a port from a companion host controller. When this occurs, the routing logic to the companion HC effectively signals a disconnect to the companion HC's port status and control register.

11.1.3.3.2.3.2 Companion HC Owner

Entry to this state occurs whenever one of the following events occur:

- When the PortOwner field transitions from zero to one.
- When the HS-mode HC's Configure Flag (CF) is equal to zero.

On entry to this state, the routing logic allows the companion HC port register to detect a device connect. Normal port enumeration proceeds.

11.1.3.3.2.4 Port Power

The Port Power Control (PPC) bit in the USB_HCSPARAMS register indicates whether the USB 2.0 host controller has port power control (see).

When this bit is zero, then the host controller does not support software control of port power switches. When in this configuration, the port power is always available and the companion host controllers must implement functionality consistent with port power always on. When the *PPC* bit is one, then the host controller implementation includes port power switches. Each available switch has an output enable, which is referred to in this discussion as PortPowerOutputEnable (PPE). PPE is controlled based on the state of the combination bits PPC bit, EHCI Configured (CF)-bit and individual Port Power (PP) bits.

The following table describes the summary behavioral model.

Table 11-35. Port Power Enable Control Rules

CF	CHC[1] (PP)	EHC[2] (PP)	Owner	PPE[3]	Description
0	0	X	CHC	0	When the EHCI controller is not configured, the port is owned by the companion host controller. When the companion HC's port power select is off, then the port power is off.
0	1	X	CHC	1	Similar to previous entry. When the companion HC's port power select is on, then the port power is on.
1	0	0	CHC	0	Port owner has port power turned off, the power to port is off.
1	0	0	EHC	0	Port owner has port power turned off, the power to port is off.
1	0	1	EHC	1	Port owner has port power on, so power to port is on.
1	0	1	CHC	1	If either HC has port power turned on, the power to the port is on.

Table continues on the next page...

Table 11-35. Port Power Enable Control Rules (continued)

1	1	0	EHC	1	If either HC has port power turned on, the power to the port is on.
1	1	0	CHC	1	Port owner has port power on, so power to port is on.
1	1	1	CHC	1	Port owner has port power on, so power to port is on.
1	1	1	EHC	1	Port owner has port power on, so power to port is on.

1. CHC (Companion Host Controller).
2. EHC (EHCI Host Controller).
3. PPE (Port Power Enable). This bit actually turns on the port power switch (if one exists).

11.1.3.3.2.5 Port Reporting Over-Current

Host controllers are by definition power providers on USB. Whether the ports are considered high- or low-powered is a platform implementation issue. Each EHCI USB_PORTSC1 register has an over-current status and over-current change bit.

The functionality of these bits are specified in the USB Specification Revision 2.0.

The over current detection and limiting logic usually resides outside the host controller logic. This logic may be associated with one or more ports. When this logic detects an over-current condition it is made available to both the companion and EHCI ports. The effect of an over-current status on a companion host controller port is beyond the scope of this document.

The over-current condition effects the following bits in the USB_PORTSC1 register on the EHCI port:

- Over-current Active bits are set to one. When the over-current condition goes away, the Over-current Active bit transitions from one to zero.
- Over-current Change bits are set to one. On every transition of the Over-current Active bit the host controller sets the Over-current Change bit to one. Software sets the Over-current Change bit to zero by writing one to this bit.
- Port Enabled/Disabled bit is set to zero. When this change bit gets set to one, then the Port Change Detect bit in the USB_USBSTS register is set to one.
- Port Power (PP) bits may optionally be set to zero. There is no requirement in USB that a power provider shut off power in an over current condition. It is sufficient to limit the current and leave power applied. When the Over-current Change bit transitions from zero to one, the host controller also sets the Port Change Detect bit in the USB_USBSTS register to one. In addition, if the Port Change Interrupt Enable bit in the USB_USBINTR register is one, then the host controller issues an interrupt to the system. Refer to Table 11-36 for summary behavior for over-current detection

when the host controller is halted (suspended from a device component point of view).

11.1.3.3.3 Suspend/Resume-Host Operational Model

The EHCI host controller provides an equivalent suspend and resume model as that defined for individual ports in a USB 2.0 Hub.

Control mechanisms are provided to allow system software to suspend and resume individual ports. The mechanisms allow the individual ports to be resumed completely through software initiation. Other control mechanisms are provided to parameterize the host controller's response (or sensitivity) to external resume events. In this discussion, host-initiated, or software initiated resumes are called Resume Events/Actions. Bus-initiated resume events are called wake-up events. The classes of wake-up events are:

- Remote-wake-up enabled device asserts resume signaling. In similar kind to USB 2.0 Hubs, EHCI controllers must always respond to explicit device resume signaling and wake-up the system (if necessary).
- Port connect and disconnect and over-current events. Sensitivity to these events can be turned on or off by using the per-port control bits in the USB_PORTSC1 registers.

Selective suspend is a feature supported by every USB_PORTSC1 register. It is used to place specific ports into a suspend mode. This feature is used as a functional component for implementing the appropriate power management policy implemented in a particular operating system. When system software intends to suspend the entire bus, it should selectively suspend all enabled ports, then shut off the host controller by setting the Run/Stop bit in the USB_USBCMD register to zero. The EHCI sub-block can then be placed into a lower device state through the PCI power management interface (see Appendix A, Enhanced Host Controller Interface Specification for Universal Serial Bus, Revision 0.95, November 2000, Intel Corporation. http://www.intel.com).

When a wake event occurs, the system resumes operation and system software eventually set the Run/Stop bit to one and resume the suspended ports. Software must not set the Run/Stop bit to one until it is confirmed that the clock to the host controller is stable. This is usually confirmed in a system implementation in that all of the clocks in the system are stable before the Arm platform is restarted. So, by definition, if software is running, clocks in the system are stable and the Run/Stop bit in the USB_USBCMD register can be set to one. Minimum system software delays are also defined in the PCI Power Management Specification. Refer to PCI Power Management Specification for more information.

11.1.3.3.3.1 Port Suspend/Resume

System software places individual ports into suspend mode by writing one into the appropriate USB_PORTSC1 Suspend bit. Software must only set the Suspend bit when the port is in the enabled state (Port Enabled bit is one) and the EHCI is the port owner (PortOwner bit is zero).

The host controller may evaluate the Suspend bit immediately or wait until a micro-frame or frame boundary occurs. If evaluated immediately, the port is not suspended until the current transaction (if one is executing) completes. Therefore, there may be several micro-frames of activity on the port until the host controller evaluates the Suspend bit. The host controller must evaluate the Suspend bit at least every frame boundary.

System software can initiate a resume on a selectively suspended port by writing one to the Force Port Resume bit. Software should not attempt to resume a port unless the port reports that it is in the suspended state (see Port Status & Control (USB_*n*PORTSC1)). If system software sets Force Port Resume bit to one when the port is not in the suspended state, the resulting behavior is undefined. In order to assure proper USB device operation, software must wait for at least 10 ms after a port indicates that it is suspended (Suspend bit is one) before initiating a port resume through the Force Port Resume bit. When Force Port Resume bit is one, the host controller sends resume signaling down the port. System software times the duration of the resume (nominally 20 ms) then sets the Force Port Resume bit to zero. When the host controller receives the write to transition Force Port Resume to zero, it completes the resume sequence as defined in the USB specification, and sets both the Force Port Resume and Suspend bits to zero. Software-initiated port resumes do not affect the Port Change Detect bit in the USB_USBSTS register nor do they cause an interrupt if the Port Change Interrupt Enable bit in the USB_USBINTR register is one. An external USB event may also initiate a resume. The wake events are defined above. When a wake event occurs on a suspended port, the resume signaling is detected by the port and the resume is reflected downstream within 100 µsec. The port's Force Port Resume bit is set to one and the Port Change Detect bit in the USB_USBSTS register is set to one. If the Port Change Interrupt Enable bit in the USB_USBINTR register is one the host controller issues a hardware interrupt.

System software observes the resume event on the port, delays a port resume time (nominally 20 ms), then terminates the resume sequence by writing zero to the Force Port Resume bit in the port. The host controller receives the write of zero to Force Port Resume, terminates the resume sequence and sets Force Port Resume and Suspend port bits to zero. Software can determine that the port is enabled (not suspended) by sampling the USB_PORTSC1 register and observing that the Suspend and Force Port Resume bits are zero. Software must ensure that the host controller is running (that is HCHalted bit in the USB_USBSTS register is zero), before terminating a resume by writing zero to a

port's Force Port Resume bit. If HCHalted is one when Force Port Resume is set to zero, then SOFs do not occur down the enabled port and the device returns to suspend mode in a maximum of 10 msec.

The table below summarizes the wake-up events. Whenever a resume event is detected, the Port Change Detect bit in the USB_USBSTS register is set to one. If the Port Change Interrupt Enable bit is one in the USB_USBINTR register, the host controller generates an interrupt on the resume event. Software acknowledges the resume event interrupt by clearing the Port Change Detect status bit in the USB_USBSTS register.

Table 11-36. Behavior During Wake-up Events

Port Status and Signaling Type	Signaled Port Response	Device State	
		D0	Not D0
Port disabled, resume K-State received	No Effect	N/A	N/A
Port suspended, resume K-State received	Resume reflected downstream on signaled port. Force Port Resume status bit in USB_PORTSC1 register is set to one. Port Change Detect bit in USB_USBSTS register set to one.	[1], [2]	[2]
Port is enabled, disabled or suspended, and the port's WKDSCNNT_E bit is one. A disconnect is detected.	Depending in the initial port state, theUSB_ PORTSC1 Connected Enable status bits are set to zero, and the Connect Change status bit is set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [2]	[2]
Port is enabled, disabled or suspended, and the port's WKDSCNNT_E bit is zero. A disconnect is detected.	Depending on the initial port state, the USB_PORTSC1 Connect and Enable status bits are set to zero, and the Connect Change status bit is set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [3]	[3]
Port is not connected and the port's WKCNNT_E bit is one. A connect is detected.	USB_PORTSC1 Connect Status and Connect Status Change bits are set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [2]	[2]
Port is not connected and the port's WKCNNT_E bit is zero. A connect is detected.	USB_PORTSC1 Connect Status and Connect Status Change bits are set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [3]	[3]
Port is connected and the port's WKOC_E bit is one. An over-current condition occurs.	USB_PORTSC1 Over-current Active, Over-current Change bits are set to one. If Port Enable/Disable bit is one, it is set to zero. Port Change Detect bit in the USB_USBSTS register is set to one	[1], [2]	[2]
Port is connected and the port's WKOC_E bit is zero. An over-current condition occurs.	USB_PORTSC1 Over-current Active, Over-current Change bits are set to one. If Port Enable/Disable bit is one, it is set to zero. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [3]	[3]

[1] Hardware interrupt issued if Port Change Interrupt Enable bit in the USB_USBINTR register is one.

[2] PME# asserted if enabled (Note: PME Status must always be set to one).

[3] PME# not asserted.

11.1.3.3.4 Schedule Traversal Rules

The host controller executes transactions for devices using a simple, shared-memory schedule.

The schedule is comprised of a few data structures, organized into two distinct lists. The data structures are designed to provide the maximum flexibility required by USB, minimize memory traffic and hardware / software complexity.

System software maintains two schedules for the host controller: a periodic schedule and an asynchronous schedule. The root of the periodic schedule is the USB_PERIODICLISTBASE register (see Frame List Base Address (USB_*n*PERIODICLISTBASE))/ Device Address (USB_*n*DEVICEADDR). The USB_PERIODICLISTBASE register is the physical memory base address of the periodic frame list. The periodic frame list is an array of physical memory pointers. The objects referenced from the frame list must be valid schedule data structures as defined in Host Data Structures. In each micro-frame, if the periodic schedule is enabled (see Periodic scheduling threshold) then the host controller must execute from the periodic schedule before executing from the asynchronous schedule. It only executes from the asynchronous schedule after it encounters the end of the periodic schedule. The host controller traverses the periodic schedule by constructing an array offset reference from the USB_PERIODICLISTBASE and the USB_FRINDEX registers (see the following figure). It fetches the element and begins traversing the graph of linked schedule data structures.

The end of the periodic schedule is identified by a next link pointer of a schedule data structure having its T-bit set to one. When the host controller encounters a T-Bit set to one during a horizontal traversal of the periodic list, it interprets this as an End-Of-Periodic-List mark. This causes the host controller to cease working on the periodic schedule and transitions immediately to traversing the asynchronous schedule. After the transition, the host controller executes from the asynchronous schedule until the end of the micro-frame.

The following figure illustrates the derivation of pointer into frame list array.

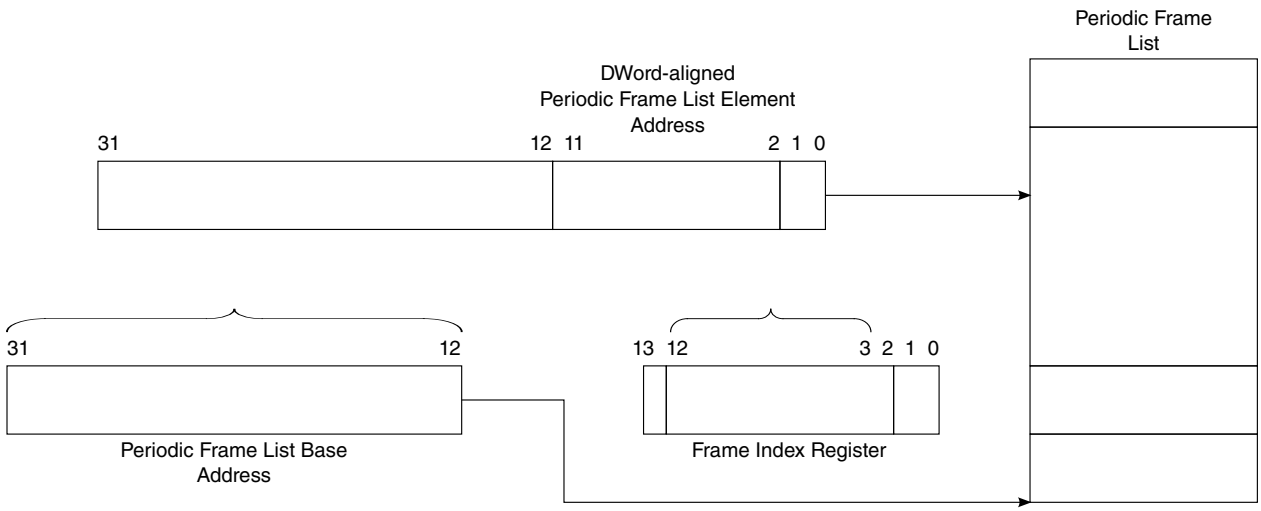


Figure 11-7. Derivation of Pointer into Frame List Array

When the host controller determines that it is the time to execute from the asynchronous list, it uses the operational register USB_ASYNCLISTADDR to access the asynchronous schedule, see the figure below.

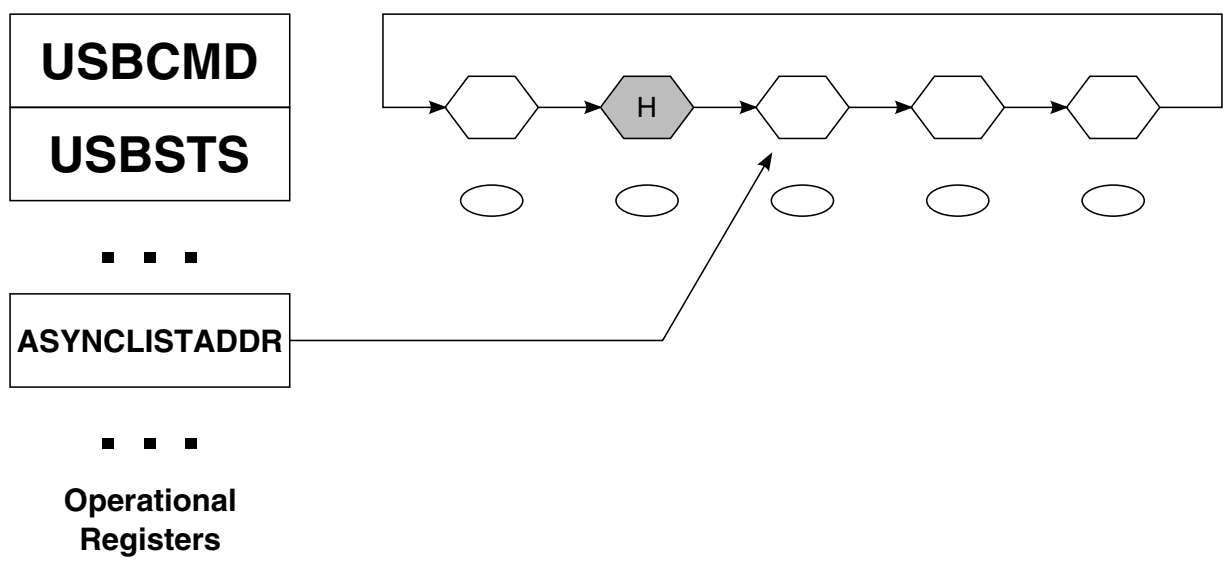


Figure 11-8. General Format of Asynchronous Schedule List

The USB_ASYNCLISTADDR register contains a physical memory pointer to the next queue head. When the host controller makes a transition to executing the asynchronous schedule, it begins by reading the queue head referenced by the USB_ASYNCLISTADDR register. Software must set queue head horizontal pointer T-bits to zero for queue heads in the asynchronous schedule. See Asynchronous Schedule for complete operational details.

11.1.3.3.4.1 Example - Preserving Micro-Frame Integrity

One of the requirements of a USB host controller is to maintain Frame Integrity. This means that the HC must preserve the micro-frame boundaries.

For example, SOF packets must be generated on time (within the specified allowable jitter), and High-speed EOF1,2 thresholds must be enforced. The end of micro-frame timing points EOF1 and EOF2 are clearly defined in the USB Specification Revision 2.0. One implication of this responsibility is that the HC must ensure that it does not start transactions that do not complete before the end of the micro-frame. More precisely, no transactions should be started by the host controller, which do not complete in their entirety before the EOF1 point. In order to enforce this rule, the host controller must check each transaction before it starts to ensure that it completes before the end of the micro-frame.

So, what exactly needs to be involved in this check? Fundamentally, the transaction data payload, plus bit stuffing, plus transaction overhead must be taken into consideration. It is possible to be extremely accurate on how much time the next transaction takes. Take OUTs for an example. The host controller must fetch all of the OUT data from memory in order to send it onto the USB bus. A host controller implementation could pre-fetch all of the OUT data, and pre-compute the actual number of bits in the token and data packets. In addition, the system knows the depth of the target endpoint, so it could closely estimate turnaround time for handshake. In addition, the host controller knows the size of a handshake packet. Pre-computing effects of bit stuffing and summing up the other overhead numbers can allow the host controller to know exactly whether there is enough bus time, before EOF1 to complete the OUT transaction. To accomplish this particular approach takes an inordinate amount of time and hardware complexity.

The alternative is to make a reasonable guess whether the next transaction can be started. An example approximation algorithm is described below. This example algorithm relies on the EHCI policy that periodic transactions are scheduled first in the micro-frame. It is a reasonable assumption that software never over-commits the micro-frame to periodic transactions greater than the specification allowable 80%. In the available remaining 20% bandwidth, the host controller has some ability (in this example) to decide whether or not to execute a transaction. The result of this algorithm is that sometimes, under some circumstances a transaction is not executed that could have been executed. However, under all circumstances, a transaction is never started unless there is enough time in the frame to complete the transaction.

11.1.3.3.4.1.1 Transaction Fit - A Best-Fit Approximation Algorithm

A curve is calculated which represents the latest start time for every packet size, at which software schedules the start of a periodic transaction.

This curve is the 80% bandwidth curve. Another curve is calculated which is the absolute, latest permitted start time for every packet size. This curve represents the absolute latest time, that a transaction of each packet size can be started and completed, in the micro-frame. A plot of these two curves are illustrated in Figure 11-9. The plot Y-axis represents the number of byte-times left in a frame.

The space between the 80% and the Last Start plots is bandwidth reclamation area. In this algorithm the host controller may skip transactions during this time if it is prudent.

The Best-Fit Approximation method plots a function ($f(x)$) between the 80% and Last Start curves. The function $f(x)$ adds a constant to every transaction's maximum packet size and the result compared with the number of bytes left in the frame. The constant represents an approximation of the effects of bit stuffing and protocol overhead. The host controller starts transactions whose results land above the function curve. The host controller will not start transactions whose results land below the function curve.

The following figure illustrates the Best-Fit Approximation.

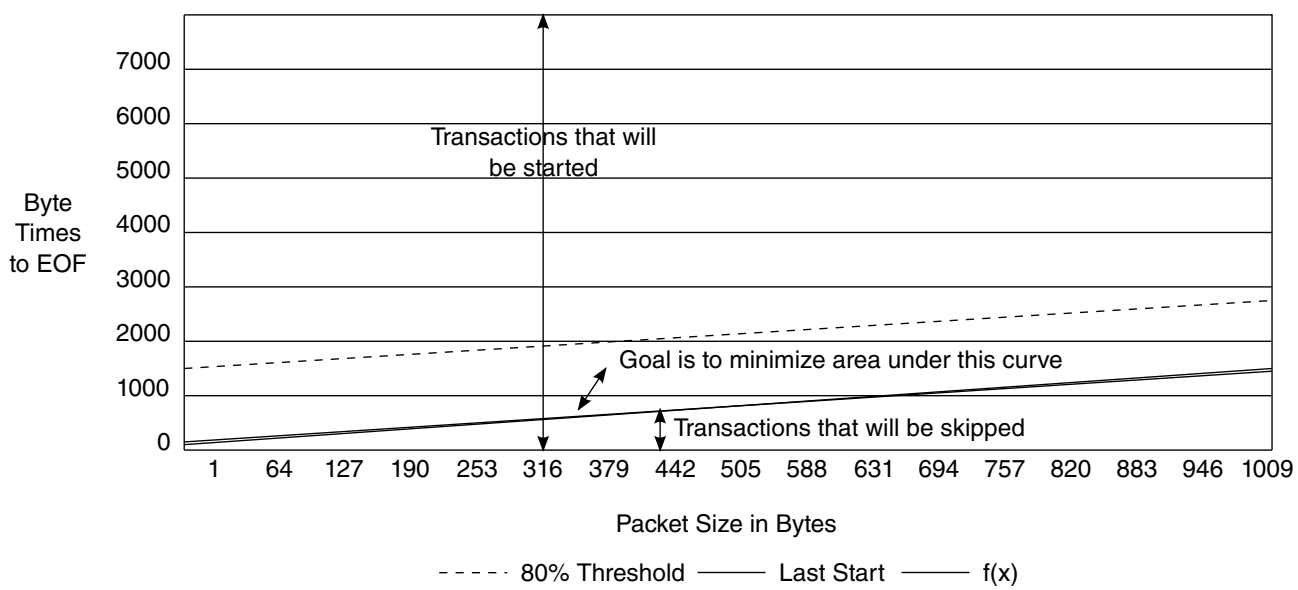


Figure 11-9. Best Fit Approximation

The LastStart line was calculated in this example to assume the absolute worst-case bus overhead per transaction. The particular transaction used is a start-split, zero-length OUT transaction with a handshake. Summaries of the component parts are listed in the table below. The component times were derived from the protocol timings defined in the USB Specification Revision 2.0.

Table 11-37. Example Worse-case Transaction Timing Components

Component	Bit time	Byte Time	Explanation
Split Token	76	9.5	Split token as defined in USB core specification. Includes sync, token, eop, and so on.
Host 2 Host IPG	88	11	Number of bit times required between consecutive host packets.
Token	67	8.375	Token as defined in USB core specification. Includes sync, token, eop, and so on.
Host 2 Host IPG	88	11	Token as defined in USB core specification. Includessync, token, eop, and so on.
Data Packet (0 data bytes)	66.7	8.34	Zero-length data packet. Includes sync, PID, crc16, eop, and so on.
Turnaround time	721	90.125	Time for packet initiator (Host) to see the beginning of a response to a transmitted packet.
Handshake packet	48	6	Handshake packet as defined in USB core specification. Includes sync, PID, eop, and so on.
		144	Total

The exact details of the function (f(x)) are up to the particular implementation. However, it should be obvious that the goal is to minimize the area under the curve between the approximation function and the Last Start curve, without dipping below the LastStart line, while at the same time keeping the check as simple as possible for hardware implementation. The f(x) in Figure 11-9 was constructed using the following pseudo-code test on each transaction size data point. This algorithm assumes that the host controller keeps track of the remaining bits in the frame.

```
Alorithm CheckTransactionWillFit (MaximumPacketSize, HC_BytesLeftInFrame)
Begin
Local Temp = MaximumPacketSize + 192
Local rvalue = TRUE
If MaximumPacketSize >= 128 then
     Temp += 128
End If
If Temp > HC_BytesLeftInFrame then
     Rvalue = FALSE
End If
Return rvalue
End
```

This algorithm takes two inputs, the current maximum packet size of the transaction and the hardware counter of the number of bytes left in the current micro-frame. It unconditionally adds a simple constant of 192 to the maximum packet size to account for a first-order effect of transaction overhead and bit stuffing. If the transaction size is greater than or equal to 128 bytes, then an additional constant of 128 is added to the

running sum to account for the additional worst-case bit stuffing of payloads larger than 128. An inflection point was inserted at 128 because the $f(x)$ plot was getting close to the LastStart line.

11.1.3.3.5 Periodic Schedule Frame Boundaries vs Bus Frame Boundaries

The USB Specification Revision 2.0 requires that the frame boundaries (SOF frame number changes) of the high-speed bus and the full- and low-speed bus(s) below USB 2.0 Hubs be strictly aligned.

Super-imposed on this requirement is that USB 2.0 Hubs manage full- and low-speed transactions through a micro-frame pipeline (see start- (SS) and complete- (CS) splits illustrated in the following figure). A simple, direct projection of the frame boundary model into the host controller interface schedule architecture creates tension (complexity for both hardware and software) between the frame boundaries and the scheduling mechanisms required to service the full- and low-speed transaction translator periodic pipelines.

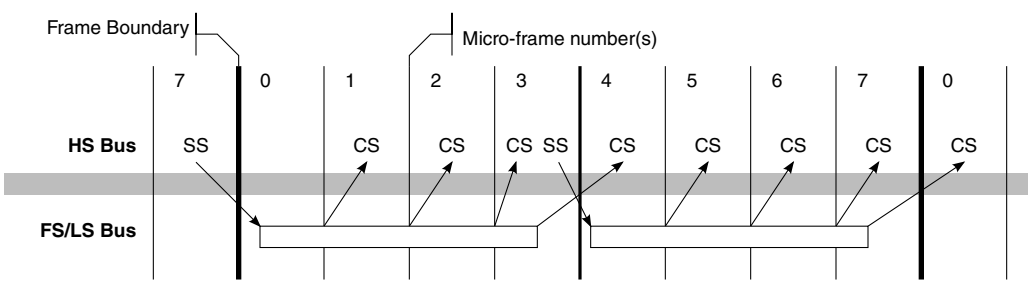


Figure 11-10. Frame Boundary Relationship between HS bus and FS/LS Bus

The simple projection, as the above figure illustrates, introduces frame-boundary wrap conditions for scheduling on both the beginning and end of a frame. In order to reduce the complexity for hardware and software, the host controller is required to implement one micro-frame phase shift for its view of frame boundaries. The phase shift eliminates the beginning of frame and frame-wrap scheduling boundary conditions.

The implementation of this phase shift requires that the host controller use one register value for accessing the periodic frame list and another value for the frame number value included in the SOF token. These two values are separate, but tightly coupled. The periodic frame list is accessed through the Frame List Index Register (USB_FRINDEX) documented in USB Frame Index (USB_*n*FRINDEX) and initially illustrated in Schedule Traversal Rules. Bits FRINDEX[2:0], represent the micro-frame number. The SOF value is coupled to the value of FRINDEX[13:3]. Both FRINDEX[13:3] and the SOF value are increment based on FRINDEX[2:0]. It is required that the SOF value be delayed from the FRINDEX value by one micro-frame. The one micro-frame delay yields host controller

periodic schedule and bus frame boundary relationship as illustrated in the following figure. This adjustment allows software to trivially schedule the periodic start and complete-split transactions for full-and low-speed periodic endpoints, using the natural alignment of the periodic schedule interface. The reasons for selecting this phase-shift are beyond the scope of this specification.

The following figure illustrates how periodic schedule data structures relate to schedule frame boundaries and bus frame boundaries. To aid the presentation, two terms are defined: The host controller's view of the 1 msec boundaries is called H-Frames. The high-speed bus's view of the 1 msec boundaries is called B-Frames.

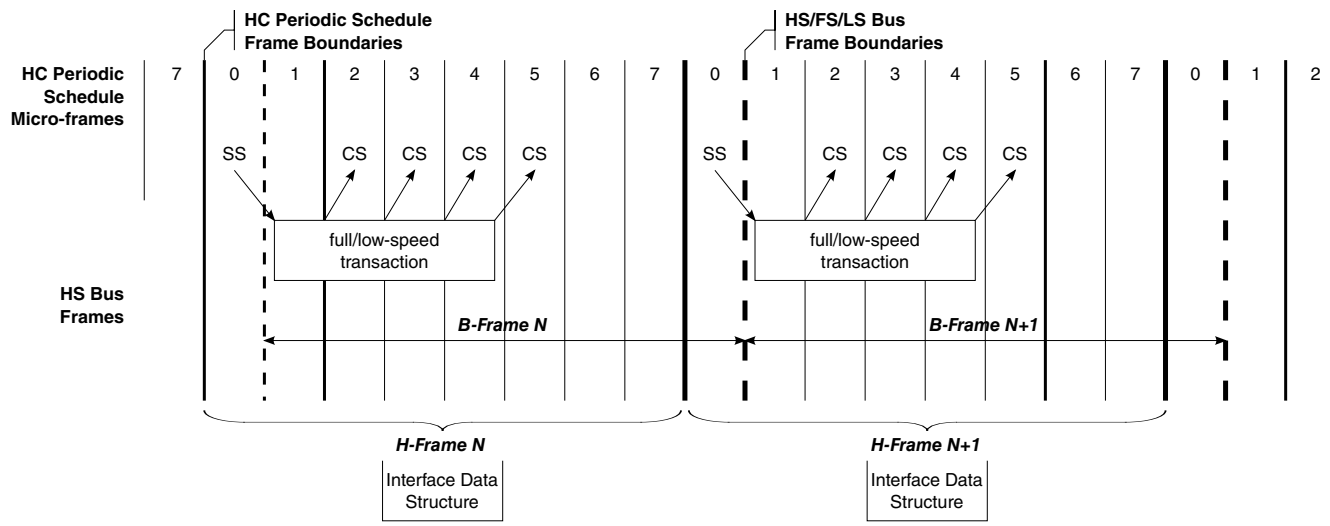


Figure 11-11. Relationship of Periodic Schedule Frame Boundaries to Bus Frame Boundaries

H-Frame boundaries for the host controller correspond to increments of FRINDEX[13:3]. Micro-frame numbers for the H-Frame are tracked by FRINDEX[2:0]. B-Frame boundaries are visible on the high-speed bus through changes in the SOF token's frame number. Micro-frame numbers on the high-speed bus are only derived from the SOF token's frame number (that is the high-speed bus sees eight SOFs with the same frame number value). H-Frames and B-Frames have the fixed relationship (that is B-Frames lag H-Frames by one micro-frame time) illustrated in the figure above. The host controller's periodic schedule is naturally aligned to H-Frames. Software schedules transactions for full- and low-speed periodic endpoints relative the H-Frames. The result is these transactions execute on the high-speed bus at exactly the right time for the USB 2.0 Hub periodic pipeline. As described in USB Frame Index (USB_*n*FRINDEX), the SOF Value can be implemented as a shadow register (in this example, called SOFV), which lags the FRINDEX register bits [13:3] by one micro-frame count. This lag behavior can be

accomplished by incrementing FRINDEX[13:3] based on carry-out on the 7 to 0 increment of FRINDEX[2:0] and incrementing SOFV based on the transition of 0 to 1 of FRINDEX[2:0].

Software is allowed to write to FRINDEX. USB Frame Index (USB_*n*FRINDEX) provides the requirements that software should adhere when writing a new value in FRINDEX.

The table below illustrates the required relationship between the value of FRINDEX and the value of SOFV.

Table 11-38. Operation of FRINDEX and SOFV (SOF Value Register)

Current			Next		
FRINDEX[F]	SOFV	FRINDEX[mF]	FRINDEX[F]	SOFV	FRINDEX[mF]
N	N	111b	N+1	N	000b
N+1	N	000b	N+1	N+1	001b
N+1	N+1	001b	N+1	N+1	010b
N+1	N+1	010b	N+1	N+1	011b
N+1	N+1	011b	N+1	N+1	100b
N+1	N+1	100b	N+1	N+1	101b
N+1	N+1	101b	N+1	N+1	110b
N+1	N+1	110b	N+1	N+1	111b

NOTE

Where [F] = [13:3]; [µF] = [2:0]

11.1.3.3.6 Periodic Schedule

The periodic schedule traversal is enabled or disabled through the Periodic Schedule Enable bit in the USB_USBCMD register. If the Periodic Schedule Enable bit is set to zero, then the host controller simply does not try to access the periodic frame list through the USB_PERIODICLISTBASE register. Likewise, when the Periodic Schedule Enable bit is one, then the host controller does use the USB_PERIODICLISTBASE register to traverse the periodic schedule.

The host controller will not react to modifications to the Periodic Schedule Enable immediately. In order to eliminate conflicts with split transactions, the host controller evaluates the Periodic Schedule Enable bit only when FRINDEX[2:0] is zero. System software must not disable the periodic schedule if the schedule contains an active split transaction work item that spans the 000b micro-frame. These work items must be removed from the schedule before the Periodic Schedule Enable bit is written to zero.

The Periodic Schedule Status bit in the USB_USBSTS register indicates status of the periodic schedule. System software enables (or disables) the periodic schedule by writing one (or zero) to the Periodic Schedule Enable bit in the USB_USBCMD register. Software then can poll the Periodic Schedule Status bit to determine when the periodic schedule has made the desired transition. Software must not modify the Periodic Schedule Enable bit unless the value of the Periodic Schedule Enable bit equals that of the Periodic Schedule Status bit.

The periodic schedule is used to manage all isochronous and interrupt transfer streams. The base of the periodic schedule is the periodic frame list. Software links schedule data structures to the periodic frame list to produce a graph of scheduled data structures. The graph represents an appropriate sequence of transactions.

The following figure illustrates isochronous transfers (using iTDs and siTDs) with a period of one are linked directly to the periodic frame list. Interrupt transfers (are managed with queue heads) and isochronous streams with periods other than one are linked following the period-one iTD/siTDs. Interrupt queue heads are linked into the frame list ordered by poll rate. Longer poll rates are linked first (for example, closest to the periodic frame list), followed by shorter poll rates, with queue heads with a poll rate of one, on the very end.

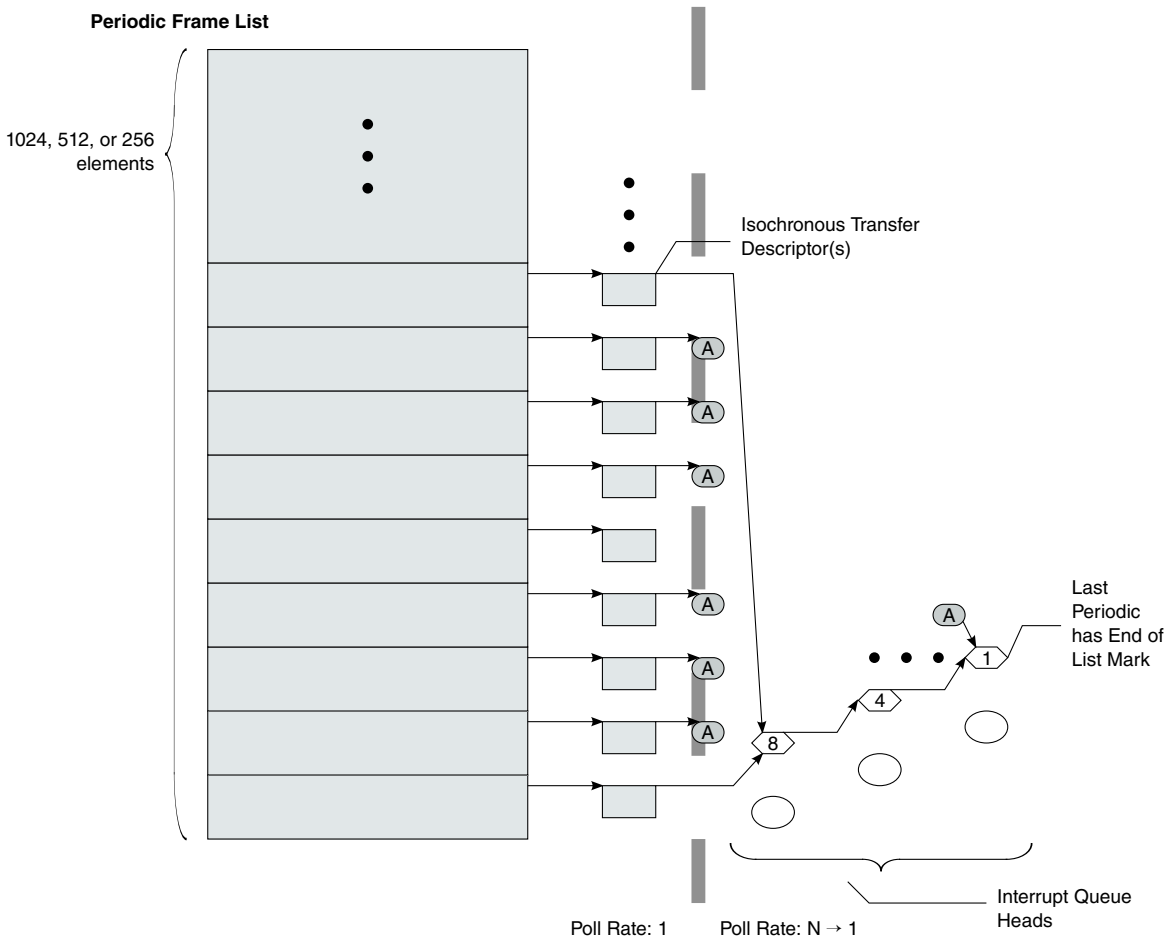


Figure 11-12. Example Periodic Schedule

11.1.3.3.7 Managing Isochronous Transfers Using iTDs

The structure of an iTD is presented in Isochronous (High-Speed) Transfer Descriptor (iTD). The four distinct sections to an iTD:

- The first field is the Next Link Pointer. This field is for schedule linkage purposes only.
- Transaction description array. This area is an eight-element array. Each element represents control and status information for one micro-frame's worth of transactions for a single high-speed isochronous endpoint.
- The buffer page pointer array is a 7-element array of physical memory pointers to data buffers. These are 4 K aligned pointers to physical memory.
- Endpoint capabilities. This area utilizes the unused low-order 12 bits of the buffer page pointer array. The fields in this area are used across all transactions executed for this iTD, including endpoint addressing, transfer direction, maximum packet size and high-bandwidth multiplier.

11.1.3.3.7.1 Host Controller Operational Model for iTDs

The host controller uses FRINDEX register bits [12:3] to index into the periodic frame list. This means that the host controller visits each frame list element eight consecutive times before incrementing to the next periodic frame list element. Each iTD contains eight transaction descriptions, which map directly to FRINDEX register bits [2:0]. Therefore, each transaction descriptor corresponds to one micro-frame. Each iTD can span 8 micro-frames worth of transactions.

When the host controller fetches an iTD, it uses FRINDEX register bits [2:0] to index into the transaction description array.

If the active bit in the Status field of the indexed transaction description is set to zero, the host controller ignores the iTD and follows the Next pointer to the next schedule data structure.

When the indexed active bit is one, the host controller continues to parse the iTD. It stores the indexed transaction description and the general endpoint information (device address, endpoint number, maximum packet size, and so on.). It also uses the Page Select (PG) field to index the buffer pointer array, storing the selected buffer pointer and the next sequential buffer pointer. For example, if PG field is 0, then the host controller stores Page 0 and Page 1.

The host controller constructs a physical data buffer address by concatenating the current buffer pointer (as selected using the current transaction description's PG field) and the transaction description's Transaction Offset field. The host controller uses the endpoint addressing information and I/O-bit to execute a transaction to the appropriate endpoint. When the transaction is complete, the host controller clears the active bit and writes back any additional status information to the Status field in the currently selected transaction description.

The data buffer associated with the iTD must be virtually contiguous memory. Seven page pointers are provided to support eight high-bandwidth transactions regardless of the starting packet's offset alignment into the first page. A starting buffer pointer (physical memory address) is constructed by concatenating the page pointer (for example, page 0 pointer) selected by the active transaction descriptions' PG (for example, value: 00B) field with the transaction offset field. As the transaction moves data, the host controller must detect when an increment of the current buffer pointer crosses a page boundary. When this occurs the host controller simply replaces the current buffer pointer's page portion with the next page pointer (for example, page 1 pointer) and continues to move data. The size of each bus transaction is determined by the value in the Maximum Packet Size field. An iTD supports high-bandwidth pipes through the Mult (multiplier) field. When the Mult field is 1, 2, or 3, the host controller executes the specified number of

Maximum Packet sized bus transactions for the endpoint in the current micro-frame. In other words, the Mult field represents a transaction count for the endpoint in the current micro-frame. If the Mult field is zero, the operation of the host controller is undefined. The transfer description is used to service all transactions indicated by the Mult field.

For OUT transfers, the value of the Transaction X Length field represents the total bytes to be sent during the micro-frame. The Mult field must be set by software to be consistent with Transaction X Length and Maximum Packet Sixe. The host controller sends the bytes in Maximum Packet Size'd portions. After each transaction, the host controller decrements its local copy of Transaction X Length by Maximum Packet Size. The number of bytes the host controller sends is always Maximum Packet Size or Transaction X Length, whichever is less. The host controller advances the transfer state in the transfer description, updates the appropriate record in the iTD and moves to the next schedule data structure. The maximum sized transaction supported is 3 x 1024 bytes.

For IN transfers, the host controller issues Mult transactions. It is assumed that software has properly initialized the iTD to accommodate all of the possible data. During each IN transaction, the host controller must use Maximum Packet Size to detect packet babble errors. The host controller keeps the sum of bytes received in the Transaction X Length field. After all transactions for the endpoint have completed for the micro-frame, Transaction X Length contains the total bytes received. If the final value of Transaction X Length is less than the value of Maximum Packet Size, then less data than was allowed for was received from the associated endpoint. This short packet condition does not set the USBINT bit in the USB_USBSTS register to one. The host controller will not detect this condition. If the device sends more than Transaction X Length or Maximum Packet Size bytes (whichever is less), then the host controller sets the Babble Detected bit to one and set the Active bit to zero. Note, that the host controller is not required to update the iTD field Transaction X Length in this error scenario. If the Mult field is greater than one, then the host controller automatically executes the value of Mult transactions. The host controller will not execute all Mult transactions if:

- The endpoint is an OUT and Transaction X Length goes to zero before all the Mult transactions have executed (ran out of data), or
- The endpoint is an IN and the endpoint delivers a short packet, or an error occurs on a transaction before Mult transactions have been executed. The end of micro-frame may occur before all of the transaction opportunities have been executed. When this happens, the transfer state of the transfer description is advanced to reflect the progress that was made, the result written back to the iTD and the host controller proceeds to processing the next micro-frame. Refer to Appendix D for a table summary of the host controller required behavior for all the high-bandwidth transaction cases.

11.1.3.3.7.2 Software Operational Model for iTDs

A client buffer request to an isochronous endpoint may span 1 to N micro-frames. When N is larger than one, system software may have to use multiple iTDs to read or write data with the buffer (if N is larger than eight, it must use more than one iTD).

The following figure illustrates the simple model of how a client buffer is mapped by system software to the periodic schedule (that is the periodic frame list and a set of iTDs). On the right is the client description of its request. The description includes a buffer base address plus additional annotations to identify which portions of the buffer should be used with each bus transaction. In the middle is the iTD data structures used by the system software to service the client request. Each iTD can be initialized to service up to 24 transactions, organized into eight groups of up to three transactions each. Each group maps to one micro-frame's worth of transactions. The EHCI controller does not provide per-transaction results within a micro-frame. It treats the per-micro-frame transactions as a single logical transfer. On the left is the host controller's frame list. System software establishes references from the appropriate locations in the frame list to each of the appropriate iTDs. If the buffer is large, then system software can use a small set of iTDs to service the entire buffer. System software can activate the transaction description records (contained in each iTD) in any pattern required for the particular data stream.

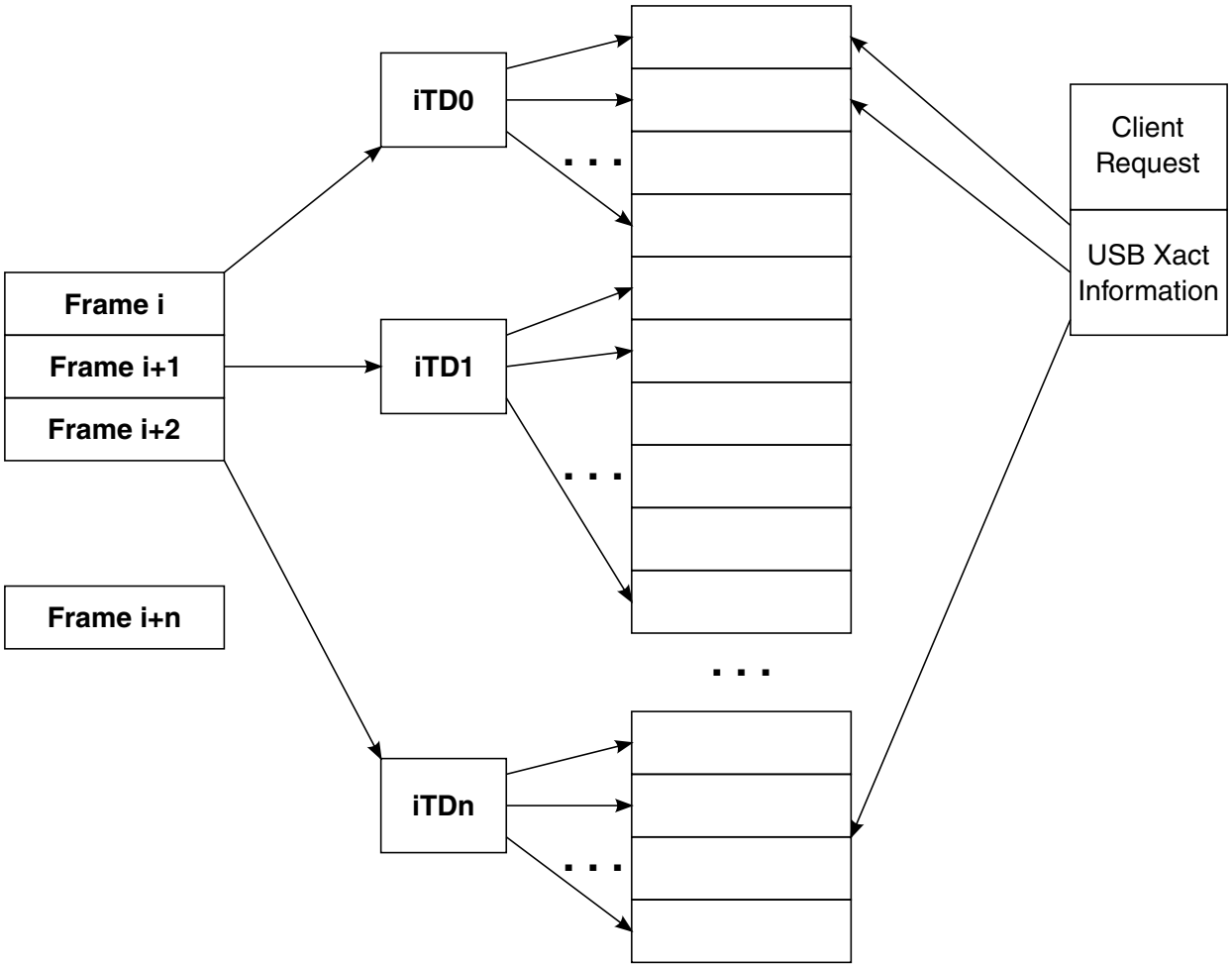


Figure 11-13. Example Association of iTDs to Client Request Buffer

As noted above, the client request includes a pointer to the base of the buffer and offsets into the buffer to annotate which buffer sections are to be used on each bus transaction that occurs on this endpoint. System software must initialize each transaction description in an iTD to ensure it uses the correct portion of the client buffer. For example, for each transaction description, the PG field is set to index the correct physical buffer page pointer and the Transaction Offset field is set relative to the correct buffer pointer page (for example, the same one referenced by the PG field). When the host controller executes a transaction it selects a transaction description record based on FRINDEX[2:0]. It then uses the current Page Buffer Pointer (as selected by the PG field) and concatenates to the transaction offset field. The result is a starting buffer address for the transaction. As the host controller moves data for the transaction, it must watch for a page wrap condition and properly advance to the next available Page Buffer Pointer. System software must not use the Page 6 buffer pointer in a transaction description where the length of the transfer wraps a page boundary. Doing so yields undefined behavior. The host controller hardware is not required to 'alias' the page selector to Page zero. USB 2.0 isochronous

endpoints can specify a period greater than one. Software can achieve the appropriate scheduling by linking iTDs into the appropriate frames (relative to the frame list) and by setting appropriate transaction description elements active bits to one.

11.1.3.3.7.2.1 Periodic scheduling threshold

The Isochronous Scheduling Threshold field in the USB_HCCPARAMS capability register is an indicator to system software as to how the host controller pre-fetches and effectively caches schedule data structures.

It is used by system software when adding isochronous work items to the periodic schedule. The value of this field indicates to system software the minimum distance it can update isochronous data (relative to the current location of the host controller execution in the periodic list) and still have the host controller process them.

The iTD and siTD data structures each describe 8 micro-frames worth of transactions. The host controller is allowed to cache one (or more) of these data structures in order to reduce memory traffic. Three basic caching models that account for the fact the isochronous data structures span 8 micro-frames. The three caching models are: no caching, micro-frame caching and frame caching.

When software is adding new isochronous transactions to the schedule, it always performs a read of the USB_FRINDEX register to determine the current frame and micro-frame the host controller is currently executing. Of course, there is no information about where in the micro-frame the host controller is, so a constant uncertainty-factor of one micro-frame has to be assumed. Combining the knowledge of where the host controller is executing with the knowledge of the caching model allows the definition of simple algorithms for how closely software can reliably work to the executing host controller.

No caching is indicated with a value of zero in the Isochronous Scheduling Threshold field. The host controller may pre-fetch data structures during a periodic schedule traversal (per micro-frame) but always dumps any accumulated schedule state at the end of the micro-frame. At the appropriate time relative to the beginning of every micro-frame, the host controller always begins schedule traversal from the frame list. Software can use the value of the USB_FRINDEX register (plus the constant 1 uncertainty-factor) to determine the approximate position of the executing host controller. When no caching is selected, software can add an isochronous transaction as near as 2 micro-frames in front of the current executing position of the host controller.

Frame caching is indicated with a non-zero value in bit [7] of the Isochronous Scheduling Threshold field. In the frame-caching model, system software assumes that the host controller caches one (or more) isochronous data structures for an entire frame (8 micro-frames). Software uses the value of the USB_FRINDEX register (plus the constant 1

uncertainty) to determine the current micro-frame/frame (assume modulo 8 arithmetic in adding the constant 1 to the micro-frame number). For any current frame N, if the current micro-frame is 0 to 6, then software can safely add isochronous transactions to Frame N + 1. If the current micro-frame is 7, then software can add isochronous transactions to Frame N + 2.

Micro-frame caching is indicated with a non-zero value in the least-significant 3 bits of the Isochronous Scheduling Threshold field. System software assumes the host controller caches one or more periodic data structures for the number of micro-frames indicated in the Isochronous Scheduling Threshold field. For example, if the count value were 2, then the host controller keeps a window of 2 micro-frames worth of state (current micro-frame, plus the next) on-chip. On each micro-frame boundary, the host controller releases the current micro-frame state and begins accumulating the next micro-frame state.

11.1.3.3.8 Asynchronous Schedule

The Asynchronous schedule traversal is enabled or disabled through the Asynchronous Schedule Enable bit in the USB_USBCMD register.

If the Asynchronous Schedule Enable bit is set to zero, then the host controller simply does not try to access the asynchronous schedule through the USB_ASYNCLISTADDR register. Likewise, when the Asynchronous Schedule Enable bit is one, then the host controller does use the USB_ASYNCLISTADDR register to traverse the asynchronous schedule. Modifications to the Asynchronous Schedule Enable bit are not necessarily immediate. Rather the new value of the bit is taken into consideration the next time the host controller needs to use the value of the USB_ASYNCLISTADDR register to get the next queue head.

The Asynchronous Schedule Status bit in the USB_USBSTS register indicates status of the asynchronous schedule. System software enables (or disables) the asynchronous schedule by writing one (or zero) to the Asynchronous Schedule Enable bit in the USB_USBCMD register. Software then can poll the Asynchronous Schedule Status bit to determine when the asynchronous schedule has made the desired transition. Software must not modify the Asynchronous Schedule Enable bit unless the value of the Asynchronous Schedule Enable bit equals that of the Asynchronous Schedule Status bit.

The asynchronous schedule is used to manage all Control and Bulk transfers. Control and Bulk transfers are managed using queue head data structures. The asynchronous schedule is based at the USB_ASYNCLISTADDR register. The default value of the USB_ASYNCLISTADDR register after reset is undefined and the schedule is disabled when the Asynchronous Schedule Enable bit is zero.

Software may only write this register with defined results when the schedule is disabled. For example, Asynchronous Schedule Enable bit in the USB_USBCMD and the Asynchronous Schedule Status bit in the USB_USBSTS register are zero. System software enables execution from the asynchronous schedule by writing a valid memory address (of a queue head) into this register. Then software enables the asynchronous schedule by setting the Asynchronous Schedule Enable bit to one. The asynchronous schedule is actually enabled when the Asynchronous Schedule Status bit is one.

When the host controller begins servicing the asynchronous schedule, it begins by using the value of the USB_ASYNCLISTADDR register. It reads the first referenced data structure and begins executing transactions and traversing the linked list as appropriate. When the host controller completes processing the asynchronous schedule, it retains the value of the last accessed queue head's horizontal pointer in the USB_ASYNCLISTADDR register. Next time the asynchronous schedule is accessed, this is the first data structure that is serviced. This provides round-robin fairness for processing the asynchronous schedule.

A host controller completes processing the asynchronous schedule when one of the following events occur:

- The end of a micro-frame occurs.
- The host controller detects an empty list condition (see Empty Asynchronous Schedule Detection)
- The schedule has been disabled through the Asynchronous Schedule Enable bit in the USB_USBCMD register.

The queue heads in the asynchronous list are linked into a simple circular list as shown in Figure 11-8. Queue head data structures are the only valid data structures that may be linked into the asynchronous schedule. An isochronous transfer descriptor (iTD or siTD) in the asynchronous schedule yields undefined results.

The maximum packet size field in a queue head is sized to accommodate the use of this data structure for all non-isochronous transfer types. The USB Specification, Revision 2.0 specifies the maximum packet sizes for all transfer types and transfer speeds. System software should always parameterize the queue head data structures according to the core specification requirements.

11.1.3.3.8.1 Adding Queue Heads to Asynchronous Schedule

This is a software requirement section.

There are two independent events for adding queue heads to the asynchronous schedule. The first is the initial activation of the asynchronous list. The second is inserting a new queue head into an activated asynchronous list.

Activation of the list is simple. System software writes the physical memory address of a queue head into the USB_ASYNCLISTADDR register, then enables the list by setting the Asynchronous Schedule Enable bit in the USB_USBCMD register to one.

When inserting a queue head into an active list, software must ensure that the schedule is always coherent from the host controllers' point of view. This means that the system software must ensure that all queue head pointer fields are valid. For example, qTD pointers have T-Bits set to one or reference valid qTDs and the Horizontal Pointer references a valid queue head data structure. The following algorithm represents the functional requirements:

```
InsertQueueHead (pQHeadCurrent, pQueueHeadNew)
          --
          -- Requirement: all inputs must be properly initialized.
          --
          -- pQHeadCurrent is a pointer to a queue head that is
          -- already in the active list
          -- pQHeadNew is a pointer to the queue head to be added
          --
          -- This algorithm links a new queue head into a existing
          -- list
          --
          pQueueHeadNew.HorizontalPointer  =  pQueueHeadCurrent.HorizontalPointer
          pQueueHeadCurrent.HorizontalPointer =  physicalAddressOf(pQueueHeadNew)
End InsertQueueHead
```

11.1.3.3.8.2 Removing Queue Heads from Asynchronous Schedule

This is a software requirement section.

There are two independent events for removing queue heads from the asynchronous schedule. The first is shutting down (deactivating) the asynchronous list. The second is extracting a single queue head from an activated list.

Software deactivates the asynchronous schedule by setting the Asynchronous Schedule Enable bit in the USB_USBCMD register to zero. Software can determine when the list is idle when the Asynchronous Schedule Status bit in the USB_USBSTS register is zero. The normal mode of operation is that software removes queue heads from the asynchronous schedule without shutting it down. Software must not remove an active queue head from the schedule. Software should first deactivate all active qTDs, wait for the queue head to go inactive, then remove the queue head from the asynchronous list. Software removes a queue head from the asynchronous list through the following algorithm. As illustrated, the unlinking is quite easy. Software merely must ensure all of the link pointers reachable by the host controller are kept consistent.

```
UnlinkQueueHead (pQHeadPrevious, pQueueHeadToUnlink, pQHeadNext)
          --
          -- Requirement: all inputs must be properly initialized.
          --
          -- pQHeadPrevious is a pointer to a queue head that
          -- references the queue head to remove
```

```
          -- pQHeadToUnlink is a pointer to the queue head to be
          -- removed
          -- pQheadNext is a pointer to a queue head still in the
          -- schedule. Software provides this pointer with the
          -- following strict rules:
          --          if the host software is one queue head, then
          --          pQHeadNext must be the same as
          --          QueueheadToUnlink.HorizontalPointer. If the host
          --          software is unlinking a consecutive series of
          --          queue heads, QHeadNext must be set by software to
          --          the  queue head remaining in the schedule.
          --
          -- This algorithm unlinks a queue head from a circular list
          --
          pQueueHeadPrevious.HorizontalPointer =  pQueueHeadToUnlink.HorizontalPointer
                                  pQueueHeadToUnlink.HorizontalPointer = pQHeadNext
End UnlinkQueueHead
```

If software removes the queue head with the H-bit set to one, it must select another queue head still linked into the schedule and set its H-bit to one. This should be completed before removing the queue head. The requirement is that software keep one queue head in the asynchronous schedule, with its H-bit set to one. At the point software has removed one or more queue heads from the asynchronous schedule, it is unknown whether the host controller has a cached pointer to them. Similarly, it is unknown how long the host controller might retain the cached information, as it is implementation dependent and may be affected by the actual dynamics of the schedule load. Therefore, once software has removed a queue head from the asynchronous list, it must retain the coherency of the queue head (link pointers, and so on). It cannot disturb the removed queue heads until it knows that the host controller does not have a local copy of a pointer to any of the removed data structures.

The method software uses to determine when it is safe to modify a removed queue head is to handshake with the host controller. The handshake mechanism allows software to remove items from the asynchronous schedule, then execute a simple, lightweight handshake that is used by software as a key that it can free (or reuse) the memory associated the data structures it has removed from the asynchronous schedule.

The handshake is implemented with three bits in the host controller. The first bit is a command bit (Interrupt on Async Advance Doorbell bit in the USB_USBCMD register) that allows software to inform the host controller that something has been removed from its asynchronous schedule. The second bit is a status bit (Interrupt on Async Advance bit in the USB_USBSTS register) that the host controller sets after it has released all on-chip state that may potentially reference one of the data structures just removed. When the host controller sets this status bit to one, it also sets the command bit to zero. The third bit is an interrupt enable (Interrupt on Async Advance bit in the USB_USBINTR register) that is matched with the status bit. If the status bit is one and the interrupt enable bit is one, then the host controller asserts a hardware interrupt.

The figure below illustrates a general example. In this example, consecutive queue heads (B and C) are unlinked from the schedule using the algorithm above. Before the unlink operation, the host controller has a copy of queue head A.

The unlink algorithm requires that as software unlinks each queue head, the unlinked queue head is loaded with the address of a queue head that remains in the asynchronous schedule.

When the host controller observes that doorbell bit being set to one, it makes a note of the local reachable schedule information. In this example, the local reachable schedule information includes both queue heads (A and B). It is sufficient that the host controller can set the status bit (and clear the doorbell bit) as soon as it has traversed beyond current reachable schedule information (that is traversed beyond queue head (B) in this example). The following figure illustrates the generic queue head unlink scenario.

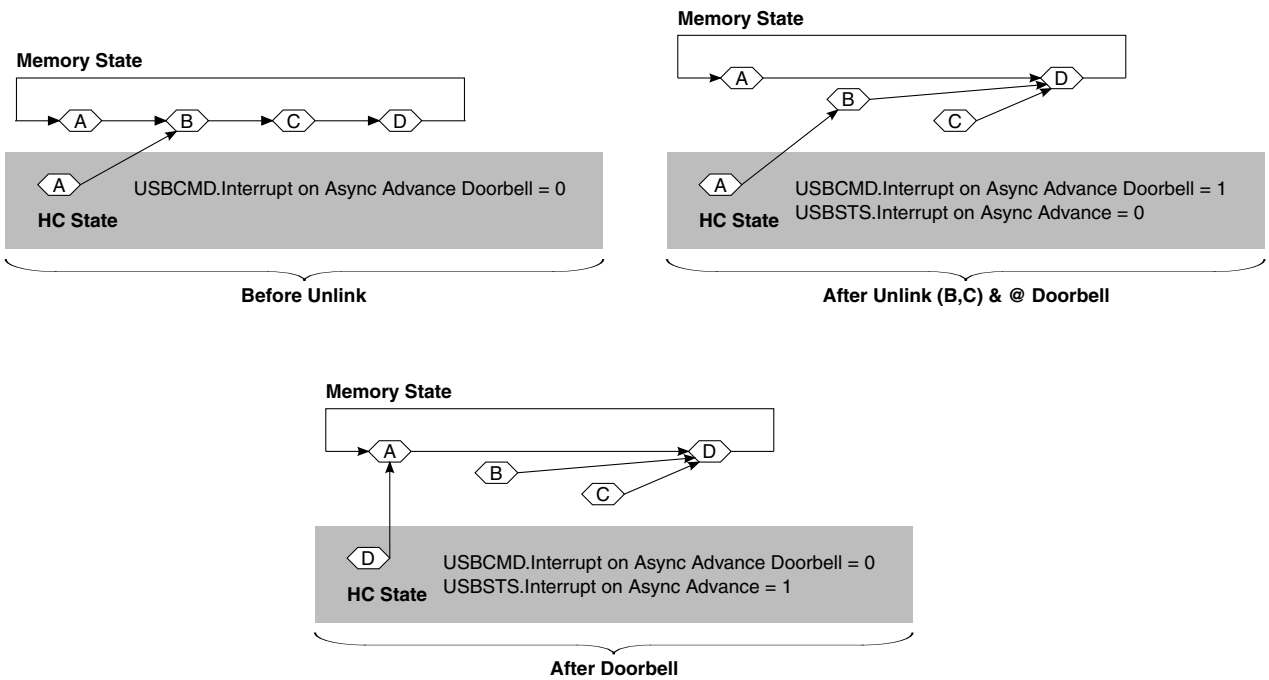


Figure 11-14. Generic Queue Head Unlink Scenario

Alternatively, a host controller implementation is allowed to traverse the entire asynchronous schedule list (for example, observed the head of the queue (twice)) before setting the Advance on Async status bit to one.

Software may re-use the memory associated with the removed queue heads after it observes the Interrupt on Async Advance status bit is set to one, following assertion of the doorbell. Software should acknowledge the Interrupt on Async Advance status as indicated in the USB_USBSTS register, before using the doorbell handshake again.

11.1.3.3.8.3 Empty Asynchronous Schedule Detection

The Enhanced Host Controller Interface uses two bits to detect when the asynchronous schedule is empty.

The queue head data structure (see Table 11-24) defines an *H-bit* in the queue head, which allows software to mark a queue head as being the *head* of the reclaim list. The Enhanced Host Controller Interface also keeps a 1-bit flag in the USB_USBSTS register (*Reclamation*) that is set to zero when the Enhanced Interface Host Controller observes a queue head with the H-bit set to one. The reclamation flag in the status register is set to one when any USB transaction from the asynchronous schedule is executed (or whenever the asynchronous schedule starts, see Asynchronous schedule traversal: *Start Event*).

If the Enhanced Host Controller Interface ever encounters an *H-bit* of one and a *Reclamation* bit of zero, the EHCI controller simply stops traversal of the asynchronous schedule.

An example illustrating the H-bit in a schedule is shown in the following figure.

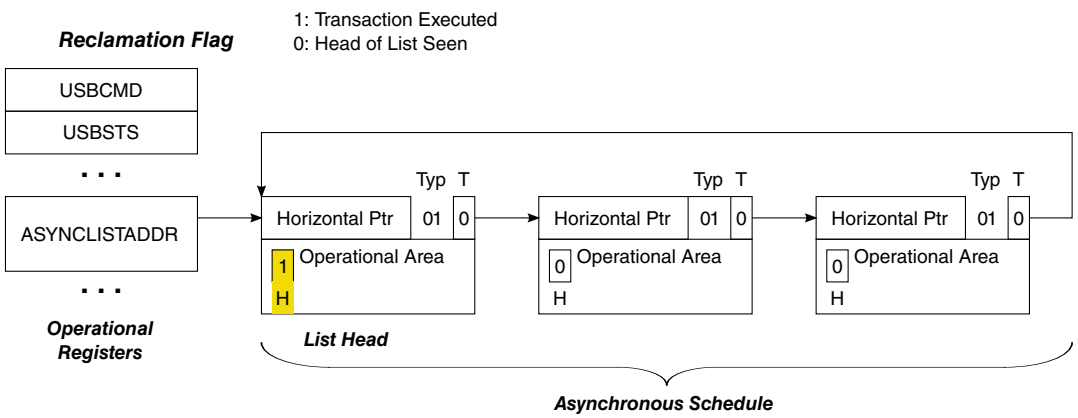


Figure 11-15. Asynchronous Schedule List w/Annotation to Mark Head of List

Software must ensure there is at most one queue head with the *H-bit* set to one, and that it is always coherent with respect to the schedule.

11.1.3.3.8.4 Restarting Asynchronous Schedule Before EOF

There are many situations where the host controller will detect an empty list *long* before the end of the micro-frame.

It is important to remember that under many circumstances the schedule traversal has stopped due to Nak/Nyet responses from all endpoints.

An example of particular interest is when a start-split for a bulk endpoint occurs early in the micro-frame. Given the EHCI simple traversal rules, the complete-split for that transaction may Nak/Nyet out very quickly. If it is the only item in the schedule, then the host controller ceases traversal of the Asynchronous schedule very early in the micro-frame. In order to provide reasonable service to this endpoint, the host controller should issue the complete-split before the end of the current micro-frame, instead of waiting until the next micro-frame. When the reason for host controller idling asynchronous schedule traversal is because of empty list detection, it is mandatory the host controller implement a 'waking' method to resume traversal of the asynchronous schedule. An example method is described below.

11.1.3.3.8.4.1 Example Method for Restarting Asynchronous Schedule Traversal

The reason for idling the host controller when the list is empty is to keep the host controller from unnecessarily occupying too much memory bandwidth. The question is: *how long should the host controller stay idle before restarting*?

The answer in this example is based on deriving a manifest constant, which is the amount of time the host controller will stay idle before restarting traversal. In this example, the manifest constant is called *AsyncSchedSleepTime*, and has a value of 10 µsec. The value is derived based on the analysis in Example Derivation for *AsyncSchedSleepTime*, The traversal algorithm is simple:

- Traverse the Asynchronous schedule until the either an End-Of-micro-Frame event occurs, or an empty list is detected. If the event is an End-of-micro-Frame, go attempt to traverse the Periodic schedule. If the event is an empty list, then set a sleep timer and go to a *schedule sleep* state.
- When the sleep timer expires, set working context to the Asynchronous Schedule start condition and go to *schedule active* state. The start context allows the HC to reload *Nakcnt* fields, and so on. So the HC has a chance to run for more than one iteration through the schedule.

This process simply repeats itself each micro-frame. The figure below illustrates a sample state machine to manage the active and sleep states of the Asynchronous Schedule traversal policy. There are three states: Actively traversing the Asynchronous schedule, Sleeping, and Not Active. The last two are similar in terms of interaction with the Asynchronous schedule, but the Not Active state means that the host controller is busy with the Periodic schedule or the Asynchronous schedule is not enabled. The Sleeping state is specifically a special state where the host controller is just waiting for a period of time before resuming execution of the Asynchronous schedule.

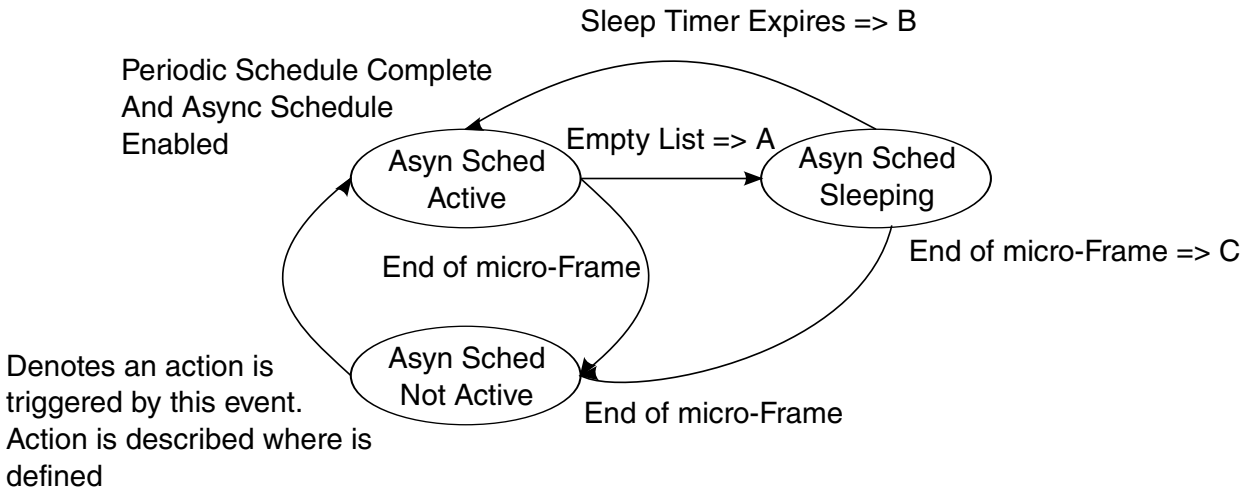


Figure 11-16. Example State Machine for Managing Asynchronous Schedule Traversal

The actions referred to in the figure above are defined in the following table.

Table 11-39. Asynchronous Schedule SM Transition Actions

Action	Action Description Label
A	On detection of the empty list, the host controller sets the *AsynchronousTraversalSleepTimer* to *AsyncSchedSleepTime*.
B	When the *AsynchronousTraversalSleepTimer* expires, the host controller sets the *Reclamation* bit in the USBSTS register to one and moves the Nak Counter reload state machine to WaitForListHead (see Nak Count Reload Control).
C	The host controller cancels the sleep timer (*AsynchronousTraversalSleepTimer*).

11.1.3.3.8.4.2 Async Sched Not Active

This is the initial state of the traversal state machine after a host controller reset. The traversal state machine does not leave this state when the *Asynchronous Schedule Enable* bit in the USB_USBCMD register is zero.

This state is entered from Async Sched Active or Async Sched Sleeping states when the end-of-micro-frame event is detected.

11.1.3.3.8.4.3 Async Sched Active

This state is entered from the Async Sched Not Active state when the periodic schedule is not active. It is also entered from the Async Sched Sleeping states when the *AsyncrhonousTraversalSleepTimer* expires. On every transition into this state, the host controller sets the *Reclamation* bit in the USB_USBSTS register to one.

While in this state, the host controller continually traverses the asynchronous schedule until either the end of micro-frame or an empty list condition is detected.

11.1.3.3.8.4.4 Async Sched Sleeping

The state is entered from the Async Sched Active state when a schedule empty condition is detected. On entry to this state, the host controller sets the *AsynchronousTraversalSleepTimer* to *AsyncSchedSleepTime*.

11.1.3.3.8.4.5 Example Derivation for AsyncSchedSleepTime

The derivation is based on analysis of what work the host controller could be doing next.

It assumes the host controller does not keep any state about what work is possibly pending in the asynchronous schedule. The schedule could contain any mix of the possible combinations of high- full- or low-speed control and bulk requests.

The table below summarizes some of the typical 'next transactions' that could be in the schedule, and the amount of time (for example *footprint*, or *wall clock*) the transaction takes to complete.

Table 11-40. Typical Low-/Full-speed Transaction Times

Transaction Attributes		Footprint (time)	Description
Speed	HS	11.9 ms	Maximum foot print for a worst-case, full-sized bulk data transaction.
Size	512	9.45 ms	Maximum footprint for an approximate best-case, full-sized bulk data transaction.
Type	Bulk		
Speed	FS	~50 ms	Approximate typical for full-sized bulk data. An 8-byte low-speed is about 2x, or between 90 and 100 ms.
Size	64		
Type	Bulk		
Speed	FS	~12 ms	Approximate typical for 8-byte bulk/control (that is setup)
Size	8		
Type	Cntrl		

A *AsyncSchedSleepTime* value of 10 µs provides a reasonable relaxation of the system memory load and still provides a good level of service for the various transfer types and payload sizes. For example, say we detect an empty list after issuing a start-split for a 64-byte full-speed bulk request. Assuming this is the only thing in the list, the host controller gets the results of the full-speed transaction from the hub during the fifth complete-split request. If the full-speed transaction was an IN and it nak'd, the 10 µs sleep period would allow the host controller to get the NAK results on the first complete-split.

11.1.3.3.8.5 Asynchronous schedule traversal: *Start Event*

Once the HC has *idled* itself through the empty schedule detection (Section 0), it will naturally *activate* and begin processing from the Periodic Schedule at the beginning of each micro-frame.

In addition, it may have idled itself early in a micro-frame. When this occurs (idles early in the micro-frame) the HC must occasionally *re-activate* during the micro-frame and traverse the asynchronous schedule to determine whether any progress can be made. The requirements and method for this restart are described in Restarting Asynchronous Schedule Before EOF . Asynchronous schedule *Start Event*s are defined to be:

- Whenever the host controller transitions from the periodic schedule to the asynchronous schedule. If the periodic schedule is disabled and the asynchronous schedule is enabled, then the beginning of the micro-frame is equivalent to the transition from the periodic schedule, or
- The asynchronous schedule traversal restarts from a sleeping state (see Restarting Asynchronous Schedule Before EOF).

11.1.3.3.8.6 Reclamation Status Bit (USBSTS Register)

The operation of the empty asynchronous schedule detection feature (see Empty Asynchronous Schedule Detection) depends on the proper management of the *Reclamation* bit in the USB_USBSTS register. The host controller tests for an empty schedule just after it fetches a new queue head while traversing the asynchronous schedule (see Fetch Queue Head).

The host controller is required to set the *Reclamation* bit to one whenever an asynchronous schedule traversal *Start Event*, as documented in Asynchronous schedule traversal: *Start Event*, occurs. The *Reclamation* bit is also set to one whenever the host controller executes a transaction while traversing the asynchronous schedule (see Execute Transaction). The host controller sets the *Reclamation* bit to zero whenever it finds a queue head with its *H-bit* set to one. Software should only set a queue head's *H-bit* if the queue head is in the asynchronous schedule. If software sets the *H-bit* in an interrupt queue head to one, the resulting behavior is undefined. The host controller may set the *Reclamation* bit to zero when executing from the periodic schedule.

11.1.3.3.9 Operational Model for Nak Counter

This section describes the operational model for the *NakCnt* field defined in a queue head.

See Queue Head Initialization for more information. Software should not use this feature for interrupt queue heads. This rule is not required to be enforced by the host controller.

USB protocol has built-in flow control through the Nak response by a device. There are several scenarios, beyond the Ping feature, where an endpoint may naturally Nak or Nyet the majority of the time. An example is the host controller management of the split transaction protocol for control and bulk endpoints. All bulk endpoints (High- or Full-speed) are serviced through the same asynchronous schedule. The time between the *Start-split* transaction and the first *Complete-split* transaction could be very short (that is like when the endpoint is the only one in the asynchronous schedule). The hub NYETs (effectively Naks) the *Complete-split* transaction until the classic transaction is complete. This could result in the host controller thrashing memory, repeatedly fetching the queue head and executing the transaction to the Hub, which does not complete until after the transaction on the classic bus completes.

The two component fields in a queue head to support the throttling feature: a counter field (*NakCnt*), and a counter reload field (*RL*). *NakCnt* is used by the host controller as one of the criteria to determine whether or not to execute a transaction to the endpoint. The two operational modes associated with this counter:

- Not Used- This mode is set when the *RL* field is zero. The host controller ignores the *NakCnt* field for any execution of transactions through a queue head with an *RL* field of zero. Software must use this selection for interrupt endpoints.
- Nak Throttle Mode- This mode is selected when the *RL* field is non-zero. In this mode, the value in the *NakCnt* field represents the maximum number of Nak or Nyet responses the host controller tolerates on each endpoint. In this mode, the HC decrements the *NakCnt* field based on the token/handshake criteria listed in the table below. The host controller must reload *NakCnt* when the endpoint successfully moves data (for example, policy to reward device for moving data).

The following table describes the NakCnt field adjustment rules.

Table 11-41. NakCnt Field Adjustment Rules

Token	Handshake	
	Handshake NAK	NYET
IN/PING	decrement *NakCnt*	N/A (protocol error)
OUT	decrement *NakCnt*	No Action[1] Start
Split	decrement *NakCnt*	N/A (protocol error)
Complete Split	No Action	Decrement *NakCnt*

1. Recommended behavior on this response is to reload *NakCnt*

In summary, system software enables the counter by setting the reload field (*RL*) to a non-zero value. The host controller may execute a transaction if *NakCnt* is non-zero. The host controller does not execute a transaction if *NakCnt* is zero. The reload mechanism is described in detail in Nak Count Reload Control .

NOTE

When all queue heads in the Asynchronous Schedule either exhausts all transfers or all NakCnt's go to zero, then the host controller detects an empty Asynchronous Schedule and idle schedule traversal (see Empty Asynchronous Schedule Detection).

Any time the host controller begins a new traversal of the Asynchronous Schedule, a *Start Event* is assumed, see Asynchronous schedule traversal: *Start Event*. Every time a Start-Event occurs, the Nak Count reload procedure is enabled.

11.1.3.3.9.1 Nak Count Reload Control

When the host controller reaches the *Execute Transaction* state for a queue head (meaning that it has an active operational state), it checks to determine whether the *NakCnt* field should be reloaded from *RL* (see Execute Transaction). If the answer is yes, then *RL* is copied into *NakCnt*. After the reload or if the reload is not active, the host controller evaluates whether to execute the transaction.

The host controller must reload nak counters (*NakCnt* see Table 11-24) in queue heads during the first pass through the reclamation list after an asynchronous schedule Start Event (see Asynchronous schedule traversal: *Start Event* for the definition of the Start Event). The Asynchronous Schedule should have at most one queue head marked as the head (see Figure 11-15).

The following figure illustrates an example state machine that satisfies the operational requirements of the host controller detecting the first pass through the Asynchronous Schedule. This state machine is maintained internal to the host controller and is only used to gate reloading of the nak counter during the queue head traversal state: Execute Transaction (see the figure below). The host controller does not perform the nak counter reload operation if the RL field (see Table 11-24) is set to zero.

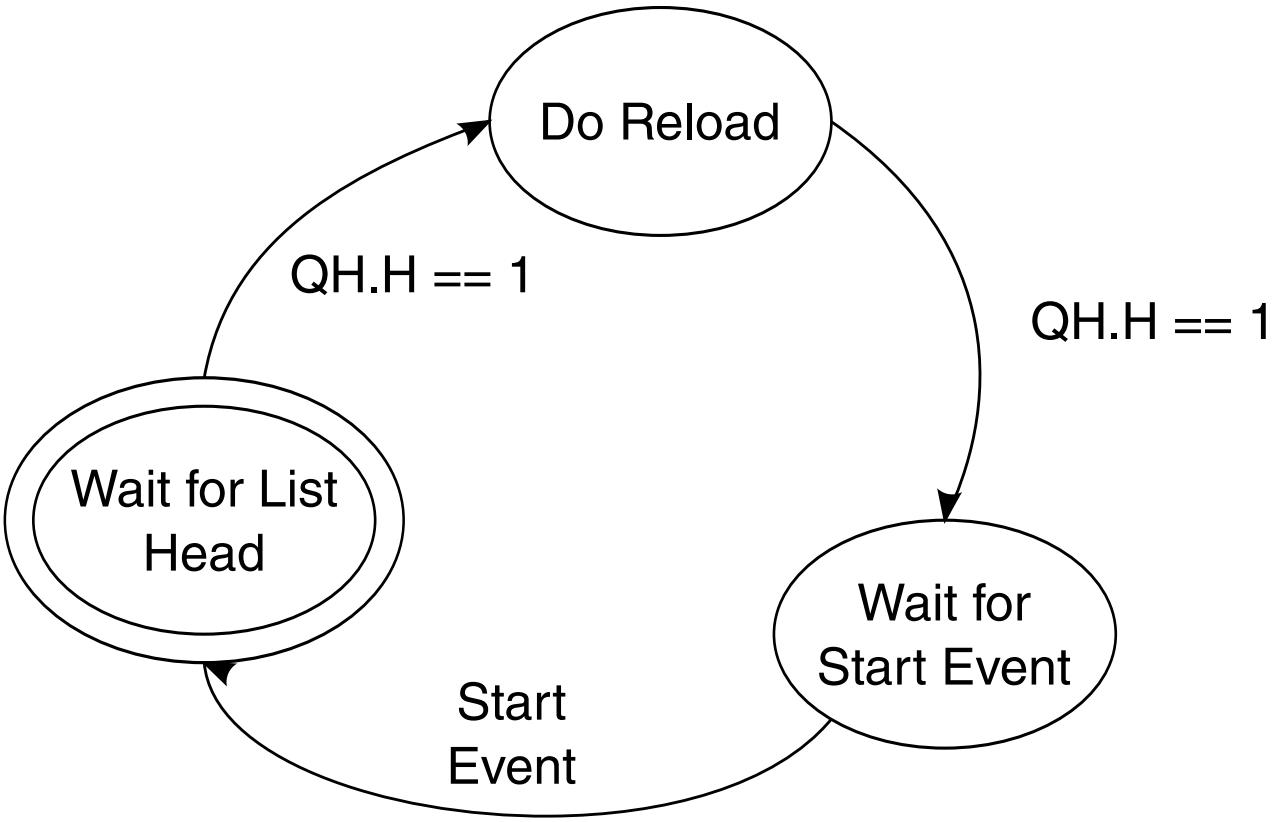


Figure 11-17. Example HC State Machine for Controlling Nak Counter Reloads

11.1.3.3.9.1.1 Wait for List Head

This is the initial state.

The state machine enters this state from Wait for Start Event when a start event as defined in Asynchronous schedule traversal: *Start Event* occurs.

The purpose of this state is to wait for the first observation of the head of the Asynchronous Schedule.

This occurs when the host controller fetches a queue head whose *H-bit* is set to one.

11.1.3.3.9.1.2 Do Reload

This state is entered from the Wait for List Head state when the host controller fetches a queue head with the *H-bit* set to one. While in this state, the host controller performs nak counter reloads for every queue head visited that has a non-zero nak reload value (*RL*) field.

11.1.3.3.9.1.3 Wait for Start Event

This state is entered from the *Do Reload* state when a queue head with the *H-bit* set to one is fetched. While in this state, the host controller does not perform nak counter reloads.

11.1.3.3.10 Managing Control/Bulk/Interrupt Transfers through Queue Heads

This section presents an overview of how the host controller interacts with queuing data structures.

Queue heads use the Queue Element Transfer Descriptor (qTD) structure. One queue head is used to manage the data stream for one endpoint. The queue head structure contains static endpoint characteristics and capabilities. It also contains a working area from where individual bus transactions for an endpoint are executed (see Overlay area defined in Table 11-24). Each qTD represents one or more bus transactions, which is defined in the context of this specification as a *transfer*.

The general processing model for the host controller's use of a queue head is simple:

- read a queue head,
- execute a transaction from the overlay area,
- write back the results of the transaction to the overlay area,
- move to the next queue head.

If the host controller encounters errors during a transaction, the host controller sets one (or more) of the error reporting bits in the queue head's *Status* field. The *Status* field accumulates all errors encountered during the execution of a qTD (for example, the error bits in the queue head *Status* field are 'sticky' until the transfer (qTD) has completed). This state is always written back to the source qTD when the transfer is complete. On transfer (for example, buffer or halt conditions) boundaries, the host controller must auto-advance (without software intervention) to the next qTD. Additionally, the hardware must be able to halt the queue so no additional bus transactions occurs for the endpoint and the host controller does not advance the queue.

An example host controller operational state machine of a queue head traversal is illustrated in the following figure. This state machine is a model for how a host controller should traverse a queue head. The host controller must be able to advance the queue from the *Fetch QH* state in order to avoid all hardware/software race conditions. This simple mechanism allows software to simply link qTDs to the queue head and *activate* them, then the host controller always *find* them if/when they are reachable. The figure below illustrates the Host Controller Queue Head Traversal State Machine.

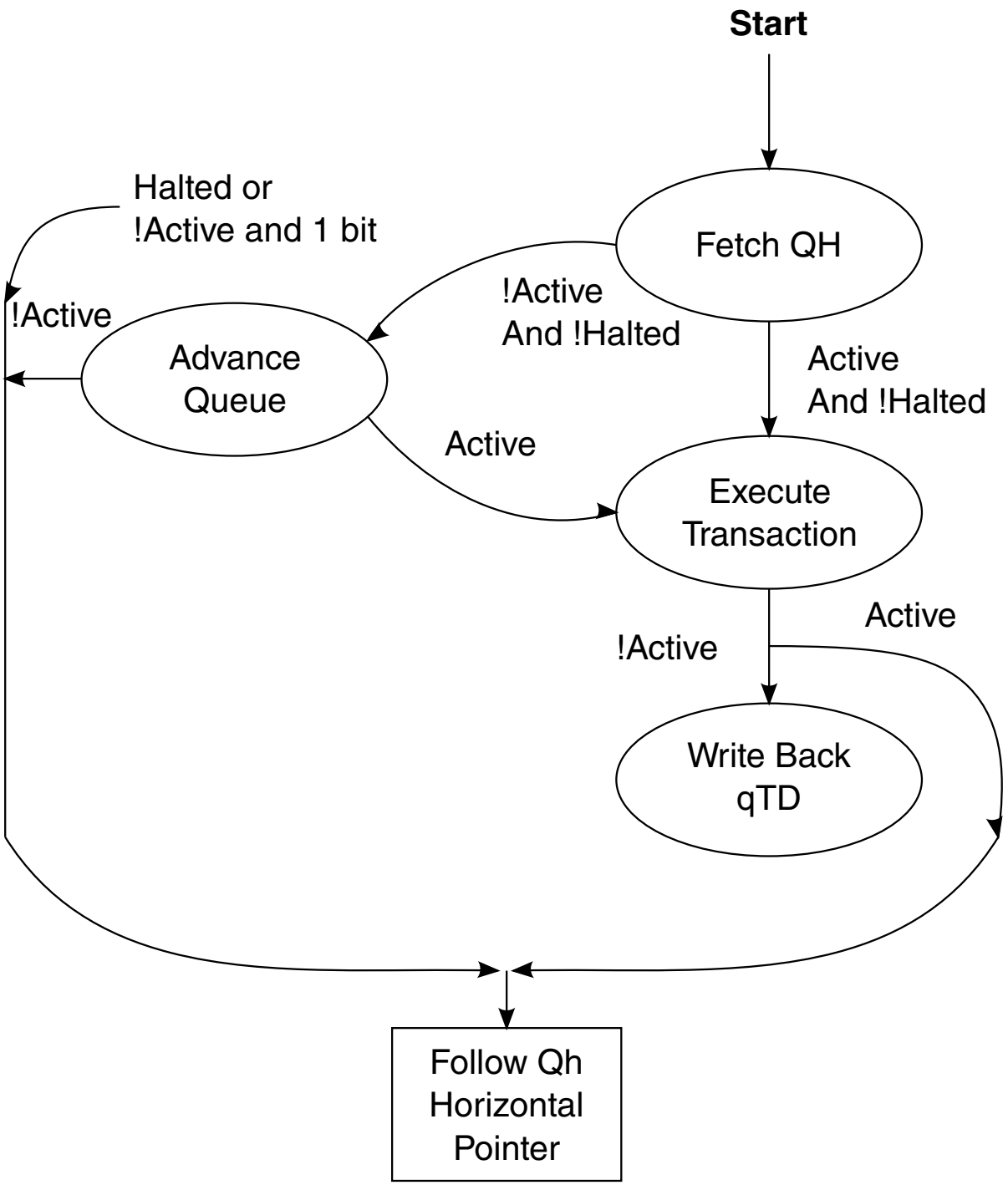


Figure 11-18. Host Controller Queue Head Traversal State Machine

This traversal state machine applies to all queue heads, regardless of transfer type or whether split transactions are required. The following sections describe each state. Each state description describes the entry criteria. The Execute Transaction state (see Execute

Transaction) describes the basic requirements for all endpoints. Split Transactions for Asynchronous Transfers and Split Transaction Interrupt describe details of the required extensions to the Execute Transaction state for endpoints requiring split transactions.

NOTE

Prior to software placing a queue head into either the periodic or asynchronous list, software must ensure the queue head is properly initialized. Minimally, the queue head should be initialized to the following (see Section Queue Head for layout of a queue head):

Valid static endpoint state.

- For the very first use of a queue head, software may zero-out the queue head transfer overlay, then set the *Next qTD Pointer* field value to reference a valid qTD.

11.1.3.3.10.1 Fetch Queue Head

A queue head can be referenced from the physical address stored in the ASYNCLISTADDR Register (see Next Asynch. Address (USB_*n*ASYNCLISTADDR))/ Endpoint List Address (USB_*n*ENDPTLISTADDR) Additionally, it may be referenced from the *Next LinkPointer* field of an iTD, siTD, FSTN or another Queue Head. If the referencing link pointer has the *Typ* field set to indicate a queue head, it is assumed to reference a queue head structure as defined in Table 11-24.

While in this state, the host controller performs operations to implement empty schedule detection (see Empty Asynchronous Schedule Detection) and Nak Counter reloads (see Operational Model for Nak Counter). After the queue head has been fetched, the host controller conducts the following queries for empty schedule detection:

- If queue head is not an interrupt queue head (that is *S-mask* is zero), and
- The *H-bit* is one, and
- The *Reclamation* bit in the USBSTS register is zero.

When these criteria are met, the host controller stops traversing the asynchronous list (as described in Empty Asynchronous Schedule Detection). When the criteria are not met, the host controller continues schedule traversal. If the queue head is not an interrupt and the *H-bit* is one and the *Reclamation* bit is one, then the host controller sets the *Reclamation* bit in the USBSTS register to zero before completing this state. The operations for reloading of the Nak Counter are described in detail in Operational Model for Nak Counter.

This state is complete when the queue head has been read on-chip.

11.1.3.3.10.2 Advance Queue

To advance the queue, the host controller must find the next qTD, adjust pointers, perform the overlay and write back the results to the queue head.

This state is entered from the FetchQHD state if the overlay *Active* and *Halt* bits are set to zero. On entry to this state, the host controller determines which next pointer to use to fetch a qTD, fetches a qTD and determines whether or not to perform an overlay.

NOTE

If the *I-bit* is one and the *Active* bit is zero, the host controller immediately skips processing of this queue head, exits this state and uses the horizontal pointer to the next schedule data structure. If the field *Bytes to Transfer* is not zero and the *T-bit* in the *Alternate Next qTD Pointer* is set to zero, then the host controller uses the *Alternate Next qTD Pointer*. Otherwise, the host controller uses the *NextqTD Pointer*. If *NextqTD Pointer's T-bit* is set to one, then the host controller exits this state and uses the horizontal pointer to the next schedule data structure.

Using the selected pointer the host controller fetches the referenced qTD. If the fetched qTD has its *Active* bit set to one, the host controller moves the pointer value used to reach the qTD (*Next* or *Alternate Next*) to the *Current qTD Pointer* field, then performs the overlay. If the fetched qTD has its *Active* bit set to zero, the host controller aborts the queue advance and follows the queue head's horizontal pointer to the next schedule data structure.

The host controller performs the overlay based on the following rules:

- The value of the data toggle (*dt*) field in the overlay area depends on the value of the *data toggle control* (*dtc*) bit (see Table 11-26).
- If the *EPS* field indicates the endpoint is a high-speed endpoint, the *Ping* state field is preserved by the host controller. The value of this field is not changed as a result of the overlay.
- *C-prog-mask* field is set to zero (field from incoming qTD is ignored, as is the current contents of the overlay area).
- *Frame Tag* field is set to zero (field from incoming qTD is ignored, as is the current contents of the overlay area).
- *NakCnt* field in the overlay area is loaded from the *RL* field in the queue head's Static Endpoint State.
- All other areas of the overlay are set by the incoming qTD.

The host controller exits this state when it has committed the write to the queue head.

11.1.3.3.10.3 Execute Transaction

The host controller enters this state from the Fetch Queue Head state only if the *Active* bit in *Status* field of the queue head is set to one.

On entry to this state, the host controller executes a few pre-operations, then checks some pre-condition criteria before committing to executing a transaction for the queue head.

The pre-operations performed and the pre-condition criteria depend on whether the queue head is an interrupt endpoint. The host controller can determine that a queue head is an interrupt queue head when the queue head's *S-mask* field contains a non-zero value. It is the responsibility of software to ensure the *S-mask* field is appropriately initialized based on the transfer type. There are other criteria that must be met if the *EPS* field indicates that the endpoint is a low- or full-speed endpoint, see Split Transactions for Asynchronous Transfers and Split Transaction Interrupt .

11.1.3.3.10.3.1 Interrupt Transfer Pre-condition Criteria

If the queue head is for an interrupt endpoint (for example, non-zero *S-mask* field), then the FRINDEX[2:0] field must identify a bit in the *S-mask* field that has one in it.

For example, an *S-mask* value of 00100000b would evaluate to true only when FRINDEX[2:0] is equal to 101b. If this condition is met then the host controller considers this queue head for a transaction.

11.1.3.3.10.3.2 Asynchronous Transfer Pre-operations and Pre-condition Criteria

If the queue head is not for an interrupt endpoint (for example, zero *S-mask* field), then the host controller performs one pre-operation and then evaluates one pre-condition criteria.

The pre-operation is:

Checks the Nak counter reload state (Operational Model for Nak Counter). It may be necessary for the host controller to reload the Nak Counter field. The reload is performed at this time.

The pre-condition evaluated is:

- Whether or not the *NakCnt* field has been reloaded, the host controller checks the value of the *NakCnt* field in the queue head. If *NakCnt* is non-zero, or if the *Reload Nak Counter* field is zero, then the host controller considers this queue head for a transaction.

11.1.3.3.10.3.3 Transfer Type Independent Pre-operations

Regardless of the transfer type, the host controller always performs at least one pre-operation and evaluates one pre-condition. The pre-operation is:

- A host controller internal transaction (down) counter *qHTransactionCounter* is loaded from the queue head's *Mult* field. A host controller implementation is allowed to ignore this for queue heads on the asynchronous list. It is mandatory for interrupt queue heads. Software should ensure that the *Mult* field is set appropriately for the transfer type.

The pre-conditions evaluated are:

- The host controller determines whether there is enough time in the micro-frame to complete this transaction (see Transaction Fit - A Best-Fit Approximation Algorithm for an example evaluation method). If there is not enough time to complete the transaction, the host controller exits this state.
- If the value of *qHTransactionCounter* for an interrupt endpoint is zero, then the host controller exits this state.

When the pre-operations are complete and pre-conditions are met, the host controller sets the *Reclamation* bit in the USBSTS register to one and then begins executing one or more transactions using the endpoint information in the queue head. The host controller iterates *qHTransactionCounter* times in this state executing transactions. After each transaction is executed, *qHTransactionCounter* is decremented by one. The host controller exits this state when one of the following events occurs:

- The *qHTransactionCounter* decrements to zero, or
- The endpoint responds to the transaction with any handshake other than an ACK,[4] or
- The transaction experiences a transaction error, or
- The *Active* bit in the queue head goes to zero, or
- There is not enough time in the micro-frame left to execute the next transaction(see Transaction Fit - A Best-Fit Approximation Algorithm) for example method for implementing the frame boundary test).

NOTE

For a high-bandwidth interrupt OUT endpoint, the host controller may optionally immediately retry the transaction if it fails.

The results of each transaction is recorded in the on-chip overlay area. If data was successfully moved during the transaction, the transfer state in the overlay area is advanced. To advance queue head's transfer state, the *Total Bytes to Transfer* field is decremented by the number of bytes moved in the transaction, the data toggle bit (*dt*) is toggled, the current page offset is advanced to the next appropriate value (for example,

advanced by the number of bytes successfully moved), and the *C_Page* field is updated to the appropriate value (if necessary). See Buffer Pointer List Use for Data Streaming with qTDs .

NOTE

The *Total Bytes To Transfer* field may be zero when all the other criteria for executing a transaction are met. When this occurs, the host controller executes zero-length transaction to the endpoint. If the *PID_Code* field indicates an IN transaction and the device delivers data, the host controller detects a packet babble condition, set the *babble* and *halted* bits in the *Status* field, set the *Active* bit to zero, write back the results to the source qTD, then exit this state.

In the event an IN token receives a data PID mismatch response, the host controller must ignore the received data (for example not advance the transfer state for the bytes received). Additionally, if the endpoint is an interrupt IN, then the host controller must record that the transaction occurred (for example, decrement *qHTransactionCounter*). It is recommended (but not required) the host controller continue executing transactions for this endpoint if the resultant value of *qHTransactionCounter* is greater than one.

If the response to the IN bus transaction is a Nak (or Nyet) and *RL* is non-zero, *NakCnt* is decremented by one. If *RL* is zero, then no write-back by the host controller is required (for a transaction receiving a Nak or Nyet response and the value of *CErr* did not change). Software should set the *RL* field to zero if the queue head is an interrupt endpoint. Host controller hardware is not required to enforce this rule or operation.

After the transaction has finished and the host controller has completed the post processing of the results (advancing the transfer state and possibly *NakCnt*, the host controller writes back the results of the transaction to the queue head's overlay area in main memory).

The number of bytes moved during an IN transaction depends on how much data the device endpoint delivers. The maximum number of bytes a device can send is *MaximumPacket Size*. The number of bytes moved during an OUT transaction is either *Maximum Packet Length* bytes or *Total Bytes to Transfer*, whichever is less.

If there was a transaction error during the transaction, the transfer state (as defined above) is not advanced by the host controller. The *CErr* field is decremented by one and the status field is updated to reflect the type of error observed. Transaction errors are summarized in Transaction Error .

The following events causes the host controller to clear the *Active* bit in the queue head's overlay status field. When the *Active* bit transitions from one to zero, the transfer in the overlay is considered complete. The reason for the transfer completion (clearing the *Active* bit) determines the next state.

- *CErr* field decrements to zero. When this occurs the *Halted* bit is set to one and *Active* is set to zero. This results in the hardware not advancing the queue and the pipe halts. Software must intercede to recover.
- The device responds to the transaction with a STALL PID. When this occurs, the *Halted* bit is set to one and the *Active* bit is set to zero. This results in the hardware not advancing the queue and the pipe halts. Software must intercede to recover.
- The *Total Bytes to Transfer* field is zero after the transaction completes.
  - For a zero length transaction, it was zero before the transaction was started. When this condition occurs, the *Active* bit is set to zero.
- The PID code is an IN, and the number of bytes moved during the transaction is less than the *Maximum Packet Length*. When this occurs, the *Active* bit is set to zero and a short packet condition exists. The short-packet condition is detected during the Advance Queue state. Refer to Split Transactions for additional rules for managing low- and full-speed transactions.

With the exception of a NAK response (when *RL* field is zero), the host controller always writes the results of the transaction back to the overlay area in main memory. This includes when the transfer completes. For a high-speed endpoint, the queue head information written back includes minimally the following fields: The *PID Code* field indicates an IN and the device sends more than the expected number of bytes (for example *Maximum Packet Length* or *Total Bytes to Transfer* bytes, whichever is less) (for example a packet babble). This results in the host controller setting the *Halted* bit to one.

- NakCnt, dt, Total Bytes to Transfer, C_Page, Status, CERR, and Current Offset

For a low- or full-speed device the queue head information written back also includes the fields:

- C-prog-mask, FrameTag and S-bytes.

The duration of this state depends on the time it takes to complete the transaction(s) and the status write to the overlay is committed.

11.1.3.3.10.3.4 Halting a Queue Head

A halted endpoint is defined only for the transfer types that are managed through queue heads (control, bulk and interrupt).

The following events indicate that the endpoint has reached a condition where no more activity can occur without intervention from the driver:

- An endpoint may return a STALL handshake during a transaction,
- A transaction had three consecutive error conditions, or
- A Packet Babble error occurs on the endpoint.

When any of these events occur (for a queue head) the Host Controller halts the queue head and set the USBERRINT status bit in the USB_n_USBSTS register to one. To halt the queue head, the *Active* bit is set to zero and the *Halted* bit is set to one. There may be other error status bits that are set when a queue is halted. The host controller always writes back the overlay area to the source qTD when the transfer is complete, regardless of the reason (normal completion, short packet or halt). The host controller does not advance the transfer state on a transaction that results in a *Halt* condition (for example no updates necessary for *Total Bytes to Transfer*, *C_Page*, *Current Offset,* and *dt*). The host controller must update *CErr* as appropriate. When a queue head is halted, the *USB Error Interrupt* bit in the USB_n_USBSTS register is set to one. If the *USB Error Interrupt Enable* bit in the USB_n_USBINTR register is set to one, a hardware interrupt is generated at the next interrupt threshold.

11.1.3.3.10.3.5 Asynchronous Schedule Park Mode

Asynchronous Schedule Park mode is a special execution mode that can be enabled by system software, where the host controller is permitted to execute more than one bus transaction from a high-speed queue head in the Asynchronous schedule before continuing horizontal traversal of the Asynchronous schedule.

This feature has no effect on queue heads or other data structures in the Periodic schedule. This feature is similar in intent as the *Mult* feature that is used in the Periodic schedule. Where-as the *Mult* feature is a characteristic that is tunable for each endpoint; park-mode is a policy that is applied to all high-speed queue heads in the asynchronous schedule. It is essentially the specification of an iterator for consecutive bus transactions to the same endpoint. All of the rules for managing bus transactions and the results of those as defined in Execute Transaction apply. This feature merely specifies how many consecutive times the host controller is permitted to execute from the same queue head before moving to the next queue head in the Asynchronous List. This feature should allow the host controller to attain better bus utilization for those devices that are capable of moving data at maximum rate, while at the same time providing a fair service to all endpoints.

A host controller exports its capability to support this feature to system software by setting the *Asynchronous Schedule Park Capability* bit in the USB_n_HCCPARAMs register to one. This information keys system software that the *Asynchronous Schedule Park Mode Enable* and *Asynchronous Schedule Park Mode Count* fields in the USB_n_USBCMD register are modifiable. System software enables the feature by writing a one to the *Asynchronous Schedule Park Mode Enable* bit.

When park-mode is not enabled (for example *Asynchronous Schedule Park Mode Enable* bit in the USB_n_USBCMD register is zero), the host controller must not execute more than one bus transaction per high-speed queue head, per traversal of the asynchronous schedule. When park-mode is enabled, the host controller must not apply the feature to a queue head whose *EPS* field indicates a Low/Full-speed device (for example only one bus transaction is allowed from each Low/Full-speed queue head per traversal of the asynchronous schedule). Park-mode may only be applied to queue heads in the Asynchronous schedule whose *EPS* field indicates that it is a high-speed device.

The host controller must apply park mode to queue heads whose *EPS* field indicates a high-speed endpoint. The maximum number of consecutive bus transactions a host controller may execute on a high-speed queue head is determined by the value in the *Asynchronous Schedule Park Mode Count* field in the USB_n_USBCMD register. Software must not set *Asynchronous Schedule Park Mode Enable* bit to one and also set *Asynchronous Schedule Park Mode Count* field to zero. The resulting behavior is not defined. An example behavioral example describes the operational requirements for the host controller implementing park-mode. This feature does not affect how the host controller handles the bus transaction as defined in Execute Transaction . It only effects how many consecutive bus transactions for the current queue head can be executed. All boundary conditions, error detection and reporting applies as usual. This feature is similar in concept to the use of the *Mult* field for high-bandwidth Interrupt for queue heads in the Periodic Schedule.

The host controller effectively loads an internal down-counter *PM-Count* from *Asynchronous Schedule Park Mode Count* when *Asyncrhonous Schedule Park Mode Enable* bit is one, and a high-speed queue head is first fetched and meets all the criteria for executing a bus transaction. After the bus transaction, *PM-Count* is decremented. The host controller may continue to execute bus transactions from the current queue head until *PM-Count* goes to zero, an error is detected, the buffer for the current transfer is exhausted or the endpoint responds with a flow-control or STALL handshake.

The following table summarizes the responses that effect whether the host controller continues with another bus transaction for the current queue head.

Table 11-42. Actions for Park Mode, based on Endpoint Response and Residual Transfer State

PID	Endpoint Response	Transfer State after Transaction		Action
		PM-Count	Bytes to Transfer	
IN	DATA[0,1] w/Maximum Packet sized data	Not zero	Not Zero	Allowed to perform another bus transaction.[1, 2]
		Not zero	Zero	Retire qTD and move to next QH
		Zero	Don't care	Move to next QH.

Table continues on the next page...

Table 11-42. Actions for Park Mode, based on Endpoint Response and Residual Transfer State (continued)

	DATA[0,1] w/short packet	Don't care	Don't care	Retire qTD and move to next QH.
	NAK	Don't care	Don't care	Move to next QH.
	STALL, XactErr	Don't care	Don't care	Move to next QH.
OUT	ACK	Not zero	Not Zero	Allowed to perform another bus transaction.[2]
		Not zero	Zero	Retire qTD and move to next QH
		Zero	Don't' care	Move to next QH.
	NYET, NAK	Don't care	Don't care	Move to next QH.
	STALL, XactErr	Don't care	Don't care	Move to next QH
PING	ACK	Not Zero	Not Zero	Allowed to perform another bus transaction. [2]
	NAK	Don't care	Don't care	Move to next QH
	STALL, XactErr	Don't care	Don't care	Move to next QH

1. The host controller may continue to execute bus transactions from the current high-speed queue head (if *PM-Count* is not equal to zero), if a PID mismatch is detected (for example expected DATA1 and received DATA0, or visa-versa).
2. This specification does not *require* that the host controller execute another bus transaction when *PM-Count* is non-zero. Implementations are encouraged to make appropriate complexity and performance trade-offs.

11.1.3.3.10.4 Write Back qTD

This state is entered from the Execute Transaction state when the *Active* bit is set to zero.

The source data for the write-back is the transfer results area of the queue head overlay area (see Table 11-42).

The host controller uses the *Current qTD Pointer* field as the target address for the qTD.

The queue head transfer result area is written back to the transfer result area of the target qTD. This state is also referred to as: qTD retirement. The fields that must be written back to the source qTD include *Total Bytes to Transfer*, *Cerr*, and *Status*.

The duration of this state depends on when the qTD write-back is committed.

11.1.3.3.10.5 Follow Queue Head Horizontal Pointer

The host controller must use the horizontal pointer in the queue head to the next schedule data structure when any of the following conditions exist:

- If the *Active* bit is one on exit from the Execute Transaction state, or
- When the host controller exits the Write Back qTD state, or

- If the Advance Queue state fails to advance the queue because the target qTD is not active, or
- If the *Halted* bit is one on exit from the Fetch QH state.

There is no functional requirement that the host controller wait until the current transaction is complete before using the horizontal pointer to read the next linked data structure. However, it must wait until the current transaction is complete before executing the next data structure.

11.1.3.3.10.6 Buffer Pointer List Use for Data Streaming with qTDs

A qTD has an array of buffer pointers, which is used to reference the data buffer for a transfer. This specification requires that the buffer associated with the transfer be *virtually contiguous*.

This means: if the buffer spans more than one physical page, it must obey the following rules (the figure below illustrates an example):

- The first portion of the buffer must begin at some offset in a page and extend through the end of the page.
- The remaining buffer cannot be allocated in small chunks scattered around memory. For each 4 K chunk beyond the first page, each buffer portion matches to a full 4 K page. The final portion, which may only be large enough to occupy a portion of a page, must start at the top of the page and be contiguous within that page.

The buffer pointer list in the qTD is long enough to support a maximum transfer size of 20 K bytes. This case occurs when all five buffer pointers are used and the first offset is zero. A qTD handles a 16 Kbyte buffer with any starting buffer alignment.

The host controller uses the field *C_Page* field as an index value to determine which buffer pointer in the list should be used to start the current transaction. The host controller uses a different buffer pointer for each physical page of the buffer. This is always true, even if the buffer is physically contiguous.

The host controller must detect when the current transaction spans a page boundary and automatically move to the next available buffer pointer in the page pointer list. The next available pointer is reached by incrementing *C_Page* and pulling the next page pointer from the list. Software must ensure there are sufficient buffer pointers to move the amount of data specified in the *Bytes to Transfer* field.

The following figure illustrates a nominal example of how System software would initialize the buffer pointers list and the *C_Page* field for a transfer size of 16383 bytes. *C_Page* is set to zero. The upper 20-bits of Page 0 references the start of the physical

page. *Current Offset* (the lower 12-bits of queue head Dword 7) holds the offset in the page for example 2049 (for example 4096-2047). The remaining page pointers are set to reference the beginning of each subsequent 4 K page.

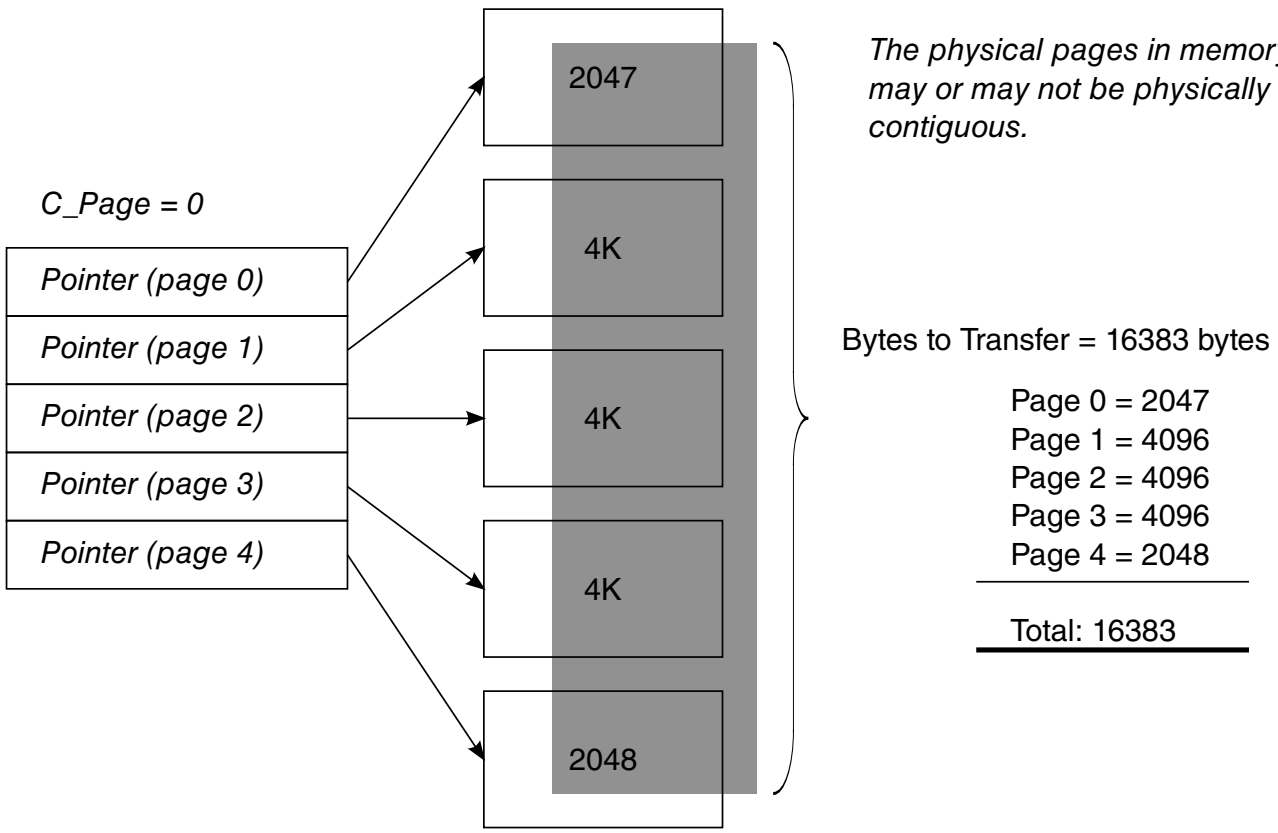


Figure 11-19. Example Mapping of qTD Buffer Pointers to Buffer Pages

For the first transaction on the qTD (assuming a 512-byte transaction), the host controller uses the first buffer pointer (page 0 because *C_Page* is set to zero) and concatenates the *Current Offset* field. The 512 bytes are moved during the transaction, the *Current Offset* and *Total Bytes to Transfer* are adjusted by 512 and written back to the queue head working area.

During the 4th transaction, the host controller needs 511 bytes in page 0 and one byte in page 1. The host controller increments *C_Page* (to 1) and use the page 1 pointer to move the final byte of the transaction. After the 4th transaction, the active page pointer is the page 1 pointer and *Current Offset* has rolled to one, and both are written back to the overlay area. The transactions continue for the rest of the buffer, with the host controller automatically moving to the next page pointer (that is *C_Page*) when necessary. The three conditions for how the host controller handles *C_Page*:

- The current transaction does not span a page boundary. The value of *C_Page* is not adjusted by the host controller.

- The current transaction does span a page boundary. The host controller must detect the page cross condition and advance to the next buffer while streaming data to/from the USB.
- The current transaction completes on a page boundary (that is the last byte moved for the current transaction is the last byte in the page for the current page pointer). The host controller must increment *C_Page* before writing back status for the transaction.

NOTE

The only valid adjustment the host controller may make to *C_Page* is to increment by one.

11.1.3.3.10.7 Adding Interrupt Queue Heads to the Periodic Schedule

The link path(s) from the periodic frame list to a queue head establishes in which frames a transaction can be executed for the queue head. Queue heads are linked into the periodic schedule so they are polled at the appropriate rate.

System software sets a bit in a queue head's *S-Mask* to indicate which micro-frame within 1 msec period a transaction should be executed for the queue head. Software must ensure that all queue heads in the periodic schedule have *S-Mask* set to a non-zero value. An *S-mask* with zero value in the context of the periodic schedule yields undefined results.

If the desired poll rate is greater than one frame, system software can use a combination of queue head linking and *S-Mask* values to spread interrupts of equal poll rates through the schedule so that the periodic bandwidth is allocated and managed in the most efficient manner possible. Some examples are illustrated in the following table.

Table 11-43. Example Periodic Reference Patterns for Interrupt Transfers with 2ms Poll Rate

Frame # Reference Sequence	Description
0, 2, 4, 6, 8, and so on *S-Mask* = 01h	A queue head for the *bInterval* of 2 msec (16 micro-frames) is linked into the periodic schedule so that it is reachable from the periodic frame list locations indicated in the previous column. In addition, the *S-Mask* field in the queue head is set to 01h, indicating that the transaction for the endpoint should be executed on the bus during micro-frame 0 of the frame.
0, 2, 4, 6, 8, and so on *S-Mask* = 02h	Another example of a queue head with a *bInterval* of 2 msec is linked into the periodic frame list at exactly the same interval as the previous example. However, the *S-Mask* is set to 02h indicating that the transaction for the endpoint should be executed on the bus during micro-frame 1 of the frame.

11.1.3.3.10.8 Managing Transfer Complete Interrupts from Queue Heads

The host controller sets an interrupt to be signaled at the next interrupt threshold when the completed transfer (qTD) has an *Interrupt on Complete* (*IOC*) bit set to one, or whenever a transfer (qTD) completes with a short packet.

If system software needs multiple qTDs to complete a client request (that is like a control transfer) the intermediate qTDs do not require interrupts. System software may only need a single interrupt to notify it that the complete buffer has been transferred. System software may set IOC's to occur more frequently. A motivation for this may be that it wants early notification so that interface data structures can be re-used in a timely manner.

11.1.3.3.11 Ping Control

USB 2.0 defines an addition to the protocol for high-speed devices called Ping. Ping is required for all USB 2.0 High-speed bulk and control endpoints.

Ping is not allowed for a split-transaction stream. This extension to the protocol eliminates the bad side-effects of Naking OUT endpoints. The *Status* field has a *Ping State* bit, which the host controller uses to determine the *next* actual PID it uses in the next transaction to the endpoint (see the table below).

The Ping State bit is only managed by the host controller for queue heads that meet the following criteria:

- Queue head is not an interrupt and
- *EPS* field equals High-Speed and
- *PIDCode* field equals OUT

The following table illustrates the state transition table for the host controller's responsibility for maintaining the PING protocol. Refer to Chapter 8 in the USB Specification Revision 2.0 for detailed description on the Ping protocol.

Table 11-44. Ping Control State Transition Table

Event			
Current	Host	Device	Next
Do Ping	PING	Nak	Do Ping
Do Ping	PING	Ack	Do OUT
Do Ping	PING	XactErr [1]	Do Ping
Do Ping	PING	Stall	N/C [2] Do
OUT	OUT	Nak	Do Ping
Do OUT	OUT	Nyet	Do Ping
Do OUT	OUT	Ack	Do OUT

Table continues on the next page...

Table 11-44. Ping Control State Transition Table (continued)

Do OUT	OUT	XactErr[1]	Do Ping
Do OUT	OUT	Stall	N/C [2]

1. Transaction Error (XactErr) is any time the host misses the handshake.
2. No transition change required for the Ping State bit. The Stall handshake results in the endpoint being halted (for example Active set to zero and Halt set to one). Software intervention is required to restart queue. 3 A Nyet response to an OUT means that the device has accepted the data, but cannot receive any more at this time. Host must advance the transfer state and additionally, transition the Ping State bit to Do Ping. The Ping State bit has the following encoding:

Table 11-45. Ping State bit Encoding

Value	Meaning
0B	Do OUT The host controller uses an OUT PID during the next bus transaction to this endpoint.
1B	Do Ping The host controller uses a PING PID during the next bus transaction to this endpoint.

The defined ping protocol (see USB 2.0 Specification, Chapter 8) allows the host to be *imprecise* on the initialization of the ping protocol (that is start in *Do OUT* when we don't know whether there is space on the device or not). The host controller manages the *Ping State* bit. System software sets the initial value in the queue head when it initializes a queue head. The host controller preserves the *Ping State* bit across all queue advancements. This means that when a new qTD is written into the queue head overlay area, the previous value of the *Ping State* bit is preserved.

11.1.3.3.12 Split Transactions

USB 2.0 defines extensions to the bus protocol for managing USB 1.x data streams through USB 2.0 Hubs.

This section describes how the host controller uses the interface data structures to manage data streams with full- and low-speed devices, connected below USB 2.0 hub, utilizing the split transaction protocol.

Refer to USB 2.0 Specification for the complete definition of the split transaction protocol. Full- and Low-speed devices are enumerated identically as high-speed devices, but the transactions to the Full- and Low-speed endpoints use the split-transaction protocol on the high-speed bus. The split transaction protocol is an encapsulation of (or wrapper around) the Full- or Low-speed transaction. The high-speed wrapper portion of the protocol is addressed to the USB 2.0 Hub and Transaction Translator below which the Full- or Low-speed device is attached.

The EHCI interface uses dedicated data structures for managing full-speed isochronous data streams (see Split Transaction Isochronous Transfer Descriptor (siTD)). Control, Bulk and Interrupt are managed using the queuing data structures (see Queue Head). The interface data structures need to be programmed with the device address and the

Transaction Translator number of the USB 2.0 Hub operating as the Low-/Full-speed host controller for this link. The following sections describe the details of how the host controller must process and manage the split transaction protocol.

11.1.3.3.12.1 Split Transactions for Asynchronous Transfers

A queue head in the asynchronous schedule with an *EPS* field indicating a full-or low-speed device indicates to the host controller that it must use split transactions to stream data for this queue head.

All full-speed bulk and full-, low-speed control are managed through queue heads in the asynchronous schedule.

Software must initialize the queue head with the appropriate device address and port number for the transaction translator that is serving as the full/low-speed host controller for the links connecting the endpoint. Software must also initialize the split transaction state bit (*SplitXState*) to Do-Start-Split. Finally, if the endpoint is a control endpoint, then system software must set the *Control Transfer Type* (*C*) bit in the queue head to one. If this is not a control transfer type endpoint, the *C* bit must be initialized by software to be zero. This information is used by the host controller to properly set the Endpoint Type (ET) field in the split transaction bus token. When the *C* bit is zero, the split transaction token's ET field is set to indicate a bulk endpoint. When the *C* bit is one, the split transaction token's ET field is set to indicate a control endpoint. Refer to Chapter 8 of USB Specification Revision 2.0 for details.

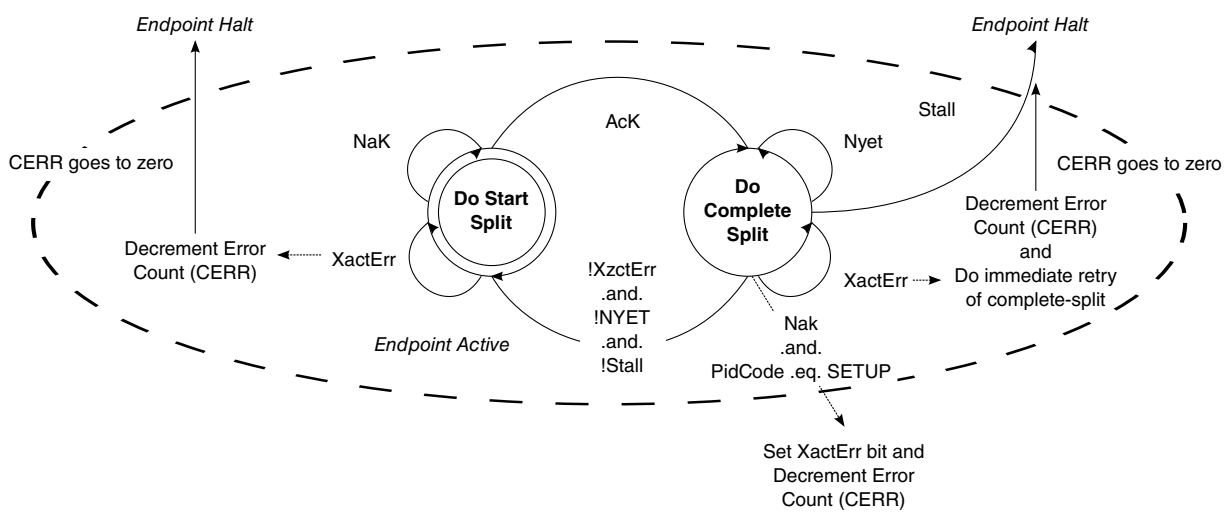


Figure 11-20. Host Controller Asynchronous Schedule Split-Transaction State Machine

11.1.3.3.12.1.1 Asynchronous - Do Start Split

This is the state which software must initialize a full- or low-speed asynchronous queue head. This state is entered from the Do Complete Split state only after a complete-split transaction receives a valid response from the transaction translator that is not a Nyet handshake.

For queue heads in this state, the host controller executes a start-split transaction to the appropriate transaction translator. If the bus transaction completes without an error and *PidCode* indicates an IN or OUT transaction, then the host controller reloads the error counter (*CErr*). If it is a successful bus transaction and the *PidCode* indicates a SETUP, the host controller does not reload the error counter. If the transaction translator responds with a Nak, the queue head is left in this state, and the host controller proceeds to the next queue head in the asynchronous schedule.

If the host controller times out the transaction (no response, or bad response) the host controller decrements *Cerr* and proceeds to the next queue head in the asynchronous schedule.

11.1.3.3.12.1.2 Asynchronous - Do Complete Split

This state is entered from the Do Start Split state only after a start-split transaction receives an Ack handshake from the transaction translator.

For queue heads in this state, the host controller executes a complete-split transaction to the appropriate transaction translator. If the transaction translator responds with a Nyet handshake, the queue head is left in this state, the error counter is reset and the host controller proceeds to the next queue head in the asynchronous schedule. When a Nyet handshake is received for a bus transaction where the queue head's *PidCode* indicates an IN or OUT, the host controller reloads the error counter (*CErr*). When a Nyet handshake is received for a complete-split bus transaction where the queue head's *PidCode* indicates a SETUP, the host controller must not adjust the value of *CErr*.

Independent of *PIDCode*, the following responses have the effects:

- Transaction Error (XactErr). Timeout or data CRC failure, and so on. The error counter (*Cerr*) is decremented by one and the complete split transaction is *immediately* retried (if possible). If there is not enough time in the micro-frame to execute the retry, the host controller MUST ensure that the next time the host controller begins executing from the Asynchronous schedule, it must begin executing from this queue head. If another start-split (for some other endpoint) is sent to the transaction translator before the complete-split is really completed, the transaction translator could dump the results (which were never delivered to the host). This is why the core specification states the retries must be immediate. A method to accomplish this behavior is to not advance the asynchronous schedule. When the host

controller returns to the asynchronous schedule in the next micro-frame, the first transaction from the schedule is the retry for this endpoint.

If *Cerr* went to zero, the host controller must halt the queue.

- NAK. The target endpoint Nak'd the full- or low-speed transaction. The state of the transfer is not advanced and the state is exited. If the *PidCode* is a SETUP, then the Nak response is a protocol error. The *XactErr* status bit is set to one and the *CErr* field is decremented.
- STALL. The target endpoint responded with a STALL handshake. The host controller sets the *halt* bit in the status byte, retires the qTD but does not attempt to advance the queue.

If the *PidCode* indicates an IN, then any of following responses are expected:

- DATA0/1. On reception of data, the host controller ensures the PID matches the expected data toggle and checks CRC. If the packet is *good*, the host controller advances the state of the transfer, for example move the data pointer by the number of bytes received, decrement *BytesToTransfer* field by the number of bytes received, and toggle the *dt* bit. The host controller then exit this state. The response and advancement of transfer may trigger other processing events, such as retirement of the qTD and advancement of the queue.

If the data sequence PID does not match the expected, the data is ignored, the transfer state is not advanced and this state is exited. If the *PidCode* indicates an OUT/SETUP, then any of following responses are expected:

- ACK. The target endpoint accepted the data, so the host controller must advance the state of the transfer. The *Current Offset* field is incremented by *Maximum Packet Length* or *Bytes to Transfer*, whichever is less. The field *Bytes To Transfer* is decremented by the same amount and the data toggle bit (*dt*) is toggled. The host controller then exit this state.
- Advancing the transfer state may cause other processing events such as retirement of the qTD and advancement of the queue (see Managing Control/Bulk/Interrupt Transfers through Queue Heads).

11.1.3.3.12.2 Split Transaction Interrupt

Split-transaction Interrupt-IN/OUT endpoints are managed through the same data structures used for high-speed interrupt endpoints. They both co-exist in the periodic schedule.

Queue heads/qTDs offer the set of features required for reliable data delivery, which is characteristic to interrupt transfer types. The split-transaction protocol is managed completely within this defined functional transfer framework. For example, for a high-speed endpoint, the host controller visits a queue head, execute a high-speed transaction (if criteria are met) and advance the transfer state (or not) depending on the results of the entire transaction. For low- and full-speed endpoints, the details of the *execution* phase are different (that is takes more than one bus transaction to complete), but the remainder of the operational framework is intact. This means that the transfer advancement, and so on, occurs as defined in Managing Control/Bulk/Interrupt Transfers through Queue Heads, but only occurs on the completion of a split transaction.

11.1.3.3.12.2.1 Split Transaction Scheduling Mechanisms for Interrupt

Full- and low-speed Interrupt queue heads have an *EPS* field indicating full- or low-speed and have a non-zero *S-mask* field.

The host controller can detect this combination of parameters and assume the endpoint is a periodic endpoint. Low- and full-speed interrupt queue heads require the use of the split transaction protocol. The host controller sets the Endpoint Type (ET) field in the split token to indicate the transaction is an interrupt. These transactions are managed through a transaction translator's periodic pipeline. Software should not set these fields to indicate the queue head is an interrupt unless the queue head is used in the periodic schedule.

System software manages the per/transaction translator periodic pipeline by budgeting and scheduling exactly during which micro-frames the start-splits and complete-splits for each endpoint occurs. The characteristics of the transaction translator are such that the high-speed transaction protocol must execute during explicit micro-frames, or the data or response information in the pipeline is lost.

The following figure illustrates the general scheduling boundary conditions that are supported by the EHCI periodic schedule and queue head data structure. The S and ^{C}X labels indicate micro-frames where software can schedule start-splits and complete splits (respectively).

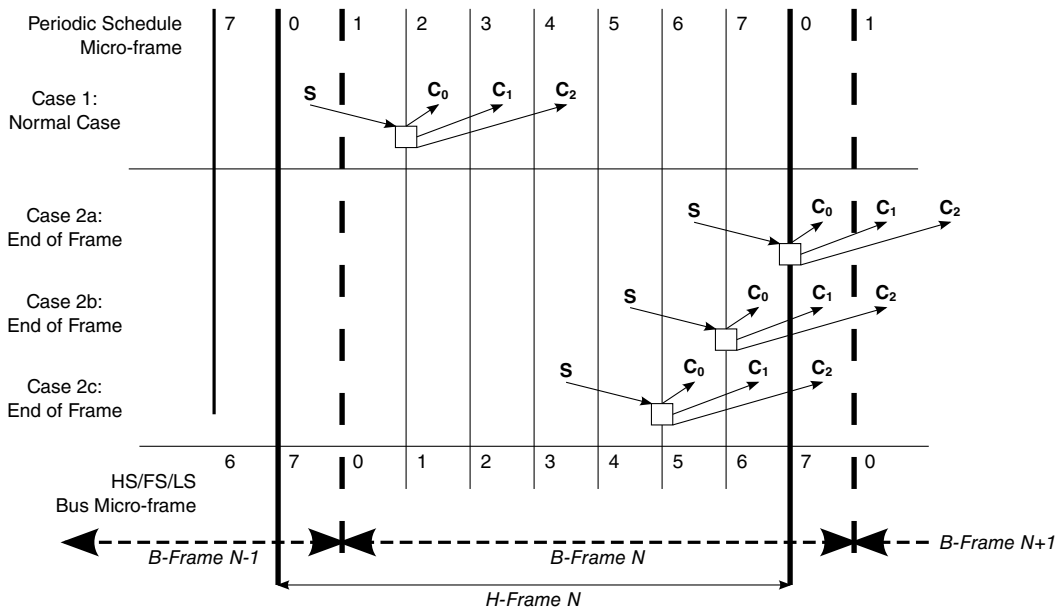


Figure 11-21. Split Transaction, Interrupt Scheduling Boundary Conditions

The scheduling cases are:

- Case 1: The normal scheduling case is where the entire split transaction is completely bounded by a frame (*H-Frame* in this case).
- Case 2a through Case 2c: The USB 2.0 Hub pipeline rules states clearly, when and how many complete-splits must be scheduled to account for earliest to latest execution on the full/low-speed link. The complete-splits may span the *H-Frame* boundary when the start-split is in micro-frame 4 or later. When this occurs, the *H-Frame* to *B-Frame* alignment requires that the queue head be reachable from consecutive periodic frame list locations. System software cannot build an efficient schedule that satisfies this requirement unless it uses FSTNs.

The figure below illustrates the general layout of the periodic schedule.

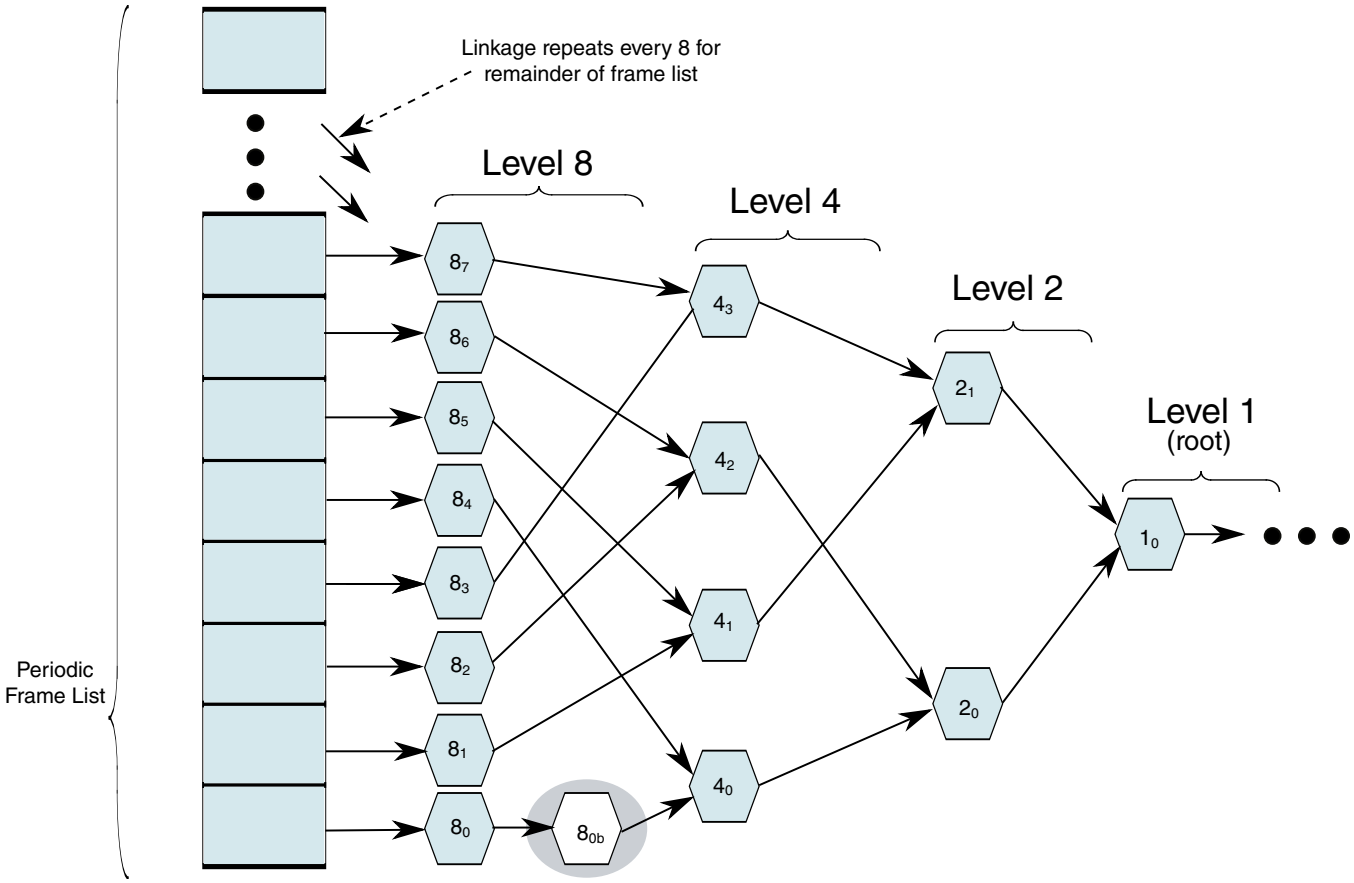


Figure 11-22. General Structure of EHCI Periodic Schedule Utilizing Interrupt Spreading

The periodic frame list is effectively the leaf level a binary tree, which is always traversed leaf to root. Each level in the tree corresponds to a 2^N poll rate. Software can efficiently manage periodic bandwidth on the USB by *spreading* interrupt queue heads that have the same poll rate requirement across all the available paths from the frame list. For example, system software can schedule eight poll rate 8 queue heads and account for them once in the high-speed bus bandwidth allocation.

When an endpoint is allocated an execution footprint that spans a frame boundary, the queue head for the endpoint must be reachable from consecutive locations in the frame list. An example would be if 8_{0b} where such an endpoint. Without additional support on the interface, to get 8_{0b} reachable at the correct time, software would have to link 8_1 to 8_{0b}. It would then have to move 4_1 and everything linked after into the same path as 4_0. This upsets the integrity of the binary tree and disallows the use of the spreading technique.

FSTN data structures are used to preserve the integrity of the binary-tree structure and enable the use of the spreading technique. Host Controller Operational Model for FSTNs defines the hardware and software operational model requirements for using FSTNs.

The following queue head fields are initialized by system software to instruct the host controller when to execute portions of the split-transaction protocol:

- *SplitXState*. This is single bit residing in the *Status* field of a queue head (see Table 11-22). This bit is used to track the current state of the split transaction.
- *Frame S-mask*. This is a bit-field where-in system software sets a bit corresponding to the micro-frame (within an *H-Frame*) that the host controller should execute a start-split transaction. This is always qualified by the value of the *SplitXState* bit in the *Status* field of the queue head. For example, referring to Figure 11-21, case one, the *S-mask* would have a value of 00000001b indicating that if the queue head is traversed by the host controller, and the *SplitXState* indicates Do_Start, and the current micro-frame as indicated by FRINDEX[2:0] is 0, then execute a start-split transaction.
- *Frame C-mask.* This is a bit-field where system software sets one or more bits corresponding to the micro-frames (within an *H-Frame*) that the host controller should execute complete-split transactions. The interpretation of this field is always qualified by the value of the *SplitXState* bit in the *Status* field of the queue head. For example, referring to Figure 11-21, case one, the *C-mask* would have a value of 00011100b indicating that if the queue head is traversed by the host controller, and the *SplitXState* indicates Do_Complete, and the current micro-frame as indicated by FRINDEX[2:0] is 2, 3, or 4, then execute a complete-split transaction. It is software's responsibility to ensure that the translation between *H-Frames* and *B-Frames* is correctly performed when setting bits in *S-mask* and *C-mask*

11.1.3.3.12.2.2 Host Controller Operational Model for FSTNs

The FSTN data structure is used to manage Low/Full-speed interrupt queue heads that need to be reached from consecutive frame list locations (that is boundary cases 2a through 2c).

An FSTN is essentially a *back pointer*, similar in intent to the back pointer field in the siTD data structure (see siTD Back Link Pointer).

This feature provides software a simple primitive to save a schedule position, redirect the host controller to traverse the necessary queue heads in the previous frame, then restore the original schedule position and complete normal traversal.

The four components to the use of FSTNs:

- FSTN data structure.
- A *Save Place* indicator. This is always an FSTN with its *Back Path Link Pointer.T-bit* set to zero.

- A *Restore* indicator. This is always an FSTN with its *Back Path Link Pointer.T-bit* set to one.
- Host controller FSTN traversal rules.

When the host controller encounters an FSTN during micro-frames 2 through 7 it simply follows the node's *Normal Path Link Pointer* to access the next schedule data structure.

NOTE

The FSTN's *Normal Path Link Pointer.T-bit* may set to one, which the host controller must interpret as the end of periodic list mark.

When the host controller encounters a *Save-Place* FSTN in micro-frames 0 or 1, it saves the value of the *Normal Path Link Pointer* and set an internal flag indicating that it is executing in *Recovery Path* mode. *Recovery Path* mode modifies the host controller's rules for how it traverses the schedule and limits which data structures is considered for execution of bus transactions. The host controller continues executing in *Recovery Path* mode until it encounters a *Restore* FSTN or it determines that it has reached the end of the micro-frame (see details in the list below).

The rules for schedule traversal and limited execution while in *Recovery Path* mode are:

- Always follow the *Normal Path Link Pointer* when it encounters an FSTN that is a *Save-Place* indicator. The host controller must not recursively follow *Save-Place* FSTNs. Therefore, while executing in *Recovery Path* mode, it must never follow an FSTN's *Back Path Link Pointer*.
- Do not process an siTD or, iTD data structure. Simply follow its *Next Link Pointer*.
- Do not process a QH (Queue Head) whose *EPS* field indicates a high-speed device. Simply follow its *Horizontal Link Pointer*.
- When a QH's *EPS* field indicates a Full/Low-speed device, the host controller considers only it for execution if its *SplitXState* is DoComplete (note: this applies whether the *PID Code* indicates an IN or an OUT). See Execute Transaction and Tracking Split Transaction Progress for Interrupt Transfers for a complete list of additional conditions that must be met in general for the host controller to issue a bus transaction.
  - The host controller must not execute a Start-split transaction while executing in *Recovery Path* mode. See Periodic Isochronous - Do Complete Split for special handling when in *Recovery Path* mode.
- Stop traversing the *recovery path* when it encounters an FSTN that is a *Restore* indicator. The host controller unconditionally uses the saved value of the *Save-Place* FSTN's *Normal Path Link Pointer* when returning to the normal path traversal. The

host controller must clear the context of executing a *Recovery Path* when it restores schedule traversal to the *Save-Place* FSTN's *Normal Path Link Pointer*.

- If the host controller determines that there is not enough time left in the micro-frame to complete processing of the periodic schedule, it abandons traversal of the recovery path, and clears the context of executing a recovery path. The result is that at the start of the next consecutive micro-frame, the host controller starts traversal at the frame list.

An example traversal of a periodic schedule that includes FSTNs is illustrated in the following figure.

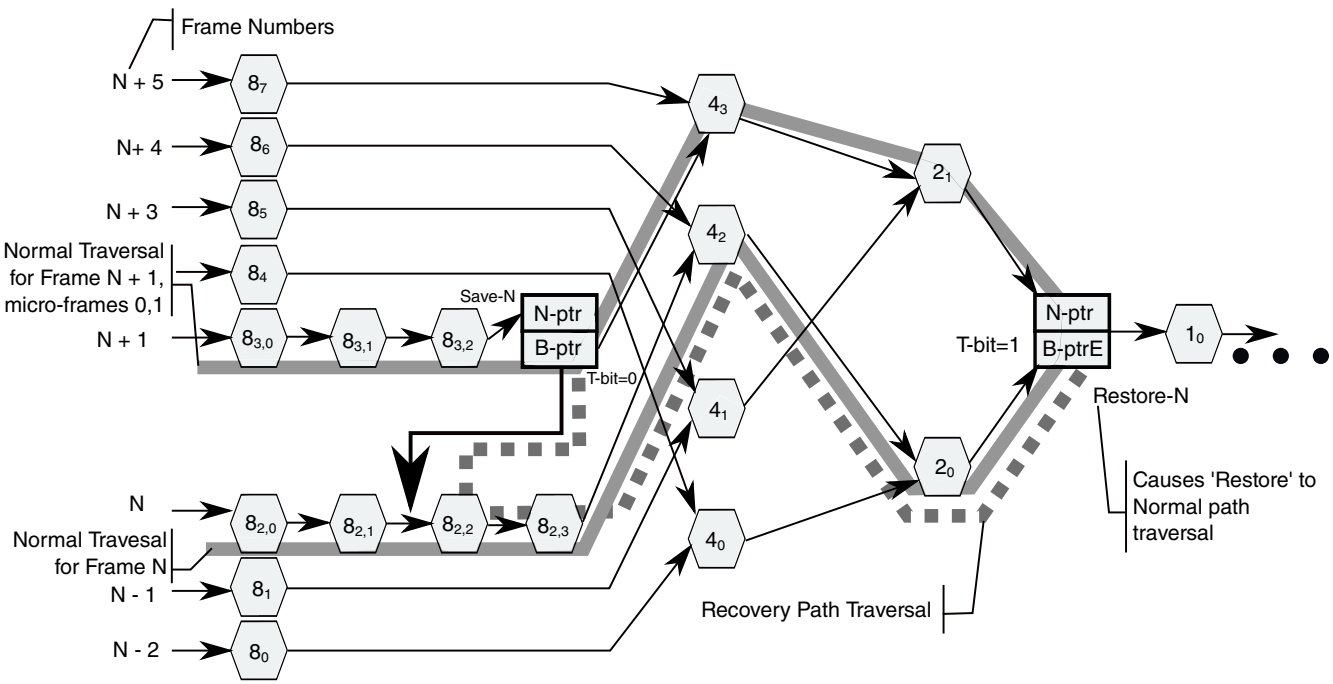


Figure 11-23. Example Host Controller Traversal of Recovery Path via FSTNs

In frame N+1 (micro-frames 0 and 1), when the host controller encounters Save-Path FSTN (Save-N), it observes that Save-N.Back Path Link Pointer.T-bit is zero (definition of a Save-Path indicator). The host controller saves the value of Save-N.Normal Path Link Pointer and follows Save-N.Back Path Link Pointer. At the same time, it sets an internal flag indicating that it is now in *Recovery Path* mode (the recovery path is annotated in the figure above with a large dashed line). The host controller continues traversing data structures on the recovery path and executing only those bus transactions as noted above, on the recovery path until it reaches Restore FSTN (Restore-N). Restore-N.Back Path Link Pointer.T-bit is set to one (definition of a Restore indicator), so the host controller exits *Recovery Path* mode by clearing the internal *Recovery Path* mode flag and commences (restores) schedule traversal using the saved value of the *Save-Place* FSTN's *Normal Path Link Pointer* (for example Save-N.Normal Path Link Pointer). The nodes traversed during these micro-frames include: {$8_{3.0}$, $8_{3.1}$, $8_{3.2}$, Save-A, $8_{2.2}$, $8_{2.3}$, 4_2,

2_0, Restore-N, 4_3, 2_1, Restore-N, 1_0 …}. The nodes on the recovery-path are in bold. In frame N (micro-frames 0-7), for this example, the host controller traverses all of the schedule data structures utilizing the *Normal Path Link Pointers* in any FSTNs it encounters. This is because the host controller has not yet encountered a *Save-Place* FSTN so it not executing in *Recovery Path* mode. When it encounters the *Restore* FSTN, (Restore-N), during micro-frames 0 and 1, it uses Restore-N.Normal Path Link Pointer to traverse to the next data structure (that is normal schedule traversal). This is because the host controller must use a Restore FSTN's *Normal Path Link Pointer* when not executing in a *Recovery-Path* mode. The nodes traversed during frame N include: {$8_{2.0}$, $8_{2.1}$, $8_{2.2}$, $8_{2.3}$, 4_2, 2_0, Restore-N, 1_0 …}.

In frame N+1 (micro-frames 2-7), when the host controller encounters Save-Path FSTN Save-N, it unconditionally follows Save-N.Normal Path Link Pointer. The nodes traversed during these micro-frames include: {$8_{3.0}$, $8_{3.1}$, $8_{3.2}$, Save-A, 4_3, 2_1, Restore-N, 1_0 …}.

11.1.3.3.12.2.3 Software Operational Model for FSTNs

Software must create a consistent, coherent schedule for the host controller to traverse.

When using FSTNs, system software must adhere to the following rules:

- Each *Save-Place* indicator requires a matching *Restore* indicator.
  - The *Save-Place* indicator is an FSTN with a valid *Back Path Link Pointer and T-bit* equal to zero.
    - *Back Path Link Pointer.Typ* field must be set to indicate the referenced data structure is a queue head. The *Restore* indicator is an FSTN with its *Back Path Link Pointer.T-bit* set to one.
  - A *Restore* FSTN may be matched to one or more *Save-Place* FSTNs. For example, if the schedule includes a poll-rate 1 level, then system software only needs to place a *Restore* FSTN at the beginning of this list in order to match all possible *Save-Place* FSTNs.
- If the schedule does not have elements linked at a poll-rate level of one, and one or more *Save-Place* FSTNs are used, then System Software must ensure the *Restore* FSTN's *Normal Path Link Pointer*'s *T-bit* is set to one, as this is used to mark the end of the periodic list.
- When the schedule does have elements linked at a poll rate level of one, a *Restore* FSTN must be the first data structure on the poll rate one list. All traversal paths from the frame list converge on the poll-rate one list. System software must ensure that *Recovery Path* mode is exited before the host controller is allowed to traverse the poll rate level one list.
- A *Save-Place* FSTN's *Back Path Link Pointer* must reference a queue head data structure. The referenced queue head must be reachable from the previous frame list

location. In other words, if the *Save-Place* FSTN is reachable from frame list offset N, then the FSTN's *Back Path Link Pointer* must reference a queue head that is reachable from frame list offset N-1.

Software should make the schedule as efficient as possible. What this means in this context is that software should have no more than one *Save-Place* FSTN reachable in any single frame. Note there is times when two (or more, depending on the implementation) could exist as full/low-speed footprints change with bandwidth adjustments. This could occur, for example when a bandwidth re-balance causes system software to move the *Save-Place* FSTN from one poll rate level to another. During the transition, software must preserve the integrity of the previous schedule until the new schedule is in place.

11.1.3.3.12.2.4 Tracking Split Transaction Progress for Interrupt Transfers

To correctly maintain the data stream, the host controller must be able to detect and report errors where data is lost.

For interrupt-IN transfers, data is lost when it makes it into the USB 2.0 hub, but the USB 2.0 host system is unable to get it from the USB 2.0 Hub and into the system before it expires from the transaction translator pipeline.

When a lost data condition is detected, the queue must be halted, thus signaling system software to recover from the error. A data-loss condition exists whenever a start-split is issued, accepted and successfully executed by the USB 2.0 Hub, but the complete-splits get unrecoverable errors on the high-speed link, or the complete-splits do not occur at the correct times. One reason complete-splits might not occur at the right time would be due to host-induced system hold-offs that cause the host controller to miss bus transactions because it cannot get timely access to the schedule in system memory.

The same condition can occur for an interrupt-OUT, but the result is not an endpoint halt condition, but rather effects only the progress of the transfer. The queue head has the following fields to track the progress of each split transaction. These fields are used to keep incremental state about which (and when) portions have been executed.

- *C-prog-mask*. This is an eight-bit bit-vector where the host controller keeps track of which complete-splits have been executed. Due to the nature of the Transaction Translator periodic pipeline, the complete-splits need to be executed in-order. The host controller needs to detect when the complete-splits have not been executed in order. This can only occur due to system hold-offs where the host controller cannot get to the memory-based schedule. *C-prog-mask* is a simple bit-vector that the host controller sets one of the *C-prog*-mask bits for each complete-split executed. The bit position is determined by the micro-frame number in which the complete-split was executed. The host controller always checks *C-prog-mask* before executing a

complete-split transaction. If the previous complete-splits have not been executed then it means one (or more) have been skipped and data has potentially been lost.
- *FrameTag*. This field is used by the host controller during the complete-split portion of the split transaction to tag the queue head with the frame number (*H-Frame* number) when the next complete split must be executed.
- *S-bytes*. This field can be used to store the number of data payload bytes sent during the start-split (if the transaction was an OUT). The *S-bytes* field must be used to accumulate the data payload bytes received during the complete-splits (for an IN).

11.1.3.3.12.2.5 Split Transaction Execution State Machine for Interrupt

In the following presentation, all references to micro-frame are in the context of a micro-frame within an *H-Frame.*

As with asynchronous Full- and Low-speed endpoints, a split-transaction state machine is used to manage the split transaction sequence.

Aside from the fields defined in the queue head for scheduling and tracking the split transaction, the host controller calculates one internal mechanism that is also used to manage the split transaction. The internal calculated mechanism is:

- *cMicroFrameBit* is a single-bit encoding of the current micro-frame number. It is an eight-bit value calculated by the host controller at the beginning of every micro-frame. It is calculated from the three least significant bits of the *FRINDEX* register (that is, *cMicroFrameBit* = (1 shifted-left(*FRINDEX*[2:0])). The *cMicroFrameBit* has at most one bit asserted, which always corresponds to the current micro-frame number. For example, if the current micro-frame is 0, then *cMicroFrameBit* will equal 00000001b. The variable *cMicroFrameBit* is used to compare against the *S-mask* and *C-mask* fields to determine whether the queue head is marked for a start- or complete-splt transaction for the current micro-frame.

The following figure illustrates the state machine for managing a complete interrupt split transaction. There are two phases to each split transaction. The first is a single start-split transaction, which occurs when the *SplitXState* is at Do_Start and the single bit in *cMicroFrameBit* has a corresponding bit active in *QH.S-mask*. The transaction translator does not acknowledge the receipt of the periodic start-split, so the host controller unconditionally transitions the state to Do_Complete. Due to the available jitter in the transaction translator pipeline, there will be more than one complete-split transaction scheduled by software for the Do_Complete state. This translates simply to the fact that there are multiple bits set to a one in the *QH.C-mask* field.

The host controller keeps the queue head in the Do_Complete state until the split transaction is complete (see definition below), or an error condition triggers the *three-strikes-rule* (for example, after the host tries the same transaction three times, and each

encounters an error, the host controller will stop retrying the bus transaction and halt the endpoint, thus requiring system software to detect the condition and perform system-dependent recovery).

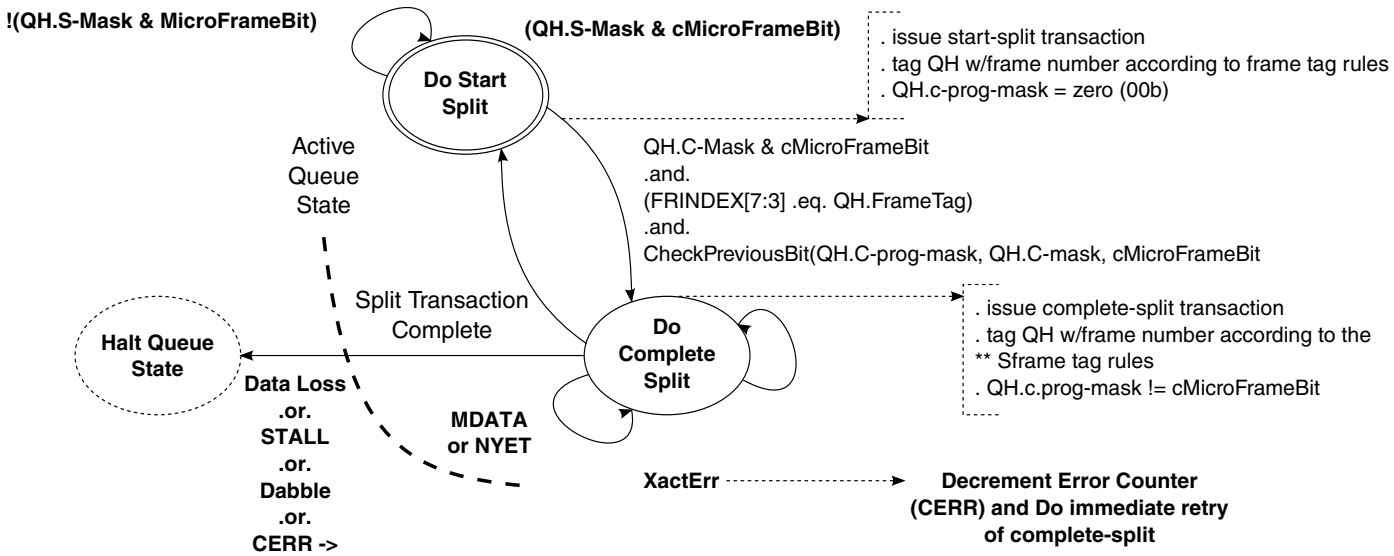


Figure 11-24. Split Transaction State Machine for Interrupt

See Previous Section for the frame tag management rules.

Periodic Interrupt - Do Start Split

This is the state software must initialize a full- or low-speed interrupt queue head *StartXState* bit. This state is entered from the Do_Complete Split state only after the split transaction is complete. This occurs when one of the following events occur: The transaction translator responds to a complete-split transaction with one of the following:

- NAK. A NAK response is a propagation of the full- or low-speed endpoint's NAK response.
- ACK. An ACK response is a propagation of the full- or low-speed endpoint's ACK response. Only occurs on an OUT endpoint.
- DATA 0/1. Only occurs for INs. Indicates that this is the last of the data from the endpoint for this split transaction.
- ERR. The transaction on the low-/full-speed link below the transaction translator had a failure (for example, timeout, bad CRC, etc.).
- NYET (and Last). The host controller issued the last complete-split and the transaction translator responded with a NYET handshake. This means that the start-split was not correctly received by the transaction translator, so it never executed a transaction to the full- or low-speed endpoint, see Section Periodic Isochronous - Do Complete Split for the definition of 'Last'.

Each time the host controller visits a queue head in this state (once within the Execute Transaction state), it performs the following test to determine whether to execute a start-split.

- *QH.S-mask* is bit-wise anded with *cMicroFrameBit*.

If the result is non-zero, then the host controller will issue a start-split transaction. If the *PIDCode* field indicates an IN transaction, the host controller must zero-out the *QH.S-bytes* field. After the split-transaction has been executed, the host controller sets up state in the queue head to track the progress of the complete-split phase of the split transaction. Specifically, it records the expected frame number into *QH.FrameTag* field (see Section), set *C-prog-mask* to zero (00h), and exits this state. Note that the host controller must not adjust the value of *CErr* as a result of completion of a start-split transaction.

Periodic Interrupt - Do Complete Split

This state is entered unconditionally from the Do Start Split state after a start-split transaction is executed on the bus. Each time the host controller visits a queue head in this state (once within the Execute Transaction state), it checks to determine whether a complete-split transaction should be executed now.

There are four tests to determine whether a complete-split transaction should be executed.

- Test A. *cMicroFrameBit* is bit-wise anded with *QH.C-mask* field. A non-zero result indicates that software scheduled a complete-split for this endpoint, during this micro-frame.
- Test B. *QH.FrameTag* is compared with the current contents of *FRINDEX*[7:3]. An equal indicates a match.
- Test C. The complete-split progress bit vector is checked to determine whether the previous bit is set, indicating that the previous complete-split was appropriately executed. An example algorithm for this test is provided below:

```
Algorithm Boolean CheckPreviousBit(QH.C-prog-mask, QH.C-mask, cMicroFrameBit)
Begin
-- Return values:
-- TRUE - no error
-- FALSE - error
--
Boolean rvalue = TRUE;
previousBit = cMicroframeBit logical-rotate-right(1)
-- Bit-wise anding previousBit with C-mask indicates
-- whether there was an intent
-- to send a complete split in the previous micro-frame. So,
-- if the
-- 'previous bit' is set in C-mask, check C-prog-mask to
-- make sure it
-- happened.
If (previousBit bitAND QH.C-mask)then
                                If not(previousBit bitAND QH.C-prog-mask) then
                      rvalue = FALSE;
          End if
End If
-- If the C-prog-mask already has a one in this bit position,
```

```
-- then an aliasing
-- error has occurred. It will probably get caught by the
-- FrameTag Test, but
-- at any rate it is an error condition that as detectable here
-- should not allow
-- a transaction to be executed.
If (cMicroFrameBit bitAND QH.C-prog-mask) then
 rvalue = FALSE;
End if
return (rvalue)
End Algorithm
```

- Test D. Check to see if a start-split should be executed in this micro-frame. Note this is the same test performed in the Do Start Split state (see Section Periodic Isochronous - Do Start Split). Whenever it evaluates to TRUE and the controller is NOT processing in the context of a *Recovery Path* mode, it means a start-split should occur in this micro-frame. Test D and Test A evaluating to TRUE at the same time is a system software error. Behavior is undefined.

If (A .and. B .and. C .and. not(D)) then the host controller will execute a complete-split transaction. When the host controller commits to executing the complete-split transaction, it updates *QH.C-prog-mask* by bit-ORing with *cMicroFrameBit*. On completion of the complete-split transaction, the host controller records the result of the transaction in the queue head and sets *QH.FrameTag* to the expected *H-Frame* number (see Section). The effect to the state of the queue head and thus the state of the transfer depends on the response by the transaction translator to the complete-split transaction. The following responses have the effects (note that any responses that result in decrementing of the *CErr* will result in the queue head being halted by the host controller if the result of the decrement is zero):

- NYET (and Last). On each NYET response, the host controller checks to determine whether this is the last complete-split for this split transaction. Last is defined in this context as the condition where all of the scheduled complete-splits have been executed. If it is the last complete-split (with a NYET response), then the transfer state of the queue head is not advanced (never received any data) and this state exited. The transaction translator must have responded to all the clompete-splits with NYETs, meaning that the start-split issued by the host controller was not received. The start-split should be retried at the next poll period.
- The test for whether this is the Last complete split can be performed by XOR *QH.C-mask* with *QH.C-prog-mask*. If the result is all zeros then all complete-splits have been executed. When this condition occurs, the *XactErr* status bit is set to a one and the *CErr* field is decremented.
- NYET (and not Last). See above description for testing for Last. The complete-split transaction received a NYET response from the transaction translator. Do not update any transfer state (except for *C-prog-mask* and *FrameTag*) and stay in this state. The host controller must not adjust *CErr* on this response.

- Transaction Error (XactErr). Timeout, data CRC failure, etc. The *CErr* field is decremented and the *XactErr* bit in the *Status* field is set to a one. The complete split transaction is *immediately* retried (if *Cerr* is non-zero).If there is not enough time in the micro-frame to complete the retry and the endpoint is an IN, or *CErr* is decremented to a zero from a one, the queue is halted. If there is not enough time in the micro-frame to complete the retry and the endpoint is an OUT and *CErr* is not zero, then this state is exited (that is, return to Do Start Split). This results in a retry of the entire OUT split transaction, at the next poll period. Refer to Chapter 11 Hubs (specifically the section full- and low-speed Interrupts) in the USB Specification Revision 2.0 for detailed requirements on why these errors must be immediately retried.
- ACK. This can only occur if the target endpoint is an OUT. The target endpoint ACK'd the data and this response is a propagation of the endpoint ACK up to the host controller. The host controller must advance the state of the transfer. The *Current Offset* field is incremented by *Maximum Packet Length* or *Bytes to Transfer*, whichever is less. The field *Bytes To Transfer* is decremented by the same amount. And the data toggle bit (*dt*) is toggled. The host controller will then exit this state for this queue head. The host controller must reload *CErr* with maximum value on this response. Advancing the transfer state may cause other process events such as retirement of the qTD and advancement of the queue (see Section Managing Control/Bulk/Interrupt Transfers through Queue Heads).
- MDATA. This response will only occur for an IN endpoint. The transaction translator responded with zero or more bytes of data and an MDATA PID. The incremental number of bytes received is accumulated in *QH.S-bytes*. The host controller must not adjust *CErr* on this response.
- DATA0/1. This response may only occur for an IN endpoint. The number of bytes received is added to the accumulated byte count in *QH.S-bytes*. The state of the transfer is advanced by the result and the host controller will exit this state for this queue head.
- Advancing the transfer state may cause other processing events such as retirement of the qTD and advancement of the queue (see Section Managing Control/Bulk/Interrupt Transfers through Queue Heads).
- If the data sequence PID does not match the expected, the entirety of the data received in this split transaction is ignored, the transfer state is not advanced and this state is exited.
- NAK. The target endpoint Nak'd the full- or low-speed transaction. The state of the transfer is not advanced, and this state is exited. The host controller must reload *CErr* with maximum value on this response.

- ERR. There was an error during the full- or low-speed transaction. The ERR status bit is set to a one, *Cerr* is decremented, the state of the transfer is not advanced, and this state is exited.
- STALL. The queue is halted (an exit condition of the Execute Transaction state). The status field bits: *Active* bit is set to zero and the *Halted* bit is set to a one and the qTD is retired. Responses which are not enumerated in the list or which are received out of sequence are illegal and may result in undefined host controller behavior. The other possible combinations of tests A, B, C, and D may indicate that data or response was lost. The table below lists the possible combinations and the appropriate action.

Table 11-46. Interrupt IN/OUT Do Complete Split State Execution Criteria

Condition	Action	Description
not(A) not(D)	Ignore QHD	Neither a start nor complete-split is scheduled for the current micro-frame.Host controller should continue walking the schedule.
A not(C)	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	Progress bit check failed. These means a complete-split has been missed. There is the possibility of lost data. If *PIDCode* is an IN, then the Queue head must be halted. If *PIDCode* is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect *CER*R. In either case, set the *Missed Micro-frame* bit in the status field to a one.
A not(B) C	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	*QH.FrameTag* test failed. This means that exactly one or more *H-Frames* have been skipped. This means complete-splits and have missed. There is the possibility of lost data. If *PIDCode* is an IN, then the Queue head must be halted. If *PIDCode* is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect *CER*R. In either case, set the *Missed Micro-frame* bit in the status field to a one.
A B C not(D)	Execute complete-split	This is the non-error case where the host controller executes a complete-split transaction.
D	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	This is a degenerate case where the start-split was issued, but all of the complete-splits were skipped and all possible intervening opportunities to detect the missed data failed to fire. If *PIDCode* is an IN, then the Queue head must be halted. If *PIDCode* is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect *CER*R. In either case, set the *Missed Micro-frame* bit in the status field to a one. Note: When executing in the context of a *Recovery Path* mode, the host controller is allowed to process the queue head and take the actions indicated above, or it may wait until the queue head is visited in the

Table 11-46. Interrupt IN/OUT Do Complete Split State Execution Criteria

		normal processing mode. Regardless, the host controller must not execute a start-split in the context of a executing in a *Recovery Path* mode.

Managing QH.FrameTag Field

The *QH.FrameTag* field in a queue head is completely managed by the host controller. The rules for setting *QH.FrameTag* are simple:

- Rule 1: If transitioning from Do Start Split to Do Complete Split and the current value of *FRINDEX*[2:0] is 6 *QH.FrameTag* is set to *FRINDEX*[7:3] + 1. This accommodates split transactions whose start-split and complete-splits are in different *H-Frames* (case 2a, see Figure 11-21).
- Rule 2: If the current value of *FRINDEX*[2:0] is 7, *QH.FrameTag* is set to *FRINDEX*[7:3] + 1. This accommodates staying in Do Complete Split for cases 2a, 2b, and 2c (Figure 11-21).
- Rule 3: If transitioning from Do_Start Split to Do Complete Split and the current value of *FRINDEX*[2:0] is not 6, or currently in Do Complete Split and the current value of (*FRINDEX*[2:0]) is not 7, *FrameTag* is set to *FRINDEX*[7:3]. This accommodates all other cases (Figure 11-21).

11.1.3.3.12.2.6 Rebalancing the periodic schedule

System software must occasionally adjust a periodic queue head's S-mask and C-mask fields during operation.

This need occurs when adjustments to the periodic schedule create a new bandwidth budget and one or more queue head's are assigned new execution footprints (that is, new S-mask and C-mask values).

It is imperative that System software must not update these masks to new values in the midst of a split transaction. In order to avoid any race conditions with the update, the EHCI host controller provides a simple assist to system software. System software sets the *Inactivate-on-next-Transaction* (*I*) bit to a one to signal the host controller that it intends to update the S-mask and C-mask on this queue head. System software will then wait for the host controller to observe the *I-bit* is a one and transition the *Active* bit to a zero. The rules for how and when the host controller sets the *Active* bit to zero are enumerated below:

- If the *Active* bit is a zero, no action is taken. The host controller does not attempt to advance the queue when the *I-bit* is a one.
- If the *Active* bit is a one and the *SplitXState* is DoStart (regardless of the value of *S-mask*), the host controller will simply set *Active* bit to a zero. The host controller is

not required to write the transfer state back to the *current* qTD. Note that if the *S-mask* indicates that a start-split is scheduled for the current micro-frame, the host controller must not issue the start-split bus transaction. It must set the *Active* bit to zero.

System software must save transfer state before setting the *I-bit* to a one. This is required so that it can correctly determine what transfer progress (if any) occurred after the *I-bit* was set to a one and the host controller executed its final bus-transaction and set *Active* to a zero.

After system software has updated the S-mask and C-mask, it must then reactivate the queue head. Because the *Active* bit and the *I-bit* cannot be updated with the same write, system software needs to use the following algorithm to coherently re-activate a queue head that has been stopped via the *I-bit*.

1. Set the *Halted* bit to a one, then
2. Set the *I-bit* to a zero, then
3. Set the *Active* bit to a one and the *Halted* bit to a zero in the same write.

Setting the *Halted* bit to a one inhibits the host controller from attempting to advance the queue between the time the *I-bit* goes to a zero and the *Active* bit goes to a one.

11.1.3.3.12.3 Split Transaction Isochronous

Full-speed isochronous transfers are managed using the split-transaction protocol through a USB 2.0 transaction translator in a USB2.0 Hub. The EHCI controller utilizes siTD data structure to support the special requirements of isochronous split-transactions.

This data structure uses the scheduling model of isochronous TDs (iTD, Section Isochronous (High-Speed) Transfer Descriptor (iTD)) (see Section Managing Isochronous Transfers Using iTDs for the operational model of iTDs) with the contiguous data feature provided by queue heads. This simple arrangement allows a single isochronous scheduling model and adds the additional feature that all data received from the endpoint (per split transaction) must land into a contiguous buffer.

11.1.3.3.12.3.1 Split Transaction Scheduling Mechanisms for Isochronous

Full-speed isochronous transactions are managed through a transaction translator's periodic pipeline. As with full- and low-speed interrupt, system software manages each transaction translator's periodic pipeline by budgeting and scheduling exactly during which micro-frames the start-splits and complete-splits for each full-speed isochronous endpoint occur.

The requirements described in Section Split Transaction Scheduling Mechanisms for Interrupt apply. The following figure illustrates the general scheduling boundary conditions that are supported by the EHCI periodic schedule. The ^{S}X and ^{C}X labels indicate micro-frames where software can schedule start- and complete-splits (respectively). The *H-Frame* boundaries are marked with a large, solid bold vertical line. The *B-Frame* boundaries are marked with a large, bold, dashed line. The bottom of the figure illustrates the relationship of an siTD to the *H-Frame*.

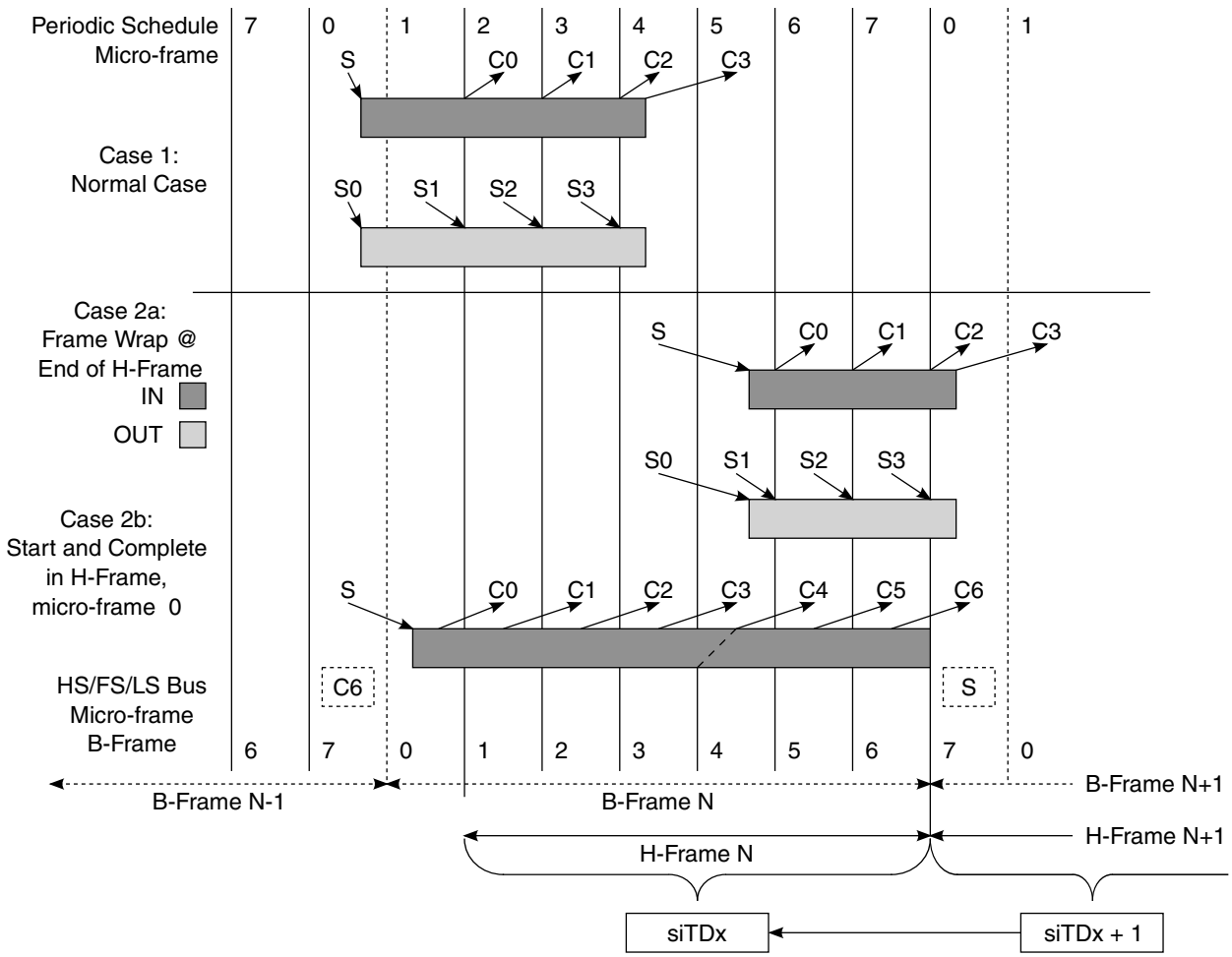


Figure 11-25. Split Transaction, Isochronous Scheduling Boundary Conditions

When the endpoint is an isochronous OUT, there are only start-splits, and no complete-splits. When the endpoint is an isochronous IN, there is at most one start-split and one to *N* complete-splits. The scheduling boundary cases are:

- *Case 1*: The entire split transaction is completely bounded by an *H-Frame*. For example: the start-splits and complete-splits are all scheduled to occur in the same *H-Frame*.

- *Case 2a*: This boundary case is where one or more (at most two) complete-splits of a split transaction IN are scheduled across an *H-Frame* boundary. This can only occur when the split transaction has the possibility of moving data in *B-Frame,* micro-frames 6 or 7 (*H-Frame* micro-frame 7 or 0). When an *H-Frame* boundary wrap condition occurs, the scheduling of the split transaction spans more than one location in the periodic list.(For example, it takes two siTDs in adjacent periodic frame list locations to fully describe the scheduling for the split transaction.)
- Although the scheduling of the split transaction may take two data structures, all of the complete-splits for each full-speed IN isochronous transaction must use only one data pointer. For this reason, siTDs contain a back pointer, the use of which is described below.
- Software must never schedule full-speed isochronous OUTs across an *H-Frame* boundary.
- *Case 2b*: This case can only occur for a very large isochronous IN. It is the only allowed scenario where a start-split and complete-split for the same endpoint can occur in the same micro-frame. Software must enforce this rule by scheduling the large transaction first. Large is defined to be anything larger than 579 byte maximum packet size.

A subset of the same mechanisms employed by full- and low-speed interrupt queue heads are employed in siTDs to schedule and track the portions of isochronous split transactions. The following fields are initialized by system software to instruct the host controller when to execute portions of the split transaction protocol.

- *SplitXState*. This is a single bit residing in the *Status* field of an siTD (see Figure 11-26). This bit is used to track the current state of the split transaction. The rules for managing this bit are described in Section Split Transaction Execution State Machine for Interrupt.
- *Frame S-mask*. This is a bit-field where-in system software sets a bit corresponding to the micro-frame (within an *H-Frame*) that the host controller should execute a start-split transaction. This is always qualified by the value of the *SplitXState* bit. For example, referring to the IN example in Figure 11-25, case one, the *S-mask* would have a value of 00000001b indicating that if the siTD is traversed by the host controller, and the *SplitXState* indicates Do Start Split, and the current micro-frame as indicated by USB_n_FRINDEX[2:0] is 0, then execute a start-split transaction.
- *Frame C-mask.* This is a bit-field where system software sets one or more bits corresponding to the micro-frames (within an *H-Frame*) that the host controller should execute complete-split transactions. The interpretation of this field is always qualified by the value of the *SplitXState* bit. For example, referring to the IN example in Figure 11-25, case one, the *C-mask* would have a value of 00111100b indicating that if the siTD is traversed by the host controller, and the *SplitXState* indicates Do

Complete Split, and the current micro-frame as indicated by *USB_n_FRINDEX*[2:0] is 2, 3, 4, or 5, then execute a complete-split transaction.

- *Back Pointer*. This field in a siTD is used to complete an IN split-transaction using the previous *H-Frame*'s siTD. This is only used when the scheduling of the complete-splits span an *H-Frame* boundary.

There exists a one-to-one relationship between a high-speed isochronous split transaction (including all start- and complete-splits) and one full-speed isochronous transaction. An siTD contains (amongst other things) buffer state and split transaction scheduling information. An siTD's buffer state always maps to one full-speed isochronous data payload. This means that for any full-speed transaction payload, a single siTD's data buffer must be used. This rule applies to both IN an OUTs. An siTD's scheduling information usually also maps to one high-speed isochronous split transaction. The exception to this rule is the *H-Frame* boundary wrap cases mentioned above.

The siTD data structure describes at most, one frame's worth of high-speed transactions and that description is strictly bounded within a frame boundary. The figure below illustrates some examples. On the top are examples of the full-speed transaction footprints for the boundary scheduling cases described above. In the middle are time-frame references for both the *B-Frames* (HS/FS/LS Bus) and the *H-Frames*. On the bottom is illustrated the relationship between the scope of an siTD description and the time references. Each *H-Frame* corresponds to a single location in the periodic frame list. The implication is that each siTD is reachable from a single periodic frame list location at a time.

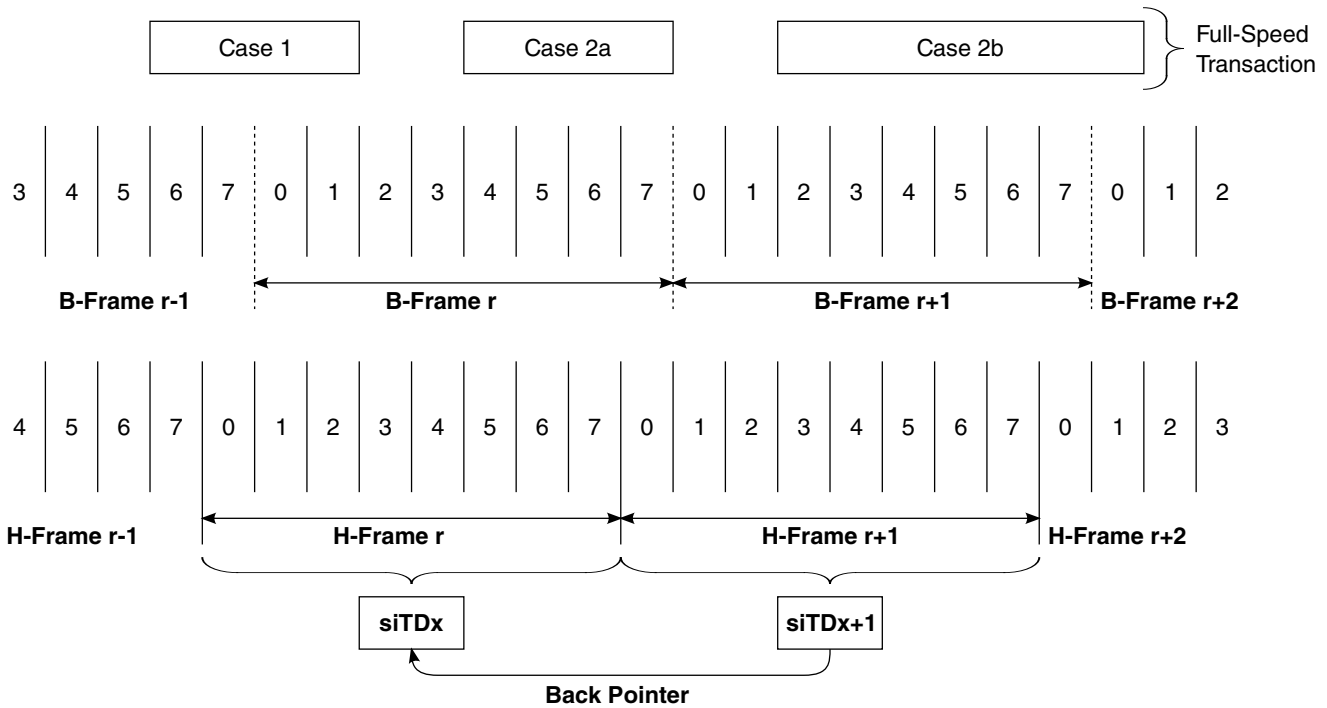


Figure 11-26. siTD Scheduling Boundary Examples

Each case is described below:

- *Case 1*: One siTD is sufficient to describe and complete the isochronous split transaction because the whole isochronous split transaction is tightly contained within a single *H-Frame*.
- *Case 2a, 2b*: Although both INs and OUTs can have these footprints, OUTs always take only one siTD to schedule. However, INs (for these boundary cases) require two siTDs to complete the scheduling of the isochronous split transaction $siTD_X$ is used to always issue the start-split and the first *N* complete-splits. The full-speed transaction (for these cases) can deliver data on the full-speed bus segment during micro-frame 7 of $H\text{-}Frame_{Y+1}$, or micro-frame 0 of $H\text{-}Frame_{Y+2}$. The complete splits are scheduled using $siTD_{X+2}$ (not shown). The complete-splits to extract this data must use the buffer pointer from $siTD_{X+1}$. The only way for the host controller to reach $siTD_{X+1}$ from $H\text{-}Frame_{Y+2}$is to use $siTD_{X+2}$'s back pointer. The host controller rules for when to use the back pointer are described is Section Periodic Isochronous - Do Complete Split .

Software must apply the following rules when calculating the schedule and linking the schedule data structures into the periodic schedule:

- Software must ensure that an isochronous split-transaction is started so that it will complete before the end of the *B-Frame*.
- Software must ensure that for a single full-speed isochronous endpoint, there is never a start-split and complete-split in *H-Frame, micro-frame 1*. This is mandated as a rule so that case 2a and case 2b can be discriminated. According to the core USB specification, the long isochronous transaction illustrated in Case 2b, could be scheduled so that the start-split was in micro-frame 1 of *H-Frame* N and the last complete-split would need to occur in micro-frame 1 of H-Frame N+1. However, it is impossible to discriminate between cases 2a and case 2b, which has significant impact on the complexity of the host controller.

11.1.3.3.12.3.2 Tracking Split Transaction Progress for Isochronous Transfers

To correctly maintain the data stream, the host controller must be able to detect and report errors where device to host data is lost. Isochronous endpoints do not employ the concept of a halt on error, however the host is required to identify and report per-packet errors observed in the data stream. This includes schedule traversal problems (skipped micro-frames), timeouts and corrupted data received.

In similar kind to interrupt split-transactions, the portions of the split transaction protocol must execute in the micro-frames they are scheduled. The queue head data structure used to manage full- and low-speed interrupt has several mechanisms for tracking when portions of a transaction have occurred. Isochronous transfers use siTDs, for their transfers, and the data structures are only reachable via the schedule in the exact micro-frame in which they are required (so all the mechanism employed for tracking in queue heads is not required for siTDs). Software has the option of reusing siTD several times in the complete periodic schedule. However, it must ensure that the results of split transaction N are consumed and the siTD reinitialized (activated) before the host controller gets back to the siTD (in a future micro-frame).

Split-transaction isochronous OUTs utilize a low-level protocol to indicate which portions of the split transaction data have arrived. Control over the low-level protocol is exposed in an siTD via the fields *Transaction Position* (*TP*) and *Transaction Count* (*T-count*). If the entire data payload for the OUT split transaction is larger than 188 bytes, there will be more than one start-split transaction, each of which require proper annotation. If host hold-offs occur, then the sequence of annotations received from the host will not be complete, which is detected and handled by the transaction translator. See Section Periodic Isochronous - Do Start Split for a description on how these fields are used during a sequence of start-split transactions.

The fields *siTD.T-Count* and *siTD.TP* are used by the host controller to drive and sequence the transaction position annotations. It is the responsibility of system software to properly initialize these fields in each siTD. Once the budget for a split-transaction

isochronous endpoint is established, *S-mask*, *T-Count*, and *TP* initialization values for all the siTD associated with the endpoint are constant. They remain constant until the budget for the endpoint is recalculated by software and the periodic schedule adjusted.

For IN-endpoints, the transaction translator simply annotates the response data packets with enough information to allow the host controller to identify the last data. As with split transaction Interrupt, it is the host controller's responsibility to detect when it has missed an opportunity to execute a complete-split. The following field in the siTD is used to track and detect errors in the execution of a split transaction for an IN isochronous endpoint.

- *C-prog-mask*. This is an eight-bit bit-vector where the host controller keeps track of which complete-splits have been executed. Due to the nature of the Transaction Translator periodic pipeline, the complete-splits need to be executed in-order. The host controller needs to detect when the complete-splits have not been executed in order. This can only occur due to system hold-offs where the host controller cannot get to the memory-based schedule. *C-prog-mask* is a simple bit-vector that the host controller sets a bit for each complete-split executed. The bit position is determined by the micro-frame (USB_n_FRINDEX[2:0]) number in which the complete-split was executed. The host controller always checks *C-prog-mask* before executing a complete-split transaction. If the previous complete-splits have not been executed, then it means one (or more) have been skipped and data has potentially been lost. System software is required to initialize this field to zero before setting an siTD's *Active* bit to a one.

If a transaction translator returns with the final data before all of the complete-splits have been executed, the state of the transfer is advanced so that the remaining complete-splits are not executed. Refer to Section Asynchronous - Do Complete Split for a description on how the state of the transfer is advanced. It is important to note that an IN siTD is retired based solely on the responses from the Transaction Translator to the complete-split transactions. This means, for example, that it is possible for a transaction translator to respond to a complete-split with an MDATA PID. The number of bytes in the MDATA's data payload could cause the siTD field *Total Bytes to Transfer* to decrement to zero. This response can occur, before all of the scheduled complete-splits have been executed. In other interface, data structures (for example, high-speed data streams through queue heads), the transition of *Total Bytes to Transfer* to zero signals the end of the transfer and results in setting of the *Active* bit to zero. However, in this case, the result has not been delivered by the Transaction Translator and the host must continue with the next complete-split transaction to extract the residual transaction state. This scenario occurs because of the pipeline rules for a Transaction Translator (see Chapter 11 of the Universal Serial Bus Revision 2.0). In summary the periodic pipeline rules require that on a micro-frame boundary, the Transaction Translator will hold the final two bytes received (if it has not seen an End Of Packet (EOP)) in the full-speed bus pipe stage and give the

remaining bytes to the high-speed pipeline stage. At the micro-frame boundary, the Transaction Translator could have received the entire packet (including both CRC bytes) but not received the packet EOP. In the next micro-frame, the Transaction Translator will respond with an MDATA and send all of the data bytes (with the two CRC bytes being held in the full-speed pipeline stage). This could cause the siTD to decrement its *Total Bytes to Transfer* field to zero, indicating it has received all expected data. The host must still execute one more (scheduled) complete-split transaction in order to extract the results of the full-speed transaction from the Transaction Translator (for example, the Transaction Translator may have detected a CRC failure, and this result must be forwarded to the host).

If the host experiences hold-offs that cause the host controller to skip one or more (but not all) scheduled split transactions for an isochronous OUT, then the protocol to the transaction translator will not be consistent and the transaction translator will detect and react to the problem. Likewise, for host hold-offs that cause the host controller to skip one or more (but not all) scheduled split transactions for an isochronous IN, the *C-prog-mask* is used by the host controller to detect errors. However, if the host experiences a hold-off that causes it to skip all of an siTD, or an siTD expires during a host hold off (for example, a hold-off occurs and the siTD is no longer reachable by the host controller in order for it to report the hold-off event), then system software must detect that the siTDs have not been processed by the host controller (that is, state not advanced) and report the appropriate error to the client driver.

11.1.3.3.12.3.3 Split Transaction Execution State Machine for Isochronous

In the following presentation, all references to micro-frame are in the context of a micro-frame within an *H-Frame.*

If the *Active* bit in the *Status* byte is a zero, the host controller will ignore the siTD and continue traversing the periodic schedule. Otherwise the host controller will process the siTD as specified below. A split transaction state machine is used to manage the split-transaction protocol sequence. The host controller uses the fields defined in Section Tracking Split Transaction Progress for Interrupt Transfers , plus the variable *cMicroFrameBit* defined in Section Split Transaction Execution State Machine for Interrupt to track the progress of an isochronous split transaction. The figure below illustrates the state machine for managing an siTD through an isochronous split transaction. Bold, dotted circles denote the state of the *Active* bit in the *Status* field of a siTD. The Bold, dotted arcs denote the transitions between these states. Solid circles denote the states of the split transaction state machine and the solid arcs denote the transitions between these states. Dotted arcs and boxes reference actions that take place either as a result of a transition or from being in a state.

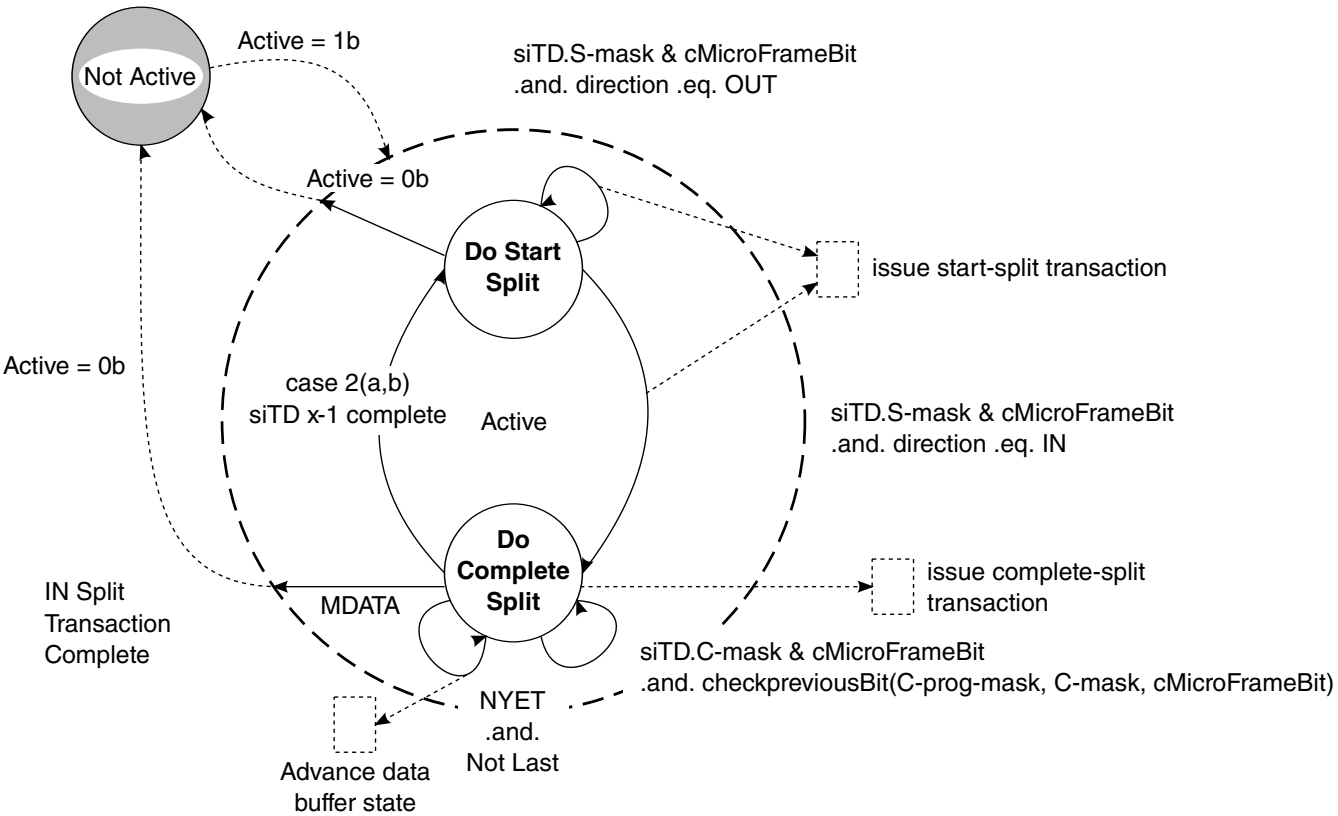


Figure 11-27. Split Transaction State Machine for Isochronous

11.1.3.3.12.3.4 Periodic Isochronous - Do Start Split

Isochronous split transaction OUTs use only this state.

An siTD for a split-transaction isochronous IN is either initialized to this state, or the siTD transitions to this state from Do Complete Split when a case 2a (IN) or 2b scheduling boundary isochronous split-transaction completes.

Each time the host controller reaches an active siTD in this state, it checks the *siTD.S-mask* against *cMicroFrameBit*. If there is a one in the appropriate position, the siTD will execute a start-split transaction. By definition, the host controller cannot *reach* an siTD at the wrong time. If the *I/O* field indicates an IN, then the start-split transaction includes only the extended token plus the full-speed token. Software must initialize the *siTD.Total Bytes To Transfer* field to the number of bytes expected. This is usually the maximum packet size for the full-speed endpoint. The host controller exits this state when the start-split transaction is complete.

The remainder of this section is specific to an isochronous OUT endpoint (that is, the *I/O* field indicates an OUT). When the host controller executes a start-split transaction for an isochronous OUT it includes a data payload in the start-split transaction. The memory

buffer address for the data payload is constructed by concatenating *siTD.Current Offset* with the page pointer indicated by the page selector field (*siTD.P*). A zero in this field selects Page 0 and a 1 selects Page 1. During the start-split for an OUT, if the data transfer crosses a page boundary during the transaction, the host controller must detect the page cross, update the *siTD.P*-bit from a zero to a one, and begin using the *siTD.Page 1* with *siTD.Current Offset* as the memory address pointer. The field *siTD.TP* is used to annotate each start-split transaction with the indication of which part of the split-transaction data the current payload represents (ALL, BEGIN, MID, END). In all cases the host controller simply uses the value in *siTD.TP* to mark the start-split with the correct transaction position code.

T-Count is always initialized to the number of start-splits for the current frame. *TP* is always initialized to the first required transaction position identifier. The scheduling boundary case (see Figure 11-26) is used to determine the initial value of *TP*. The initial cases are summarized in the following table.

Table 11-47. Initial Conditions for OUT siTD's TP and T-count Fields

Case	T-count	TP	Description
1, 2a	=1	ALL	When the OUT data payload is less than (or equal to) 188 bytes, only one start-split is required to move the data. The one start-split must be marked with an ALL.
1, 2a	!=1	BEGIN	When the OUT data payload is greater than 188 bytes more than one start-split must be used to move the data. The initial start-split must be marked with a BEGIN.

After each start-split transaction is complete, the host controller updates *T-Count* and *TP* appropriately so that the next start-split is correctly annotated.

The table below illustrates all of the *TP* and *T-count* transitions, which must be accomplished by the host controller.

.

Table 11-48. Transaction Position (TP)/Transaction Count (T-Count) Transition Table

TP	T-count next	TP next	Description
ALL	0	N/A	Transition from ALL, to done.
BEGIN	1	END	Transition from BEGIN to END. Occurs when *T-count* starts at 2.
BEGIN	!=1	MID	Transition from BEGIN to MID. Occurs when *T-count* starts at greater than 2.
MID	!=1	MID	*TP* stays at MID while *T-count* is not equal to 1 (that is, greater than 1). This case can occur for any of the scheduling boundary cases where the *T-count* starts greater than 3.
MID	1	END	Transition from MID to END. This case can occur for any of the scheduling boundary cases where the *T-count* starts greater than 2.

The start-split transactions do not receive a handshake from the transaction translator, so the host controller always advances the transfer state in the siTD after the bus transaction is complete. To advance the transfer state the following operations take place:

- The *siTD.Total Bytes To Transfer* and the *siTD.Current Offset* fields are adjusted to reflect the number of bytes transferred.
- The *siTD.P* (page selector) bit is updated appropriately.
- The *siTD.TP* and *siTD.T-count* fields are updated appropriately as defined in Table 11-48.

These fields are then written back to the memory based siTD. The *S-mask* is fixed for the life of the current budget. As mentioned above, *TP* and *T-count* are set specifically in each siTD to reflect the data to be sent from this siTD. Therefore, regardless of the value of *S-mask*, the actual number of start-split transactions depends on *T-count* (or equivalently, *Total Bytes to Transfer*). The host controller must set the *Active* bit to a zero when it detects that all of the schedule data has been sent to the bus. The preferred method is to detect when *T-Count* decrements to zero as a result of a start-split bus transaction. Equivalently, the host controller can detect when *Total Bytes to Transfer* decrements to zero. Either implementation must ensure that if the initial condition is *Total Bytes to Transfer* equal to zero and *T-count* is equal to a one, then the host controller will issue a single start-split, with a zero-length data payload. Software must ensure that *TP*, *T-count* and *Total Bytes to Transfer* are set to deliver the appropriate number of bus transactions from each siTD. An inconsistent combination will yield undefined behavior.

If the host experiences hold-offs that cause the host controller to skip start-split transactions for an OUT transfer, the state of the transfer will not progress appropriately. The transaction translator will observe protocol violations in the arrival of the start-splits for the OUT endpoint (that is, the transaction position annotation will be incorrect as received by the transaction translator).

Example scenarios are described in Section Split Transaction for Isochronous - Processing Examples.

A host controller implementation can optionally track the progress of an OUT split transaction by setting appropriate bits in the *siTD.C-prog-mask* as it executes each scheduled start-split. The *checkPreviousBit*() algorithm defined in Periodic Isochronous - Do Complete Split can be used prior to executing each start-split to determine whether start-splits were skipped. The host controller can use this mechanism to detect missed micro-frames. It can then set the siTD's *Active* bit to zero and stop execution of this siTD. This saves on both memory and high-speed bus bandwidth.

11.1.3.3.12.3.5 Periodic Isochronous - Do Complete Split

This state is only used by a split-transaction isochronous IN endpoint.

This state is entered unconditionally from the Do Start State after a start-split transaction is executed for an IN endpoint. Each time the host controller visits an siTD in this state, it conducts a number of tests to determine whether it should execute a complete-split transaction. The individual tests are listed below. The sequence they are applied depends on which micro-frame the host controller is currently executing which means that the tests might not be applied until after the siTD referenced from the back pointer has been fetched.

- Test A. *cMicroFrameBit* is bit-wise anded with *siTD.C-mask* field. A non-zero result indicates that software scheduled a complete-split for this endpoint, during this micro-frame. This test is always applied to a newly fetched siTD that is in this state.
- Test B. The *siTD.C-prog-mask* bit vector is checked to determine whether the previous complete splits have been executed. An example algorithm is below (this is slightly different than the algorithm used in Section Periodic Isochronous - Do Complete Split). The sequence in which this test is applied depends on the current value of USB_n_FRINDEX[2:0]. If USB_n_FRINDEX[2:0] is 0 or 1, it is not applied until the back pointer has been used. Otherwise it is applied immediately.

```
Algorithm Boolean CheckPreviousBit(siTD.C-prog-mask, siTD.C-mask, cMicroFrameBit)
Begin
          Boolean rvalue = TRUE;
          previousBit = cMicroFrameBit rotate-right(1)
          -- Bit-wise anding previousBit with C-mask indicates whether there was an intent
          -- to send a complete split in the previous micro-frame. So, if the
          -- 'previous bit' is set in C-mask, check C-prog-mask to make sure it
          -- happened.
          if previousBit bitAND siTD.C-mask then
                    if not (previousBit bitAND siTD.C-prog-mask) then
                              rvalue = FALSE
                    End if
          End if
          Return rvalue
End Algorithm
```

If Test A is true and USB_n_FRINDEX[2:0] is zero or one, then this is a case 2a or 2b scheduling boundary (see Figure 11-25). See Section Periodic Isochronous - Do Complete Split for details in handling this condition.

If Test A and Test B evaluate to true, then the host controller will execute a complete-split transaction using the transfer state of the current siTD. When the host controller commits to executing the complete-split transaction, it updates *QH.C-prog-mask* by bit-ORing with *cMicroFrameBit*. The transfer state is advanced based on the completion status of the complete-split transaction. To advance the transfer state of an IN siTD, the host controller must:

- Decrement the number of bytes received from *siTD.Total Bytes To Transfer,*
- Adjust *siTD.Current Offset* by the number of bytes received,

- Adjust *siTD.P* (page selector) field if the transfer caused the host controller to use the next page pointer, and
- Set any appropriate bits in the *siTD.Status* field, depending on the results of the transaction.

Note that if the host controller encounters a condition where *siTD.Total Bytes To Transfer* is zero, and it receives more data, the host controller must not write the additional data to memory. The *siTD.Status.Active* bit must be set to zero and the *siTD.Status.Babble Detected* bit must be set to a one. The fields *siTD.Total Bytes To Transfer*, *siTD.Current Offset*, and *siTD.P* (page selector) are not required to be updated as a result of this transaction attempt.

The host controller must accept (assuming good data packet CRC and sufficient room in the buffer as indicated by the value of *siTD.Total Bytes To Transfer*) MDATA and DATA0/1 data payloads up to and including 192 bytes. A host controller implementation may optionally set *siTD.Status Active* to a zero and *siTD.Status.Babble Detected* to a one when it receives and MDATA or DATA0/1 with a data payload of more than 192 bytes. The following responses have the noted effects:

- ERR. The full-speed transaction completed with a time-out or bad CRC and this is a reflection of that error to the host. The host controller sets the *ERR* bit in the *siTD.Status* field and sets the *Active* bit to a zero.
- Transaction Error (XactErr). The complete-split transaction encounters a Timeout, CRC16 failure, etc. The *siTD.Status* field *XactErr* field is set to a one and the complete-split transaction must be retried immediately. The host controller must use an internal error counter to count the number of retries as a counter field is not provided in the siTD data structure. The host controller will not retry more than two times. If the host controller exhausts the retries or the end of the micro-frame occurs, the *Active* bit is set to zero.
- DATAx (0 or 1). This response signals that the final data for the split transaction has arrived. The transfer state of the siTD is advanced and the *Active* bit is set to a zero. If the *Bytes To Transfer* field has not decremented to zero (including the reception of the data payload in the DATAx response), then less data than was expected, or allowed for was actually received. This *short packet* event does not set the USBINT status bit in the USBSTS register to a one. The host controller will not detect this condition.
- NYET (and Last). On each NYET response, the host controller also checks to determine whether this is the last complete-split for this split transaction. Last was defined in Section Periodic Isochronous - Do Complete Split . If it is the last complete-split (with a NYET response), then the transfer state of the siTD is not advanced (never received any data) and the *Active* bit is set to a zero. No bits are set in the *Status* field because this is essentially a skipped transaction. The transaction translator must have responded to all the scheduled clompete-splits with NYETs,

meaning that the start-split issued by the host controller was not received. This result should be interpreted by system software as if the transaction was completely skipped. The test for whether this is the last complete split can be performed by XORing C-mask with C-prog-mask. A zero result indicates that all complete-splits have been executed.

- MDATA (and Last). See above description for testing for Last. This can only occur when there is an error condition. Either there has been a babble condition on the full-speed link, which delayed the completion of the full-speed transaction, or software set up the *S-mask* and/or *C-mask*s incorrectly. The host controller must set *XactErr* bit to a one and the *Active* bit is set to a zero.
- NYET (and not Last). See above description for testing for Last. The complete-split transaction received a NYET response from the transaction translator. Do not update any transfer state (except for *C-prog-mask*) and stay in this state.
- MDATA (and not Last). The transaction translator responds with an MDATA when it has partial data for the split transaction. For example, the full-speed transaction data payload spans from micro-frame X to X+1 and during micro-frame X, the transaction translator will respond with an MDATA and the data accumulated up to the end of micro-frame X. The host controller advances the transfer state to reflect the number of bytes received.

If Test A succeeds, but Test B fails, it means that one or more of the complete-splits have been skipped. The host controller sets the *Missed Micro-Frame* status bit and sets the *Active* bit to a zero.

11.1.3.3.12.3.6 Complete-Split for Scheduling Boundary Cases 2a, 2b

Boundary cases 2a and 2b (INs only) (see Figure 11-25) require that the host controller use the transaction state context of the previous siTD to finish the split transaction. The table below enumerates the transaction state fields.

Table 11-49. Summary siTD Split Transaction State

Buffer State	Status	Execution Progress
Total Bytes To Transfer P (page select) Current Offset TP (transaction position) T-count (transaction count)	All bits in the status field	C-prog-mask

NOTE

TP and *T-count* are used only for Host to Device (OUT) endpoints.

If software has budgeted the schedule of this data stream with a frame wrap case, then it must initialize the *siTD.Back Pointer* field to reference a valid siTD and will have the *siTD.Back Pointer.T-bit* in the *siTD.Back Pointer*

field set to a zero. Otherwise, software must set the *siTD.Back Pointer.T-bit* in the *siTD.Back Pointer* field to a one. The host controller's rules for interpreting when to use the *siTD.Back Pointer* field are listed below. These rules apply only when the siTD's *Active* bit is a one and the *SplitXState* is Do Complete Split.

- When cMicroFrameBit is a 1h and the siTDX.Back Pointer.T-bit is a zero, or
- If cMicroFrameBit is a 2h and siTDX.S-mask[0] is a zero

When either of these conditions apply, then the host controller must use the transaction state from $siTD_{X-1}$.

In order to access $siTD_{X-1}$, the host controller reads on-chip the siTD referenced from $siTD_X$.*Back Pointer*.

The host controller must save the entire state from $siTD_X$ while processing $siTD_{X-1}$. This is to accommodate for case 2b processing. The host controller must not recursively walk the list of *siTD.Back Pointers*.

If $siTD_{X-1}$ is active (*Active* bit is a one and *SplitXStat* is Do Complete Split), then both Test A and Test B are applied as described above. If these criteria to execute a complete-split are met, the host controller executes the complete split and evaluates the results as described above. The transaction state (see Table 11-49) of $siTD_{X-1}$ is appropriately advanced based on the results and written back to memory. If the resultant state of $siTD_{X-1}$'s *Active* bit is a one, then the host controller returns to the context of $siTD_X$, and follows its next pointer to the next schedule item. No updates to $siTD_X$ are necessary.

If $siTD_{X-1}$ is active (*Active* bit is a one and *SplitXStat* is Do Start Split), then the host controller must set *Active* bit to a zero and *Missed Micro-Frame* status bit to a one and the resultant status written back to memory.

If $siTD_{X-1}$*'sActive* bit is a zero, (because it was zero when the host controller first visited $siTD_{X-1}$ via $siTD_X$*'s* back pointer, it transitioned to zero as a result of a detected error, or the results of $siTD_{X-1}$*'s* complete-split transaction transitioned it to zero), then the host controller returns to the context of $siTD_X$ and transitions its *SplitXState* to Do Start Split. The host controller then determines whether the case 2b start split boundary condition exists (that is, if *cMicroframeBit* is a 1b and $siTD_X$.*S-mask*[0] is a 1b). If this criterion is met the host controller immediately executes a start-split transaction and appropriately advances the transaction state of $siTD_X$, then follows $siTD_X$.*Next Pointer* to the next schedule item. If the criterion is not met, the host controller simply follows $siTD_X$.*Next Pointer* to the next schedule item. Note that in the case of a 2b boundary case, the split-transaction of $siTD_{X-1}$ will have its *Active* bit set to zero when the host controller returns

to the context of $siTD_X$. Also, note that software should not initialize an siTD with *C-mask* bits 0 and 1 set to a one and an *S-mask* with bit zero set to a one. This scheduling combination is not supported and the behavior of the host controller is undefined.

11.1.3.3.12.3.7 Split Transaction for Isochronous - Processing Examples

There is an important difference between how the hardware/software manages the isochronous split transaction state machine and how it manages the asynchronous and interrupt split transaction state machines.

The asynchronous and interrupt split transaction state machines are encapsulated within a single queue head. The progress of the data stream depends on the progress of each split transaction. In some respects, the split-transaction state machine is sequenced via the Execute Transaction queue head traversal state machine (see Figure 11-18).

Isochronous is a pure time-oriented transaction/data stream. The interface data structures are optimized to efficiently describe transactions that need to occur at specific times. The isochronous split-transaction state machine must be managed across these time-oriented data structures. This means that system software must correctly describe the scheduling of split-transactions across more than one data structure.

Then the host controller must make the appropriate state transitions at the appropriate times, in the correct data structures.

For example, the table below illustrates a couple of frames worth of scheduling required to schedule a case 2a full-speed isochronous data stream.

Table 11-50. Example Case 2a - Software Scheduling siTDs for an IN Endpoint

siTDX		Micro-Frames								Initial
#	Masks	0	1	2	3	4	5	6	7	SplitXState
X	S-Mask	-	-	-	-	1	-	-	-	Do Start Split
	C-Mask	1	1	-	-	-	-	1	1	
X+1	S-Mask	-	-	-	-	1	-	-	-	Do Complete Split
	C-Mask	1	1					1	1	
X+2	S-Mask	-	-	-	-	1	-	-	-	Do Complete Split
	C-Mask	1	1					1	1	
X+3	S-Mask	Repeats previous pattern								Do Complete Split
	C-Mask									

This example shows the first three siTDs for the transaction stream. Because this is the case-2a frame-wrap case, *S-masks* of all siTDs for this endpoint have a value of 10h (a one bit in micro-frame 4) and *C-mask* value of C3h (one-bits in micro-frames 0,1, 6 and 7). Additionally, sofware ensures that the *Back Pointer* field of each siTD references the appropriate siTD data structure (and the *Back PointerT-bits* are set to zero).

The initial *SplitXState* of the first siTD is Do Start Split. The host controller will visit the first siTD eight times during frame X. The C-mask bits in micro-frames 0 and 1 are ignored because the state is Do Start Split. During micro-frame 4, the host controller determines that it can run a start-split (and does) and changes *SplitXState* to Do Complete Split. During micro-frames 6 and 7, the host controller executes complete-splits. Notice the siTD for frame X+1 has its *SplitXState* initialized to Do Complete Split. As the host controller continues to traverse the schedule during *H-Frame* X+1, it will visit the second siTD eight times. During micro-frames 0 and 1 it will detect that it must execute complete-splits.

During *H-Frame* X+1, micro-frame 0, the host controller detects that $siTD_{X+1}$'s *Back Pointer.T-bit* is a zero, saves the state of $siTD_{X+1}$ and fetches $siTD_X$. It executes the complete split transaction using the transaction state of $siTD_X$. If the $siTD_X$ split transaction is complete, siTD's *Active* bit is set to zero and results written back to $siTD_X$. The host controller retains the fact that $siTD_X$ is retired and transitions the *SplitXState* in the $siTD_{X+1}$ to Do Start Split. At this point, the host controller is prepared to execute the start-split for $siTD_{X+1}$ when it reaches micro-frame 4. If the split-transaction completes early (transaction-complete is defined in Section Periodic Isochronous - Do Complete Split), that is, before all the scheduled complete-splits have been executed, the host controller will transition $siTD_X$.*SplitXState* to Do Start Split early and naturally skip the remaining scheduled complete-split transactions. For this example, $siTD_{X+1}$ does not receive a DATA0 response until *H-Frame* X+2, micro-frame 1.

During *H-Frame X+2,* micro-frame 0, the host controller detects that $siTD_{X+2}$'s *Back Pointer.T-bit* is a zero, saves the state of $siTD_{X+2}$ and fetches $siTD_{X+1}$. As described above, it executes another split transaction, receives an MDATA response, updates the transfer state, but does not modify the *Active* bit. The host controller returns to the context of $siTD_{X+2}$, and traverses its next pointer without any state change updates to $siTD_{X+2}$. S

During *H-Frame X+2*, micro-frame 1, the host controller detects $siTD_{X+2}$'s *S-mask[0]* is a zero, saves the state of $siTD_{X+2}$ and fetches $siTD_{X+1}$. It executes another complete-split transaction, receives a DATA0 response, updates the transfer state and sets the *Active* bit to a zero. It returns to the state of $siTD_{X+2}$ and changes its *SplitXState* to Do Start Split. At this point, the host controller is prepared to execute start-splits for $siTD_{X+2}$ when it reaches micro-frame 4.

11.1.3.3.13 Host Controller Pause

When the host controller's *HCHalted* bit in the USBSTS register is a zero, the host controller is sending SOF (Start OF Frame) packets down all enabled ports.

When the schedules are enabled, the EHCI host controller will access the schedules in main memory each micro-frame. This constant pinging of main memory is known to create Arm platform power management problems for mobile systems. Specifically, mobile systems aggressively manage the state of the Arm platform, based on recent history usage. In the more aggressive power saving modes, the Arm platform can disable its caches. Current PC architectures assume that bus-master accesses to main memory must be cache-coherent. So, when bus masters are busy touching memory, the Arm platform power management software can detect this activity over time and inhibit the transition of the Arm platform into its lowest power savings mode. USB controllers are bus-masters and the frequency at which they access their memory-based schedules keeps the Arm platform power management software from placing the Arm platform into its lowest power savings state.

USB Host controllers don't access main memory when they are suspended. However, there are a variety of reasons why placing the USB controllers into suspend won't work, but they are beyond the scope of this document. The base requirement is that the USB controller needs to be kept out of main memory, while at the same time, the USB bus is kept from going into suspend.

EHCI controllers provide a large-grained mechanism that can be manipulated by system software to change the memory access pattern of the host controller. System software can manipulate the schedule enable bits in the USBCMD register to turn on/off the scheduling traversal. A software heuristic can be applied to implement an on/off duty cycle that allows the USB to make reasonable progress and allow the Arm platform power management to get the Arm platform into its lowest power state. This method is not intended to be applied at all times to throttle USB, but should only be applied in very specific configurations and usage loads. For example, when only a keyboard or mouse is attached to the USB, the heuristic could detect times when the USB is attempting to move data only very infrequently and can adjust the duty cycle to allow the Arm platform to reach its low power state for longer periods of time. Similarly, it could detect increases in the USB load and adjust the duty cycle appropriately, even to the point where the schedules are never disabled. The assumption here is that the USB is moving data and the Arm platform will be required to process the data streams.

It is suggested that in order to provide a complete solution for the system, the companion host controllers should also provide a similar method to allow system software to inhibit the companion host controller from accessing its shared memory based data structures (schedule lists or otherwise).

11.1.3.3.14 Port Test Modes -Host Operational Model

EHCI host controllers must implement the port test modes Test J_State, Test K_State, Test_Packet, Test Force_Enable, and Test SE0_NAK as described in the USB Specification Revision 2.0.

The system is only allowed to test ports that are owned by the EHCI controller (for example, *CF-bit* is a one and *PortOwner* bit is a zero). System software is allowed to have at most one port in test mode at a time. Placing more than one port in test mode will yield undefined results. The required, per port test sequence is (assuming the *CF-bit* in the USB_n_CONFIGFLAG register is a one):

- Disable the periodic and asynchronous schedules by setting the *Asynchronous Schedule Enable* and *Periodic Schedule Enable* bits in the USBCMD register to a zero.
- Place all enabled root ports into the suspended state by setting the *Suspend* bit in each appropriate USB_n_PORTSC register to a one.
- Set the *Run/Stop* bit in the USBCMD register to a zero and wait for the *HCHalted* bit in the USBSTS register, to transition to a one. Note that an EHCI host controller implementation may optionally allow port testing with the *Run/Stop* bit set to a one. However, all host controllers must support port testing with *Run/Stop* set to a zero and *HCHalted* set to a one.
- Set the *Port Test Control* field in the port under test PORTSC register to the value corresponding to the desired test mode. If the selected test is Test_Force_Enable, then the *Run/Stop* bit in the USBCMD register must then be transitioned back to one, in order to enable transmission of SOFs out of the port under test.
- When the test is complete, system software must ensure the host controller is halted (*HCHalted* bit is a one) then it terminates and exits test mode by setting *HCReset* to a one.

11.1.3.3.15 Interrupts-Host Operational Model

The EHCI Host Controller hardware provides interrupt capability based on a number of sources.

There are several general groups of interrupt sources:

- Interrupts as a result of executing transactions from the schedule (success and error conditions),
- Host controller events (Port change events, etc.), and
- Host Controller error events

All transaction-based sources are maskable through the Host Controller's Interrupt Enable register (USBINTR, see Section). Additionally, individual transfer descriptors can be marked to generate an interrupt on completion. This section describes each interrupt source and the processing that occurs in response to the interrupt.

During normal operation, interrupts may be immediate or deferred until the next interrupt threshold occurs. The interrupt threshold is a tunable parameter via the *Interrupt Threshold Control* field in the USBCMD register. The value of this register controls when the host controller will generate an interrupt on behalf of normal transaction execution. When a transaction completes during an interrupt interval period, the interrupt signaling the completion of the transfer will not occur until the interrupt threshold occurs. For example, the default value is eight micro-frames. This means that the host controller will not generate interrupts any more frequently than once every eight micro-frames.

Section Host System Error details effects of a host system error.

If an interrupt has been scheduled to be generated for the current interrupt threshold interval, the interrupt is not signaled until after the status for the last complete transaction in the interval has been written back to host memory. This may sometimes result in the interrupt not being signaled until the next interrupt threshold.

Initial interrupt processing is the same, regardless of the reason for the interrupt. When an interrupt is signaled by the hardware, Arm platform control is transferred to host controller's USB interrupt handler. The precise mechanism to accomplish the transfer is OS specific. For this discussion it is just assumed that control is received. When the interrupt handler receives control, its first action is to reads the USBSTS (USB Status Register). It then acknowledges the interrupt by clearing all of the interrupt status bits by writing ones to these bit positions. The handler then determines whether the interrupt is due to schedule processing or some other event. After acknowledging the interrupt, the handler (via an OS-specific mechanism), schedules a deferred procedure call (DPC) which will execute later. The DPC routine processes the results of the schedule execution. The precise mechanisms used are beyond the scope of this document.

Note: the host controller is not required to de-assert a currently active interrupt condition when software sets the interrupt enables (in the USBINR register, see Section) to a zero. The only reliable method software should use for acknowledging an interrupt is by transitioning the appropriate status bits in the USBSTS register (Section) from a one to a zero.

11.1.3.3.15.1 Transfer/Transaction Based Interrupts

These interrupt sources are associated with transfer and transaction progress. They are all dependent on the next interrupt threshold.

11.1.3.3.15.1.1 Transaction Error

A transaction error is any error that caused the host controller to think that the transfer did not complete successfully.

The table below lists the events/responses that the host can observe as a result of a transaction. The effects of the error counter and interrupt status are summarized in the following paragraphs. Most of these errors set the *XactErr* status bit in the appropriate interface data structure.

There is a small set of protocol errors that relate only when executing a queue head and fit under the umbrella of a WRONG PID error that are significant to explicitly identify. When these errors occur, the *XactErr* status bit in the queue head is set and the *CErr* field is decremented. When the *PIDCode* indicates a SETUP, the following responses are protocol errors and result in *XactErr* bit being set to a one and the *CErr* field being decremented.

- *EPS* field indicates a high-speed device and it returns a Nak handshake to a SETUP.
- *EPS* field indicates a high-speed device and it returns a Nyet handshake to a SETUP.
- *EPS* field indicates a low- or full-speed device and the complete-split receives a Nak handshake.

Table 11-51. Summary of Transaction Errors

Event / Result	Queue Head/qTD/iTD/siTD Side-effects		USB Status Register (USBSTS)
	Cerr	Status Field	USBERRINT
CRC	-1	XactErr set to a one.	1[1]
Timeout	-1	XactErr set to a one.	1[1]
Bad PID[2]	-1	XactErr set to a one.	1[1]
Babble	N/A	Section Serial Bus Babble	1
Buffer Error	N/A	Section Data Buffer Error	

1. If occurs in a queue head, then *USBERRINT* is asserted only when *CErr* counts down from a one to a zero. In addition the queue is halted, see Halting a Queue Head.
2. The host controller received a response from the device, but it could not recognize the PID as a valid PID.

11.1.3.3.15.1.2 Serial Bus Babble

When a device transmits more data on the USB than the host controller is expecting for this transaction, it is defined to be babbling. In general, this is called a *Packet Babble*.

When a device sends more data than the *Maximum Length* number of bytes, the host controller sets the *Babble Detected* bit to a one and halts the endpoint if it is using a queue head (see Halting a Queue Head). *Maximum Length* is defined as the minimum of *Total Bytes to Transfer* and *Maximum Packet Size*. The *CErr* field is not decremented for

a packet babble condition (only applies to queue heads). A babble condition also exists if IN transaction is in progress at High-speed EOF2 point. This is called a frame babble. A frame babble condition is recorded into the appropriate schedule data structure. In addition, the host controller must disable the port to which the frame babble is detected.

The *USBERRINT* bit in the USB_n_USBSTS register is set to a one and if the *USB Error Interrupt Enable* bit in the USB_n_USBINTR register is a one, then a hardware interrupt is signaled to the system at the next interrupt threshold. The host controller must never start an OUT transaction that will babble across a micro-frame EOF.

NOTE

When a host controller detects a data PID mismatch, it must either: disable the packet babble checking for the duration of the bus transaction or do packet babble checking based solely on *Maximum Packet Size*. The USB core specification defines the requirements on a data receiver when it receives a data PID mismatch (for example, expects a DATA0 and gets a DATA1 or visa-versa). In summary, it must ignore the received data and respond with an ACK handshake, in order to advance the transmitter's data sequence.

The EHCI interface allows System software to provide buffers for a Control, Bulk or Interrupt IN endpoint that are not an even multiple of the maximum packet size specified by the device. Whenever a device misses an ACK for an IN endpoint, the host and device are out of synchronization with respect to the progress of the data transfer. The host controller may have advanced the transfer to a buffer that is less than maximum packet size. The device will re-send its maximum packet size data packet, with the original data PID, in response to the next IN token. In order to properly manage the bus protocol, the host controller must disable the packet babble check when it observes the data PID mismatch.

11.1.3.3.15.1.3 Data Buffer Error

This event indicates that an overrun of incoming data or a underrun of outgoing data has occurred for this transaction.

This would generally be caused by the host controller not being able to access required data buffers in memory within necessary latency requirements. These conditions are not considered transaction errors, and do not effect the error count in the queue head. When these errors do occur, the host controller records the fact the error occurred by setting the *Data Buffer Error* bit in the queue head, iTD or siTD.

If the data buffer error occurs on a non-isochronous IN, the host controller will not issue a handshake to the endpoint. This will force the endpoint to resend the same data (and data toggle) in response to the next IN to the endpoint.

If the data buffer error occurs on an OUT, the host controller must corrupt the end of the packet so that it cannot be interpreted by the device as a good data packet. Simply truncating the packet is not considered acceptable. An acceptable implementation option is to 1's complement the CRC bytes and send them. There are other options suggested in the Transaction Translator section of the USB Specification Revision 2.0.

11.1.3.3.15.1.4 USB Interrupt (Interrupt on Completion (IOC))

Transfer Descriptors (iTDs, siTDs, and queue heads (qTDs)) contain a bit that can be set to cause an interrupt on their completion. The completion of the transfer associated with that schedule item causes the USB Interrupt (USBINT) bit in the USB_n_USBSTS register to be set to a one.

In addition, if a short packet is encountered on an IN transaction associated with a queue head, then this event also causes USBINT to be set to a one. If the USB Interrupt Enable bit in the USB_n_USBINTR register is set to a one, a hardware interrupt is signaled to the system at the next interrupt threshold. If the completion is because of errors, the *USBERRINT* bit in the USB_n_USBSTS register is also set to a one.

11.1.3.3.15.1.5 Short Packet

Reception of a data packet that is less than the endpoint's Max Packet size during Control, Bulk or Interrupt transfers signals the completion of the transfer. Whenever a short packet completion occurs during a queue head execution, the *USBINT* bit in the USB_n_USBSTS register is set to a one.

If the *USB Interrupt Enable* bit is set in the USB_n_USBINTR register, a hardware interrupt is signaled to the system at the next interrupt threshold.

11.1.3.3.15.2 Host Controller Event Interrupts

These interrupt sources are independent of the interrupt threshold (with the one exception being the Interrupt on Async Advance, see Section Interrupt on Async Advance).

11.1.3.3.15.2.1 Port Change Events

Port registers contain status and status change bits. When the status change bits are set to a one, the host controller sets the *Port Change Detect* bit in the USBSTS register to a one.

If the *Port Change Interrupt Enable* bit in the USB_n_USBINTR register is a one, then the host controller will issue a hardware interrupt. The port status change bits include:

- Connect Status Change
- Port Enable/Disable Change
- Over-current Change
- Force Port Resume

11.1.3.3.15.2.2 Frame List Rollover

This event indicates that the host controller has wrapped the frame list. The current programmed size of the frame list effects how often this interrupt occurs.

If the frame list size is 1024, then the interrupt will occur every 1024 milliseconds, if it is 512, then it will occur every 512 milliseconds, etc. When a frame list rollover is detected, the host controller sets the *Frame List Rollover* bit in the USB.USBSTS register to a one. If the *Frame List Rollover Enable* bit in the USB.USBINTR register is set to a one, the host controller issues a hardware interrupt. This interrupt is not delayed to the next interrupt threshold.

11.1.3.3.15.2.3 Interrupt on Async Advance

This event is used for deterministic removal of queue heads from the asynchronous schedule. Whenever the host controller advances the on-chip context of the asynchronous schedule, it evaluates the value of the *Interrupt on Async Advance Doorbell* bit in the USB.USBCMD register.

If it is a one, it sets the *Interrupt on Async Advance* bit in the USB.USBSTS register to a one. If the *Interrupton Async Advance Enable* bit in the USB.USBINTR register is a one, the host controller issues a hardware interrupt at the next interrupt threshold. A detailed explanation of this feature is described in Section Removing Queue Heads from Asynchronous Schedule .

11.1.3.3.15.2.4 Host System Error

The host controller is a bus master and any interaction between the host controller and the system may experience errors.

The type of host error may be catastrophic to the host controller (such as a Master Abort) making it impossible for the host controller to continue in a coherent fashion.In the presence of non-catastrophic host errors, such as parity errors, the host controller could potentially continue operation. The recommended behavior for these types of errors is to escalate it to a catastrophic error and halt the host controller. Host-based error must result in the following actions:

- The *Run/Stop* bit in the USB.USBCMD register is set to a zero.
- The following bits in the USB.USBSTS register are set:
  - *Host System Error* bit is to a one.
  - *HCHalted* bit is set to a one.
- If the *Host System Error Enable* bit in the USB.USBINTR register is a one, then the host controller will issue a hardware interrupt. This interrupt is not delayed to the next interrupt threshold. The following table summarizes the required actions taken on the various host errors.

Table 11-52. Summary Behavior of EHCI Host Controller on Host System Errors

Cycle Type	Master Abort	Target Abort	Data Phase Parity
Frame list pointer fetch (read)	Fatal	Fatal	Fatal [o]
siTD fetch (read)	Fatal	Fatal	Fatal [o]
siTD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
iTD fetch (read)	Fatal	Fatal	Fatal [o]
iTD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
qTD fetch (read)	Fatal	Fatal	Fatal [o]
qHD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
Data write	Fatal [o]	Fatal [o]	Fatal [o]
Data read	Fatal	Fatal	Fatal [o]

Potentially, a host controller implementation could continue operation without a halt. However, the recommended behavior is to halt the host controller.

NOTE

After a *Host System Error*, Software must reset the host controller through *HCReset* in the USB.USBCMD register before re-initializing and restarting the host controller.

11.1.3.4 EHCI Deviation

For the purposes a dual-role Host/Device controller with support for On-The-Go applications, it is necessary to deviate from the EHCI specification. Enhanced Host Controller Interface Specification for Universal Serial Bus, Revision 0.95, November 2000, Intel Corporation. http://www.intel.com. Device operation & On-The-Go operation is not specified in the EHCI and thus the implementation supported in this core is proprietary.

The host mode operation of the core is near EHCI compatible with few minor differences documented in this section.

The particulars of the deviations occur in the areas summarized here:

- Embedded Transaction Translator - Allows direct attachment of FS and LS devices in host mode without the need for a companion controller.
- Device operation - In host mode the device operational registers are generally disabled and thus device mode is mostly transparent when in host mode. However, there are a couple exceptions documented in the following sections.
- Embedded design interface - This core does not have a PCI Interface and therefore the PCI configuration registers described in the EHCI specification are not applicable.
- On-The-Go Operation - This design includes an On-The-Go controller for Port #1.

11.1.3.4.1 Embedded Transaction Translator Function

The OTG controller supports directly connected full and low speed devices without requiring a companion controller by including the capabilities of a USB 2.0 high speed hub transaction translator.

Although there is no separate Transaction Translator block in the system, the transaction translator function normally associated with a high speed hub has been implemented within the DMA and Protocol engine blocks. The embedded transaction translator function is an extension to EHCI interface, but makes use of the standard data structures and operational models that exist in the EHCI specification to support full and low speed devices.

11.1.3.4.1.1 Capability Registers

The following additions have been added to the capability registers to support the embedded Transaction Translator Function:

- N_TT added to USB.HCSPARAMS - Host Control Structural Parameters
- N_PTT added to USB.HCSPARAMS - Host Control Structural Parameters

11.1.3.4.1.2 Operational Registers

The following additions have been added to the operational registers to support the embedded TT:

- Addition of two-bit Port Speed (PSPD) to the register.

11.1.3.4.1.3 Discovery-EHCI Deviation

In a standard EHCI controller design, the EHCI host controller driver detects a Full speed (FS) or Low speed (LS) device by noting if the port enable bit is set after the port reset operation.

The port enable will only be set in a standard EHCI controller implementation after the port reset operation and when the host and device negotiate a High-Speed connection (that is, Chirp completes successfully).

Because this controller has an embedded Transaction Translator, the port enable will always be set after the port reset operation regardless of the result of the host device chirp result and the resulting port speed will be indicated by the PSPD field in USB.PORTSCx.

Therefore, the standard EHCI host controller driver requires an alteration to handle directly connected Full and Low speed devices or hubs.

The change is a fundamental one in that is summarized in the following table.

Table 11-53. Summary of EHCI

Standard EHCI	EHCI with embedded Transaction Translator
After port enable bit is set following a connection and reset sequence, the device/hub is assumed to be HS.	After port enable bit is set following a connection and reset sequence, the device/hub speed is noted from USB.PORTSCx.
FS and LS devices are assumed to be downstream from a HS hub thus, all port-level control is performed through the Hub Class to the nearest Hub.	FS and LS device can be either downstream from a HS hub or directly attached. When the FS/LS device is downstream from a HS hub, then port-level control is done using the Hub Class through the nearest Hub. When a FS/LS device is directly attached, then port-level control is accomplished using USB.PORTSCx.
FS and LS devices are assumed to be downstream from a HS hub with HubAddr=X. [where HubAddr > 0 and HubAddr is the address of the Hub where the bus transitions from HS to FS/LS (ie. Split target hub)]	FS and LS device can be either downstream from a HS hub with HubAddr = X [HubAddr > 0] or directly attached [where HubAddr = 0 and HubAddr is the address of the Root Hub where the bus transitions from HS to FS/LS (ie. Split target hub is the root hub)]

11.1.3.4.1.4 Data Structures

The same data structures used for FS/LS transactions though a HS hub are also used for transactions through the Root Hub with sm embedded Transaction Translator.

Here it is demonstrated how the Hub Address and Endpoint Speed fields should be set for directly attached FS/LS devices and hubs:

1. QH (for direct attach FS/LS) - Async. (Bulk/Control Endpoints) Periodic (Interrupt)
    - Hub Address = 0
    - Transactions to direct attached device/hub.
        - QH.EPS = Port Speed
    - Transactions to a device downstream from direct attached FS hub.

- QH.EPS = Downstream Device Speed

NOTE

When QH.EPS = 01 (LS) and PORTSCx.PSPD = 00 (FS), a LS-pre-pid will be sent before the transmitting LS traffic.

Maximum Packet Size must be less than or equal 64 or undefined behaviour may result.

2. siTD (for direct attach FS) - Periodic (ISO Endpoint)
   - All FS ISO transactions:
     - Hub Address = 0
     - siTD.EPS = 00 (full speed)
       - Maximum Packet Size must less than or equal to 1023 or undefined behaviour may result.

11.1.3.4.1.5 Operational Model

The operational models are well defined for the behavior of the Transaction Translator (see USB 2.0 specification. Universal Serial Bus Specification, Revision 2.0, April 2000, Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips. http://www.usb.org) and for the EHCI controller moving packets between system memory and a USB-HS hub. Because the embedded Transaction Translator exists within the host controller there is no physical bus between EHCI host controller driver and the USB FS/LS bus. These sections will briefly discuss the operational model for how the EHCI and Transaction Translator operational models are combined without the physical bus between. The following sections assume the reader is familiar with both the EHCI and USB 2.0 Transaction Translator operational models.

11.1.3.4.1.5.1 Micro- frame Pipeline

The EHCI operational model uses the concept of H-frames and B-frames to describe the pipeline between the Host (H) and the Bus (B). The embedded Transaction Translator shall use the same pipeline algorithms specified in the USB 2.0 specification for a Hub-based Transaction Translator.

All periodic transfers always begin at B-frame 0 (after SOF) and continue until the stored periodic transfers are complete. As an example of the micro-frame pipeline implemented in the embedded Transaction Translator, all periodic transfers that are tagged in EHCI to execute in H-frame 0 will be ready to execute on the bus in B-frame 0.

It is important to note that when programming the S-mask and C-masks in the EHCI data structures to schedule periodic transfers for the embedded Transaction Translator, the EHCI host controller driver must follow the same rules specified in EHCI for programming the S-mask and C-mask for downstream Hub-based Transaction Translators.

Once periodic transfers are exhausted, any stored asynchronous transfer will be moved. Asynchronous transfers are opportunistic in that they shall execute whenever possible and their operation is not tied to H-frame and B-frame boundaries with the exception that an asynchronous transfer can not babble through the SOF (start of B-frame 0.)

11.1.3.4.1.5.2 Split State Machines

The start and complete split operational model differs from EHCI slightly because there is no bus medium between the EHCI controller and the embedded Transaction Translator.

Where a start or complete-split operation would occur by requesting the split to the HS hub, the start/complete split operation is simple an internal operation to the embedded Transaction Translator. The following table summarizes the conditions where handshakes are emulated from internal state instead of actual handshakes to HS split bus traffic.

Table 11-54. Summary of the Conditons of Handshakes[1]

Condition	Emulate TT Response
Start-Split: All asynchronous buffers full.	NAK
Start-Split: All periodic buffers full.	ERR
Start-Split: Success for start of Async. Transaction.	ACK
Start-Split: Start Periodic Transaction.	No Handshake (Ok)
Complete-Split: Failed to find transaction in queue.	Bus Time Out
Complete-Split: Transaction in Queue is Busy.	NYET
Complete-Split: Transaction in Queue is Complete.	[Actual Handshake from LS/FS device]

1. The un-shaded cells represent Start-Splits and the shaded cells represent Complete-Splits

11.1.3.4.1.5.3 Asynchronous Transaction Scheduling and Buffer Management

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

- Sequencing is provided & a packet length estimator ensures no full-speed/low-speed packet babbles into SOF time.
- Transaction tracking for 2 data pipes.

11.1.3.4.1.5.3.1 USB 2.0 - 11.17.3

- Sequencing is provided & a packet length estimator ensures no full-speed/low-speed packet babbles into SOF time.

11.1.3.4.1.5.3.2 USB 2.0 - 11.17.4

- Transaction tracking for 2 data pipes.

11.1.3.4.1.5.4 Periodic Transaction Scheduling and Buffer Management

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

- Abort of pending start-splits
  - EOF (and not started in micro-frames 6)
  - Idle for more than 4 micro-frames
- Abort of pending complete-splits
  - EOF
  - Idle for more than 4 micro-frames
- Transaction tracking for up to 16 data pipes.
- Complete-split transaction searching.

NOTE

There is no data schedule mechanism for these transactions other than the micro-frame pipeline. The embedded TT assumes the number of packets scheduled in a frame does not exceed the frame duration (1 ms) or else undefined behavior may result.

11.1.3.4.1.5.4.1 USB 2.0 - 11.18.6.[1-2]

- Abort of pending start-splits
  - EOF (and not started in micro-frames 6)
  - Idle for more than 4 micro-frames
- Abort of pending complete-splits
  - EOF
  - Idle for more than 4 micro-frames

11.1.3.4.1.5.4.2 USB 2.0 - 11.18.[7-8]

- Transaction tracking for up to 16 data pipes.
- Complete-split transaction searching.

NOTE

There is no data schedule mechanism for these transactions other than the micro-frame pipeline. The embedded TT assumes

the number of packets scheduled in a frame does not exceed the frame duration (1 ms) or else undefined behavior may result.

11.1.3.4.1.5.5 Multiple Transaction Translators

The maximum number of embedded Transaction Translators that is currently supported is one as indicated by the N_TT field in the register.

11.1.3.4.2 Device Operation

The co-existence of a device operational controller within the host controller has little effect on EHCI compatibility for host operation except as noted in this section.

11.1.3.4.2.1 USB_USBMODE Register

Given that the dual-role controller is initialized in neither host nor device mode, the USB Device Mode (USB_*n*USBMODE) register must be programmed for host operation before the EHCI host controller driver can begin EHCI host operations.

11.1.3.4.2.2 Non-Zero Fields the Register File

Some of the reserved fields and reserved addresses in the capability registers and operational register have use in device mode, the following must be adhered to:

- Write operations to all EHCI reserved fields (some of which are device fields) with the operation registers should always be written to zero. This is an EHCI requirement of the device controller driver that must be adhered to.
- Read operations by the host controller must properly mask EHCI reserved fields (some of which are device fields) because fields that are used exclusive for device are undefined in host mode .

11.1.3.4.2.3 SOF Interrupt

This SOF Interrupt used for device mode is shared as a free running 125us interrupt for host mode.

EHCI does not specify this interrupt but it has been added for convenience and as a potential software time base. See USB Status Register (USB_*n*USBSTS) and Interrupt Enable Register (USB_*n*USBINTR) registers.

11.1.3.4.3 Embedded Design Interface

This is an Embedded USB Host Controller as defined by the EHCI specification and thus does not implement the PCI configuration registers.

11.1.3.4.3.1 Frame Adjust Register

Given that the optional PCI configuration registers are not included in this implementation, there is no corresponding bit level timing adjustments like is provided by the Frame Adjust register in the PCI configuration registers. Starts of micro-frames are timed precisely to 125 us using the transceiver clock as a reference clock. That is, a 60 Mhz transceiver clock for 8-bit physical interfaces & full-speed serial interfaces or 30 Mhz transceiver clock for 16-bit physical interfaces.

11.1.3.4.4 Miscellaneous variations from EHCI

11.1.3.4.4.1 Programmable Physical Interface Behaviour

This design supports multiple Physical interfaces which can operate in differing modes when the core is configured with software programmable Physical Interface Modes. Software programmability allows the selection of the Physical interface part during the board design phase instead of during the chip design phase. The control bits for selecting the Physical Interface operating mode have been added to the Port Status & Control (USB_*n*PORTSC1) register providing a capability that is not defined by EHCI.

11.1.3.4.4.2 Discovery

11.1.3.4.4.2.1 Port Reset

The port connect methods specified by EHCI require setting the port reset bit in the Port Status & Control (USB_*n*PORTSC1) register for a duration of 10ms. Due to the complexity required to support the attachment of devices that are not high speed there are counter already present in the design that can count the 10ms reset pulse to alleviate the requirement of the software to measure this duration. Therefore, the basic connection is then summarized as the following:

- [Port Change Interrupt] Port connect change occurs to notify the host controller driver that a device has attached.
- Software shall write a '1' to reset the device.
- Software shall write a '0' to reset the device after 10 ms.

- This step, which is necessary in a standard EHCI design, may be omitted with this implementation. Should the EHCI host controller driver attempt to write a '0' to the reset bit while a reset is in progress, the write will simple be ignored and the reset will continue until completion.

- [Port Change Interrupt] Port enable change occurs to notify the host controller that the device in now operational and at this point the port speed has been determined.

11.1.3.4.4.2.2 Port Speed Detection

After the port change interrupt indicates that a port is enabled, the EHCI stack should determine the port speed. Unlike the EHCI implementation, which will re-assign the port owner for any device that does not connect at High-Speed, this host controller supports direct attach of non High-Speed devices.

Therefore, the following differences are important regarding port speed detection:

- Port Owner is read-only and always reads 0.
- A 2-bit Port Speed indicator has been added to PORTSC to provide the current operating speed of the port to the host controller driver.
- A 1-bit High Speed indicator has been added to PORTSC to signify that the port is in High-Speed vs. Full/Low Speed - *This information is redundant with the 2-bit Port Speed indicator above.*

11.1.3.4.4.3 Port Test Mode

Port Test Control mode behaves fully as described in EHCI. An alternate host controller driver procedure is not necessary or supported.

11.1.3.5 Device Data Structures

This section defines the interface data structures used to communicate control, status, and data between Device Controller Driver (DCD) Software and the Device Controller.

The data structure definitions in this chapter support a 32-bit memory buffer address space. The interface consists of device Queue Heads and Transfer Descriptors.

NOTE

Software must ensure that no interface data structure reachable by the Device Controller spans a 4K-page boundary.

The data structures defined in the chapter are (from the device controller's perspective) a mix of read-only and read/ writable fields. The device controller must preserve the read-only fields on all data structure writes.

The figure below shows the organization of the EndPoint Queue Head.

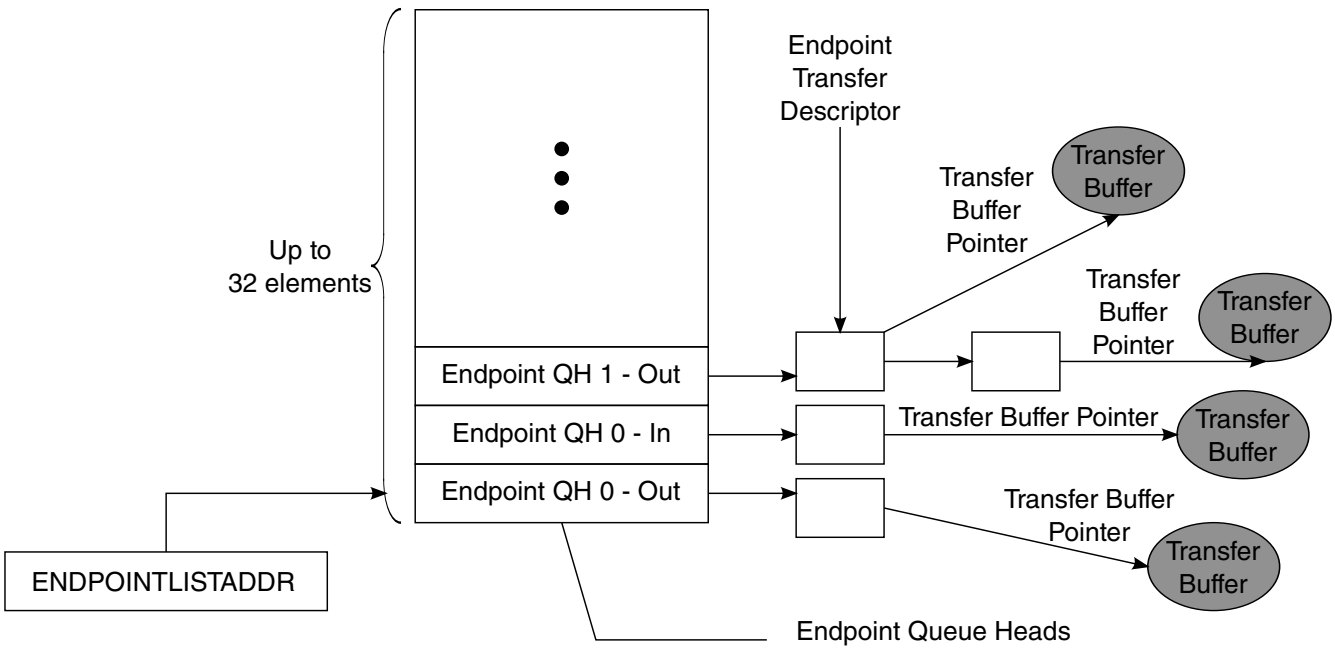


Figure 11-28. EndPoint Queue Head Organization

Endpoint queue heads are arranged in an array in a continuous area of memory pointed to by the USB.ENDPOINTLISTADDR pointer. The even -numbered device queue heads in the list support receive endpoints (OUT/SETUP) and the odd-numbered queue heads in the list are used for transmit endpoints (IN/INTERRUPT). The device controller will index into this array based upon the endpoint number received from the USB bus. All information necessary to respond to transactions for all primed transfers is contained in this list so the Device Controller can readily respond to incoming requests without having to traverse a linked list.

NOTE

The Endpoint Queue Head List must be aligned to a 2k boundary.

11.1.3.5.1 Endpoint Queue Head (dQH)

The device Endpoint Queue Head (dQH) is where all transfers for a given endpoint are managed. The dQH is a 48-byte data structure, but must be aligned on 64-byte boundaries.

During priming of an endpoint, the dTD (device transfer descriptor) is copied into the overlay area of the dQH, which starts at the nextTD pointer DWord and continues through the end of the buffer pointers DWords. After a transfer is complete, the dTD status DWord is updated in the dTD pointed to by the currentTD pointer. While a packet is in progress, the overlay area of the dQH is used as a staging area for the dTD so that the Device Controller can access needed information with little minimal latency.

Table 11-55. Endpoint Queue Head (dQH)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Mult		zlt	0		Maximum Packet Length											ios	0														
Current dTD Pointer																											0				
Next dTD Pointer																											0				T[1]
0	Total Bytes															ioc	0			MultO		0		Status							
Buffer Pointer (Page 0)																				Current Offset											
Buffer Pointer (Page 1)																				Reserved											
Buffer Pointer (Page 2)																				Reserved											
Buffer Pointer (Page 3)																				Reserved											
Buffer Pointer (Page 4)[1]																				Reserved											
Reserved																															
Set-up Buffer Bytes 3…0																															
Set-up Buffer Bytes 7…4																															

1. Transfer overlay starts at T and continues through Buffer Pointer (Page 4).

Host Controller Read/Write

Host Controller Read Only

11.1.3.5.1.1 Endpoint Capabilities/Characteristics

This DWord specifies static information about the endpoint, in other words, this information does not change over the lifetime of the endpoint. Device Controller software should not attempt to modify this information while the corresponding endpoint is enabled.

Table 11-56 describes the endpoint capabilities.

Table 11-56. Endpoint Capabilities/Characteristics

Bit	Description
31-30	Mult. This field is used to indicate the number of packets executed per transaction description as given by the following: 00 - Execute N Transactions as demonstrated by the USB variable length packet protocol where N is computed using the Maximum Packet Length (dQH) and the Total Bytes field (dTD) 01 Execute 1 Transaction. 10 Execute 2 Transactions. 11 Execute 3 Transactions. **NOTE:** Non-ISO endpoints must set Mult="00". ISO endpoints must set Mult="01", "10", or "11" as needed.
29	Zero Length Termination Select. This bit is used to indicate when a zero length packet is used to terminate transfers where to total transfer length is a multiple . This bit is not relevant for Isochronous 0 - Enable zero length packet to terminate transfers equal to a multiple of the Maximum Packet Length. (default). 1 - Disable the zero length packet on transfers that are equal in length to a multiple Maximum Packet Length.
28-27	Reserved. These bit reserved for future use and should be set to zero.
26-16	Maximum Packet Length. This directly corresponds to the maximum packet size of the associated endpoint (wMaxPacketSize). The maximum value this field may contain is 0x400 (1024).
15	Interrupt On Setup (IOS). This bit is used on control type endpoints to indicate if USBINT is set in response to a setup being received.
14-0	Reserved. Bits reserved for future use and should be set to zero.

11.1.3.5.1.2 Transfer Overlay-Endpoint Queue Head

The seven DWords in the overlay area represent a transaction working space for the device controller.

The general operational model is that the device controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then it will not read the associated endpoint.

After an endpoint is readied, the dTD will be copied into this queue head overlay area by the device controller. Until a transfer is expired, software must not write the queue head overlay area or the associated transfer descriptor. When the transfer is complete, the device controller will write the results back to the original transfer descriptor and advance the queue.

See dTD for a description of the overlay fields.

11.1.3.5.1.3 Current dTD Pointer

The current dTD pointer is used by the device controller to locate the transfer in progress. This word is for Device Controller (hardware) use only and should not be modified by DCD software.

The following table describes the dTD Pointer.

Table 11-57. Next dTD Pointer

Bit	Description
31-5	Current dTD. This field is a pointer to the dTD that is represented in the transfer overlay area. This field will be modified by the Device Controller to next dTD pointer during endpoint priming or queue advance.
4-0	Reserved. Bit reserved for future use and should be set to zero.

11.1.3.5.1.4 Set-up Buffer

The set-up buffer is dedicated storage for the 8-byte data that follows a set-up PID.

NOTE

Each endpoint has a TX and an RX dQH associated with it, and only the RX queue head is used for receiving setup data packets.

The following table describes the Multiple Mode Control.

Table 11-58. Multiple Mode Control (HCCPARAMS)

DWord	Bits	Description
1	31-0	Setup Buffer 0. This buffer contains bytes 3 to 0 of an incoming setup buffer packet and is written by the device controller to be read by software.
2	31-0	Setup Buffer 1. This buffer contains bytes 7 to 4 of an incoming setup buffer packet and is written by the device controller to be read by software.

11.1.3.5.2 Endpoint Transfer Descriptor (dTD)

The dTD describes to the device controller the location and quantity of data to be sent/ received for a given transfer.

The DCD should not attempt to modify any field in an active dTD except the Next Like Pointer, which should only be modified as described in section Managing Transfers with Transfer Descriptors.

Table below shows the Endpoint Transfer Descriptor (dTD).

Table 11-59. Endpoint Transfer Descriptor (dTD)

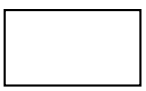
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Next Link Pointer																											0				T
0	Total Bytes															ioc	0			MultO		0		Status							

Table continues on the next page...

Table 11-59. Endpoint Transfer Descriptor (dTD) (continued)

Buffer Pointer (Page 0)	Current Offset
Buffer Pointer (Page 1)	0 Frame Number
Buffer Pointer (Page 2)	Reserved
Buffer Pointer (Page 3)	Reserved
Buffer Pointer (Page 4)	Reserved

Host Controller Read/Write



Host Controller Read Only

The following table describes the dTD Pointer.

Table 11-60. Next dTD Pointer

Bit	Description
31-5	Next Transfer Element Pointer. This field contains the physical memory address of the next dTD to be processed. The field corresponds to memory address signals [31:5], respectively.
4-1	Reserved. Bits reserved for future use and should be set to zero.
0	Terminate (T). 1=pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Device Controller that there are no more valid entries in the queue.

The following table describes the dTD Token.

Table 11-61. dTD Token

Bit	Description
31	Reserved. Bit reserved for future use and should be set to zero.
30-16	Total Bytes. This field specifies the total number of bytes to be moved with this transfer descriptor. This field is decremented by the number of bytes actually moved during the transaction and only on the successful completion of the transaction. The maximum value software may store in the field is 5*4K (5000H). This is the maximum number of bytes 5 page pointers can access. Although it is possible to create a transfer up to 20K this assumes the 1st offset into the first page is 0. When the offset cannot be predetermined, crossing past the 5th page can be guaranteed by limiting the total bytes to 16K**. Therefore, the maximum recommended transfer is 16K (4000H). If the value of the field is zero when the host controller fetches this transfer descriptor (and the active bit is set), the device controller executes a zero-length transaction and retires the transfer descriptor. It is not a requirement for IN transfers that Total Bytes To Transfer be an even multiple of *Maximum Packet Length*. If software builds such a transfer descriptor for an IN transfer, the last transaction will always be less that *Maximum Packet Length*.
15	Interrupt On Complete (IOC). This bit is used to indicate if USBINT is to be set in response to device controller being finished with this dTD.
14-12	Reserved. Bits reserved for future use and should be set to zero.

Table continues on the next page...

Table 11-61. dTD Token (continued)

11-10	Multiplier Override (MultO). This field can be used for transmit ISO's (ie. ISO-IN) to override the multiplier in the QH. This field must be zero for all packet types that are not transmit-ISO. Example: if QH.multiplier = 3; Maximum packet size = 8; Total Bytes = 15; MultiO = 0 [default] Three packets are sent: {Data2(8); Data1(7); Data0(0)} if QH.multiplier = 3; Maximum packet size = 8; Total Bytes = 15; MultiO = 2 Two packets are sent: {Data1(8); Data0(7)} For maximal efficiency, software should compute MultO = greatest integer of (Total Bytes / Max. Packet Size) except for the case when Total Bytes = 0; then MultO should be 1. Note: Non-ISO and Non-TX endpoints must set MultO = "00".
9-8	Reserved. Bits reserved for future use and should be set to zero.
7-0	Status. This field is used by the Device Controller to communicate individual command execution states back to the Device Controller software. This field contains the status of the last transaction performed on this qTD. The bit encodings are: Bit Status Field Description 7 Active. 6 Halted. 5 Data Buffer Error. 3 Transaction Error. 4, 2, 0 Reserved.

The table below describes the dTD Buffer Page Pointer List.

Table 11-62. dTD Buffer Page Pointer List

Bit	Description
31-12	Buffer Pointer. Selects the page offset in memory for the packet buffer. Non virtual memory systems will typically set the buffer pointers to a series of incrementing integers.
0,11-0	Current Offset. Offset into the 4kb buffer where the packet is to begin.
1,10-0	Frame Number. Written by the device controller to indicate the frame number in which a packet finishes. This is typically be used to correlate relative completion times of packets on an ISO endpoint.

11.1.3.6 Device Operational Model

The function of the device operation is to transfer a request in the memory image to and from the Universal Serial Bus.

Using a set of linked list transfer descriptors, pointed to by a queue head, the device controller will perform the data transfers. The following sections explain the use of the device controller from the device controller driver (DCD) point-of-view and further describe how specific USB bus events relate to status changes in the device controller programmer's interface.

11.1.3.6.1 Device Controller Initialization

After hardware reset, the device is disabled until the Run/Stop bit is set to a '1'. In the disabled state, the pull-up on the USB D+ is not active which prevents an attach event from occurring. At a minimum, it is necessary to have the queue heads setup for endpoint zero before the device attach occurs.

Shortly after the device is enabled, a USB reset will occur followed by setup packet arriving at endpoint 0. A Queue head must be prepared so that the device controller can store the incoming setup packet.

In order to initialize a device, the software should perform the following steps:

- Set Controller Mode in the USB.USBMODE register to device mode.

NOTE

Transitioning from host mode to device mode requires a device controller reset before modifying USB.USBMODE.

- Allocate and Initialize device queue heads in system memory.
  - Minimum: Initialize device queue heads 0 Tx & 0 Rx.

NOTE

All device queue heads for control endpoints must be initialized before the endpoint is enabled. Non-Control device queue heads before the endpoint can be used.

  - For information on device queue heads, refer to section Device Data Structures.
- Configure USB.ENDPOINTLISTADDR Pointer.
  - For additional information on USB.ENDPOINTLISTADDR, refer to the register table.
- Enable the microprocessor interrupt associated with the USB core.
  - Recommended: enable all device interrupts including: USBINT, USBERRINT, Port Change Detect, USB Reset Received, DCSuspend.
  - For a list of available interrupts refer to the Interrupt Enable Register (USB_*n*USBINTR) and the USB Status Register (USB_*n*USBSTS) register tables.
- Set Run/Stop bit to Run Mode.

- After the Run bit is set and the device is connected to a host, a Bus Reset will be issued by host downstream port. The DCD must monitor the reset event and adjust the software state as described in the Bus Reset section of the Port State and Control section below.

NOTE

Endpoint 0 is designed as a control endpoint only and does not need to be configured using ENDPTCTRL0 register.

It is also not necessary to prime Endpoint 0 initially because the first packet received will always be a setup packet. The contents of the first setup packet will require a response in accordance with USB device framework (Chapter 9) command set.

11.1.3.6.2 Port State and Control

From a chip or system reset, the device controller enters the *powered* state. A transition from the *powered* state to the *attach* state occurs when the Run/Stop bit is set to a '1'.

After receiving a reset on the bus, the port will enter the *defaultFS or defaultHS* state in accordance with the reset protocol described in Appendix C.2 of the USB Specification Rev. 2.0.

The following state diagram depicts the state of a USB 2.0 device.

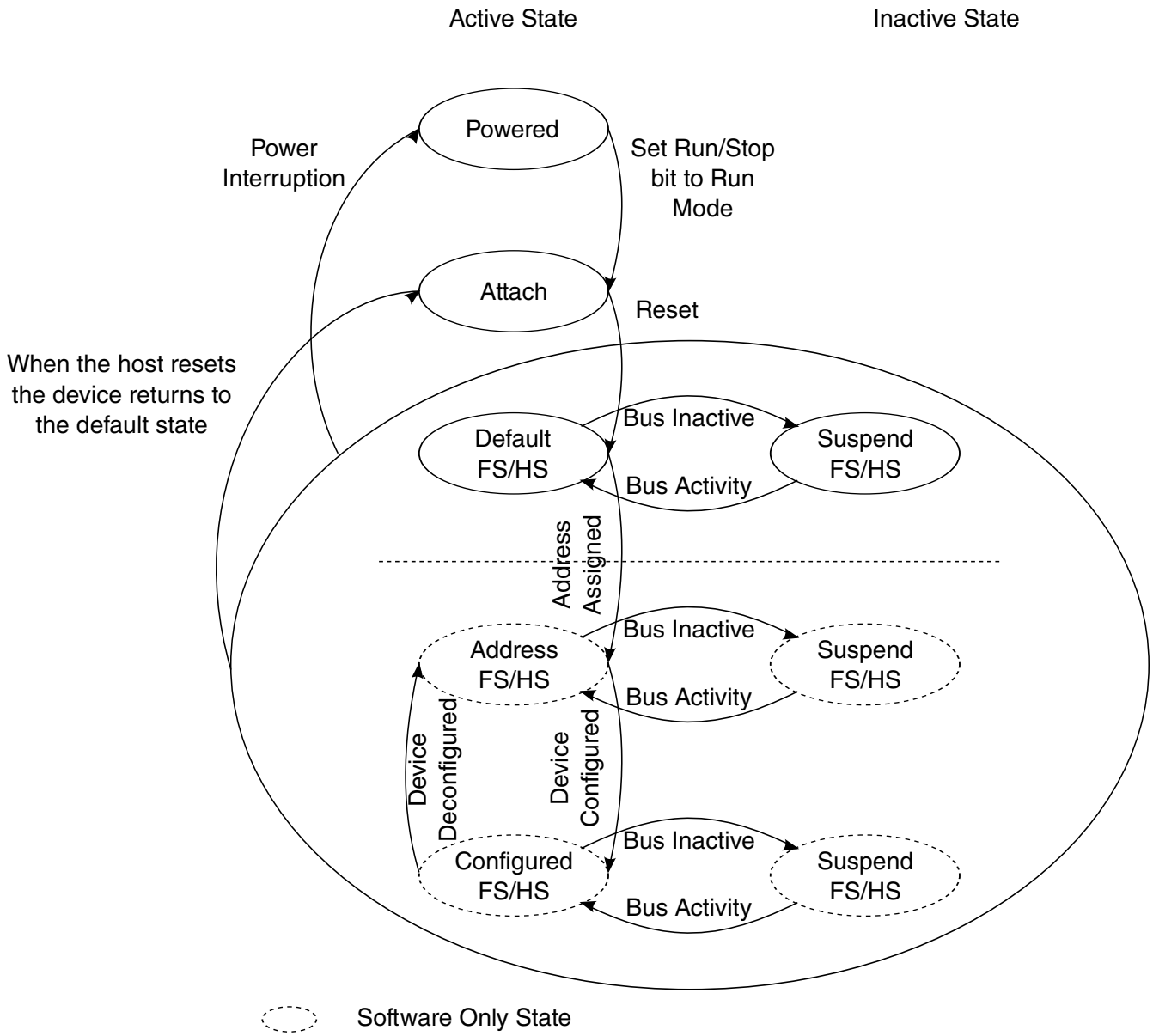


Figure 11-29. Device State Diagram

States *powered, attach, defaultFS/HS, suspendFS/HS* are implemented in the device controller and are communicated to the DCD using the following status bits:

The following table describes the Device Controller State Information Bits.

Table 11-63. Device Controller State Information Bits

Bit	Register
DCSuspend	
USB Reset Received	
Port Change Detect	
High-Speed Port	

It is the responsibility of the DCD to maintain a state variable to differentiate between the *DefaultFS/HS* state and the *Address/Configured* states. Change of state from *Default* to *Address* and the *Configured* states is part of the enumeration process described in the device framework section of the USB 2.0 Specification.

As a result of entering the *Address* state, the device address register (DEVICEADDR) must be programmed by the DCD.

Entry into the *Configured* indicates that all endpoints to be used in the operation of the device have been properly initialized by programming the USB_UOG_ENDPTCTRLx registers and initializing the associated queue heads.

11.1.3.6.2.1 Bus Reset

A bus reset is used by the host to initialize downstream devices.

When a bus reset is detected, the device controller will renegotiate its attachment speed, reset the device address to 0, and notify the DCD by interrupt (assuming the USB Reset Interrupt Enable is set). After a reset is received, all endpoints (except endpoint 0) are disabled and any primed transactions will be cancelled by the device controller. The concept of priming will be clarified below, but the DCD must perform the following tasks when a reset is received:

Clear all setup token semaphores by reading the Endpoint Status (USB_*n*ENDPTSTAT) register and writing the same value back to the Endpoint Status (USB_*n*ENDPTSTAT) register.

Clear all the endpoint complete status bits by reading the Endpoint Complete (USB_*n*ENDPTCOMPLETE) register and writing the same value back to the Endpoint Complete (USB_*n*ENDPTCOMPLETE) register.

Cancel all primed status by waiting until all bits in the Endpoint Prime (USB_*n*ENDPTPRIME) are 0 and then writing 0xFFFFFFFF to Endpoint Flush (USB_*n*ENDPTFLUSH).

Read the reset bit in the Port Status & Control (USB_*n*PORTSC1) register and make sure that it is still active. A USB reset will occur for a minimum of 3 ms and the DCD must reach this point in the reset cleanup before end of the reset occurs, otherwise a hardware reset of the device controller is recommended (rare.)

- A hardware reset can be performed by writing a one to the device controller reset bit in the USBCMD reset. Note: a hardware reset will cause the device to detach from the bus by clearing the Run/Stop bit. Thus, the DCD must completely re-initialize the device controller after a hardware reset.

Free all allocated dTDs because they will no longer be executed by the device controller. If this is the first time the DCD is processing a USB reset event, then it is likely that no dTDs have been allocated.

At this time, the DCD may release control back to the OS because no further changes to the device controller are permitted until a Port Change Detect is indicated.

After a Port Change Detect, the device has reached the default state and the DCD can read the Port Status & Control (USB_*n*PORTSC1) to determine if the device is operating in FS or HS mode. At this time, the device controller has reached normal operating mode and DCD can begin enumeration according to the USB Chapter 9 - Device Framework.

NOTE

The device DCD may use the FS/HS mode information to determine the bandwidth mode of the device

In some applications, it may not be possible to enable one or more pipes while in FS mode. *Beyond the data rate issue, there is no difference in DCD operation between FS and HS modes.*

11.1.3.6.2.2 Suspend/Resume

The detials of suspend and resume are explained in these sections.

11.1.3.6.2.2.1 Suspend

Suspend Description

In order to conserve power, USB devices automatically enter the suspended state when the device has observed no bus traffic for a specified period. When suspended, the USB device maintains any internal status, including its address and configuration. Attached devices must be prepared to suspend at any time they are powered, regardless of if they have been assigned a non-default address, are configured, or neither. Bus activity may cease due to the host entering a suspend mode of its own. In addition, a USB device shall also enter the suspended state when the hub port it is attached to is disabled.

A USB device exits suspend mode when there is bus activity. A USB device may also request the host to exit suspend mode or selective suspend by using electrical signaling to indicate remote wakeup. The ability of a device to signal remote wakeup is optional. If the USB device is capable of remote wakeup signaling, the device must support the ability of the host to enable and disable this capability. When the device is reset, remote wakeup signaling must be disabled.

Suspend Operational Model

The device controller moves into the suspend state when suspend signaling is detected or activity is missing on the upstream port for more than a specific period. After the device controller enters the suspend state, the DCD is notified by an interrupt (assuming *DC Suspend Interrupt* is enabled). When the *DCSuspend* bit in the Port Status & Control (USB_*n*PORTSC1) is set to a '1', the device controller is suspended.

DCD response when the device controller is suspended is application specific and may involve switching to low power operation.

Information on the bus power limits in suspend state can be found in USB 2.0 specification.

NOTE

Review system level clocking issues defined in section (Ref: Signals-Clocking) for the clocking requirements of a suspended device controller.

11.1.3.6.2.2.2 Resume

If the device controller is suspended, its operation is resumed when any non-idle signaling is received on its upstream facing port. In addition, the device can signal the system to resume operation by forcing resume signaling to the upstream port.

Resume signaling is sent upstream by writing a '1' to the Resume bit in the in the Port Status & Control (USB_*n*PORTSC1) while the device is in suspend state. Sending resume signal to an upstream port should cause the host to issue resume signaling and bring the suspended bus segment (one more devices) back to the active condition.

NOTE

Before resume signaling can be used, the host must enable it by using the Set Feature command defined in device framework (chapter 9) of the USB 2.0 Specification.

11.1.3.6.3 Managing Endpoints

The USB 2.0 specification defines an endpoint, also called a device endpoint or an address endpoint as a uniquely addressable portion of a USB device that can source or sink data in a communications channel between the host and the device.

The endpoint address is specified by the combination of the endpoint number and the endpoint direction.

The channel between the host and an endpoint at a specific device represents a data pipe. Endpoint 0 for a device is always a *control* type data channel used for device discovery and enumeration. Other types of endpoints support by USB include *bulk, interrupt*, and

isochronous. Each endpoint type has specific behavior related to packet response and error handling. More detail on endpoint operation can be found in the USB 2.0 specification.

The USB OTG device controller hardware supports up to 8 endpoint numbers.

Each endpoint direction is essentially independent and can be configured with differing behavior in each direction. For example, the DCD can configure endpoint 1-IN to be a bulk endpoint and endpoint 1-OUT to be an isochronous endpoint. This helps to conserve the total number of endpoints required for device operation. The only exception is that control endpoints must use both directions on a single endpoint number to function as a control endpoint. Endpoint 0 is, for example, is always a control endpoint and uses the pair of directions.

Each endpoint direction requires a *queue head* allocated in memory. To support the 8 endpoint numbers, 16 *queue heads* are required. The operation of an endpoint and use of *queue heads* are described later in this document.

11.1.3.6.3.1 Endpoint Initialization

After hardware reset, all endpoints except endpoint zero are uninitialized and disabled. The DCD must configure and enable each endpoint by writing to configuration bit in the USB_UOG_ENDPTCTRLx register.

Each 32-bit USB_UOG_ENDPTCTRLx is split into an upper and lower half. The lower half of USB_UOG_ENDPTCTRLx is used to configure the receive or OUT endpoint and the upper half is likewise used to configure the corresponding transmit or IN endpoint. Control endpoints must be configured the same in both the upper and lower half of the USB_UOG_ENDPTCTRLx register otherwise the behavior is undefined. The following table shows how to construct a configuration word for endpoint initialization. The following table shows the fields and values for the Device Controller Endpoint initialization.

Table 11-64. Device Controller Endpoint Initialization

Field	Value
Data Toggle Reset	1
Data Toggle Inhibit	0
Endpoint Type	00 Control 01 Isochronous 10 Bulk 11 Interrupt
Endpoint Stall	0

11.1.3.6.3.2 Stalling

There are two occasions where the device controller may need to return to the host a STALL.

The first occasion is the functional stall, which is a condition set by the DCD as described in the USB 2.0 device framework. A functional stall is only used on non-control endpoints and can be enabled in the device controller by setting the endpoint stall bit in the USB_UOG_ENDPTCTRLx register associated with the given endpoint and the given direction. In a functional stall condition, the device controller will continue to return STALL responses to all transactions occurring on the respective endpoint and direction until the endpoint stall bit is cleared by the DCD.

A protocol stall, unlike a function stall, is used on control endpoints is automatically cleared by the device controller at the start of a new control transaction (setup phase). When enabling a protocol stall, the DCD should enable the stall bits (both directions) as a pair. A single write to the USB_UOG_ENDPTCTRLx register can ensure that both stall bits are set at the same instant.

NOTE

Any write to the USB_UOG_ENDPTCTRLx register during operational mode must preserve the endpoint type field (that is, perform a read-modify-write).

The following table shows the response matrix for the Device Controller Stall.

Table 11-65. Device Controller Stall Response Matrix

USB Packet	Endpoint Stall Bit.	Effect on STALL bit.	USB Response
SETUP packet received by a non-control endpoint.	N/A	None.	STALL
IN/OUT/PING packet received by a non-control endpoint.	'1'	None.	STALL
IN/OUT/PING packet received by a non-control endpoint.	'0'	None.	ACK/ NAK/ NYET
SETUP packet received by a control endpoint.	N/A	Cleared	ACK
IN/OUT/PING packet received by a control endpoint	'1'	None	STALL
IN/OUT/PING packet received by a control endpoint.	'0'	None.	ACK/ NAK/ NYET

11.1.3.6.3.3 Data Toggle

Data toggle is a mechanism to maintain data coherency between host and device for any given data pipe.

For more information on data toggle, refer to the USB 2.0 specification.

11.1.3.6.3.3.1 Data Toggle Reset

The DCD may reset the data toggle state bit and cause the data toggle sequence to reset in the device controller by writing a '1' to the data toggle reset bit in the USB_UOG_ENDPTCTRLx register.

This should only be necessary when configuring/initializing an endpoint or returning from a STALL condition.

11.1.3.6.3.3.2 Data Toggle Inhibit

NOTE

This feature is for test purposes only and should never be used during normal device controller operation.

Setting the *data toggle Inhibit bit* active ('1') causes the device controller to ignore the data toggle pattern that is normally sent and accept all incoming data packets regardless of the data toggle state.

In normal operation, the device controller checks the DATA0/DATA1 bit against the data toggle to determine if the packet is valid. If Data PID does not match the data toggle state bit maintained by the device controller for that endpoint, the Data toggle is considered not valid. If the data toggle is not valid, the device controller assumes the packet was already received and discards the packet (not reporting it to the DCD). To prevent the host controller from re-sending the same packet, the device controller will respond to the error packet by acknowledging it with either an ACK or NYET response.

11.1.3.6.3.3.3 Priming Transmit Endpoints

Priming a transmit endpoint will cause the device controller to fetch the device transfer descriptor (dTD) for the transaction pointed to by the device queue head (dQH).

After the dTD is fetched, it will be stored in the dQH until the device controller completes the transfer described by the dTD. Storing the dTD in the dQH allows the device controller to fetch the operating context needed to handle a request from the host without the need to follow the linked list, starting at the dQH when the host request is received.

After the device has loaded the dTD, the leading data in the packet is stored in a FIFO in the device controller. This FIFO is split into virtual channels so that the leading data can be stored for any endpoint up to the maximum number of endpoints configured at device synthesis time.

After a priming request is complete, an endpoint state of primed is indicated in the USB_UOG_ENDPTSTATUS register. For a primed transmit endpoint, the device controller can respond to an IN request from the host and meet the stringent bus turnaround time of High Speed USB.

Because only the leading data is stored in the device controller FIFO, it is necessary for the device controller to begin filling in behind leading data after the transaction starts. The FIFO must be sized to account for the maximum latency that can be incurred by the system memory bus. More information about FIFO sizing is presented in section .

11.1.3.6.3.3.4 Priming Receive Endpoints

Priming receive endpoints is identical to priming of transmit endpoints from the point of view of the DCD. At the device controller the major difference in the operational model is that there is no data movement of the leading packet data simply because the data is to be received from the host.

Note as part of the architecture, the FIFO for the receive endpoints is not partitioned into multiple channels like the transmit FIFO. Thus, the size of the RX FIFO does not scale with the number of endpoints.

11.1.3.6.4 Operational Model For Packet Transfers

All transactions on the USB bus are initiated by the host and in turn, the device must respond to any request from the host within the turnaround time stated in the USB 2.0 Specification.

At USB 1.1 Full or Low Speed rates, this turnaround time was significant and the USB 1.1 device controllers were architected so that the device controller could access main memory or interrupt a host protocol processor in order to respond to the USB 1.1 transaction. The architecture of the USB 2.0 device controller must be different because same methods will not meet USB 2.0 High-speed turnaround time requirements by simply increasing clock rate.

A USB host will send requests to the device controller in an order that can not be precisely predicted as a single pipeline, so it is not possible to prepare a single packet for the device controller to execute. However, the order of packet requests is predictable when the endpoint number and direction is considered. For example, if endpoint 3 (transmit direction) is configured as a bulk pipe, then we can expect the host will send IN

requests to that endpoint. This device controller is architected in such a way that it can prepare packets for each endpoint/direction in anticipation of the host request. The process of preparing the device controller to send or receive data in response to host initiated transaction on the bus is referred to as "priming" the endpoint. This term will be used throughout the following documentation to describe the device controller operation so the DCD can be architected properly use priming. Further, note that the term "flushing" is used to describe the action of clearing a packet that was queued for execution.

11.1.3.6.4.1 Interrupt/Bulk Endpoint Operational Model

The behaviors of the device controller for interrupt and bulk endpoints are identical.

All valid IN and OUT transactions to bulk pipes will handshake with a NAK unless the endpoint had been primed. Once the endpoint has been primed, data delivery will commence.

A dTD will be retired by the device controller when the packets described in the transfer descriptor have been completed. Each dTD describes N packets to be transferred according to the USB Variable Length transfer protocol. The formula and table on the following page describe how the device controller computes the number and length of the packets to be sent/received by the USB vary according to the total number of bytes and maximum packet length.

With Zero Length Termination (ZLT) = 0

N = INT(Number Of Bytes/Max. Packet Length) + 1

With Zero Length Termination (ZLT) = 1

N = MAXINT(Number Of Bytes/Max. Packet Length)

Table 11-66. Variable Length Transfer Protocol Example (ZLT = 0)

Bytes (dTD)	Max. Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	
512	256	3	256	256	0
512	512	2	512	0	

Table 11-67. Variable Length Transfer Protocol Example (ZLT = 1)

Bytes (dTD)	Max. Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	
512	256	2	256	256	
512	512	1	512		

NOTE

The MULT field in the dQH must be set to "00" for bulk, interrupt, and control endpoints.

TX-dTD is complete when:

- All packets described dTD were successfully transmitted. *** Total bytes in dTD will equal zero when this occurs.

RX-dTD is complete when:

- All packets described in dTD were successfully received. *** Total bytes in dTD will equal zero when this occurs.
- A short packet (number of bytes < maximum packet length) was received. *** This is a successful transfer completion; DCD must check Total Bytes in dTD to determine the number of bytes that are remaining. From the total bytes remaining in the dTD, the DCD can compute the actual bytes received.
- A long packet was received (number of bytes > maximum packet size) OR (total bytes received > total bytes specified). *** This is an error condition. The device controller will discard the remaining packet, and set the Buffer Error bit in the dTD. In addition, the endpoint will be flushed and the USBERR interrupt will become active.

On the successful completion of the packet(s) described by the dTD, the active bit in the dTD will be cleared and the next pointer will be followed when the Terminate bit is clear. When the Terminate bit is set, the device controller will flush the endpoint/direction and cease operations for that endpoint/direction.

On the unsuccessful completion of a packet (see long packet above), the dQH will be left pointing to the dTD that was in error. In order to recover from this error condition, the DCD must properly reinitialize the dQH by clearing the active bit and update the nextTD pointer before attempting to re-prime the endpoint.

NOTE

All packet level errors such as a missing handshake or CRC error will be retried automatically by the device controller.

There is no required interaction with the DCD for handling such errors.

11.1.3.6.4.1.1 Interrupt/Bulk Endpoint Bus Response Matrix

The table below shows the response matrix for Interrput/Bulk Endpoint Bus.

Table 11-68. Interrupt/Bulk Endpoint Bus Response Matrix

	Stall	Not Primed	Primed	Underflow	Overflow
Setup	Ignore	Ignore	Ignore	N/A	N/A
In	STALL	NAK	Transmit	BS Error	N/A
Out	STALL	NAK	Receive + NYET/ACK	N/A	NAK
Ping	STALL	NAK	ACK	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

NOTE

BS Error = Force Bit Stuff Error

NYET/ACK - NYET unless the Transfer Descriptor has packets remaining according to the USB variable length protocol then ACK.

SYSERR - System error should never occur when the latency FIFOs are correctly sized and the DCD is responsive.

11.1.3.6.4.2 Control Endpoint Operation Model

This section details the setup phase, data phase, status phase, and the control endpoint bus response matrix.

11.1.3.6.4.2.1 Setup Phase

All requests to a control endpoint begin with a setup phase followed by an optional data phase and a required status phase. The device controller will always accept the setup phase unless the setup lockout is engaged.

The setup lockout will engage so that future setup packets are ignored. Lockout of setup packets ensures that while software is reading the setup packet stored in the queue head, that data is not written as it is being read potentially causing an invalid setup packet.

The setup lockout mechanism can be disabled and a new tripwire type semaphore will ensure that the setup packet payload is extracted from the queue head without being corrupted be an incoming setup packet. This is the preferred behavior because ignoring repeated setup packets due to long software interrupt latency would be a compliance issue.

- Disable Setup Lockout by writing 1 to Setup Lockout Mode (SLOM) in USB Device Mode (USB_*n*USBMODE). (once at initialization). Setup lockout is not necessary when using the tripwire as described below.

NOTE

Leaving the Setup Lockout Mode As 0 will result in pre-2.3 hardware behavior.

- After receiving an interrupt and inspecting Endpoint Setup Status (USB_*n*ENDPTSETUPSTAT) to determine that a setup packet was received on a particular pipe:
  a. Write 1 to clear corresponding bit Endpoint Setup Status (USB_*n*ENDPTSETUPSTAT).
  b. Write 1 to Setup Tripwire (SUTW) in USB Command Register (USB_*n*USBCMD) register.
  c. Duplicate contents of dQH.SetupBuffer into local software byte array.
  d. Read Setup TripWire (SUTW) in USB Command Register (USB_*n*USBCMD) register. (if set - continue; if cleared - goto 2)
  e. Write 0 to clear Setup Tripwire (SUTW) in USB Command Register (USB_*n*USBCMD) register.
  f. Process setup packet using local software byte array copy and execute status/ handshake phases.

NOTE

After receiving a new setup packet the status and/or handshake phases may still be pending from a previous control sequence. These should be flushed & deallocated before linking a new status and/or handshake dTD for the most recent setup packet.

11.1.3.6.4.2.2 Data Phase

Following the setup phase, the DCD must create a device transfer descriptor for the data phase and prime the transfer.

After priming the packet, the DCD must verify a new setup packet has not been received by reading the USB.ENDPTSETUPSTAT register immediately verifying that the prime had completed. A prime will complete when the associated bit in the Endpoint Prime (USB_*n*ENDPTPRIME) register is zero and the associated bit in the Endpoint Status (USB_*n*ENDPTSTAT) register is a one. If a prime fails, ie. The Endpoint Prime (USB_*n*ENDPTPRIME) bit goes to zero and the Endpoint Status (USB_*n*ENDPTSTAT) bit is not set, then the prime has failed. This can only be due to improper setup of the dQH, dTD or a setup arriving during the prime operation. If a new setup packet is indicated after the ENDPTPRIME bit is cleared, then the transfer descriptor can be freed and the DCD must reinterpret the setup packet.

Should a setup arrive after the data stage is primed, the device controller will automatically clear the prime status (Endpoint Status (USB_*n*ENDPTSTAT)) to enforce data coherency with the setup packet.

NOTE

- The MULT field in the dQH must be set to "00" for bulk, interrupt, and control endpoints.
- Error handling of data phase packets is the same as bulk packets described previously.

11.1.3.6.4.2.3 Status Phase

Similar to the data phase, the DCD must create a transfer descriptor (with byte length equal zero) and prime the endpoint for the status phase.

The DCD must also perform the same checks of the USB.ENDPTSETUPSTAT as described above in the data phase.

NOTE

- The MULT field in the dQH must be set to 00 for bulk, interrupt, and control endpoints.
- Error handling of data phase packets is the same as bulk packets described previously.

11.1.3.6.4.2.4 Control Endpoint Bus Response Matrix

Shown in the following table is the device controller response to packets on a control endpoint according to the device controller state.

The table below shows the response matrix for the Control Endpoint Bus.

Table 11-69. Control Endpoint Bus Response Matrix

Token Type	Endpoint State					Setup Lockout
	Stall	Not Primed	Primed	Underflow	Overflow	
Setup	ACK	ACK	ACK	N/A	SYSERR	
In	STALL	NAK	Transmit	BS Error	N/A	N/A
Out	STALL	NAK	Receive + NYET/ACK	N/A	NAK	N/A
Ping	STALL	NAK	ACK	N/A	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore

BS Error = Force Bit Stuff Error

NYET/ACK - NYET unless the Transfer Descriptor has packets remaining according to the USB variable length protocol then ACK.

SYSERR - System error should never occur when the latency FIFOs are correctly sized and the DCD is responsive.

11.1.3.6.4.3 Isochronous Endpoint Operational Model

Isochronous endpoints are used for real-time scheduled delivery of data and their operational model is significantly different than the host throttled Bulk, Interrupt, and Control data pipes.

Real time delivery by the device controller will be accomplished by the following:

- Exactly MULT Packets per (micro) Frame are transmitted/received. Note: MULT is a two-bit field in the device Queue Head. The variable length packet protocol is not used on isochronous endpoints.
- NAK responses are not used. Instead, zero length packets are sent in response to an IN request to an unprimed endpoints. For unprimed RX endpoints, the response to an OUT transaction is to ignore the packet within the device controller.
- Prime requests always schedule the transfer described in the dTD for the next (micro) frame. If the ISO-dTD is still active after that frame, then the ISO-dTD will be held ready until executed or canceled by the DCD.

An EHCI compatible host controller uses the periodic frame list to schedule data exchanges to Isochronous endpoints. The operational model for device mode does not use such a data structure. Instead, the same dTD used for Control/Bulk/Interrupt endpoints is also used for isochronous endpoints. The difference is in the handling of the dTD.

The first difference between bulk and ISO-endpoints is that priming an ISO-endpoint is a delayed operation such that an endpoint will become primed only after a SOF is received. After the DCD writes the prime bit, the prime bit will be cleared as usual to indicate to software that the device controller completed a priming the dTD for transfer. Internal to the design, the device controller hardware masks that prime start until the next frame boundary. This behavior is hidden from the DCD but occurs so that the device controller can match the dTD to a specific (micro)frame.

Another difference with isochronous endpoints is that the transaction must wholly complete in a (micro)frame. Once an ISO transaction is started in a (micro)frame it will retire the corresponding dTD when MULT transactions occur or the device controller finds a fulfillment condition.

The transaction error bit set in the status field indicates a fulfillment error condition. When a fulfillment error occurs, the frame after the transfer failed to complete wholly, the device controller will force retire the ISO-dTD and move to the next ISO-dTD.

It is important to note that fulfillment errors are only caused due to partially completed packets. If no activity occurs to a primed ISO-dTD, the transaction will stay primed indefinitely. This means it is up to software discard transmit ISO-dTDs that pile up from a failure of the host to move the data.

Finally, the last difference with ISO packets is in the data level error handling. When a CRC error occurs on a received packet, the packet is not retried similar to bulk and control endpoints. Instead, the CRC is noted by setting the *Transaction Error* bit and the data is stored as usual for the application software to sort out.

- TX Packet Retired
  - MULT counter reaches zero.
  - Fulfillment Error [*Transaction Error* bit is set]
    - # Packets Occurred > 0 AND # Packets Occurred < MULT

NOTE

For TX-ISO, MULT Counter can be loaded with a lesser value in the dTD Multiplier Override field in hardware versions 2.3 and later. If the Multiplier Override is zero, the MULT Counter is initialized to the Multiplier in the QH.

- RX Packet Retired:
  - MULT counter reaches zero.
  - Non-MDATA Data PID is received**
    - ** Exit criteria only valid in hardware version 2.3 or later. Previous to hardware version 2.3, any PID sequence that did not match the MULT field exactly would be flagged as a transaction error due to PID mismatch or fulfillment error.
  - Overflow Error:
    - Packet received is > maximum packet length. [*Buffer Error* bit is set]
    - Packet received exceeds total bytes allocated in dTD. [*Buffer Error* bit is set]
  - Fulfillment Error [*Transaction Error* bit is set]
    - # Packets Occurred > 0 AND # Packets Occurred < MULT
  - CRC Error [*Transaction Error* bit is set]

NOTE

For ISO, when a dTD is retired, the next dTD is primed for the next frame. For continuous (micro)frame to (micro)frame

operation the DCD should ensure that the dTD linked-list is out ahead of the device controller by at least two (micro)frames.

11.1.3.6.4.3.1 Isochronous Pipe Synchronization

When it is necessary to synchronize an isochronous data pipe to the host, the (micro) frame number (USB_UOG_FRINDEX register) can be used as a marker.

To cause a packet transfer to occur at a specific (micro) frame number [N], the DCD should interrupt on SOF during frame N-1. When the USB_UOG_FRINDEX=N-1, the DCD must write the prime bit. The device controller will prime the isochronous endpoint in (micro) frame N-1 so that the device controller will execute delivery during (micro) frame N.

NOTE

Priming an endpoint towards the end of (micro) frame N-1 will not guarantee delivery in (micro) frame N. The delivery may actually occur in (micro) frame N+1 if device controller does not have enough time to complete the prime before the SOF for packet N is received.

11.1.3.6.4.3.2 Isochronous Endpoint Bus Response Matrix

The following table shows the response matrix for the Isochronous Endpoint Bus.

Table 11-70. Isochronous Endpoint Bus Response Matrix

	Stall	Not Primed	Primed	Underflow	Overflow
Setup	STALL	STALL	STALL	N/A	N/A
In	NULL Packet	NULL Packet	Transmit	BS Error	N/A
Out	Ignore	Ignore	Receive	N/A	Drop Packet
Ping	Ignore	Ignore	Ignore	Ignore	Ignore
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

1. BS Error = Force Bit Stuff Error

   NULL Packet = Zero Length Packet

11.1.3.6.5 Managing Queue Heads

The following figure shows the End Point Queue Head.

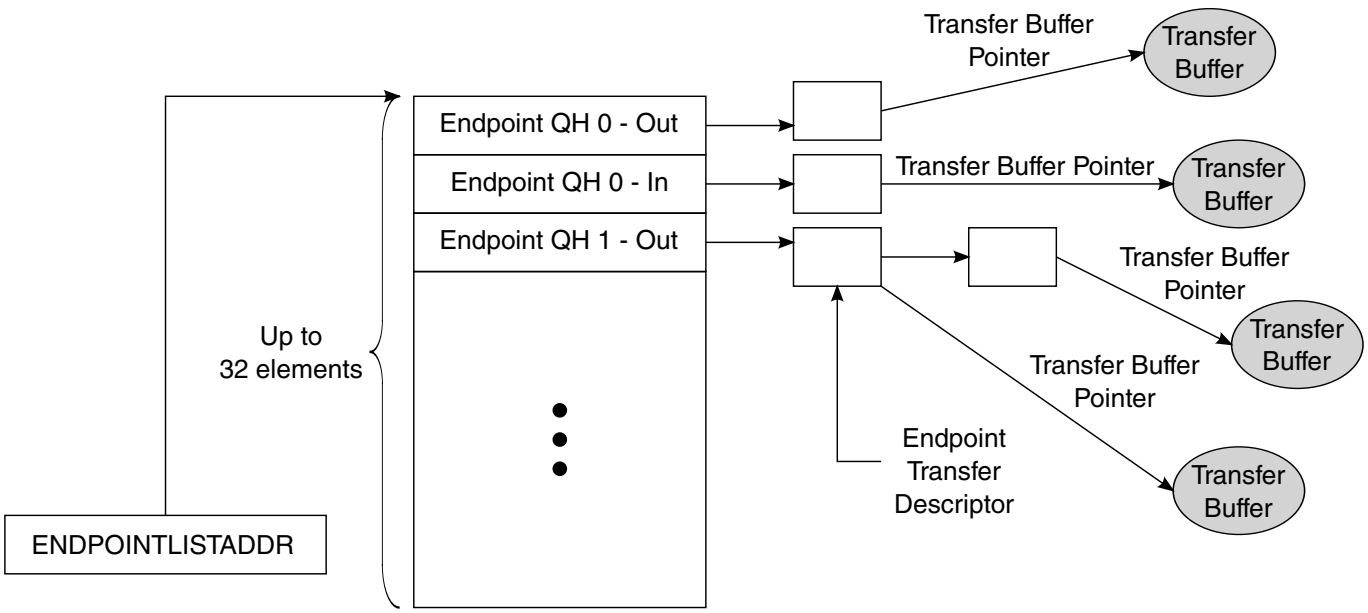


Figure 11-30. End Point Queue Head Diagram

The device queue head (dQH) points to the linked list of transfer tasks, each depicted by the device Transfer Descriptor (dTD). An area of memory pointed to by USB.ENDPOINTLISTADDR contains a group of all dQH's in a sequential list as shown in Figure 11-30. The even elements in the list of dQH's are used for receive endpoints (OUT/SETUP) and the odd elements are used for transmit endpoints (IN/INTERRUPT). Device transfer descriptors are linked head to tail starting at the queue head and ending at a terminate bit. Once the dTD has been retired, it will no longer be part of the linked list from the queue head. Therefore, software is required to track all transfer descriptors because pointers will no longer exist within the queue head once the dTD is retired (see section Software Link Pointers).

In addition to the current and next pointers and the dTD overlay examined in section Operational Model For Packet Transfers, the dQH also contains the following parameters for the associated endpoint: Multipler, Maximum Packet Length, Interrupt On Setup. The complete initialization of the dQH including these fields is demonstrated in the next section.

11.1.3.6.5.1 Queue Head Initialization

One device queue head must be initialized for each active endpoint.

To initialize a device queue head:

- Write the wMaxPacketSize field as required by the USB Chapter 9 or application specific protocol.
- Write the multiplier field to 0 for control, bulk, and interrupt endpoints. For ISO endpoints, set the multiplier to 1,2, or 3 as required bandwidth and in conjunction with the USB Chapter 9 protocol.

NOTE

In FS mode, the multiplier field can only be 1 for ISO endpoints.

- Write the next dTD Terminate bit field to 1.
- Write the Active bit in the status field to 0.
- Write the Halt bit in the status field to 0.

NOTE

The DCD must only modify dQH if the associated endpoint is not primed and there are no outstanding dTD's.

11.1.3.6.5.2 Operational Model For Setup Transfers

As discussed in section Control Endpoint Operation Model, setup transfer requires special treatment by the DCD. A setup transfer does not use a dTD but instead stores the incoming data from a setup packet in an 8-byte buffer within the dQH.

Upon receiving notification of the setup packet, the DCD should handle the setup transfer as demonstrated here:

1. Copy setup buffer contents from dQH - RX to software buffer.
2. Acknowledge setup backup by writing a "1" to the corresponding bit in ENDPTSETUPSTAT.

NOTE

- The acknowledge must occur before continuing to process the setup packet.
- After the acknowledge has occurred, the DCD must not attempt to access the setup buffer in the dQH - RX. Only the local software copy should be examined.

3. Check for pending data or status dTD's from previous control transfers and flush if any exist as discussed in section Flushing/De-priming an Endpoint.
4. Decode setup packet and prepare data phase [optional] and status phase transfer as required by the USB Chapter 9 or application specific protocol.

NOTE

It is possible for the device controller to receive setup packets before previous control transfers complete. Existing control packets in progress must be flushed and the new control packet completed.

11.1.3.6.6 Managing Transfers with Transfer Descriptors

11.1.3.6.6.1 Software Link Pointers

It is necessary for the DCD software to maintain head and tail pointers to the for the linked list of dTDs for each respective queue head.

This is necessary because the dQH only maintains pointers to the current working dTD and the next dTD to be executed. The operations described in next section for managing dTD will assume the DCD can use reference the head and tail of the dTD linked list. The following figure shows the Software Link Pointers.

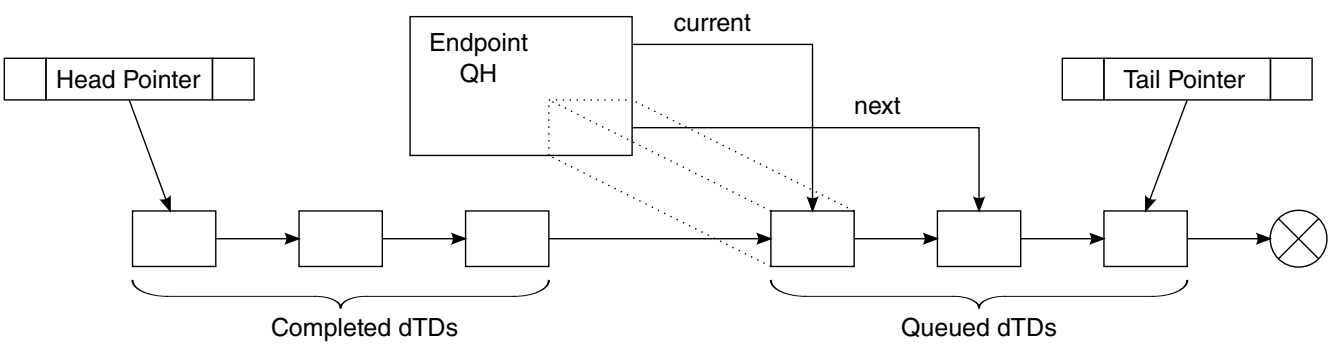


Figure 11-31. Software Link Pointers

NOTE

To conserve memory, the reserved fields at the end of the dQH can be used to store the Head & Tail pointers, but it still remains the responsibility of the DCD to maintain the pointers.

11.1.3.6.6.2 Building a Transfer Descriptor

Before a transfer can be executed from the linked list, a dTD must be built to describe the transfer.

Use the following procedure for building dTDs.

Allocate 8-DWord dTD block of memory aligned to 8-DWord boundaries. Example: bit address 4:0 would be equal to "00000"

Write the following fields:

1. Initialize first 7 DWords to 0.
2. Set the terminate bit to 1.
3. Fill in total bytes with transfer size.
4. Set the interrupt on complete if desired.
5. Initialize the status field with the active bit set to 1 and all remaining status bits set to 0.
6. Fill in buffer pointer page 0 and the current offset to point to the start of the data buffer.
7. Initialize buffer pointer page 1 through page 4 to be one greater than each of the previous buffer pointer.

11.1.3.6.6.3 Executing A Transfer Descriptor

To safely add a dTD, the DCD must be follow this procedure which will handle the event where the device controller reaches the end of the dTD list at the same time a new dTD is being added to the end of the list.

Determine whether the link list is empty: Check DCD driver to see if pipe is empty (internal representation of linked-list should indicate if any packets are outstanding).

- Case 1: Link list is empty
  a. Write dQH next pointer AND dQH terminate bit to 0 as a single DWord operation.
  b. Clear active & halt bit in dQH (in case set from a previous error).
  c. Prime endpoint by writing 1 to correct bit position in .
- Case 2: Link list is not empty
  a. Add dTD to end of linked list.
  b. Read correct prime bit in - if 1 DONE.
  c. Set ATDTW bit in USBCMD register to 1.
  d. Read correct status bit in . (store in tmp. variable for later)
  e. Read ATDTW bit in USBCMD register.
     - If 0 goto 3.
     - If 1 continue to 6.
  f. Write ATDTW bit in USBCMD register to 0.
  g. If status bit read in (3) is 1 DONE.
  h. If status bit read in (3) is 0 then Goto Case 1: Step 1.

11.1.3.6.6.4 Transfer Completion

After a dTD has been initialized and the associated endpoint primed the device controller will execute the transfer upon the host-initiated request. The DCD will be notified with a USB interrupt if the Interrupt On Complete bit was set or alternately, the DCD can poll the endpoint complete register to find when the dTD had been executed. After a dTD has been executed, DCD can check the status bits to determine success or failure.

NOTE

Multiple dTD can be completed in a single endpoint complete notification. After clearing the notification, DCD must search the dTD linked list and retire all dTDs that have finished (Active bit cleared).

By reading the status fields of the completed dTDs, the DCD can determine if the transfers completed successfully. Success is determined with the following combination of status bits:

- Active = 0
- Halted = 0
- Transaction Error = 0
- Data Buffer Error = 0

Should any combination other than the one shown above exist, the DCD must take proper action. Transfer failure mechanisms are indicated in the Device Error Matrix.

In addition to checking the status bit the DCD must read the Transfer Bytes field to determine the actual bytes transferred. When a transfer is complete, the Total Bytes transferred is by decremented by the actual bytes transferred. For Transmit packets, a packet is only complete after the actual bytes reaches zero, but for receive packets, the host may send fewer bytes in the transfer according the USB variable length packet protocol.

11.1.3.6.6.5 Flushing/De-priming an Endpoint

It is necessary for the DCD to flush to de-prime one more endpoints on a USB device reset or during a broken control transfer.

There may also be application specific requirements to stop transfers in progress. The following procedure can be used by the DCD to stop a transfer in progress:

1. Write a '1' to the corresponding bit(s) in .
2. Wait until all bits in are '0'.

- Software note: this operation may take a large amount of time depending on the USB bus activity. It is not desirable to have this wait loop within an interrupt service routine.

3. Read to ensure that for all endpoints commanded to be flushed, that the corresponding bits are now '0'. If the corresponding bits are '1' after step #2 has finished, then the flush failed as described in the following:
   - Explanation: In very rare cases, a packet is in progress to the particular endpoint when commanded flush using . A safeguard is in place to refuse the flush to ensure that the packet in progress completes successfully. The DCD may need to repeatedly flush any endpoints that fail to flush be repeating steps 1-3 until each endpoint is successfully flushed.

11.1.3.6.6.6 Device Error Matrix

The following table summarizes packet errors that are not automatically handled by the Device Controller.

Table 11-71. Device Error Matrix

Error	Direction	Packet Type	Data Buffer Error Bit	Transaction Error Bit
Overflow **	RX	Any	1	0
ISO Packet Error	RX	ISO	0	1
ISO Fulfillment Error	Both	ISO	0	1

Notice that the device controller handles all errors on Bulk/Control/Interrupt Endpoints except for a data buffer overflow. However, for ISO endpoints, errors packets are not retried and errors are tagged as indicated. The table below describes the errors.

Table 11-72. Error Descriptions

Error	Description
Overflow	Number of bytes received exceeded max. packet size or total buffer length. ** This error will also set the Halt bit in the dQH and if there are dTDs remaining in the linked list for the endpoint, then those will not be executed.
ISO Packet Error	CRC Error on received ISO packet. Contents not guaranteed to be correct.
ISO Fulfillment Error	Host failed to complete the number of packets defined in the dQH mult field within the given (micro)frame. For scheduled data delivery the DCD may need to readjust the data queue because a fulfillment error will cause Device Controller to cease data transfers on the pipe for one (micro)frame. During the "dead" (micro)frame, the Device Controller reports error on the pipe and primes for the following frame.

11.1.3.6.7 Servicing Interrupts

The interrupt service routine must consider that there are high-frequency, low-frequency operations, and error operations and order accordingly.

11.1.3.6.7.1 High-Frequency Interrupts

High frequency interrupts in particular should be handled in the order below. The most important of these is listed first because the DCD must acknowledge a setup buffer in the timeliest manner possible.

The table below describes the High frequency interrupt events.

Table 11-73. High Frequency Interrupt Events

Execution Order	Interrupt	Action
1a	USB Interrupt - USB.ENDPTSETUPSTATUS	Copy contents of setup buffer and acknowledge setup packet (as indicated in Figure 11-30 shows the End Point Queue Head). Process setup packet according to USB 2.0 Chapter 9 or application specific protocol.
1b	USB Interrupt[1] - USB.ENDPTCOMPLETE	Handle completion of dTD as indicated in Figure 11-30 shows the End Point Queue Head.
2	SOF Interrupt	Action as deemed necessary by application. This interrupt may not have a use in all applications.

1. It is likely that multiple interrupts to stack up on any call to the Interrupt Service Routine AND during the Interrupt Service Routine.

11.1.3.6.7.2 Low-Frequency Interrupts

The low frequency interrupts can be handled in any order because they do not occur often in comparison to the high-frequency interrupts.

The table below shows the Low frequency interrupt events.

Table 11-74. Low Frequency Interrupt Events

Interrupt	Action
Port Change	Change software state information.
Sleep Enable (Suspend)	Change software state information. Low power handling as necessary.
Reset Received	Change software state information. Abort pending transfers.

11.1.3.6.7.3 Error Interrupts

Error interrupts will be least frequent and should be placed last in the interrupt service routine.

The following table shows the error interrput events.

Table 11-75. Error Interrupt Events

Interrupt	Action
USB Error Interrupt	This error is redundant because it combines USB Interrupt and an error status in the dTD. The DCD will more aptly handle packet-level errors by checking dTD status field upon receipt of USB Interrupt (w/ USB.ENDPTCOMPLETE).
System Error	Unrecoverable error. Immediate Reset of core; free transfers buffers in progress and restart the DCD.

11.1.4 USB Non-Core Memory Map/Register Definition

There are two kinds of registers in the USB module: USB core registers and USB non-core registers.

USB core registers are used to control USB core functions, and more independent of USB features. Each USB controller core has its own core registers.

USB non-core registera are additional to USB core registers, and more dependent on USB features. i.MX series products vary in non-core registers.

This section describes only the USB non-core registers. For detailed descriptions of USB core registers, please refer to Register Interface .

NOTE

- For reserved bits, please preserve the value when writing (read its reset value, then write this value back)
- "USB_UOG1_", "USB_UOG2_" prefix in register name indicates it is a core register for OTG1/OTG2 controller core respectively.
- USBNC_USB_" prefix in register name indicates it is a USB non-core register.

USBNC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E4_0200	USBNC_OTG1_CTRL1	32	R/W	3000_1000h	11.1.4.1/ 2797
32E4_0204	USBNC_OTG1_CTRL2	32	R/W	0F00_0000h	11.1.4.2/ 2799
32E4_0230	USBNC_OTG1_PHY_CFG1	32	R/W	1533_71CBh	11.1.4.3/ 2802
32E4_0234	USBNC_OTG1_PHY_CFG2	32	R/W	F001_0020h	11.1.4.4/ 2806
32E4_023C	USBNC_OTG1_PHY_STATUS	32	R	0000_0000h	11.1.4.5/ 2809
32E4_0250	USBNC_ADP_CFG1	32	R/W	0000_C000h	11.1.4.6/ 2813
32E4_0254	USBNC_ADP_CFG2	32	R/W	0046_4010h	11.1.4.7/ 2815
32E4_0258	USBNC_ADP_STATUS	32	R	0000_0000h	11.1.4.8/ 2816
32E5_0200	USBNC_OTG2_CTRL1	32	R/W	3000_1000h	11.1.4.1/ 2797
32E5_0204	USBNC_OTG2_CTRL2	32	R/W	0F00_0000h	11.1.4.2/ 2799
32E5_0230	USBNC_OTG2_PHY_CFG1	32	R/W	1533_71CBh	11.1.4.3/ 2802
32E5_0234	USBNC_OTG2_PHY_CFG2	32	R/W	F001_0020h	11.1.4.4/ 2806
32E5_023C	USBNC_OTG2_PHY_STATUS	32	R	0000_0000h	11.1.4.5/ 2809

11.1.4.1 USBNC_*n*_CTRL1

The USB_n_PHY_CTRL1 register controls the integration specific features of the USB PHY modules. These features are not directly related to basic USB functionality, but control special features, interfacing on the USB ports, as well as power control and wake-up functionality.

Address: 32E4_0000h base + 200h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WIR	Reserved	WKUP_DPDM_EN	Reserved											WKUP_VBUS_EN	WKUP_ID_EN
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WKUP_SW	WKUP_SW_EN	Reserved			WIE	PWR_POL	OVER_CUR_POL	OVER_CUR_DIS	Reserved						
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

USBNC_*n*_CTRL1 field descriptions

Field	Description
31 WIR	Wake-up Interrupt Request This bit indicates that a wake-up interrupt request is received on the OTG1 port. This bit is cleared by disabling the wake-up interrupt (clearing bit "OWIE"). 1 Wake-up Interrupt Request received 0 No wake-up interrupt request received
30 Reserved	This field is reserved.
29 WKUP_DPDM_EN	Wake-up on DPDM change enable 1 (Default) DPDM changes wake-up to be enabled, it is for device only. 0 DPDM changes wake-up to be disabled only when VBUS is 0.
28–18 Reserved	This field is reserved.
17 WKUP_VBUS_EN	Wake-up on VBUS change enable 1 Enable 0 Disable
16 WKUP_ID_EN	Wake-up on ID change enable 1 Enable 0 Disable
15 WKUP_SW	Software Wake-up 1 Force wake-up 0 Inactive
14 WKUP_SW_EN	Software Wake-up Enable 1 Enable 0 Disable
13–11 Reserved	This field is reserved. Reserved. Do not write to.
10 WIE	Wake-up Interrupt Enable This bit enables or disables the OTG1 wake-up interrupt. Disabling the interrupt also clears the Interrupt request bit. Wake-up interrupt enable should be turned off after receiving a wake-up interrupt and turned on again prior to going in suspend mode 1 Interrupt Enabled 0 Interrupt Disabled
9 PWR_POL	Power Polarity This bit should be set according to PMIC Power Pin polarity. 1 PMIC Power Pin is High active. 0 PMIC Power Pin is Low active.
8 OVER_CUR_POL	Polarity of Overcurrent The polarity of OTG1/OTG2 port overcurrent event

Table continues on the next page...

USBNC_*n*_CTRL1 field descriptions (continued)

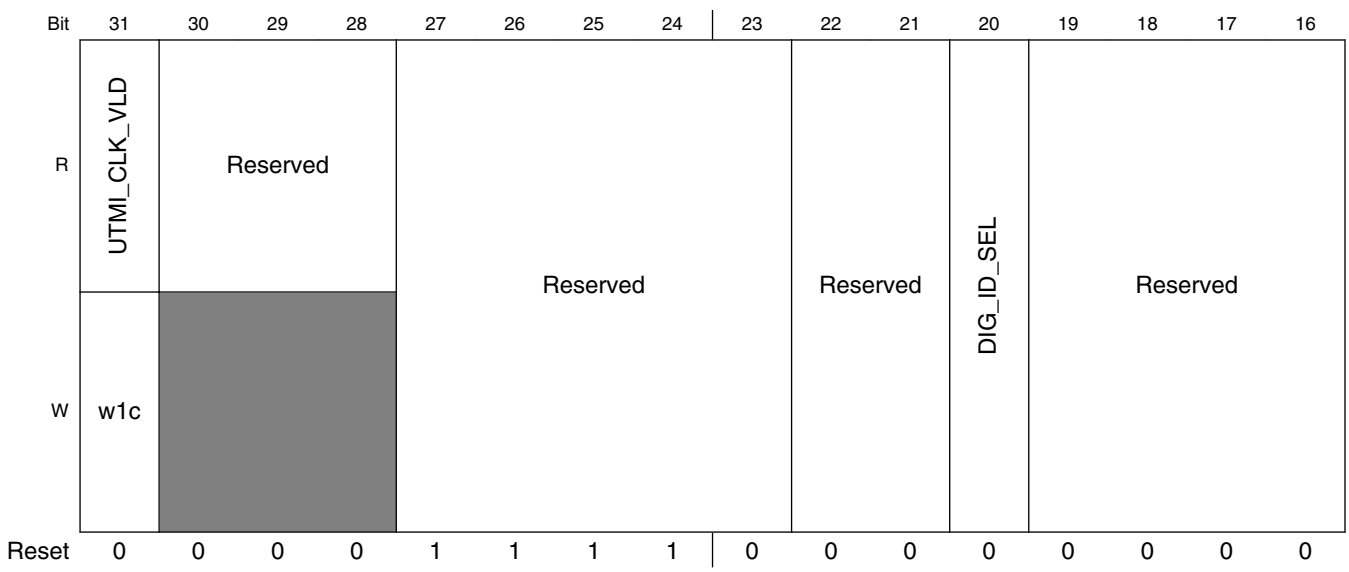
Field	Description
	1 Low active (low on this signal represents an overcurrent condition) 0 High active (high on this signal represents an overcurrent condition)
7 OVER_CUR_DIS	Disable Overcurrent Detection 1 Disables overcurrent detection 0 Enables overcurrent detection
Reserved	This field is reserved.

11.1.4.2 USBNC_*n*_CTRL2

The USB_OTGx_PHY_CTRL2 register allows observation of the UTMI Clock Valid status along with control of selected inputs to the USB OTG PHY transceiver.

Some of these input signals may be used for USB out-of-band signaling. Customers should use caution when overriding any UTMI signals since that will interfere with normal USB data communication.

Address: 32E4_0000h base + 204h offset + (65536d × i), where i=0d to 1d



Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMPULLDOWN_OVERRIDEEN	DMPULLDOWN_OVERRIDE	DPPULLDOWN_OVERRIDEEN	DPPULLDOWN_OVERRIDE	XCVRSEL_OVERRIDEEN	XCVRSEL_OVERRIDE		OPMODE_OVERRIDEEN	OPMODE_OVERRIDE		TERMSEL_OVERRIDEEN	TERMSEL_OVERRIDE	LOWSPEED_EN	AUTURESUME_EN	VBUS_SOURCE_SEL	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBNC_*n*_CTRL2 field descriptions

Field	Description
31 UTMI_CLK_VLD	Indicate whether PHY clock is valid. Write 1 to clear 0 UTMI clock to USB PHY is not toggling (Default) 1 UTMI clock to USB PHY has toggled several times
30–28 Reserved	This field is reserved.
27–23 Reserved	This field is reserved. 0B11110 Default
22–21 Reserved	This field is reserved. Customers should not change the value of this bit field.
20 DIG_ID_SEL	Selects whether the USB OTG ID pin function is connected to the PHY's USB_OTG*_ID pin or a GPIO pin. The ID pin state is observed on USBNC_n_PHY_STATUS[ID_DIG]. 0 Use the USB_OTG*_ID pin for the USB OTG ID pin detection function(default) 1 Use the pin configured by the IOMUXC_USB_OTG*_ID_SELECT_INPUT register for the USB OTG ID pin detection function
19–16 Reserved	This field is reserved. Customers should not change the value of this bit field.
15 DMPULLDOWN_OVERRIDEEN	Enables override of control of the DM pulldown resistor instead of the state set by the USB controller. 0 USB controller enables/disables the DM pulldown resistor in the USB PHY. 1 Use the value set by the USBNC_n_CTRL2[DMPULLDOWN_OVERRIDE] bit field to enable/disable the DM pulldown resistor in the USB PHY.
14 DMPULLDOWN_OVERRIDE	Controls state of DM pulldown resistor if USBNC_n_CTRL2[DMPULLDOWN_OVERRIDEEN] is set to 1'b1. 0 DM pulldown resistor disabled 1 DM pulldown resistor enabled

Table continues on the next page...

USBNC_*n*_CTRL2 field descriptions (continued)

Field	Description
13 DPPULLDOWN_ OVERRIDEEN	Enables override of control of the DP pulldown resistor instead of using the state set by the USB controller. 0 USB controller enables/disables the DP pulldown resistor in the USB PHY. 1 Use the value set by the USBNC_n_CTRL2[DPPULLDOWN_OVERRIDE] bit field to enable/disable the DP pulldown resistor in the USB PHY.
12 DPPULLDOWN_ OVERRIDE	Controls state of DP pulldown resistor if USBNC_n_CTRL2[DPPULLDOWN_OVERRIDEEN] is set to 1'b1. 0 DP pulldown resistor disabled 1 DP pulldown resistor enabled
11 XCVRSEL_ OVERRIDEEN	Enables override of control of the UTMI XcvrSelect signals to the USB OTG PHY instead of using the state normally set by the USB controller. 0 The state of the UTMI XcvrSelect signals to the USB PHY is set by the USB controller. 1 The state of the UTMI XcvrSelect signals to the USB PHY is set by the values in the USBNC_x_CTRL2[XCVRSEL_OVERRIDE] bit field.
10–9 XCVRSEL_ OVERRIDE	Controls the state of the UTMI XcvrSelect signals to the USB OTG PHY if the value of USBNC_x_CTRL2[XCVRSEL_OVERRIDEEN] is set to 1'b1. See the *UTMI+ Specification* for descriptions of the functions of the XcvrSelect signals.
8 OPMODE_ OVERRIDEEN	Enables override of control of the UTMI OpMode signals to the USB OTG PHY instead of using the state normally set by the USB controller. 0 The state of the UTMI OpMode signals to the USB PHY is set by the USB controller. 1 The state of the UTMI OpMode signals to the USB PHY is set by the values in the USBNC_x_CTRL2[OPMODE_OVERRIDE] bit field.
7–6 OPMODE_ OVERRIDE	Controls the state of the UTMI XcvrSelect signals to the USB OTG PHY if the value of USBNC_x_CTRL2[OPMODE_OVERRIDEEN] is set to 1'b1. See the *UTMI+ Specification* for descriptions of the functions of the OpMode signals.
5 TERMSEL_ OVERRIDEEN	Enables override of control of the UTMI TermSelect signal to the USB OTG PHY instead of using the state normally set by the USB controller. 0 The state of the UTMI TermSelect signal to the USB PHY is set by the USB controller. 1 The state of the UTMI TermSelect signal to the USB PHY is set by the value in the USBNC_x_CTRL2[TERMSEL_OVERRIDE] bit field.
4 TERMSEL_ OVERRIDE	Controls the state of the UTMI TermSelect signal to the USB OTG PHY if the value of USBNC_x_CTRL2[TERMSEL_OVERRIDEEN] is set to 1'b1. See the *UTMI+ Specification* for descriptions of the function of the TermSelect signal.
3 LOWSPEED_EN	Set if AUTURESUME_EN is set and works on low speed. 0 Default
2 AUTURESUME_ EN	Auto Resume Enable This bit field is for UTMI OTG PHY host mode only (UH core does not have this feature since HSIC PHY does not support auto resume). To set USB, it will send resume with 32 KHz clock when it detects remote wakeup from device. This feature is useful if device drives a very short remote wakeup (minimal value is 1 ms for USB2 spec) and system 24 MHz is off during suspend mode. 0 Default

Table continues on the next page...

USBNC_*n*_CTRL2 field descriptions (continued)

Field	Description
VBUS_SOURCE_SEL	VBUS source select used to detect a VBUS wakeup event. This bit field is for UTMI OTG PHY only (UH core and HSIC PHY have no such feature). 2'b00 vbus_valid others sess_valid

11.1.4.3 USB OTG PHY Configuration Register 1 (USBNC_*n*_PHY_CFG1)

The USB_OTGx_PHY_CFG1 register allows control of selected inputs to the USB OTG PHY.

Most of these signals are used for parametric tuning of the USB transceiver functions.

One signal is used to allow persistent control of the USB_OTG1_CHD_B external pin.

Address: 32E4_0000h base + 230h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CHRGDET_Megamix	TXPREEMPPULSETUNE0	TXPREEMPAMPTUNE0		TXRESTUNE0		TXRISETUNE0		TXVREFTUNE0				TXFSLSTUNE0			
Reset	0	0	0	1	0	1	0	1	0	0	1	1	0	0	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	0	TXHSXVTUNE0		OTGTUNE0			SQRXTUNE0		COMPDISTUNE0				FSEL			COMMONONN
Reset	0	1	1	1	0	0	0	1	1	1	0	0	1	0	1	1

USBNC_*n*_PHY_CFG1 field descriptions

Field	Description
31 CHRGDET_Megamix	USB_OTG1_CHD_B output control This signal controls the state of the USB_OTG1_CHD_B external pin together with the CHRGDET (USBNC_USB_OTG1_PHY_STATUS[29] signal during periods when the SOC is fully powered. If either CHRGDET_Megamix or CHRGDET are set to 1'b1, then USB_OTG1_CHD_B will be pulled low. If both CHRGDET_Megamix and CHRGDET are set to 1'b0, then USB_OTG1_CHD_B will be pulled high by its external open-drain pull-up resistor.

Table continues on the next page...

USBNC_*n*_PHY_CFG1 field descriptions (continued)

Field	Description
	Note: The instance of this bit field for the USB OTG2 PHY does not produce results on an external pin. The R/W register bit may still be used as a memory location to store results of Battery Charger detection tests for USB OTG2 PHY. 0 The external state of USB_OTG1_CHD_B is only controlled by the state of the CHRGDET signal 1 The external state of USB_OTG1_CHD_B is forced low
30 TXPREEMPPULSETUNE0	HS Transmitter Pre-Emphasis Duration Control This signal controls the duration for which the HS pre-emphais current is sourced on the the USB_OTG*_DP and USB_OTG*DN pins. The HS Transmitter pre-emphasis duration is defined in terms of unit amounts. One unit of pre-emphasis duration is approximately 580ps and is defined as 1X pre-emphasis duration. This signal is valid only if TXPREEMPAMPTUNE (USBNC_USB_OTGx_PHY_CFG1[29:28]) is NOT set to 2'b00. 0 2X, long pre-emphasis current duration (design default) 1 1X, short pre-emphasis current duration
29–28 TXPREEMPAMPTUNE0	HS Treansmitter Pre-Emphasis Current Control This signal controls the amount of current sourced to the USB_OTG*_DP and USB_OTG*_DN pins after a J-to-K or K-to-J transition. The HS Transmitter pre-emphasis current is defined in terms of unit amounts. One unit amount is approximately 600uA and is defined as 1X pre-emphasis current. 00 HS Transmitter pre-emphasis is disabled 01 HS Transmitter pre-emphasis circuit sources 1X pre-emphasis current (design default) 10 HS Transmitter pre-emphasis circuit sources 2X pre-emphasis current 11 HS Transmitter pre-emphasis circuit sources 3X pre-emphasis current
27–26 TXRESTUNE0	USB Source Impedance Adjustment In some applications, there can be significant series resistance on the USB DP/DN path between the USB_OTG*_DP/USB_OTG*_DN pins tns and the USB cable. This bus adjusts the driver source impedance to compensate for that added resistance. **Note:** Any setting other than the design default can cause HS/FS signal integrity changes along with variation across process, voltage, and temperature conditions that affect compliance with USB 2.0 specification limits. 00 Source impedance is increased by approximately 1.5 Ω 01 Design default 10 Source impedance is decreased by approximately 2 Ω 11 Source impedance is decreased by approximately 4 Ω
25–24 TXRISETUNE0	HS Transmitter Rise/Fall Time Adjustment This bus adjust the rise/fall times of the high-speed transmitter waveform. 00 -10% 01 Design default 10 +15% 11 +20%
23–20 TXVREFTUNE0	HS DC Voltage Level Adjustment This bus adjust the high-speed transmitter DC level voltage.

Table continues on the next page...

USBNC_*n*_PHY_CFG1 field descriptions (continued)

Field	Description
	0000 -6% 0001 -4% 0010 -2% 0011 Design default 0100 +2% 0101 +4% 0110 +6% 0111 +8% 1000 +10% 1001 +12% 1010 +14% 1011 +16% 1100 +18% 1101 +20% 1110 +22% 1111 +24%
19–16 TXFSLSTUNE0	FS/LS Source Impedance Adjustment This bus adjusts the low- and full-speed single-ended source impedance while driving high. The adjustment values listed are based on nominal process, voltage, and temperature conditions. 0000 +5% 0001 +2.5% 0011 Design default 0111 -2.5% 1111 -5% All other bit settings are reserved and should not be used.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–13 TXHSXVTUNE0	Transmitter High-Speed Crossover Adjustment This bus adjusts the voltage at which the USB_OTG*_DP and USB_OTG*_DN signals cross while transmitting in HS mode. 00 Reserved 01 -15mV 10 +15mV 11 Design default
12–10 OTGTUNE0	VBUS Valid Threshold Adjustment This bus adjust the voltage level for the VBUS VALID threshold. 000 -6% 001 -4.5% 010 -3% 011 -1.5% 100 Design default 101 +1.5%

Table continues on the next page...

USBNC_*n*_PHY_CFG1 field descriptions (continued)

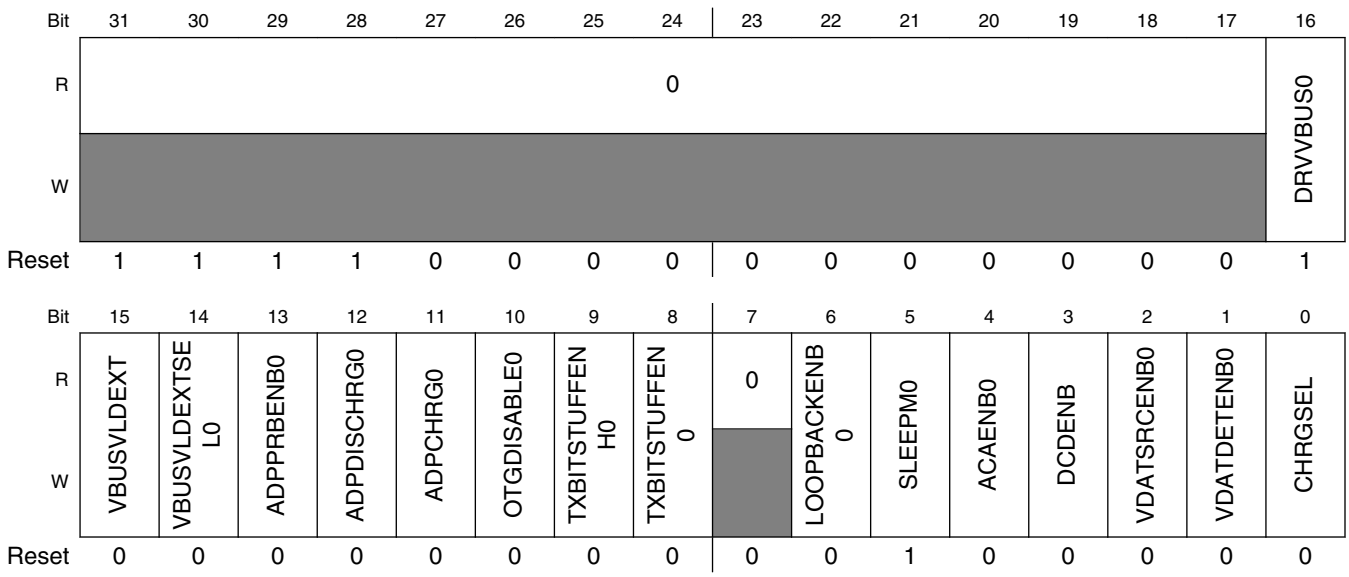
Field	Description
	110 +3% 111 +4.5%
9–7 SQRXTUNE0	Squelch Threshold Adjustment This bus adjusts the voltage level for the receiver threshold used to detect valid high-speed data. 000 +15% 001 +10% 010 +5% 011 Design default 100 -5% 101 -10% 110 -15% 111 -20%
6–4 COMPDISTUNE0	Disconnect Threshold Adjustment This bus adjusts the voltage level for the receiver threshold used to detect a disconnect event at the host. 000 -6% 001 -4.5% 010 -3% 011 -1.5% 100 Design default 101 +1.5% 110 +3% 111 +4.5%
3–1 FSEL	Reference Clock Frequency Select This signal selects the USB OTG PHY reference clock frequency for the USB PLL. **Note:** This SOC is only configured for a 24MHz clock to be routed to the USB OTG PHY PLL clock input. Customers should not change the value of this bit field. 000 9.6 MHz 001 10 MHz 010 12 MHz 011 19.2 MHz 100 20 MHz 101 24 MHz (only valid setting for this SOC) 110 Reserved 111 50 MHz
0 COMMONONN	Common Block Power-Down Control This signal controls the power-down signals in the Bias and PLL blocks when the USB OTG PHY is in Suspend or Sleep modes. 0 In Suspend or Sleep modes, the Bias and PLL blocks remain powered 1 In Suspend or Sleep modes, the Bias and PLL blocks are powered down 0

11.1.4.4 USB OTG PHY Configuration Register 2 (USBNC_*n*_PHY_CFG2)

The USB_OTGx_PHY_CFG2 register allows control of selected inputs to the USB OTG PHY.

Most of these signals are used for configuration of VBUS detection, ADP (Attach Detection Protocol), and Battery Charging ACA (Accesory Charging Adaptor) functions.

Address: 32E4_0000h base + 234h offset + (65536d × i), where i=0d to 1d



USBNC_*n*_PHY_CFG2 field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 DRVVBUS0	VBUS Valid Comparator Enable This signal controls the USB OTG PHY VBUS Valid comparator which indicates whether the voltage on the USB_OTG*_VBUS pin is below the VBUS Valid threshold. The VBUS Valid threshold is nominally 4.75V on this USB PHY. The VBUS Valid threshold can be adjusted using the USBNC_OTGn_PHY_CFG1[OTGTUNE0] bit field. Status of the VBUS Valid comparator, when it is enabled, is reported on the USBNC_OTGn_PHY_STATUS[VBUS_VLD] bit. When OTGDISABLE0 (USBNC_USB_OTGx_PHY_CFG2[10])is set to 1'b0 and DRVVBUS0 is set to 1'b1, the Bandgap circuitry and VBUS Valid comparator are powered, even in Suspend or Sleep mode. DRVVBUS0 should be reset to 1'b0 when the internal VBUS Valid comparator is not required, to reduce quiescent current in Suspend or Sleep mode. 0 The VBUS Valid comparator is disabled 1 The VBUS Valid comparator is enabled
15 VBUSVLDEXT	External VBUS Valid Indicator

Table continues on the next page...

USBNC_*n*_PHY_CFG2 field descriptions (continued)

Field	Description
	This signal is used if a VBUS Valid signal generated from a circuit outside the USB OTG PHY is desired. If the USB OTG PHY is configured for Device mode and VBUSVLDEXTSEL0 (USBNC_USB_OTGx_PHY_CFG2[14] is set to 1'b1, VBUSVLD indicated whether the VBUS signal on the USB cable is valid and enables the pull-up resistor on USB_OTG1_DP. This signal has no effect when the USB OTG PHY is configured for Host mode. 0 The VBUS signal sensed outside the USB OTG PHY is not valid, and the pull-up resistor on USB_OTG*_DP is disabled 1 The VBUS signal sensed outside the USB OTG PHY is valid, and the pull-up resistor on USB_OTG*_DP is enabled
14 VBUSVLDEXTSEL0	External VBUS Valid Select This signal selects whether the VBUSVLDEXT input (USBNC_USB_OTGx_PHY_CFG2[15]) or the internal Session Valid comparator in the USB OTG PHY is used to check if the VBUS signal on USB_OTG*_VBUS is valid and to enable the pull-up resistor on the USB_OTG*_DP pin. The activation of the pull-up resistor on the USB_OTG*_DP pin also depends on the state of the XCVRSEL, OPMODE, TERMSEL, DPPULLDOWN, and DMPULLDOWN signals in the UTMI bus per the UTMI+ specification. 0 The USB OTG PHY internal Session Valid comparator is used to enable the pull-up resistor on the USB_OTG*_DP pin 1 The VBUSVLDEXT signal is used to enable the pull-up resistor on the USB_OTG*_DP pin
13 ADPPRBENB0	ADP Probe Enable This signal determines whether the ADP (Attach Detection Protocol) Probe comparator on the USB_OTG*_VBUS pin is enabled. The ADP Probe threshold is > 0.6V and < 0.75V on this USB PHY. 0 ADP Probe comparator is disabled 1 ADP Probe comparator is enabled
12 ADPDISCHRG0	VBUS Input ADP Discharge Enable This signal controls discharging the USB_OTG*_VBUS pin during ADP. The I_{ADP_SINK} current is >1.1mA and <2.0mA. 0 Disables discharging USB_OTG*_VBUS during ADP 1 Enables discharging USB_OTG*_VBUS during ADP
11 ADPCHRG0	VBUS Input ADP Charge Enable This signal controls charging the USB_OTG*_VBUS pin during ADP. The I_{ADP_SRC} current is >1.1mA and <1.65mA. 0 Disables charging USB_OTG*_VBUS during ADP 1 Disables charging USB_OTG*_VBUS during ADP
10 OTGDISABLE0	OTG Block Disable This signal powers down the VBUS Valid comparator. It does not control power to the Session Valid comparator, ADP Probe and Sense comparators, or the ID detection circuitry.

Table continues on the next page...

USBNC_*n*_PHY_CFG2 field descriptions (continued)

Field	Description
	0 The OTG block is powered up 1 The OTG block is powered down
9 TXBITSTUFFENH0	High-Byte Transmit Bit-Stuffing Enable This controller signal controls bit stuffing on DATAINH[7:0] when OPMODE[1:0] = 2'b11. For more information on the use of this setting see UTMI+ Specification Section 2.2.1.12 . 0 Bit stuffing is disabled 1 Bit stuffing is enabled
8 TXBITSTUFFEN0	Low-Byte Transmit Bit-Stuffing Enable This controller signal controls bit stuffing on DATAIN[7:0] when OPMODE[1:0] = 2'b11. For more information on the use of this setting see UTMI+ Specification Section 2.2.1.12 . 0 Bit stuffing is disabled 1 Bit stuffing is enabled
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 LOOPBACKENB0	Loopback Test Enable This signal places the USB OTG PHY in Loopback mode, which enables the receive and transmit logic concurrently. Loopback mode is for test purposes only; it cannot be used for normal operation. 0 During data transmission, the receive logic is disabled 1 During data transmission, the receive logic is enabled
5 SLEEPM0	Sleep Mode Assertion Asserting this signal places the USB OTG PHY in Sleep mode accourding to the USB 2.0 Link Power Management (LPM) addendum to the USB 2.0 specification. In Sleep mode, the transmitter is tri-stated and the USB OTG PHY circuits used in this SOC are powered down. **Note:**The USB controller used in this SOC does not support LPM operation. Leave this bit set to 1'b1 for normal operation. 0 Sleep mode 1 Normal operating mode
4 ACAENB0	ACA USB_OTG*_ID Pin Resistance Detection Enable This signal enables the circuitry to detect resistance on the USB_OTG*_ID pin if the USB cable is connected to an ACA (Acessory Charger Adaptor). When ACAENB0 is set to 1'b1, IDPULLUP0 (USB_OTGSC[26]) is overridden and set to 1'b0 internally to the USB OTG PHY. The RID* results are reported in USBNC_USB_OTGx_PHY_STATUS[28:24]. 0 Disables detection of resistance on the USB_OTG*_ID pin 1 Enables detection of resistance on the USB_OTG*_ID pin
3 DCDENB	Data Contact Detection Enable This signal enables current sourcing on the USB_OTG*_DP pin and pull-down resistaince on the USB_OTG*_DN pin for Data Contact Detect (DCD) per the USB Battery Charging Specification revision 1.2.

Table continues on the next page...

USBNC_*n*_PHY_CFG2 field descriptions (continued)

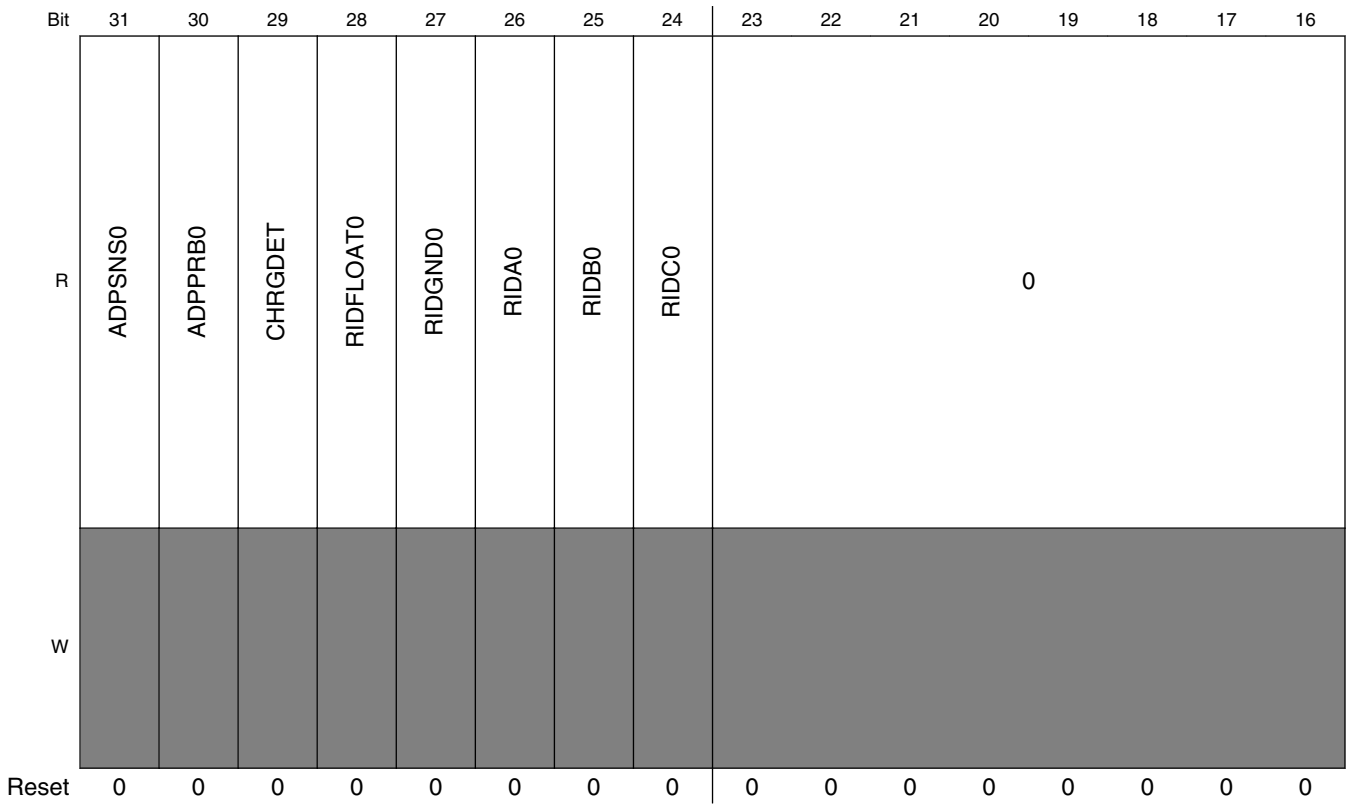
Field	Description
	Note: During normal USB operation of the USB OTG PHY, set this signal to 1'b0. 0 I_{DP_SRC} current and R_{DM_DWN} pull-down resistance are disabled 1 I_{DP_SRC} current and R_{DM_DWN} pull-down resistance are enabled
2 VDATSRCENB0	Battery Charging Source Select This signal is used to enable the VD*_SRC voltage source and ID*SINK current sink per the USB Battery Charging Specification revision 1.2. Pin selection for the voltage source and the current sink are controlled by USBNC_USB_OTGx_PHY_CFG2[0]. **Note:** During normal USB operation of the USB OTG PHY, set this signal to 1'b0. 0 VD*_SRC and ID*_SINK are disabled 1 VD*_SRC and ID*_SINK are enabled
1 VDATDETENB0	Battery Charging Attach / Connect Detection Enable This signal is used to enable attach/connect detection per the USB Battery Charging Specification revision 1.2. Results are reported by using the single-ended receiver status in USBNC_USB_OTGx_PHY_STATUS[1:0]. **Note:** During normal USB operation of the USB OTG PHY, set this signal to 1'b0. 0 IDP_SRC current source compliant to VLGC_HI connected to USB_OTG*_DP pin is disabled 1 IDP_SRC current source compliant to VLGC_HI connected to USB_OTG*_DP pin is enabled
0 CHRGSEL	Battery Charging Source Select This signal is used to select the Battery Charging detection connections per the USB Battery Charging Specification revision 1.2. It determines which of USB_OTG*_DP / USB_OTG*_DN has the VD*_SRC voltage source and which has the ID*_SINK current sink. Results are reported in USBNC_USB_OTGx_PHY_STATUS[29]. **Note:** During normal USB operation of the USB OTG PHY, set this signal to 1'b0. 0 VDP_SRC is connected to USB_OTG*_DP and IDM_SINK is connected to USB_OTG*_DN. Used for Primary Detection. 1 VDM_SRC is connected to USB_OTG*_DN and IDP_SINK is connected to USB_OTG*_DP. Used for Secondary Detection.

11.1.4.5 USB OTG PHY Status Register (USBNC_*n*_PHY_STATUS)

The USB_OTGx_PHY_STATUS register allows visibility of selected outputs of the USB OTG PHY.

These signals are used for some VBUS detection, ADP (Attach Detection Protocol), and Battery Charging ACA (Accesory Charging Adaptor) functions.

Address: 32E4_0000h base + 23Ch offset + (65536d × i), where i=0d to 1d



USBNC_*n*_PHY_STATUS field descriptions

Field	Description
31 ADPSNS0	ADP Sense Indicator This signal is output from the USB OTG PHY ADP Sense comparator and indicates whether the voltage on the USB_OTG*_VBUS pin is below the ADP sensing voltage. 0 The voltage on USB_OTG*_VBUS is below the ADP sensing voltage 1 The voltage on USB_OTG*_VBUS is above the ADP sensing voltage
30 ADPPRB0	ADP Probe Indicator This signal is output from the USB OTG PHY ADP Probe comparator and indicates whether the voltage on the USB_OTG*_VBUS pin is below the ADP probing voltage. 0 The voltage on USB_OTG*_VBUS is below the ADP probing voltage 1 The voltage on USB_OTG*_VBUS is above the ADP probing voltage
29 CHRGDET	Battery Charger Detection Output This signal indicates whether the voltage level on USB_OTG*_DP or USB_OTG*_DN is greater than VDAT_REF as defined in the USB Battery Charger, Revision 1.2 Specification. The selection of which USB data pins have the VD*_SRC and the ID*_SINK sources is controlled by register bit USBNC_USB_OTGx_CFG2[0]. The CHRGDET register bit is only valid when USBNC_USB_OTGx_CFG2[2:1] is set to 2'b11. 0 VD* < VDAT_REF 1 VD* > VDAT_REF
28 RIDFLOAT0	ACA USB_OTG*_ID Pin Resistance Indicator For USB Charger Detection, indicates whether the USB_OTG*_ID pin connected to an ACA is floating (has resistance > R_{ID_FLOAT}). 0 ACA OTG_ID pin resistance is ≤ R_{ID_A} (max) 1 ACA OTG_ID pin resistance is ≥ R_{ID_FLOAT} (min)
27 RIDGND0	ACA USB_OTG*_ID Pin Resistance Indicator For USB Charger Detection, indicates whether the USB_OTG*_ID pin connected to an ACA is grounded (has resistance < R_{ID_GND}). 0 ACA OTG_ID pin resistance is ≥ R_{ID_C} (min) 1 ACA OTG_ID pin resistance is ≤ R_{ID_GND} (max)
26 RIDA0	ACA USB_OTG*_ID Pin Resistance Indicator For USB Charger Detection, indicates whether the USB_OTG*_ID pin connected to an ACA is within the R_{ID_A}) range. 0 ACA OTG_ID pin resistance is ≥ R_{ID_FLOAT} (min) and ≤ R_{ID_B} max 1 ACA OTG_ID pin resistance is ≥ R_{ID_A} (min) and ≤ R_{ID_A} max
25 RIDB0	ACA USB_OTG*_ID Pin Resistance Indicator For USB Charger Detection, indicates whether the USB_OTG*_ID pin connected to an ACA is within the R_{ID_B}) range. 0 ACA OTG_ID pin resistance is ≥ R_{ID_A} (min) and ≤ R_{ID_C} max 1 ACA OTG_ID pin resistance is ≥ R_{ID_B} (min) and ≤ R_{ID_B} max

Table continues on the next page...

USBNC_*n*_PHY_STATUS field descriptions (continued)

Field	Description
24 RIDC0	ACA USB_OTG*_ID Pin Resistance Indicator For USB Charger Detection, indicates whether the USB_OTG*_ID pin connected to an ACA is within the R_{ID_C}) range. 0 ACA OTG_ID pin resistance is ≥ R_{ID_B} (min) and ≤ R_{ID_GND} max 1 ACA OTG_ID pin resistance is ≥ R_{ID_C} (min) and ≤ R_{ID_C} max
23–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 HOST_ DISCONNECT	Peripheral Disconnect Indicator This signal indicates to the USB_OTGx controller that a peripheral has been disconnected from or connected to the USB cable. In high-speed host operation, HOST_DISCONNECT is updated only at the end of SOF packet transmission. In FS and LS host operations, HOST_DISCONNECT is updated according to UTMI+ Specification, Section 2.2.1.10. This output is valid only when the USB OTG PHY and associated USB controller are configured to Host mode by setting USBC_n_USBMODE[1:0] to 2'b11. 0 Peripheral is connected 1 No peripheral is connected
4 ID_DIG	Micro- or Mini- A/B Plug Indicator This signal is the output from the ID pin A/B detector, which determines whether the resistance from the USB_OTG*_ID pin to VSS is grounded (10 Ω maximum for a Micro- or Mini-A plug) or floating (1 MegΩ minimum for a Micro- or Mini- B plug). The USB OTG PHY internal ID pin A/B detector circuit is enabled or disabled depending on the states of register bits USBC_n_OTGSC[5] and USBNC_USB_OTGx_PHY_CFG2[4]. Selection of whether to use the USB OTG PHY ID pin A/B detector or a GPIO pin ID detector is determined by the state of the USBNC_n_CTRL2[DIG_ID_SEL] bit field. 0 The connnected plug is a Micro- or Mini-A plug 1 The connnected plug is a Micro- or Mini-B plug
3 VBUS_VLD	VBUS Valid Indicator from USB OTG PHY This signal is the output from the USB OTG PHY VBUS Valid comparator and indicates whether the voltage on the USB_OTG*_VBUS pin is below the VBUS Valid threshold. The threshold is 4.75V but can be adjusted with OTGTUNE0 (USBNC_USB_PHY_CFG1[12:10]). The USB OTG PHY VBUS Valid comparator is enabled or disabled depending on the states of register bits USBNC_USB_OTGx_PHY_CFG2[10] and USBNC_USB_OTGx_PHY_CFG2[16]. 0 The voltage on USB_OTG*_VBUS is below the VBUS Valid threshold 1 The voltage on USB_OTG*_VBUS is above the VBUS Valid threshold
2 SESS_VLD	OTG Device Session Valid Indicator from USB OTG PHY This signal is the output from the USB OTG PHY Session Valid comparator and indicates whether the voltage on the USB_OTG*_VBUS pin is below the OTG Device Session Valid threshold. The Session Valid threshold is > 0.8V and < 2.0V on this USB PHY. 0 The voltage on USB_OTG*_VBUS is below the OTG Device Session Valid threshold 1 The voltage on USB_OTG*_VBUS is above the OTG Device Session Valid threshold

Table continues on the next page...

USBNC_*n*_PHY_STATUS field descriptions (continued)

Field	Description
LINE_STATE	Line State Indicator outputs from USB OTG PHY These signals reflect the state of the single-ended receivers as defined by the UTMI+ Specification. In Suspend or Sleep mode, this bus is a combinatatorial output (directly reflecting the current state of DN and DP, respectively. During normal high-speed packet transfers, the line indicates a high-speed J state. 00 SE0 (DP low, DN low) 01 J state for high-speed and full-speed USB traffic; K state for low-speed USB traffic (DP high, DN low) 10 K state for high-speed and full-speed USB traffic; J state for low-speed USB traffic (DP low, DN high) 11 SE1 (DP high, DN high)

11.1.4.6 USBNC_ADP_CFG1

Address: 32E4_0000h base + 250h offset = 32E4_0250h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								ADP_PRB_EN	ADP_PRB_INT_EN	ADP_SNS_INT_EN	TIMER_EN	Reserved		ADP_WAIT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

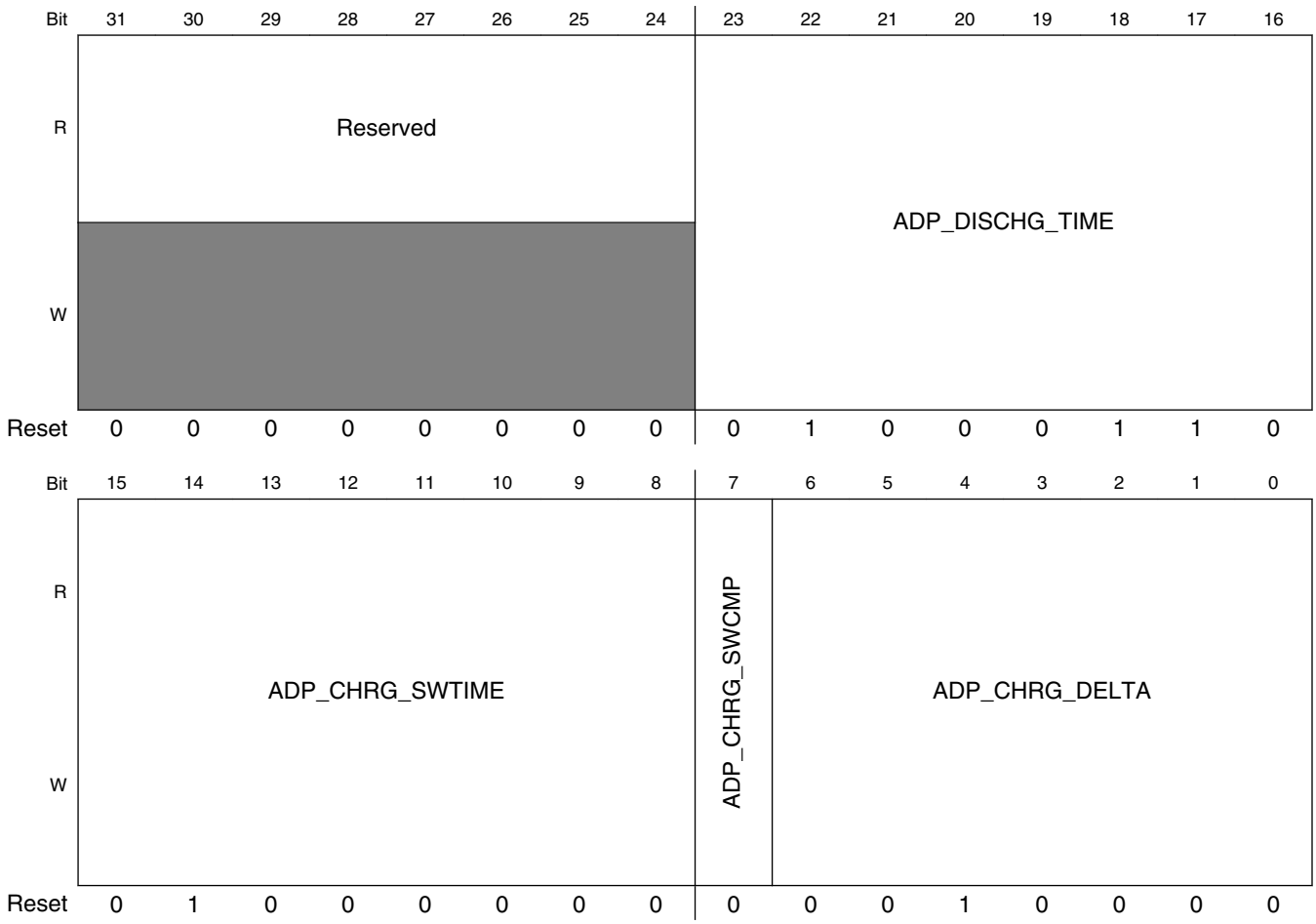
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADP_WAIT															
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBNC_ADP_CFG1 field descriptions

Field	Description
31–24 Reserved	This field is reserved. Normal read/write able register 8'b0 Default
23 ADP_PRB_EN	Set to enable the ADP probe sequence 0 Default
22 ADP_PRB_INT_EN	ADP probe interrupt enable 0 Default
21 ADP_SNS_INT_EN	ADP Sense Interrupt Enable 0 Default
20 TIMER_EN	ADP Timer Test Enable Set will start the internal 18-bit counter. 0 Default
19–18 Reserved	This field is reserved.
ADP_WAIT	Delay between 2 ADP probe, default value is about 1.5 s in 32 KHz. 18'hc000 Default

11.1.4.7 USBNC_ADP_CFG2

Address: 32E4_0000h base + 254h offset = 32E4_0254h



USBNC_ADP_CFG2 field descriptions

Field	Description
31–24 Reserved	This field is reserved. Normal read/write able register 8'b0 Default
23–16 ADP_DISCHG_TIME	ADP Discharge time Default value is about 2.2 ms in 32 KHz. 8'h46 Default
15–8 ADP_CHRG_SWTIME	ADP charge time assigned by software 8'h40 Default
7 ADP_CHRG_SWCMP	If set, HW uses ADP_CHRG_SWTIME to compare. If clear, HW compares current charge time with n-2 (see otg2 spec).

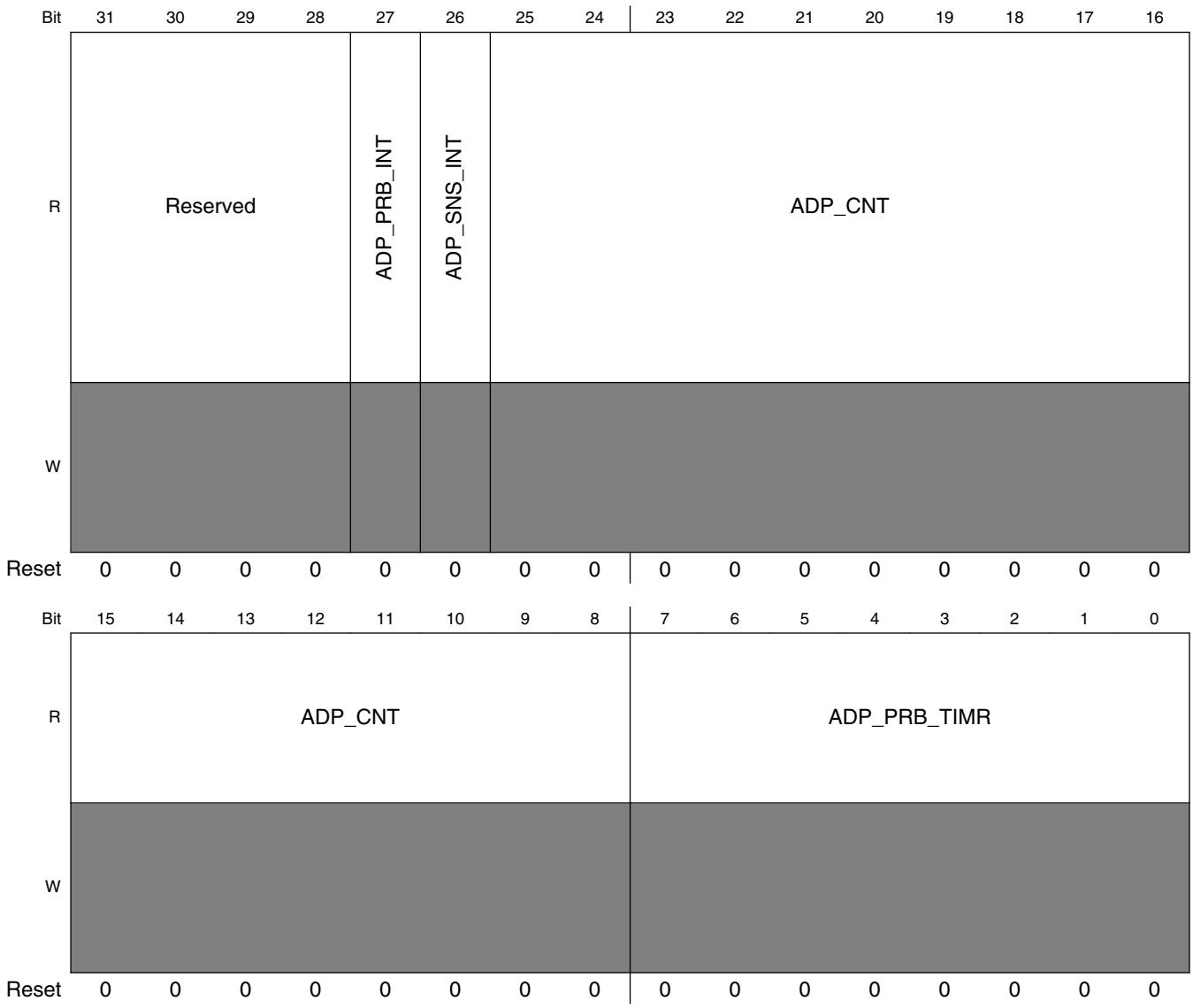
Table continues on the next page...

USBNC_ADP_CFG2 field descriptions (continued)

Field	Description
	0 Default
ADP_CHRG_ DELTA	ADP charge time compare If diffence is larger than ADP_CHG_DELTA, it generates ADP_PROBE interrupt. 7'h10 Default

11.1.4.8 USBNC_ADP_STATUS

Address: 32E4_0000h base + 258h offset = 32E4_0258h



USBNC_ADP_STATUS field descriptions

Field	Description
31–28 Reserved	This field is reserved.
27 ADP_PRB_INT	ADP Probe Interrupt Status Set when ADP charge time exceed [ADP_CHG_MIN, ADP_CHG_MAX] cleared by CLEAR ADP_PRB_INT_EN 0 Default
26 ADP_SNS_INT	ADP Sense Interrupt Status Set when PHY detect ADP sense cleared by clear ADP_SNS_INT_EN 0 Default
25–8 ADP_CNT	ADP Internal 18-bit Counter It counts when ADP_PRB_EN or TIMER_TEST_EN is set cleared if both ADP_PRB_EN and TIMER_TEST_EN are 0, or internal state machine change internal state machine: IDLE When ADP_PRB_EN is 0, change to DSCH when ADP_PRB_EN is set. DSCH ADP is discharging, change to CHRG when timer reach ADP_DSCHG_TIME. CHRG ADP is charging, change to WAIT when PHY indicate ADP_PROBE. WAIT Wait for next probe sequence. Change to DSCH when timer reach ADP_WAIT_TIME.
ADP_PRB_TIMR	ADP Probe Time It latchs the internal counter when PHY indicates ADP_PROBE. 0 Default

11.1.5 USB Core Memory Map/Register Definition

USB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E4_0000	Identification register (USB_OTG1_UOG1_ID)	32	R	E401_FA05h	11.1.5.1/ 2826
32E4_0004	Hardware General (USB_OTG1_UOG1_HWGENERAL)	32	R	0000_0015h	11.1.5.2/ 2826
32E4_0008	Host Hardware Parameters (USB_OTG1_UOG1_HWHOST)	32	R	1002_0001h	11.1.5.3/ 2828
32E4_000C	Device Hardware Parameters (USB_OTG1_UOG1_HWDEVICE)	32	R	0000_0011h	11.1.5.4/ 2828
32E4_0010	TX Buffer Hardware Parameters (USB_OTG1_UOG1_HWTXBUF)	32	R	8008_0B08h	11.1.5.5/ 2829
32E4_0014	RX Buffer Hardware Parameters (USB_OTG1_UOG1_HWRXBUF)	32	R	0000_0808h	11.1.5.6/ 2830

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E4_0080	General Purpose Timer #0 Load (USB_OTG1_UOG1_GPTIMER0LD)	32	R/W	0000_0000h	11.1.5.7/ 2830
32E4_0084	General Purpose Timer #0 Controller (USB_OTG1_UOG1_GPTIMER0CTRL)	32	R/W	0000_0000h	11.1.5.8/ 2831
32E4_0088	General Purpose Timer #1 Load (USB_OTG1_UOG1_GPTIMER1LD)	32	R/W	0000_0000h	11.1.5.9/ 2832
32E4_008C	General Purpose Timer #1 Controller (USB_OTG1_UOG1_GPTIMER1CTRL)	32	R/W	0000_0000h	11.1.5.10/ 2833
32E4_0090	System Bus Config (USB_OTG1_UOG1_SBUSCFG)	32	R/W	0000_0002h	11.1.5.11/ 2834
32E4_0100	Capability Registers Length (USB_OTG1_UOG1_CAPLENGTH)	8	R	40h	11.1.5.12/ 2835
32E4_0102	Host Controller Interface Version (USB_OTG1_UOG1_HCIVERSION)	16	R	0100h	11.1.5.13/ 2835
32E4_0104	Host Controller Structural Parameters (USB_OTG1_UOG1_HCSPARAMS)	32	R	0001_0011h	11.1.5.14/ 2836
32E4_0108	Host Controller Capability Parameters (USB_OTG1_UOG1_HCCPARAMS)	32	R	0000_0006h	11.1.5.15/ 2838
32E4_0120	Device Controller Interface Version (USB_OTG1_UOG1_DCIVERSION)	16	R	0001h	11.1.5.16/ 2840
32E4_0124	Device Controller Capability Parameters (USB_OTG1_UOG1_DCCPARAMS)	32	R	0000_0188h	11.1.5.17/ 2840
32E4_0140	USB Command Register (USB_OTG1_UOG1_USBCMD)	32	R/W	0008_0000h	11.1.5.18/ 2842
32E4_0144	USB Status Register (USB_OTG1_UOG1_USBSTS)	32	R/W	0000_0000h	11.1.5.19/ 2846
32E4_0148	Interrupt Enable Register (USB_OTG1_UOG1_USBINTR)	32	R/W	0000_0000h	11.1.5.20/ 2850
32E4_014C	USB Frame Index (USB_OTG1_UOG1_FRINDEX)	32	R/W	0000_0000h	11.1.5.21/ 2852
32E4_0154	Frame List Base Address (USB_OTG1_UOG1_PERIODICLISTBASE)	32	R/W	0000_0000h	11.1.5.22/ 2853
32E4_0154	Device Address (USB_OTG1_UOG1_DEVICEADDR)	32	R/W	0000_0000h	11.1.5.23/ 2853
32E4_0158	Next Asynch. Address (USB_OTG1_UOG1_ASYNCLISTADDR)	32	R/W	0000_0000h	11.1.5.24/ 2854
32E4_0158	Endpoint List Address (USB_OTG1_UOG1_ENDPTLISTADDR)	32	R/W	0000_0000h	11.1.5.25/ 2855
32E4_0160	Programmable Burst Size (USB_OTG1_UOG1_BURSTSIZE)	32	R/W	0000_0000h	11.1.5.26/ 2855
32E4_0164	TX FIFO Fill Tuning (USB_OTG1_UOG1_TXFILLTUNING)	32	R/W	0000_0808h	11.1.5.27/ 2856
32E4_0178	Endpoint NAK (USB_OTG1_UOG1_ENDPTNAK)	32	R/W	0000_0000h	11.1.5.28/ 2858

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E4_017C	Endpoint NAK Enable (USB_OTG1_UOG1_ENDPTNAKEN)	32	R/W	0000_0000h	11.1.5.29/ 2858
32E4_0180	Configure Flag Register (USB_OTG1_UOG1_CONFIGFLAG)	32	R/W	0000_0001h	11.1.5.30/ 2859
32E4_0184	Port Status & Control (USB_OTG1_UOG1_PORTSC1)	32	R/W	1000_0000h	11.1.5.31/ 2859
32E4_01A4	On-The-Go Status & control (USB_OTG1_UOG1_OTGSC)	32	R/W	0000_0120h	11.1.5.32/ 2866
32E4_01A8	USB Device Mode (USB_OTG1_UOG1_USBMODE)	32	R/W	0000_0000h	11.1.5.33/ 2870
32E4_01AC	Endpoint Setup Status (USB_OTG1_UOG1_ENDPTSETUPSTAT)	32	R/W	0000_0000h	11.1.5.34/ 2871
32E4_01B0	Endpoint Prime (USB_OTG1_UOG1_ENDPTPRIME)	32	R/W	0000_0000h	11.1.5.35/ 2872
32E4_01B4	Endpoint Flush (USB_OTG1_UOG1_ENDPTFLUSH)	32	R/W	0000_0000h	11.1.5.36/ 2873
32E4_01B8	Endpoint Status (USB_OTG1_UOG1_ENDPTSTAT)	32	R	0000_0000h	11.1.5.37/ 2873
32E4_01BC	Endpoint Complete (USB_OTG1_UOG1_ENDPTCOMPLETE)	32	R/W	0000_0000h	11.1.5.38/ 2874
32E4_01C0	Endpoint Control0 (USB_OTG1_UOG1_ENDPTCTRL0)	32	R/W	0080_0080h	11.1.5.39/ 2875
32E4_01C4	Endpoint Control 1 (USB_OTG1_UOG1_ENDPTCTRL1)	32	R/W	0000_0000h	11.1.5.40/ 2877
32E4_01C8	Endpoint Control 2 (USB_OTG1_UOG1_ENDPTCTRL2)	32	R/W	0000_0000h	11.1.5.41/ 2880
32E4_01CC	Endpoint Control 3 (USB_OTG1_UOG1_ENDPTCTRL3)	32	R/W	0000_0000h	11.1.5.42/ 2882
32E4_01D0	Endpoint Control 4 (USB_OTG1_UOG1_ENDPTCTRL4)	32	R/W	0000_0000h	11.1.5.43/ 2885
32E4_01D4	Endpoint Control 5 (USB_OTG1_UOG1_ENDPTCTRL5)	32	R/W	0000_0000h	11.1.5.44/ 2888
32E4_01D8	Endpoint Control 6 (USB_OTG1_UOG1_ENDPTCTRL6)	32	R/W	0000_0000h	11.1.5.45/ 2891
32E4_01DC	Endpoint Control 7 (USB_OTG1_UOG1_ENDPTCTRL7)	32	R/W	0000_0000h	11.1.5.46/ 2894
32E5_0000	Identification register (USB_OTG1_UOG2_ID)	32	R	E401_FA05h	11.1.5.1/ 2826
32E5_0004	Hardware General (USB_OTG1_UOG2_HWGENERAL)	32	R	0000_0015h	11.1.5.2/ 2826
32E5_0008	Host Hardware Parameters (USB_OTG1_UOG2_HWHOST)	32	R	1002_0001h	11.1.5.3/ 2828
32E5_000C	Device Hardware Parameters (USB_OTG1_UOG2_HWDEVICE)	32	R	0000_0011h	11.1.5.4/ 2828

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E5_0010	TX Buffer Hardware Parameters (USB_OTG1_UOG2_HWTXBUF)	32	R	8008_0B08h	11.1.5.5/ 2829
32E5_0014	RX Buffer Hardware Parameters (USB_OTG1_UOG2_HWRXBUF)	32	R	0000_0808h	11.1.5.6/ 2830
32E5_0080	General Purpose Timer #0 Load (USB_OTG1_UOG2_GPTIMER0LD)	32	R/W	0000_0000h	11.1.5.7/ 2830
32E5_0084	General Purpose Timer #0 Controller (USB_OTG1_UOG2_GPTIMER0CTRL)	32	R/W	0000_0000h	11.1.5.8/ 2831
32E5_0088	General Purpose Timer #1 Load (USB_OTG1_UOG2_GPTIMER1LD)	32	R/W	0000_0000h	11.1.5.9/ 2832
32E5_008C	General Purpose Timer #1 Controller (USB_OTG1_UOG2_GPTIMER1CTRL)	32	R/W	0000_0000h	11.1.5.10/ 2833
32E5_0090	System Bus Config (USB_OTG1_UOG2_SBUSCFG)	32	R/W	0000_0002h	11.1.5.11/ 2834
32E5_0100	Capability Registers Length (USB_OTG1_UOG2_CAPLENGTH)	8	R	40h	11.1.5.12/ 2835
32E5_0102	Host Controller Interface Version (USB_OTG1_UOG2_HCIVERSION)	16	R	0100h	11.1.5.13/ 2835
32E5_0104	Host Controller Structural Parameters (USB_OTG1_UOG2_HCSPARAMS)	32	R	0001_0011h	11.1.5.14/ 2836
32E5_0108	Host Controller Capability Parameters (USB_OTG1_UOG2_HCCPARAMS)	32	R	0000_0006h	11.1.5.15/ 2838
32E5_0120	Device Controller Interface Version (USB_OTG1_UOG2_DCIVERSION)	16	R	0001h	11.1.5.16/ 2840
32E5_0124	Device Controller Capability Parameters (USB_OTG1_UOG2_DCCPARAMS)	32	R	0000_0188h	11.1.5.17/ 2840
32E5_0140	USB Command Register (USB_OTG1_UOG2_USBCMD)	32	R/W	0008_0000h	11.1.5.18/ 2842
32E5_0144	USB Status Register (USB_OTG1_UOG2_USBSTS)	32	R/W	0000_0000h	11.1.5.19/ 2846
32E5_0148	Interrupt Enable Register (USB_OTG1_UOG2_USBINTR)	32	R/W	0000_0000h	11.1.5.20/ 2850
32E5_014C	USB Frame Index (USB_OTG1_UOG2_FRINDEX)	32	R/W	0000_0000h	11.1.5.21/ 2852
32E5_0154	Frame List Base Address (USB_OTG1_UOG2_PERIODICLISTBASE)	32	R/W	0000_0000h	11.1.5.22/ 2853
32E5_0154	Device Address (USB_OTG1_UOG2_DEVICEADDR)	32	R/W	0000_0000h	11.1.5.23/ 2853
32E5_0158	Next Asynch. Address (USB_OTG1_UOG2_ASYNCLISTADDR)	32	R/W	0000_0000h	11.1.5.24/ 2854
32E5_0158	Endpoint List Address (USB_OTG1_UOG2_ENDPTLISTADDR)	32	R/W	0000_0000h	11.1.5.25/ 2855
32E5_0160	Programmable Burst Size (USB_OTG1_UOG2_BURSTSIZE)	32	R/W	0000_0000h	11.1.5.26/ 2855

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E5_0164	TX FIFO Fill Tuning (USB_OTG1_UOG2_TXFILLTUNING)	32	R/W	0000_0808h	11.1.5.27/ 2856
32E5_0178	Endpoint NAK (USB_OTG1_UOG2_ENDPTNAK)	32	R/W	0000_0000h	11.1.5.28/ 2858
32E5_017C	Endpoint NAK Enable (USB_OTG1_UOG2_ENDPTNAKEN)	32	R/W	0000_0000h	11.1.5.29/ 2858
32E5_0180	Configure Flag Register (USB_OTG1_UOG2_CONFIGFLAG)	32	R/W	0000_0001h	11.1.5.30/ 2859
32E5_0184	Port Status & Control (USB_OTG1_UOG2_PORTSC1)	32	R/W	1000_0000h	11.1.5.31/ 2859
32E5_01A4	On-The-Go Status & control (USB_OTG1_UOG2_OTGSC)	32	R/W	0000_0120h	11.1.5.32/ 2866
32E5_01A8	USB Device Mode (USB_OTG1_UOG2_USBMODE)	32	R/W	0000_0000h	11.1.5.33/ 2870
32E5_01AC	Endpoint Setup Status (USB_OTG1_UOG2_ENDPTSETUPSTAT)	32	R/W	0000_0000h	11.1.5.34/ 2871
32E5_01B0	Endpoint Prime (USB_OTG1_UOG2_ENDPTPRIME)	32	R/W	0000_0000h	11.1.5.35/ 2872
32E5_01B4	Endpoint Flush (USB_OTG1_UOG2_ENDPTFLUSH)	32	R/W	0000_0000h	11.1.5.36/ 2873
32E5_01B8	Endpoint Status (USB_OTG1_UOG2_ENDPTSTAT)	32	R	0000_0000h	11.1.5.37/ 2873
32E5_01BC	Endpoint Complete (USB_OTG1_UOG2_ENDPTCOMPLETE)	32	R/W	0000_0000h	11.1.5.38/ 2874
32E5_01C0	Endpoint Control0 (USB_OTG1_UOG2_ENDPTCTRL0)	32	R/W	0080_0080h	11.1.5.39/ 2875
32E5_01C4	Endpoint Control 1 (USB_OTG1_UOG2_ENDPTCTRL1)	32	R/W	0000_0000h	11.1.5.40/ 2877
32E5_01C8	Endpoint Control 2 (USB_OTG1_UOG2_ENDPTCTRL2)	32	R/W	0000_0000h	11.1.5.41/ 2880
32E5_01CC	Endpoint Control 3 (USB_OTG1_UOG2_ENDPTCTRL3)	32	R/W	0000_0000h	11.1.5.42/ 2882
32E5_01D0	Endpoint Control 4 (USB_OTG1_UOG2_ENDPTCTRL4)	32	R/W	0000_0000h	11.1.5.43/ 2885
32E5_01D4	Endpoint Control 5 (USB_OTG1_UOG2_ENDPTCTRL5)	32	R/W	0000_0000h	11.1.5.44/ 2888
32E5_01D8	Endpoint Control 6 (USB_OTG1_UOG2_ENDPTCTRL6)	32	R/W	0000_0000h	11.1.5.45/ 2891
32E5_01DC	Endpoint Control 7 (USB_OTG1_UOG2_ENDPTCTRL7)	32	R/W	0000_0000h	11.1.5.46/ 2894
32E5_0000	Identification register (USB_OTG2_UOG1_ID)	32	R	E401_FA05h	11.1.5.1/ 2826
32E5_0004	Hardware General (USB_OTG2_UOG1_HWGENERAL)	32	R	0000_0015h	11.1.5.2/ 2826

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E5_0008	Host Hardware Parameters (USB_OTG2_UOG1_HWHOST)	32	R	1002_0001h	11.1.5.3/ 2828
32E5_000C	Device Hardware Parameters (USB_OTG2_UOG1_HWDEVICE)	32	R	0000_0011h	11.1.5.4/ 2828
32E5_0010	TX Buffer Hardware Parameters (USB_OTG2_UOG1_HWTXBUF)	32	R	8008_0B08h	11.1.5.5/ 2829
32E5_0014	RX Buffer Hardware Parameters (USB_OTG2_UOG1_HWRXBUF)	32	R	0000_0808h	11.1.5.6/ 2830
32E5_0080	General Purpose Timer #0 Load (USB_OTG2_UOG1_GPTIMER0LD)	32	R/W	0000_0000h	11.1.5.7/ 2830
32E5_0084	General Purpose Timer #0 Controller (USB_OTG2_UOG1_GPTIMER0CTRL)	32	R/W	0000_0000h	11.1.5.8/ 2831
32E5_0088	General Purpose Timer #1 Load (USB_OTG2_UOG1_GPTIMER1LD)	32	R/W	0000_0000h	11.1.5.9/ 2832
32E5_008C	General Purpose Timer #1 Controller (USB_OTG2_UOG1_GPTIMER1CTRL)	32	R/W	0000_0000h	11.1.5.10/ 2833
32E5_0090	System Bus Config (USB_OTG2_UOG1_SBUSCFG)	32	R/W	0000_0002h	11.1.5.11/ 2834
32E5_0100	Capability Registers Length (USB_OTG2_UOG1_CAPLENGTH)	8	R	40h	11.1.5.12/ 2835
32E5_0102	Host Controller Interface Version (USB_OTG2_UOG1_HCIVERSION)	16	R	0100h	11.1.5.13/ 2835
32E5_0104	Host Controller Structural Parameters (USB_OTG2_UOG1_HCSPARAMS)	32	R	0001_0011h	11.1.5.14/ 2836
32E5_0108	Host Controller Capability Parameters (USB_OTG2_UOG1_HCCPARAMS)	32	R	0000_0006h	11.1.5.15/ 2838
32E5_0120	Device Controller Interface Version (USB_OTG2_UOG1_DCIVERSION)	16	R	0001h	11.1.5.16/ 2840
32E5_0124	Device Controller Capability Parameters (USB_OTG2_UOG1_DCCPARAMS)	32	R	0000_0188h	11.1.5.17/ 2840
32E5_0140	USB Command Register (USB_OTG2_UOG1_USBCMD)	32	R/W	0008_0000h	11.1.5.18/ 2842
32E5_0144	USB Status Register (USB_OTG2_UOG1_USBSTS)	32	R/W	0000_0000h	11.1.5.19/ 2846
32E5_0148	Interrupt Enable Register (USB_OTG2_UOG1_USBINTR)	32	R/W	0000_0000h	11.1.5.20/ 2850
32E5_014C	USB Frame Index (USB_OTG2_UOG1_FRINDEX)	32	R/W	0000_0000h	11.1.5.21/ 2852
32E5_0154	Frame List Base Address (USB_OTG2_UOG1_PERIODICLISTBASE)	32	R/W	0000_0000h	11.1.5.22/ 2853
32E5_0154	Device Address (USB_OTG2_UOG1_DEVICEADDR)	32	R/W	0000_0000h	11.1.5.23/ 2853
32E5_0158	Next Asynch. Address (USB_OTG2_UOG1_ASYNCLISTADDR)	32	R/W	0000_0000h	11.1.5.24/ 2854

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E5_0158	Endpoint List Address (USB_OTG2_UOG1_ENDPTLISTADDR)	32	R/W	0000_0000h	11.1.5.25/ 2855
32E5_0160	Programmable Burst Size (USB_OTG2_UOG1_BURSTSIZE)	32	R/W	0000_0000h	11.1.5.26/ 2855
32E5_0164	TX FIFO Fill Tuning (USB_OTG2_UOG1_TXFILLTUNING)	32	R/W	0000_0808h	11.1.5.27/ 2856
32E5_0178	Endpoint NAK (USB_OTG2_UOG1_ENDPTNAK)	32	R/W	0000_0000h	11.1.5.28/ 2858
32E5_017C	Endpoint NAK Enable (USB_OTG2_UOG1_ENDPTNAKEN)	32	R/W	0000_0000h	11.1.5.29/ 2858
32E5_0180	Configure Flag Register (USB_OTG2_UOG1_CONFIGFLAG)	32	R/W	0000_0001h	11.1.5.30/ 2859
32E5_0184	Port Status & Control (USB_OTG2_UOG1_PORTSC1)	32	R/W	1000_0000h	11.1.5.31/ 2859
32E5_01A4	On-The-Go Status & control (USB_OTG2_UOG1_OTGSC)	32	R/W	0000_0120h	11.1.5.32/ 2866
32E5_01A8	USB Device Mode (USB_OTG2_UOG1_USBMODE)	32	R/W	0000_0000h	11.1.5.33/ 2870
32E5_01AC	Endpoint Setup Status (USB_OTG2_UOG1_ENDPTSETUPSTAT)	32	R/W	0000_0000h	11.1.5.34/ 2871
32E5_01B0	Endpoint Prime (USB_OTG2_UOG1_ENDPTPRIME)	32	R/W	0000_0000h	11.1.5.35/ 2872
32E5_01B4	Endpoint Flush (USB_OTG2_UOG1_ENDPTFLUSH)	32	R/W	0000_0000h	11.1.5.36/ 2873
32E5_01B8	Endpoint Status (USB_OTG2_UOG1_ENDPTSTAT)	32	R	0000_0000h	11.1.5.37/ 2873
32E5_01BC	Endpoint Complete (USB_OTG2_UOG1_ENDPTCOMPLETE)	32	R/W	0000_0000h	11.1.5.38/ 2874
32E5_01C0	Endpoint Control0 (USB_OTG2_UOG1_ENDPTCTRL0)	32	R/W	0080_0080h	11.1.5.39/ 2875
32E5_01C4	Endpoint Control 1 (USB_OTG2_UOG1_ENDPTCTRL1)	32	R/W	0000_0000h	11.1.5.40/ 2877
32E5_01C8	Endpoint Control 2 (USB_OTG2_UOG1_ENDPTCTRL2)	32	R/W	0000_0000h	11.1.5.41/ 2880
32E5_01CC	Endpoint Control 3 (USB_OTG2_UOG1_ENDPTCTRL3)	32	R/W	0000_0000h	11.1.5.42/ 2882
32E5_01D0	Endpoint Control 4 (USB_OTG2_UOG1_ENDPTCTRL4)	32	R/W	0000_0000h	11.1.5.43/ 2885
32E5_01D4	Endpoint Control 5 (USB_OTG2_UOG1_ENDPTCTRL5)	32	R/W	0000_0000h	11.1.5.44/ 2888
32E5_01D8	Endpoint Control 6 (USB_OTG2_UOG1_ENDPTCTRL6)	32	R/W	0000_0000h	11.1.5.45/ 2891
32E5_01DC	Endpoint Control 7 (USB_OTG2_UOG1_ENDPTCTRL7)	32	R/W	0000_0000h	11.1.5.46/ 2894

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E6_0000	Identification register (USB_OTG2_UOG2_ID)	32	R	E401_FA05h	11.1.5.1/ 2826
32E6_0004	Hardware General (USB_OTG2_UOG2_HWGENERAL)	32	R	0000_0015h	11.1.5.2/ 2826
32E6_0008	Host Hardware Parameters (USB_OTG2_UOG2_HWHOST)	32	R	1002_0001h	11.1.5.3/ 2828
32E6_000C	Device Hardware Parameters (USB_OTG2_UOG2_HWDEVICE)	32	R	0000_0011h	11.1.5.4/ 2828
32E6_0010	TX Buffer Hardware Parameters (USB_OTG2_UOG2_HWTXBUF)	32	R	8008_0B08h	11.1.5.5/ 2829
32E6_0014	RX Buffer Hardware Parameters (USB_OTG2_UOG2_HWRXBUF)	32	R	0000_0808h	11.1.5.6/ 2830
32E6_0080	General Purpose Timer #0 Load (USB_OTG2_UOG2_GPTIMER0LD)	32	R/W	0000_0000h	11.1.5.7/ 2830
32E6_0084	General Purpose Timer #0 Controller (USB_OTG2_UOG2_GPTIMER0CTRL)	32	R/W	0000_0000h	11.1.5.8/ 2831
32E6_0088	General Purpose Timer #1 Load (USB_OTG2_UOG2_GPTIMER1LD)	32	R/W	0000_0000h	11.1.5.9/ 2832
32E6_008C	General Purpose Timer #1 Controller (USB_OTG2_UOG2_GPTIMER1CTRL)	32	R/W	0000_0000h	11.1.5.10/ 2833
32E6_0090	System Bus Config (USB_OTG2_UOG2_SBUSCFG)	32	R/W	0000_0002h	11.1.5.11/ 2834
32E6_0100	Capability Registers Length (USB_OTG2_UOG2_CAPLENGTH)	8	R	40h	11.1.5.12/ 2835
32E6_0102	Host Controller Interface Version (USB_OTG2_UOG2_HCIVERSION)	16	R	0100h	11.1.5.13/ 2835
32E6_0104	Host Controller Structural Parameters (USB_OTG2_UOG2_HCSPARAMS)	32	R	0001_0011h	11.1.5.14/ 2836
32E6_0108	Host Controller Capability Parameters (USB_OTG2_UOG2_HCCPARAMS)	32	R	0000_0006h	11.1.5.15/ 2838
32E6_0120	Device Controller Interface Version (USB_OTG2_UOG2_DCIVERSION)	16	R	0001h	11.1.5.16/ 2840
32E6_0124	Device Controller Capability Parameters (USB_OTG2_UOG2_DCCPARAMS)	32	R	0000_0188h	11.1.5.17/ 2840
32E6_0140	USB Command Register (USB_OTG2_UOG2_USBCMD)	32	R/W	0008_0000h	11.1.5.18/ 2842
32E6_0144	USB Status Register (USB_OTG2_UOG2_USBSTS)	32	R/W	0000_0000h	11.1.5.19/ 2846
32E6_0148	Interrupt Enable Register (USB_OTG2_UOG2_USBINTR)	32	R/W	0000_0000h	11.1.5.20/ 2850
32E6_014C	USB Frame Index (USB_OTG2_UOG2_FRINDEX)	32	R/W	0000_0000h	11.1.5.21/ 2852
32E6_0154	Frame List Base Address (USB_OTG2_UOG2_PERIODICLISTBASE)	32	R/W	0000_0000h	11.1.5.22/ 2853

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E6_0154	Device Address (USB_OTG2_UOG2_DEVICEADDR)	32	R/W	0000_0000h	11.1.5.23/ 2853
32E6_0158	Next Asynch. Address (USB_OTG2_UOG2_ASYNCLISTADDR)	32	R/W	0000_0000h	11.1.5.24/ 2854
32E6_0158	Endpoint List Address (USB_OTG2_UOG2_ENDPTLISTADDR)	32	R/W	0000_0000h	11.1.5.25/ 2855
32E6_0160	Programmable Burst Size (USB_OTG2_UOG2_BURSTSIZE)	32	R/W	0000_0000h	11.1.5.26/ 2855
32E6_0164	TX FIFO Fill Tuning (USB_OTG2_UOG2_TXFILLTUNING)	32	R/W	0000_0808h	11.1.5.27/ 2856
32E6_0178	Endpoint NAK (USB_OTG2_UOG2_ENDPTNAK)	32	R/W	0000_0000h	11.1.5.28/ 2858
32E6_017C	Endpoint NAK Enable (USB_OTG2_UOG2_ENDPTNAKEN)	32	R/W	0000_0000h	11.1.5.29/ 2858
32E6_0180	Configure Flag Register (USB_OTG2_UOG2_CONFIGFLAG)	32	R/W	0000_0001h	11.1.5.30/ 2859
32E6_0184	Port Status & Control (USB_OTG2_UOG2_PORTSC1)	32	R/W	1000_0000h	11.1.5.31/ 2859
32E6_01A4	On-The-Go Status & control (USB_OTG2_UOG2_OTGSC)	32	R/W	0000_0120h	11.1.5.32/ 2866
32E6_01A8	USB Device Mode (USB_OTG2_UOG2_USBMODE)	32	R/W	0000_0000h	11.1.5.33/ 2870
32E6_01AC	Endpoint Setup Status (USB_OTG2_UOG2_ENDPTSETUPSTAT)	32	R/W	0000_0000h	11.1.5.34/ 2871
32E6_01B0	Endpoint Prime (USB_OTG2_UOG2_ENDPTPRIME)	32	R/W	0000_0000h	11.1.5.35/ 2872
32E6_01B4	Endpoint Flush (USB_OTG2_UOG2_ENDPTFLUSH)	32	R/W	0000_0000h	11.1.5.36/ 2873
32E6_01B8	Endpoint Status (USB_OTG2_UOG2_ENDPTSTAT)	32	R	0000_0000h	11.1.5.37/ 2873
32E6_01BC	Endpoint Complete (USB_OTG2_UOG2_ENDPTCOMPLETE)	32	R/W	0000_0000h	11.1.5.38/ 2874
32E6_01C0	Endpoint Control0 (USB_OTG2_UOG2_ENDPTCTRL0)	32	R/W	0080_0080h	11.1.5.39/ 2875
32E6_01C4	Endpoint Control 1 (USB_OTG2_UOG2_ENDPTCTRL1)	32	R/W	0000_0000h	11.1.5.40/ 2877
32E6_01C8	Endpoint Control 2 (USB_OTG2_UOG2_ENDPTCTRL2)	32	R/W	0000_0000h	11.1.5.41/ 2880
32E6_01CC	Endpoint Control 3 (USB_OTG2_UOG2_ENDPTCTRL3)	32	R/W	0000_0000h	11.1.5.42/ 2882
32E6_01D0	Endpoint Control 4 (USB_OTG2_UOG2_ENDPTCTRL4)	32	R/W	0000_0000h	11.1.5.43/ 2885
32E6_01D4	Endpoint Control 5 (USB_OTG2_UOG2_ENDPTCTRL5)	32	R/W	0000_0000h	11.1.5.44/ 2888

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E6_01D8	Endpoint Control 6 (USB_OTG2_UOG2_ENDPTCTRL6)	32	R/W	0000_0000h	11.1.5.45/ 2891
32E6_01DC	Endpoint Control 7 (USB_OTG2_UOG2_ENDPTCTRL7)	32	R/W	0000_0000h	11.1.5.46/ 2894

11.1.5.1 Identification register (USBx_*n*ID)

The ID register identifies the USB 2.0 High-Speed core and its revision.

Address: Base address + 0h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								REVISION							
W																
Reset	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		NID						Reserved		ID					
W																
Reset	1	1	1	1	1	0	1	0	0	0	0	0	0	1	0	1

USBx_*n*ID field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 REVISION	Revision number of the controller core.
15–14 -	This field is reserved. Reserved
13–8 NID	Complement version of ID
7–6 -	This field is reserved. Reserved
ID	Configuration number. This number is set to 0x05 and indicates that the peripheral is USB 2.0 High-Speed core.

11.1.5.2 Hardware General (USBx_*n*HWGENERAL)

General hardware parameters as defined in System Level Issues and Core Configuration.

NOTE

The reset value could vary from instance to instance. Please see the detail in bit field description and ignore reset value in summary table in this case!

Address: Base address + 4h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					SM		PHYM			PHYW		Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1

USBx_nHWGENERAL field descriptions

Field	Description
31–11 -	This field is reserved. Reserved
10–9 SM	Serial interface mode capability 00 No Serial Engine, always use parallel signalling. 01 Serial Engine present, always use serial signalling for FS/LS. 10 Software programmable - Reset to use parallel signalling for FS/LS 11 Software programmable - Reset to use serial signalling for FS/LS
8–6 PHYM	Transciever type. PHYM bit reset value: '0000b' for OTG controller core, '0100b' for Host-only controller core. 000 UTMI/UMTI+ 001 ULPI DDR 010 ULPI 011 Serial Only 100 Software programmable - reset to UTMI/UTMI+ 101 Software programmable - reset to ULPI DDR 110 Software programmable - reset to ULPI 111 Software programmable - reset to Serial
5–4 PHYW	Data width of the transciever connected to the controller core. PHYW bit reset value is PHYW bit reset value is '01b'. 11 Reset to 16 bit wide data bus Software programmable
-	This field is reserved. Reserved

11.1.5.3 Host Hardware Parameters (USB*x*_*n*HWHOST)

Address: Base address + 8h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												NPORT			HC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

USB*x*_*n*HWHOST field descriptions

Field	Description
31–4 -	This field is reserved. Reserved
3–1 NPORT	The Nmber of downstream ports supported by the host controller is (NPORT+1). **NOTE:** When these bits value is '000', it indicates a single-port host controller.
0 HC	Host Capable. Indicating whether host operation mode is supported or not. 1 Supported 0 Not supported

11.1.5.4 Device Hardware Parameters (USB*x*_*n*HWDEVICE)

NOTE

This register is only available in OTG core.

Address: Base address + Ch offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										DEVEP					DC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

USBx_nHWDEVICE field descriptions

Field	Description
31–6 -	This field is reserved. Reserved
5–1 DEVEP	Device Endpoint Number
0 DC	Device Capable. Indicating whether device operation mode is supported or not. 1 Supported 0 Not supported

11.1.5.5 TX Buffer Hardware Parameters (USBx_nHWTXBUF)

Address: Base address + 10h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								TXCHANADD								Reserved								TXBURST							
W																																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	0	0	0

USBx_nHWTXBUF field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 TXCHANADD	TX FIFO Buffer size is: (2^TXCHANADD) * 4 Bytes. These bits are set to '08h', so buffer size is 256*4 Bytes. For the OTG controller operating in device mode, this is the FIFO buffer size per endpoint. As the OTG controller has 8 TX endpoint, there are 8 of these buffers. For the OTG controller operating in host mode, or for Host-only controller, the entire buffer memory is used as a single TX buffer. Therefore, there is only 1 of this buffer
15–8 -	This field is reserved. Reserved
TXBURST	Default burst size for memory to TX buffer transfer. This is reset value of TXPBURST bits in USB core regsiter USB_n_BURSTSIZE. Please see Programmable Burst Size (USB_nBURSTSIZE) .

11.1.5.6 RX Buffer Hardware Parameters (USB*x*_*n*HWRXBUF)

Address: Base address + 14h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																RXADD								RXBURST							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0

USB*x*_*n*HWRXBUF field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15–8 RXADD	Buffer total size for all receive endpoints is (2^RXADD). RX Buffer size is: (2^RXADD) * 4 Bytes. These bits are set to '08h', so buffer size is 256*4 Bytes. There is a single Receive FIFO buffer in the USB controller. The buffer is shared for all endpoints for the OTG controller in device mode.
RXBURST	Default burst size for memory to RX buffer transfer. This is reset value of RXPBURST bits in USB core regsiter USB_n_BURSTSIZE. Please see Programmable Burst Size (USB_*n*BURSTSIZE) .

11.1.5.7 General Purpose Timer #0 Load (USB*x*_*n*GPTIMER0LD)

This register controls load value of the count timer in register n_GPTIMER0CTRL. Please see General Purpose Timer #0 Controller (USB_*n*GPTIMER0CTRL) .

Address: Base address + 80h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								GPTLD																							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB*x*_*n*GPTIMER0LD field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
GPTLD	General Purpose Timer Load Value These bit fields are loaded to GPTCNT bits when GPTRST bit is set '1b'. This value represents the time in microseconds minus 1 for the timer duration.

Table continues on the next page...

USBx_*n*GPTIMER0LD field descriptions (continued)

Field	Description
	Example: for a one millisecond timer, load 1000-1=999 or 0x0003E7. **NOTE:** Max value is 0xFFFFFF or 16.777215 seconds.

11.1.5.8 General Purpose Timer #0 Controller (USBx_*n*GPTIMER0CTRL)

This register contains the control for this countdown timer and a data field can be queried to determine the running count value. This timer has granularity on 1 us and can be programmed to a little over 16 seconds. There are two counter modes which are described in the register table below. When the timer counter value transitions to zero, an interrupt could be generated if enable.

Interrupt status bit is TI0 bit in n_USBSTS register (See USB Status Register (USB_*n*USBSTS)), interrupt enable bit is TIE0 bit in n_USBINTR register. (See Interrupt Enable Register (USB_*n*USBINTR) .)

Address: Base address + 84h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GPTRUN	GPTRST	Reserved					GPTMODE	GPTCNT							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPTCNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBx_*n*GPTIMER0CTRL field descriptions

Field	Description
31 GPTRUN	General Purpose Timer Run GPTCNT bits are not effected when setting or clearing this bit. 0 Stop counting 1 Run
30 GPTRST	General Purpose Timer Reset 0 No action 1 Load counter value from GPTLD bits in n_GPTIMER0LD

Table continues on the next page...

USBx_nGPTIMER0CTRL field descriptions (continued)

Field	Description
29–25 -	This field is reserved. Reserved
24 GPTMODE	General Purpose Timer Mode In one shot mode, the timer will count down to zero, generate an interrupt, and stop until the counter is reset by software; In repeat mode, the timer will count down to zero, generate an interrupt and automatically reload the counter value from GPTLD bits to start again. 0 One Shot Mode 1 Repeat Mode
GPTCNT	General Purpose Timer Counter. This field is the count value of the countdown timer.

11.1.5.9 General Purpose Timer #1 Load (USBx_nGPTIMER1LD)

This register controls load value of the count timer in register n_GPTIMER1CTRL. Please see General Purpose Timer #1 Controller (USB_nGPTIMER1CTRL) .

Address: Base address + 88h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								GPTLD																							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBx_nGPTIMER1LD field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
GPTLD	General Purpose Timer Load Value These bit fields are loaded to GPTCNT bits when GPTRST bit is set '1b'. This value represents the time in microseconds minus 1 for the timer duration. Example: for a one millisecond timer, load 1000-1=999 or 0x0003E7. **NOTE:** Max value is 0xFFFFFF or 16.777215 seconds.

11.1.5.10 General Purpose Timer #1 Controller (USB*x*_*n*GPTIMER1CTRL)

This register contains the control for this countdown timer and a data field can be queried to determine the running count value. This timer has granularity on 1 us and can be programmed to a little over 16 seconds. There are two counter modes which are described in the register table below. When the timer counter value transitions to zero, an interrupt could be generated if enable.

Interrupt status bit is TI1 bit in USB_n_USBSTS register (See USB Status Register (USB_*n*USBSTS)), interrupt enable bit is TIE1 bit in n_USBINTR register (See Interrupt Enable Register (USB_*n*USBINTR)).

Address: Base address + 8Ch offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GPTRUN	GPTRST	Reserved					GPTMODE	GPTCNT							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPTCNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB*x*_*n*GPTIMER1CTRL field descriptions

Field	Description
31 GPTRUN	General Purpose Timer Run GPTCNT bits are not effected when setting or clearing this bit. 0 Stop counting 1 Run
30 GPTRST	General Purpose Timer Reset 0 No action 1 Load counter value from GPTLD bits in USB_n_GPTIMER0LD
29–25 -	This field is reserved. Reserved
24 GPTMODE	General Purpose Timer Mode In one shot mode, the timer will count down to zero, generate an interrupt, and stop until the counter is reset by software. In repeat mode, the timer will count down to zero, generate an interrupt and automatically reload the counter value from GPTLD bits to start again.

Table continues on the next page...

USBx_nGPTIMER1CTRL field descriptions (continued)

Field	Description
	0 One Shot Mode 1 Repeat Mode
GPTCNT	General Purpose Timer Counter. This field is the count value of the countdown timer.

11.1.5.11 System Bus Config (USBx_nSBUSCFG)

Address: Base address + 90h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																													AHBBRST		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

USBx_nSBUSCFG field descriptions

Field	Description
31–3 -	This field is reserved. Reserved
AHBBRST	AHB master interface Burst configuration These bits control AHB master transfer type sequence (or priority). **NOTE:** This register overrides n_BURSTSIZE register when its value is not zero. 000 Incremental burst of unspecified length only 001 INCR4 burst, then single transfer 010 INCR8 burst, INCR4 burst, then single transfer 011 INCR16 burst, INCR8 burst, INCR4 burst, then single transfer 100 Reserved, don't use 101 INCR4 burst, then incremental burst of unspecified length 110 INCR8 burst, INCR4 burst, then incremental burst of unspecified length 111 INCR16 burst, INCR8 burst, INCR4 burst, then incremental burst of unspecified length

11.1.5.12 Capability Registers Length (USBx_nCAPLENGTH)

The Capability Registers Length register contains the address offset to the Operational registers relative to the CAPLENGTH register.

Address: Base address + 100h offset + (65536d × i), where i=0d to 1d

Bit	7	6	5	4	3	2	1	0
Read	CAPLENGTH							
Write								
Reset	0	1	0	0	0	0	0	0

USBx_nCAPLENGTH field descriptions

Field	Description
CAPLENGTH	These bits are used as an offset to add to register base to find the beginning of the Operational Register. Default value is '40h'.

11.1.5.13 Host Controller Interface Version (USBx_nHCIVERSION)

This is a 2-byte register containing a BCD encoding of the EHCI revision number supported by this host controller. The most significant byte of this register represents a major revision and the least significant byte is the minor revision.

Address: Base address + 102h offset + (65536d × i), where i=0d to 1d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	HCIVERSION															
Write																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

USBx_nHCIVERSION field descriptions

Field	Description
HCIVERSION	Host Controller Interface Version Number Default value is '10h', which means EHCI rev1.0.

11.1.5.14 Host Controller Structural Parameters (USB*x*_*n*HCSPARAMS)

The following figure shows the port steering logic capabilities of Host Control Structural Parameters (n_HCSPARAMS).

Address: Base address + 104h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				N_TT				N_PTT				Reserved			PI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	N_CC				N_PCC				Reserved			PPC	N_PORTS			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

USB*x*_*n*HCSPARAMS field descriptions

Field	Description
31–28 -	This field is reserved. Reserved
27–24 N_TT	Number of Transaction Translators (N_TT). Default value '0000b' This field indicates the number of embedded transaction translators associated with the USB2.0 host controller. These bits would be set to '0001b' for Multi-Port Host, and '0000b' for Single-Port Host.
23–20 N_PTT	Number of Ports per Transaction Translator (N_PTT). Default value '0000b' This field indicates the number of ports assigned to each transaction translator within the USB2.0 host controller. These bits would be set to equal N_PORTS for Multi-Port Host, and '0000b' for Single-Port Host.
19–17 -	This field is reserved. Reserved
16 PI	Port Indicators (P INDICATOR) This bit indicates whether the ports support port indicator control. When set to one, the port status and control registers include a read/writeable field for controlling the state of the port indicator This bit is "1b" in all controller core.
15–12 N_CC	Number of Companion Controller (N_CC). This field indicates the number of companion controllers associated with this USB2.0 host controller. These bits are '0000b' in all controller core. 0 There is no internal Companion Controller and port-ownership hand-off is not supported. 1 There are internal companion controller(s) and port-ownership hand-offs is supported.

Table continues on the next page...

USBx_*n*HCSPARAMS field descriptions (continued)

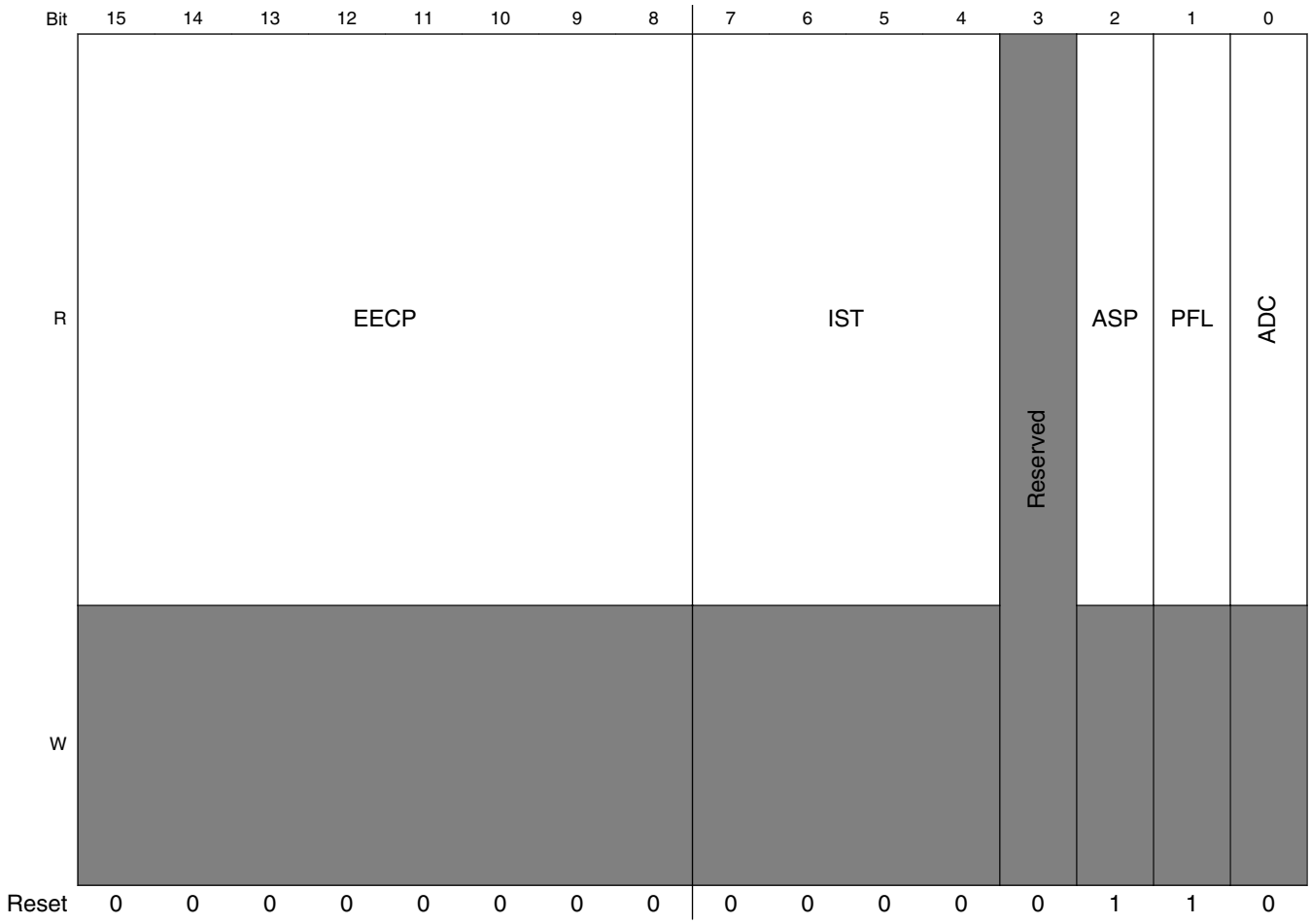
Field	Description
11–8 N_PCC	Number of Ports per Companion Controller This field indicates the number of ports supported per internal Companion Controller. It is used to indicate the port routing configuration to the system software. For example, if N_PORTS has a value of 6 and N_CC has a value of 2 then N_PCC could have a value of 3. The convention is that the first N_PCC ports are assumed to be routed to companion controller 1, the next N_PCC ports to companion controller 2, etc. In the previous example, the N_PCC could have been 4, where the first 4 are routed to companion controller 1 and the last two are routed to companion controller 2. The number in this field must be consistent with N_PORTS and N_CC. These bits are '0000b' in all controller core.
7–5 -	This field is reserved. Reserved
4 PPC	Port Power Control This field indicates whether the host controller implementation includes port power control. A one indicates the ports have port power switches. A zero indicates the ports do not have port power switches. The value of this field affects the functionality of the Port Power field in each port status and control register
N_PORTS	Number of downstream ports. This field specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational Register. Valid values are in the range of 1h to Fh. A zero in this field is undefined. These bits are always set to '0001b' because all controller cores are Single-Port Host.

11.1.5.15 Host Controller Capability Parameters (USB*x*_*n*HCCPARAMS)

This register identifies multiple mode control (time-base bit functionality), addressing capability.

Address: Base address + 108h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



USB*x*_*n*HCCPARAMS field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15–8 EECP	EHCI Extended Capabilities Pointer. This field indicates the existence of a capabilities list. A value of 00h indicates no extended capabilities are implemented. A non-zero value in this register indicates the offset in PCI configuration space of the first EHCI extended capability. The pointer value must be 40h or greater if implemented to maintain the consistency of the PCI header defined for this class of device. **NOTE:** These bits are set '00h' in all controller core.
7–4 IST	Isochronous Scheduling Threshold. This field indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule. When bit [7] is zero, the value of the least significant 3 bits indicates the number of micro-frames a host controller can hold a set of isochronous data structures (one or more) before flushing the state. When bit [7] is a one, then host software assumes the host controller may cache an isochronous data structure for an entire frame. These bits are set '00h' in all controller core.
3 -	This field is reserved. Reserved

Table continues on the next page...

USBx_nHCCPARAMS field descriptions (continued)

Field	Description
2 ASP	Asynchronous Schedule Park Capability If this bit is set to a one, then the host controller supports the park feature for high-speed queue heads in the Asynchronous Schedule. The feature can be disabled or enabled and set to a specific level by using the *Asynchronous Schedule Park Mode Enable* and *Asynchronous Schedule Park Mode Count* fields in the USBCMD register. **NOTE:** ASP bit reset value: '00b' for OTG controller core, '11b' for Host-only controller core.
1 PFL	Programmable Frame List Flag If this bit is set to zero, then the system software must use a frame list length of 1024 elements with this host controller. The USBCMD register Frame List Size field is a read-only register and must be set to zero. If set to a one, then the system software can specify and use a smaller frame list and configure the host controller via the USBCMD register Frame List Size field. The frame list must always be aligned on a 4K-page boundary. This requirement ensures that the frame list is always physically contiguous. This bit is set '1b' in all controller core.
0 ADC	64-bit Addressing Capability This bit is set '0b' in all controller core, no 64-bit addressing capability is supported.

11.1.5.16 Device Controller Interface Version (USBx_nDCIVERSION)

This register indicates the two-byte BCD encoding of the device controller interface version number.

Address: Base address + 120h offset + (65536d × i), where i=0d to 1d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	DCIVERSION															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

USBx_nDCIVERSION field descriptions

Field	Description
DCIVERSION	Device Controller Interface Version Number Default value is '01h', which means rev0.1.

11.1.5.17 Device Controller Capability Parameters (USBx_nDCCPARAMS)

These fields describe the overall device capability of the controller.

NOTE

This register is only available in OTG controller core.

Address: Base address + 124h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved							HC	DC	Reserved		DEN				
W																
Reset	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0

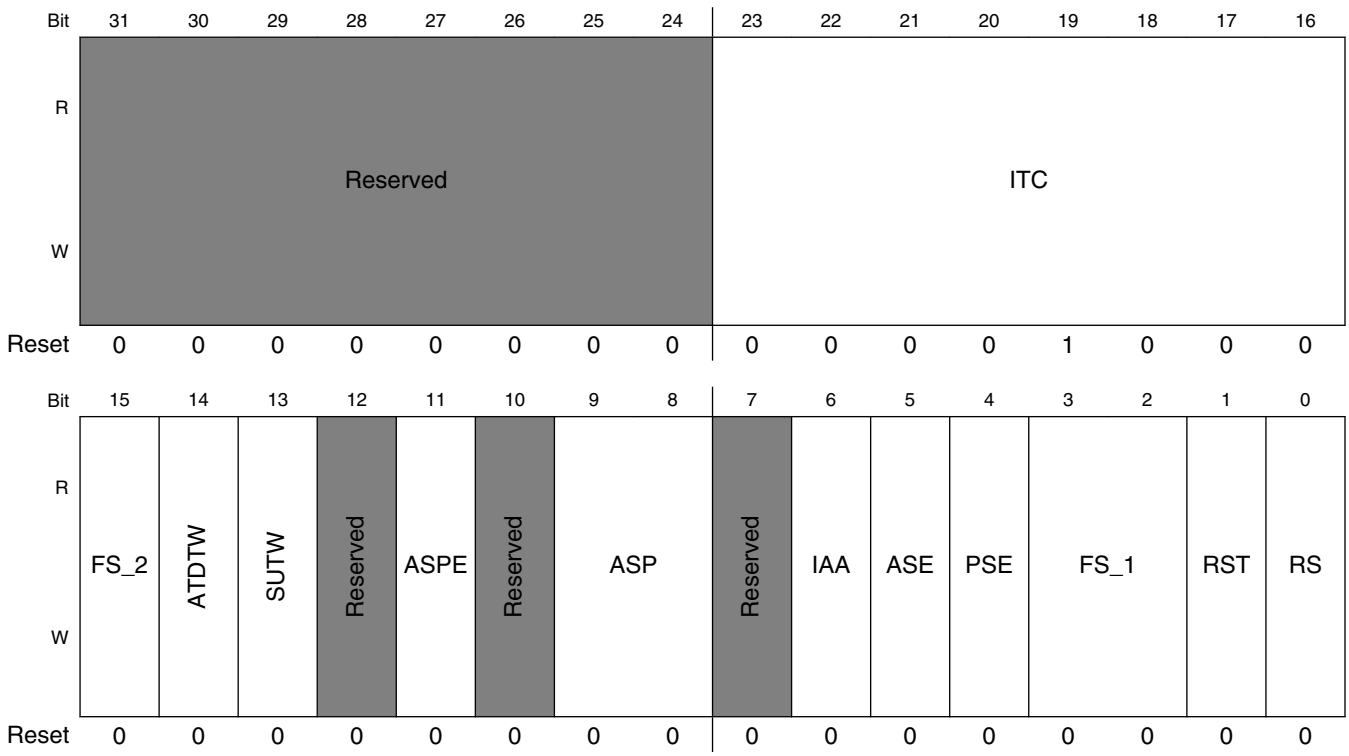
USB*x*_*n*DCCPARAMS field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 HC	Host Capable When this bit is 1, this controller is capable of operating as an EHCI compatible USB 2.0 host controller.
7 DC	Device Capable When this bit is 1, this controller is capable of operating as a USB 2.0 device.
6–5 -	This field is reserved. Reserved
DEN	Device Endpoint Number This field indicates the number of endpoints built into the device controller. If this controller is not device capable, then this field will be zero. Valid values are 0 - 15.

11.1.5.18 USB Command Register (USB*x*_*n*USBCMD)

The Command Register indicates the command to be executed by the serial bus host/device controller. Writing to the register causes a command to be executed.

Address: Base address + 140h offset + (65536d × i), where i=0d to 1d



USB*x*_*n*USBCMD field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 ITC	Interrupt Threshold Control -Read/Write. The system software uses this field to set the maximum rate at which the host/device controller will issue interrupts. ITC contains the maximum interrupt interval measured in micro-frames. Valid values are shown below. Value Maximum Interrupt Interval 0x00 Immediate (no threshold) 0x01 1 micro-frame 0x02 2 micro-frames 0x04 4 micro-frames 0x08 8 micro-frames 0x10 16 micro-frames

Table continues on the next page...

USBx_nUSBCMD field descriptions (continued)

Field	Description
	0x20 32 micro-frames 0x40 64 micro-frames
15 FS_2	Frame List Size - (Read/Write or Read Only). [host mode only] This field is Read/Write only if Programmable Frame List Flag in the HCCPARAMS registers is set to one. This field specifies the size of the frame list that controls which bits in the Frame Index Register should be used for the Frame List Current index. **NOTE:** This field is made up from USBCMD bits 15, 3 and 2. Value Meaning 000 1024 elements (4096 bytes) Default value 001 512 elements (2048 bytes) 010 256 elements (1024 bytes) 011 128 elements (512 bytes) 100 64 elements (256 bytes) 101 32 elements (128 bytes) 110 16 elements (64 bytes) 111 8 elements (32 bytes)
14 ATDTW	Add dTD TripWire - Read/Write. [device mode only] This bit is used as a semaphore to ensure proper addition of a new dTD to an active (primed) endpoint's linked list. This bit is set and cleared by software. This bit would also be cleared by hardware when state machine is hazard region for which adding a dTD to a primed endpoint may go unrecognized.
13 SUTW	Setup TripWire - Read/Write. [device mode only] This bit is used as a semaphore to ensure that the setup data payload of 8 bytes is extracted from a QH by the DCD without being corrupted. If the setup lockout mode is off (SLOM bit in USB core register n_USBMODE, see USB Device Mode (USB_nUSBMODE)) then there is a hazard when new setup data arrives while the DCD is copying the setup data payload from the QH for a previous setup packet. This bit is set and cleared by software. This bit would also be cleared by hardware when a hazard detected.
12 -	This field is reserved. Reserved
11 ASPE	Asynchronous Schedule Park Mode Enable - Read/Write. If the *Asynchronous Park Capability* bit in the HCCPARAMS register is a one, then this bit defaults to a 1h and is R/W. Otherwise the bit must be a zero and is RO. Software uses this bit to enable or disable Park mode. When this bit is one, Park mode is enabled. When this bit is a zero, Park mode is disabled. **NOTE:** ASPE bit reset value: '0b' for OTG controller core, '1b' for Host-only controller core.
10 -	This field is reserved. Reserved
9–8 ASP	Asynchronous Schedule Park Mode Count - Read/Write. If the *Asynchronous Park Capability* bit in the HCCPARAMS register is a one, then this field defaults to 3h and is R/W. Otherwise it defaults to zero and is Read-Only. It contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the

Table continues on the next page...

USBx_nUSBCMD field descriptions (continued)

Field	Description
	Asynchronous schedule before continuing traversal of the Asynchronous schedule. Valid values are 1h to 3h. Software must not write a zero to this bit when *Park Mode Enable* is a one as this will result in undefined behavior. This field is set to 3h in all controller core.
7 -	This field is reserved. Reserved
6 IAA	Interrupt on Async Advance Doorbell - Read/Write. This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Software must write a 1 to this bit to ring the doorbell. When the host controller has evicted all appropriate cached schedule states, it sets the Interrupt on Async Advance status bit in the USBSTS register. If the Interrupt on Sync Advance Enable bit in the USBINTR register is one, then the host controller will assert an interrupt at the next interrupt threshold. The host controller sets this bit to zero after it has set the Interrupt on Sync Advance status bit in the USBSTS register to one. Software should not write a one to this bit when the asynchronous schedule is inactive. Doing so will yield undefined results. This bit is only used in host mode. Writing a one to this bit when device mode is selected will have undefined results.
5 ASE	Asynchronous Schedule Enable - Read/Write. Default 0b. This bit controls whether the host controller skips processing the Asynchronous Schedule. Only the host controller uses this bit. Values Meaning 0 Do not process the Asynchronous Schedule. 1 Use the ASYNCLISTADDR register to access the Asynchronous Schedule.
4 PSE	Periodic Schedule Enable- Read/Write. Default 0b. This bit controls whether the host controller skips processing the Periodic Schedule. Only the host controller uses this bit. Values Meaning 0 Do not process the Periodic Schedule 1 Use the PERIODICLISTBASE register to access the Periodic Schedule.
3–2 FS_1	See description at bit 15
1 RST	Controller Reset (RESET) - Read/Write. Software uses this bit to reset the controller. This bit is set to zero by the Host/Device Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register. Host operation mode: When software writes a one to this bit, the Controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports. Software should not set this bit to a one when the HCHalted bit in the USBSTS register is a zero. Attempting to reset an actively running host controller will result in undefined behavior. Device operation mode:

Table continues on the next page...

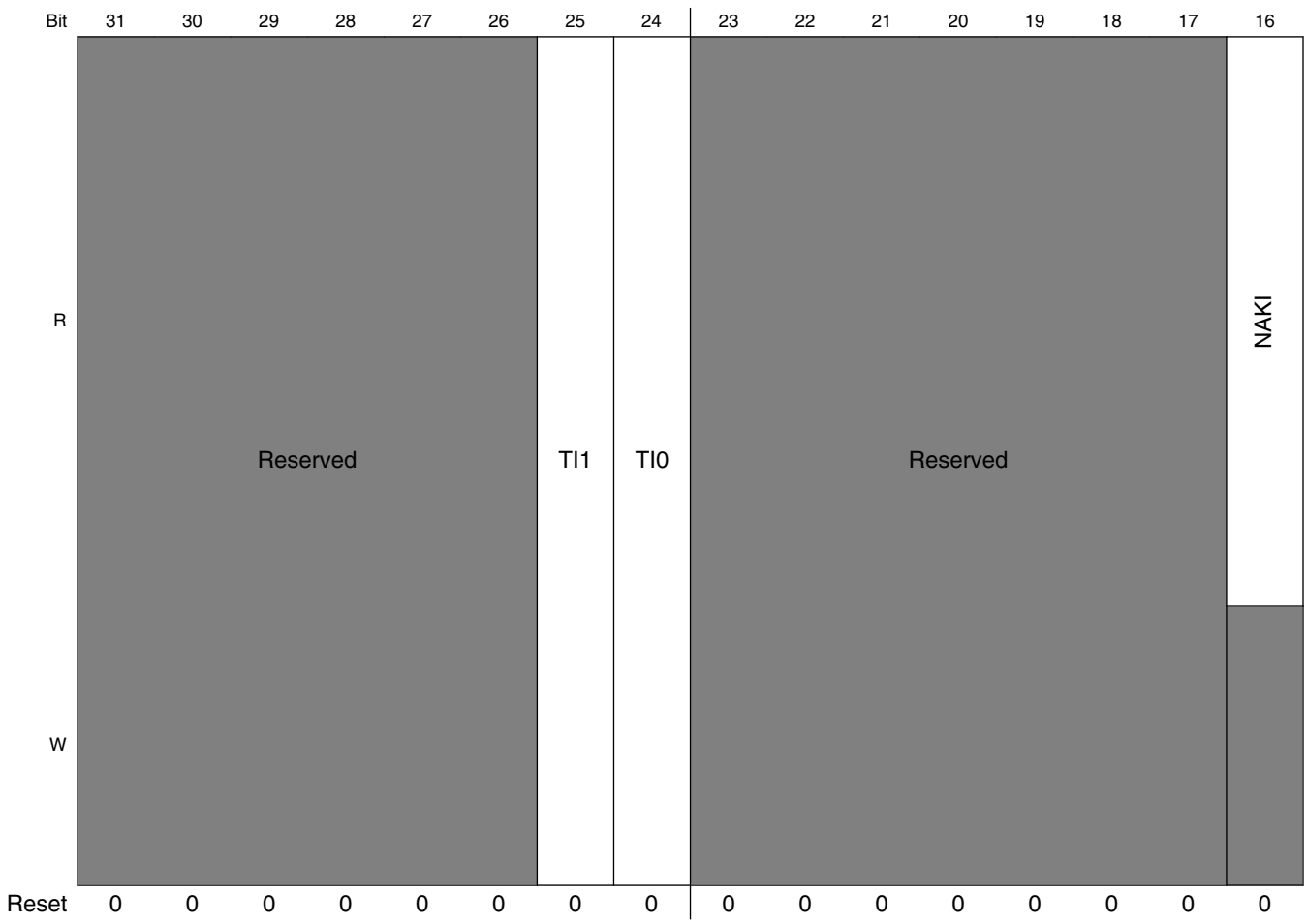
USB*x*_*n*USBCMD field descriptions (continued)

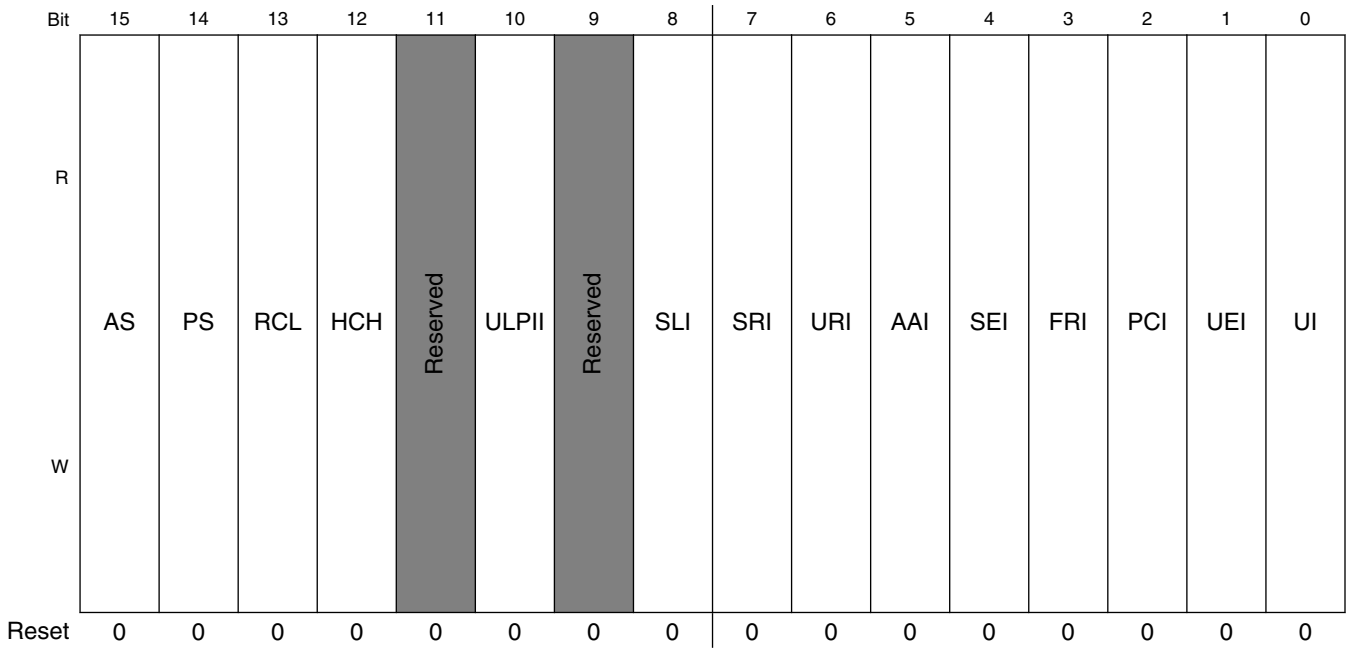
Field	Description
	When software writes a one to this bit, the Controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Writing a one to this bit when the device is in the attached state is not recommended, because the effect on an attached host is undefined. In order to ensure that the device is not in an attached state before initiating a device controller reset, all primed endpoints should be flushed and the USBCMD Run/Stop bit should be set to 0.
0 RS	Run/Stop (RS) - Read/Write. Default 0b. 1=Run. 0=Stop. Host operation mode: When set to '1b', the Controller proceeds with the execution of the schedule. The Controller continues execution as long as this bit is set to a one. When this bit is set to 0, the Host Controller completes the current transaction on the USB and then halts. The HC Halted bit in the status register indicates when the Controller has finished the transaction and has entered the stopped state. Software should not write a one to this field unless the controller is in the Halted state (that is, HCHalted in the USBSTS register is a one). Device operation mode: Writing a one to this bit will cause the controller to enable a pull-up on D+ and initiate an attach event. This control bit is not directly connected to the pull-up enable, as the pull-up will become disabled upon transitioning into high-speed mode. Software should use this bit to prevent an attach event before the controller has been properly initialized. Writing a 0 to this will cause a detach event.

11.1.5.19 USB Status Register (USB*x*_*n*USBSTS)

This register indicates various states of the Host/Device Controller and any pending interrupts. This register does not indicate status resulting from a transaction on the serial bus.

Address: Base address + 144h offset + (65536d × i), where i=0d to 1d





USBx_nUSBSTS field descriptions

Field	Description
31–26 -	This field is reserved. Reserved
25 TI1	General Purpose Timer Interrupt 1(GPTINT1)--R/WC. This bit is set when the counter in the GPTIMER1CTRL register transitions to zero, writing a one to this bit will clear it.
24 TI0	General Purpose Timer Interrupt 0(GPTINT0)--R/WC. This bit is set when the counter in the GPTIMER0CTRL register transitions to zero, writing a one to this bit clears it.
23–17 -	This field is reserved. Reserved
16 NAKI	NAK Interrupt Bit--RO. This bit is set by hardware when for a particular endpoint both the TX/RX Endpoint NAK bit and corresponding TX/RX Endpoint NAK Enable bit are set. This bit is automatically cleared by hardware when all Enabled TX/RX Endpoint NAK bits are cleared.
15 AS	Asynchronous Schedule Status - Read Only. This bit reports the current real status of the Asynchronous Schedule. When set to zero the asynchronous schedule status is disabled and if set to one the status is enabled. The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either enabled (1) or disabled (0). Only used in the host operation mode.
14 PS	Periodic Schedule Status - Read Only.

Table continues on the next page...

USB*x*_*n*USBSTS field descriptions (continued)

Field	Description
	This bit reports the current real status of the Periodic Schedule. When set to zero the periodic schedule is disabled, and if set to one the status is enabled. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either enabled (1) or disabled (0). Only used in the host operation mode.
13 RCL	Reclamation - Read Only. This is a read-only status bit used to detect an empty asynchronous schedule. Only used in the host operation mode.
12 HCH	HCHalted - Read Only. This bit is a zero whenever the Run/Stop bit is a one. The Controller sets this bit to one after it has stopped executing because of the Run/Stop bit being set to 0, either by software or by the Controller hardware (for example, an internal error). Only used in the host operation mode. Default value is '0b' for OTG core, and '1b' for Host1/Host2/Host3 core. This is because OTG core is not operating as host in default. Please see CM bit in USB_n_USBMODE register. **NOTE:** HCH bit reset value: '0b' for OTG controller core, '1b' for Host-only controller core.
11 -	This field is reserved. Reserved
10 ULPII	ULPI Interrupt - R/WC. This bit will be set '1b' by hardware when there is an event completion in ULPI viewport. This bit is usable only if the controller support UPLI interface mode.
9 -	This field is reserved. Reserved
8 SLI	DCSuspend - R/WC. When a controller enters a suspend state from an active state, this bit will be set to a one. The device controller clears the bit upon exiting from a suspend state. Only used in device operation mode.
7 SRI	SOF Received - R/WC. When the device controller detects a Start Of (micro) Frame, this bit will be set to a one. When a SOF is extremely late, the device controller will automatically set this bit to indicate that an SOF was expected. Therefore, this bit will be set roughly every 1ms in device FS mode and every 125ms in HS mode and will be synchronized to the actual SOF that is received. Because the device controller is initialized to FS before connect, this bit will be set at an interval of 1ms during the prelude to connect and chirp. In host mode, this bit will be set every 125us and can be used by host controller driver as a time base. Software writes a 1 to this bit to clear it.
6 URI	USB Reset Received - R/WC. When the device controller detects a USB Reset and enters the default state, this bit will be set to a one. Software can write a 1 to this bit to clear the USB Reset Received status bit. Only used in device operation mode.

Table continues on the next page...

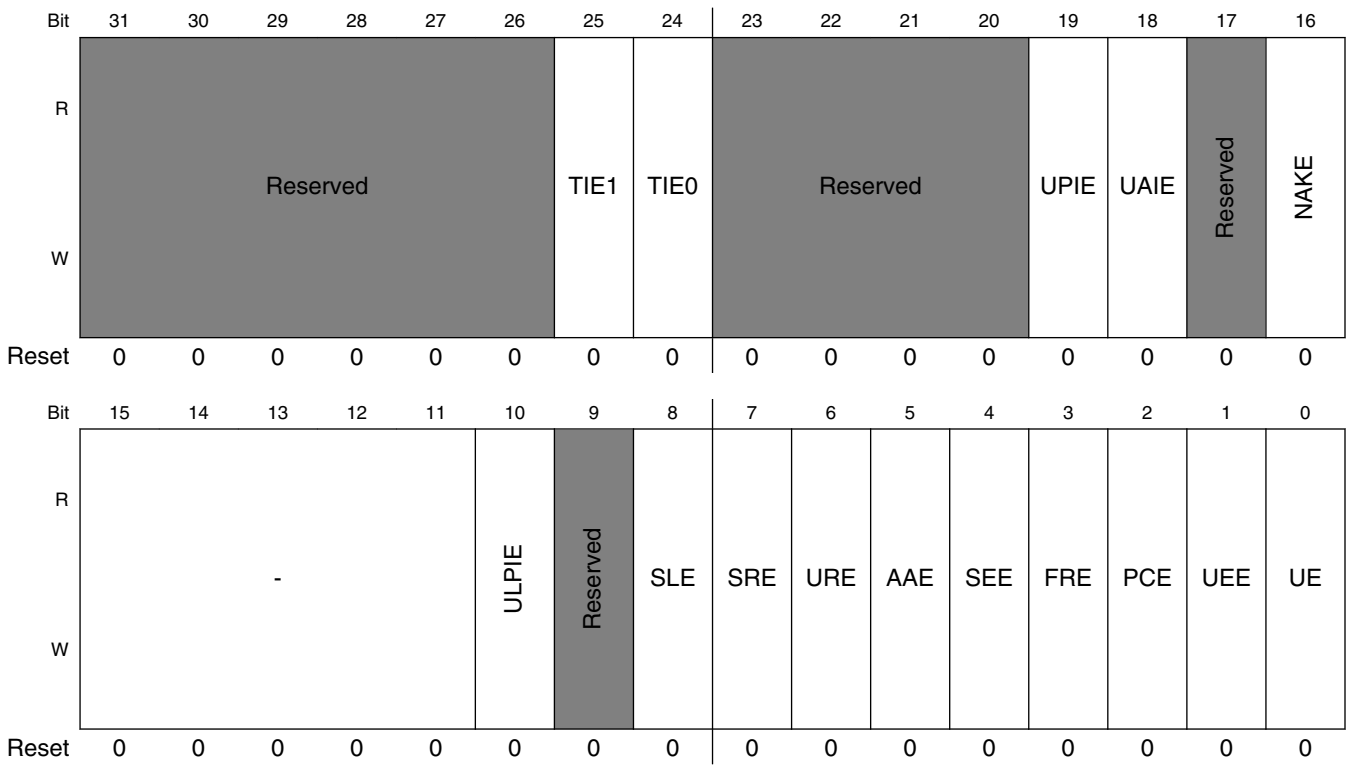
USB*x*_*n*USBSTS field descriptions (continued)

Field	Description
5 AAI	Interrupt on Async Advance - R/WC. System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a one to the Interrupt on Async Advance Doorbell bit in the n_USBCMD register. This status bit indicates the assertion of that interrupt source. Only used in host operation mode.
4 SEI	System Error- R/WC. This bit is will be set to '1b' when an Error response is seen to a read on the system interface.
3 FRI	Frame List Rollover - R/WC. The Host Controller sets this bit to a one when the Frame List Index rolls over from its maximum value to zero. The exact value at which the rollover occurs depends on the frame list size. For example. If the frame list size (as programmed in the Frame List Size field of the USB_n_USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX [13] toggles. Similarly, if the size is 512, the Host Controller sets this bit to a one every time FHINDEX [12] toggles. Only used in host operation mode.
2 PCI	Port Change Detect - R/WC. The Host Controller sets this bit to a one when on any port a Connect Status occurs, a Port Enable/ Disable Change occurs, or the Force Port Resume bit is set as the result of a J-K transition on the suspended port. The Device Controller sets this bit to a one when the port controller enters the full or high-speed operational state. When the port controller exits the full or high-speed operation states due to Reset or Suspend events, the notification mechanisms are the USB Reset Received bit and the DCSuspend bits respectively.
1 UEI	USB Error Interrupt (USBERRINT) - R/WC. When completion of a USB transaction results in an error condition, this bit is set by the Host/Device Controller. This bit is set along with the USBINT bit, if the TD on which the error interrupt occurred also had its interrupt on complete (IOC) bit set. The device controller detects resume signaling only.
0 UI	USB Interrupt (USBINT) - R/WC. This bit is set by the Host/Device Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set. This bit is also set by the Host/Device Controller when a short packet is detected. A short packet is when the actual number of bytes received was less than the expected number of bytes.

11.1.5.20 Interrupt Enable Register (USB*x*_*n*USBINTR)

The interrupts to software are enabled with this register. An interrupt is generated when a bit is set and the corresponding interrupt source is active. The USB Status register (n_USBSTS) still shows interrupt sources even if they are disabled by the n_USBINTR register, allowing polling of interrupt events by the software.

Address: Base address + 148h offset + (65536d × i), where i=0d to 1d



USB*x*_*n*USBINTR field descriptions

Field	Description
31–26 -	This field is reserved. Reserved
25 TIE1	General Purpose Timer #1 Interrupt Enable When this bit is one and the TI1 bit in n_USBSTS register is a one the controller will issue an interrupt.
24 TIE0	General Purpose Timer #0 Interrupt Enable When this bit is one and the TI0 bit in n_USBSTS register is a one the controller will issue an interrupt.
23–20 -	This field is reserved. Reserved
19 UPIE	USB Host Periodic Interrupt Enable

Table continues on the next page...

USB*x*_*n*USBINTR field descriptions (continued)

Field	Description
	When this bit is one, and the UPI bit in the n_USBSTS register is one, host controller will issue an interrupt at the next interrupt threshold.
18 UAIE	USB Host Asynchronous Interrupt Enable When this bit is one, and the UAI bit in the n_USBSTS register is one, host controller will issue an interrupt at the next interrupt threshold.
17 -	This field is reserved. Reserved
16 NAKE	NAK Interrupt Enable When this bit is one and the NAKI bit in n_USBSTS register is a one the controller will issue an interrupt.
15–11 -	These bits are reserved and should be set to zero.
10 ULPIE	ULPI Interrupt Enable When this bit is one and the UPLII bit in n_USBSTS register is a one the controller will issue an interrupt. This bit is usable only if the controller support UPLI interface mode.
9 -	This field is reserved. Reserved
8 SLE	Sleep Interrupt Enable When this bit is one and the SLI bit in n_n_USBSTS register is a one the controller will issue an interrupt. Only used in device operation mode.
7 SRE	SOF Received Interrupt Enable When this bit is one and the SRI bit in n_USBSTS register is a one the controller will issue an interrupt.
6 URE	USB Reset Interrupt Enable When this bit is one and the URI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in device operation mode.
5 AAE	Async Advance Interrupt Enable When this bit is one and the AAI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in host operation mode.
4 SEE	System Error Interrupt Enable When this bit is one and the SEI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in host operation mode.
3 FRE	Frame List Rollover Interrupt Enable When this bit is one and the FRI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in host operation mode.
2 PCE	Port Change Detect Interrupt Enable When this bit is one and the PCI bit in n_USBSTS register is a one the controller will issue an interrupt.
1 UEE	USB Error Interrupt Enable When this bit is one and the UEI bit in n_USBSTS register is a one the controller will issue an interrupt.
0 UE	USB Interrupt Enable When this bit is one and the UI bit in n_USBSTS register is a one the controller will issue an interrupt.

11.1.5.21 USB Frame Index (USBx_*n*FRINDEX)

This register is used by the host controller to index the periodic frame list. The register updates every 125 microseconds (once each micro-frame). Bits [N: 3] are used to select a particular entry in the Periodic Frame List during periodic schedule execution. The number of bits used for the index depends on the size of the frame list as set by system software in the Frame List Size field in the n_USBCMD register.

This register must be written as a DWord. Byte writes produce undefined results. This register cannot be written unless the Host Controller is in the 'Halted' state as indicated by the HCHalted bit. A write to this register while the Run/Stop hit is set to a one produces undefined results. Writes to this register also affect the SOF value.

In device mode this register is read only and, the device controller updates the FRINDEX [13:3] register from the frame number indicated by the SOF marker. Whenever a SOF is received by the USB bus, FRINDEX [13:3] will be checked against the SOF marker. If FRINDEX [13:3] is different from the SOF marker, FRINDEX [13:3] will be set to the SOF value and FRINDEX [2:0] will be set to zero (that is, SOF for 1 ms frame). If FRINDEX [13:3] is equal to the SOF value, FRINDEX [2:0] will be increment (that is, SOF for 125 us micro-frame.).

Address: Base address + 14Ch offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																FRINDEX															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBx_*n*FRINDEX field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
FRINDEX	Frame Index. The value, in this register, increments at the end of each time frame (micro-frame). Bits [N: 3] are used for the Frame List current index. This means that each location of the frame list is accessed 8 times (frames or micro-frames) before moving to the next index. The following illustrates values of N based on the value of the Frame List Size field in the USBCMD register, when used in host mode. USBCMD [Frame List Size] Number Elements N In device mode the value is the current frame number of the last frame transmitted. It is not used as an index. In either mode bits 2:0 indicate the current microframe. 000 (1024) 12 001 (512) 11

Table continues on the next page...

USBx_nFRINDEX field descriptions (continued)

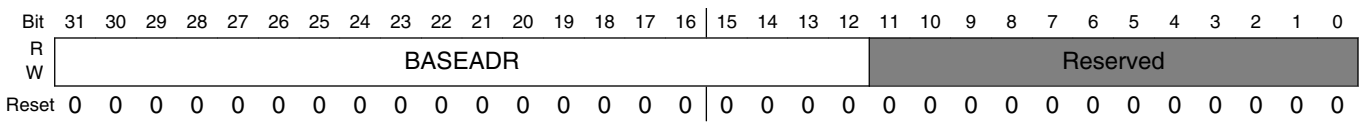
Field	Description
	010 (256) 10 011 (128) 9 100 (64) 8 101 (32) 7 110 (16) 6 111 (8) 5

11.1.5.22 Frame List Base Address (USBx_nPERIODICLISTBASE)

Host Controller only

This 32-bit register contains the beginning address of the Periodic Frame List in the system memory. HCD loads this register prior to starting the schedule execution by the Host Controller. The memory structure referenced by this physical memory pointer is assumed to be 4-Kbyte aligned. The contents of this register are combined with the Frame Index Register (USB_n_FRINDEX) to enable the Host Controller to step through the Periodic Frame List in sequence.

Address: Base address + 154h offset + (65536d × i), where i=0d to 1d



USBx_nPERIODICLISTBASE field descriptions

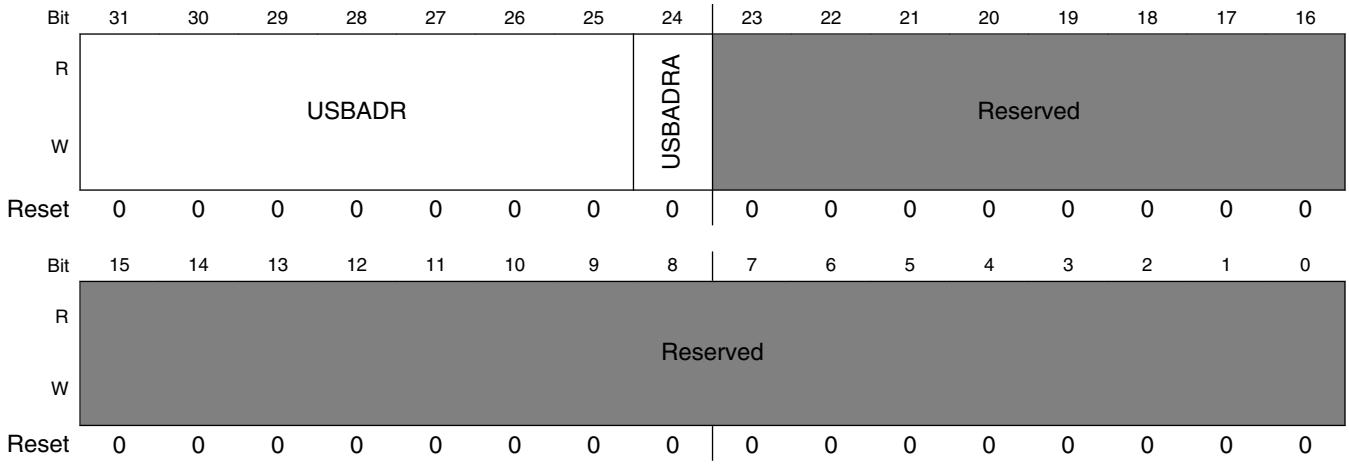
Field	Description
31–12 BASEADR	Base Address (Low). These bits correspond to memory address signals [31:12], respectively. Only used by the host controller.
-	This field is reserved. Reserved

11.1.5.23 Device Address (USBx_nDEVICEADDR)

Device Controller only

The upper seven bits of this register represent the device address. After any controller reset or a USB reset, the device address is set to the default address (0). The default address will match all incoming addresses. Software shall reprogram the address after receiving a SET_ADDRESS descriptor.

Address: Base address + 154h offset + (65536d × i), where i=0d to 1d



USBx_*n*DEVICEADDR field descriptions

Field	Description
31–25 USBADR	Device Address. These bits correspond to the USB device address
24 USBADRA	Device Address Advance. Default=0. When this bit is '0', any writes to USBADR are instantaneous. When this bit is written to a '1' at the same time or before USBADR is written, the write to the USBADR field is staged and held in a hidden register. After an IN occurs on endpoint 0 and is ACKed, USBADR will be loaded from the holding register. Hardware will automatically clear this bit on the following conditions: 1) IN is ACKed to endpoint 0. (USBADR is updated from staging register). 2) OUT/SETUP occur to endpoint 0. (USBADR is not updated). 3) Device Reset occurs (USBADR is reset to 0). **NOTE:** After the status phase of the SET_ADDRESS descriptor, the DCD has 2 ms to program the USBADR field. This mechanism will ensure this specification is met when the DCD can not write of the device address within 2ms from the SET_ADDRESS status phase. If the DCD writes the USBADR with USBADRA=1 after the SET_ADDRESS data phase (before the prime of the status phase), the USBADR will be programmed instantly at the correct time and meet the 2ms USB requirement.
-	This field is reserved. Reserved

11.1.5.24 Next Asynch. Address (USBx_*n*ASYNCLISTADDR)

Host Controller only

This 32-bit register contains the address of the next asynchronous queue head to be executed by the host. Bits [4:0] of this register cannot be modified by the system software and will always return a zero when read.

Address: Base address + 158h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ASYBASE																Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBx_nASYNCLISTADDR field descriptions

Field	Description
31–5 ASYBASE	Link Pointer Low (LPL). These bits correspond to memory address signals [31:5], respectively. This field may only reference a Queue Head (QH). Only used by the host controller.
-	This field is reserved. Reserved

11.1.5.25 Endpoint List Address (USBx_nENDPTLISTADDR)

Device Controller only

In device mode, this register contains the address of the top of the endpoint list in system memory. Bits [10:0] of this register cannot be modified by the system software and will always return a zero when read.

The memory structure referenced by this physical memory pointer is assumed 64-byte.

Address: Base address + 158h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EPBASE																					Reserved										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBx_nENDPTLISTADDR field descriptions

Field	Description
31–11 EPBASE	Endpoint List Pointer(Low). These bits correspond to memory address signals [31:11], respectively. This field will reference a list of up to 32 Queue Head (QH) (that is, one queue head per endpoint & direction).
-	This field is reserved. Reserved

11.1.5.26 Programmable Burst Size (USBx_nBURSTSIZE)

This register is used to control the burst size used during data movement on the AHB master interface. This register is ignored if AHBBRST bits in SBUSCFG register is non-zero value.

Address: Base address + 160h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																TXPBURST								RXPBURST							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB*x*_*n*BURSTSIZE field descriptions

Field	Description
31–17 -	This field is reserved. Reserved
16–8 TXPBURST	Programmable TX Burst Size. Default value is determined by TXBURST bits in n_HWTXBUF. This register represents the maximum length of a the burst in 32-bit words while moving data from system memory to the USB bus.
RXPBURST	Programmable RX Burst Size. Default value is determined by TXBURST bits in n_HWRXBUF. This register represents the maximum length of a the burst in 32-bit words while moving data from the USB bus to system memory.

11.1.5.27 TX FIFO Fill Tuning (USB*x*_*n*TXFILLTUNING)

The fields in this register control performance tuning associated with how the host controller posts data to the TX latency FIFO before moving the data onto the USB bus. The specific areas of performance include the how much data to post into the FIFO and an estimate for how long that operation should take in the target system.

Definitions:

T_0 = Standard packet overhead

T_1 = Time to send data payload

T_{ff} = Time to fetch packet into TX FIFO up to specified level.

T_s = Total Packet Flight Time (send-only) packet

$T_s = T_0 + T_1$

T_p = Total Packet Time (fetch and send) packet

$T_p = T_{ff} + T_0 + T_1$

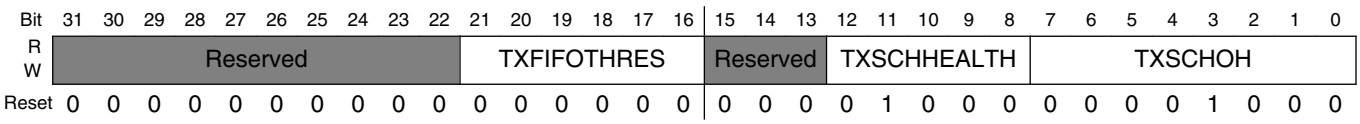
Upon discovery of a transmit (OUT/SETUP) packet in the data structures, host controller checks to ensure T_p remains before the end of the [micro]frame. If so it proceeds to pre-fill the TX FIFO. If at anytime during the pre-fill operation the time remaining the [micro]frame is < T_s then the packet attempt ceases and the packet is tried at a later time. Although this is not an error condition and the host controller will eventually recover, a

mark will be made the scheduler health counter to note the occurrence of a "back-off" event. When a back-off event is detected, the partial packet fetched may need to be discarded from the latency buffer to make room for periodic traffic that will begin after the next SOF. Too many back-off events can waste bandwidth and power on the system bus and thus should be minimized (not necessarily eliminated). Back-offs can be minimized with use of the n_TSCHHEALTH (T_{ff}) described below.

NOTE

The reset value could vary from instance to instance. Please see the detail in bit field description and ignore reset value in summary table in this case!

Address: Base address + 164h offset + (65536d × i), where i=0d to 1d



USB*x*_*n*TXFILLTUNING field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–16 TXFIFOTHRES	FIFO Burst Threshold. (Read/Write) This register controls the number of data bursts that are posted to the TX latency FIFO in host mode before the packet begins on to the bus. The minimum value is 2 and this value should be a low as possible to maximize USB performance. A higher value can be used in systems with unpredictable latency and/or insufficient bandwidth where the FIFO may underrun because the data transferred from the latency FIFO to USB occurs before it can be replenished from system memory. This value is ignored if the Stream Disable bit in USB_n_USBMODE register is set. Default value is '00h' for OTG controller core, and '02h' for Host-only controller core.
15–13 -	This field is reserved. Reserved
12–8 TXSCHHEALTH	Scheduler Health Counter. (Read/Write To Clear) This register increments when the host controller fails to fill the TX latency FIFO to the level programmed by TXFIFOTHRES before running out of time to send the packet before the next Start-Of-Frame. This health counter measures the number of times this occurs to provide feedback to selecting a proper TXSCHOH. Writing to this register will clear the counter and this counter will max. at 31. Default value is for OTG controller core, and '00h' for Host-only controller core.
TXSCHOH	Scheduler Overhead. (Read/Write) [Default = 0] This register adds an additional fixed offset to the schedule time estimator described above as Tff. As an approximation, the value chosen for this register should limit the number of back-off events captured in the TXSCHHEALTH to less than 10 per second in a highly utilized bus. Choosing a value that is too high for this register is not desired as it can needlessly reduce USB utilization. The time unit represented in this register is 1.267us when a device is connected in High-Speed Mode. The time unit represented in this register is 6.333us when a device is connected in Low/Full Speed Mode. Default value is '08h' for OTG controller core, and '00h' for Host-only controller core.

11.1.5.28 Endpoint NAK (USBx_nENDPTNAK)

Address: Base address + 178h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								EPTN								Reserved								EPRN							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBx_nENDPTNAK field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 EPTN	TX Endpoint NAK - R/WC. Each TX endpoint has 1 bit in this field. The bit is set when the device sends a NAK handshake on a received IN token for the corresponding endpoint. Bit [N] - Endpoint #[N], N is 0-7
15–8 -	This field is reserved. Reserved
EPRN	RX Endpoint NAK - R/WC. Each RX endpoint has 1 bit in this field. The bit is set when the device sends a NAK handshake on a received OUT or PING token for the corresponding endpoint. Bit [N] - Endpoint #[N], N is 0-7

11.1.5.29 Endpoint NAK Enable (USBx_nENDPTNAKEN)

Address: Base address + 17Ch offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								EPTNE								Reserved								EPRNE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBx_nENDPTNAKEN field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 EPTNE	TX Endpoint NAK Enable - R/W. Each bit is an enable bit for the corresponding TX Endpoint NAK bit. If this bit is set and the corresponding TX Endpoint NAK bit is set, the NAK Interrupt bit is set. Bit [N] - Endpoint #[N], N is 0-7
15–8 -	This field is reserved. Reserved
EPRNE	RX Endpoint NAK Enable - R/W.

Table continues on the next page...

USBx_*n*ENDPTNAKEN field descriptions (continued)

Field	Description
	Each bit is an enable bit for the corresponding RX Endpoint NAK bit. If this bit is set and the corresponding RX Endpoint NAK bit is set, the NAK Interrupt bit is set. Bit [N] - Endpoint #[N], N is 0-7

11.1.5.30 Configure Flag Register (USBx_*n*CONFIGFLAG)

Address: Base address + 180h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															CF
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

USBx_*n*CONFIGFLAG field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 CF	Configure Flag Host software sets this bit as the last action in its process of configuring the Host Controller. This bit controls the default port-routing control logic. 0 Port routing control logic default-routes each port to an implementation dependent classic host controller. 1 Port routing control logic default-routes all ports to this host controller.

11.1.5.31 Port Status & Control (USBx_*n*PORTSC1)

Host Controller

A host controller could implement one to eight port status and control registers. The number is determined by N_PORTs bits in HWSPARAMs register (please see Host Controller Structural Parameters (USB_*n*HCSPARAMS)). Software could read this parameter register to determine how many ports need service.

All controller cores on this product are Single-Port, so there is only one port status and control register for each controller core.

This register is only reset by power on reset or controller core reset. The initial conditions of a port are:

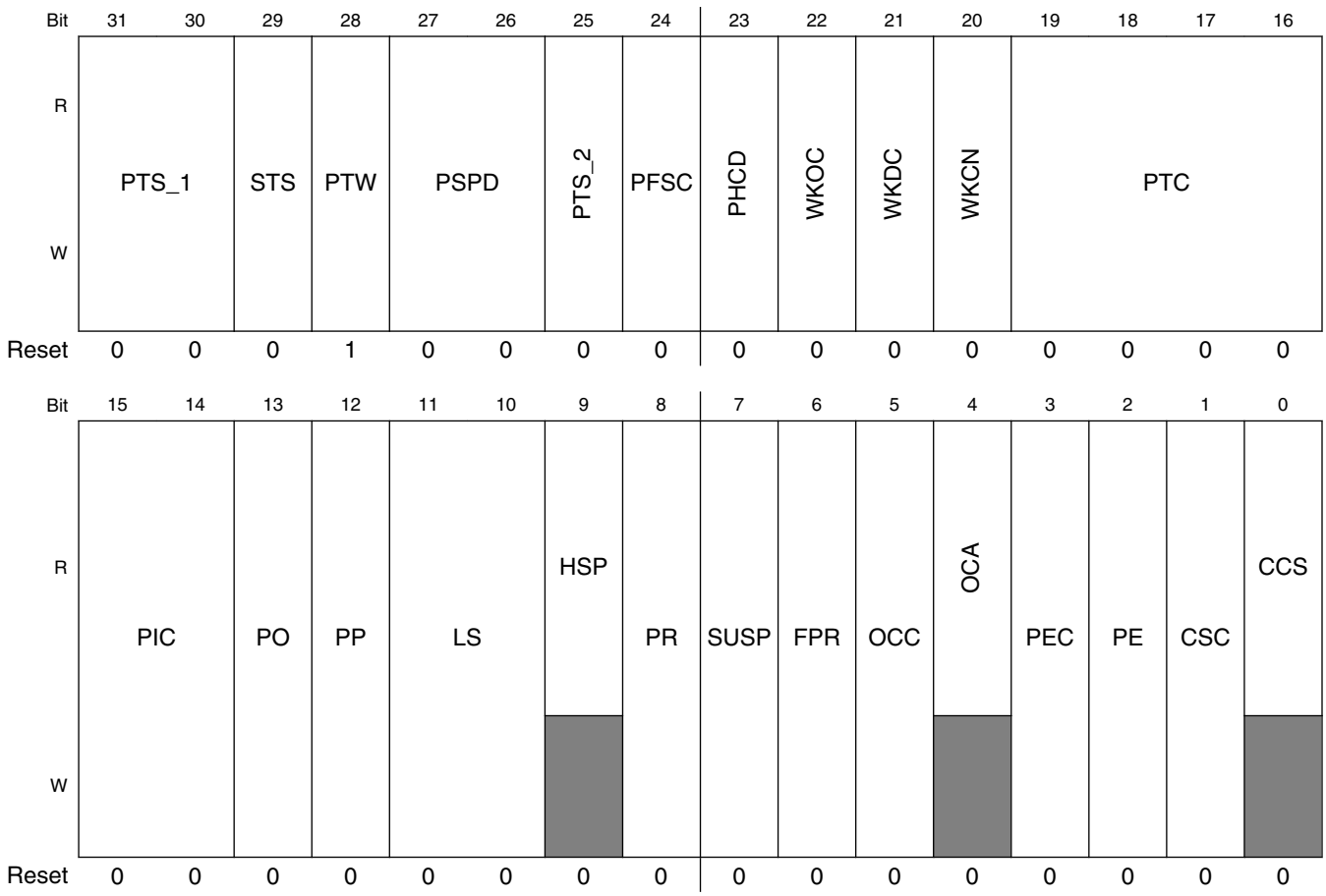
- No device connected
- Port disabled

If the port supports power control, this state remains until port power is supplied (by software).

Device Controller

A controller operating in device mode has only port register one (PORTSC1) and it does not support power control in that mode. Port control in device mode is only used for status port reset, suspend, and current connect status. It is also used to initiate test mode or force signaling and allows software to put the PHY into low power suspend mode and disable the PHY clock.

Address: Base address + 184h offset + (65536d × i), where i=0d to 1d



USBx_nPORTSC1 field descriptions

Field	Description
31–30 PTS_1	Bit field {bit25, bit31, bit30}: "000b" UTMI/UTMI+ "001b" Reserved "010b" ULPI "011b" Serial/USB 1.1 PHY/IC-USB (FS Only) "100b" HSIC Parallel Transceiver Select (bit25, bit31, bi30). For OTG core, it is Read-Only. Reset value is 000b. **NOTE:** All USB port interface modes are listed in this field description, but not all are supported. For detail feature of each controller core, please see Features . The behaviour is unknown when unsupported interface mode is selected.
29 STS	Serial Transceiver Select - Read Only Serial Transceiver Select 1 Serial Interface Engine is selected 0 Parallel Interface signals is selected Serial Interface Engine can be used in combination with UTMI+/ULPI physical interface to provide FS/LS signaling instead of the parallel interface signals. When this bit is set '1b', serial interface engine will be used instead of parallel interface signals. This bit has no effect unless PTS bits is set to select UTMI+/ULPI interface. The Serial/USB1.1 PHY/IC-USB will use the serial interface engine for FS/LS signaling regardless of this bit value.
28 PTW	Parallel Transceiver Width This bit has no effect if serial interface engine is used. For OTG1/OTG2/Host1 core, it is Read-Only. Reset value is '1b'. 0 Select the 8-bit UTMI interface [60MHz] 1 Select the 16-bit UTMI interface [30MHz]
27–26 PSPD	Port Speed - Read Only. This register field indicates the speed at which the port is operating. 00 Full Speed 01 Low Speed 10 High Speed 11 Undefined
25 PTS_2	See description at bits 31-30
24 PFSC	Port Force Full Speed Connect - Read/Write. Default = 0b. When this bit is set to '1b', the port will be forced to only connect at Full Speed, It disables the chirp sequence that allows the port to identify itself as High Speed. 1 Forced to full speed 0 Normal operation

Table continues on the next page...

USBx_*n*PORTSC1 field descriptions (continued)

Field	Description
23 PHCD	PHY Low Power Suspend - Clock Disable (PLPSCD) - Read/Write. Default = 0b. When this bit is set to '1b', the PHY clock is disabled. Reading this bit will indicate the status of the PHY clock. **NOTE:** The PHY clock cannot be disabled if it is being used as the system clock. In device mode, The PHY can be put into Low Power Suspend when the device is not running (USBCMD Run/Stop=0b) or the host has signalled suspend (PORTSC1 SUSPEND=1b). PHY Low power suspend will be cleared automatically when the host initials resume. Before forcing a resume from the device, the device controller driver must clear this bit. In host mode, the PHY can be put into Low Power Suspend when the downstream device has been put into suspend mode or when no downstream device is connected. Low power suspend is completely under the control of software. 1 Disable PHY clock 0 Enable PHY clock
22 WKOC	Wake on Over-current Enable (WKOC_E) - Read/Write. Default = 0b. Writing this bit to a one enables the port to be sensitive to over-current conditions as wake-up events. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero.
21 WKDC	Wake on Disconnect Enable (WKDSCNNT_E) - Read/Write. Default=0b. Writing this bit to a one enables the port to be sensitive to device disconnects as wake-up events. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero or in device mode.
20 WKCN	Wake on Connect Enable (WKCNNT_E) - Read/Write. Default=0b. Writing this bit to a one enables the port to be sensitive to device connects as wake-up events. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero or in device mode.
19–16 PTC	Port Test Control - Read/Write. Default = 0000b. Refer to Port Test Mode for the operational model for using these test modes and the USB Specification Revision 2.0, Chapter 7 for details on each test mode. The FORCE_ENABLE_FS and FORCE ENABLE_LS are extensions to the test mode support specified in the EHCI specification. Writing the PTC field to any of the FORCE_ENABLE_{HS/FS/LS} values will force the port into the connected and enabled state at the selected speed. Writing the PTC field back to TEST_MODE_DISABLE will allow the port state machines to progress normally from that point. **NOTE:** *Low speed operations are not supported as a peripheral device.* Any other value than zero indicates that the port is operating in test mode. Value Specific Test 0000 TEST_MODE_DISABLE 0001 J_STATE 0010 K_STATE 0011 SE0 (host) / NAK (device) 0100 Packet 0101 FORCE_ENABLE_HS 0110 FORCE_ENABLE_FS 0111 FORCE_ENABLE_LS 1000-1111 Reserved

Table continues on the next page...

USBx_nPORTSC1 field descriptions (continued)

Field	Description
15–14 PIC	Port Indicator Control - Read/Write. Default = Ob. Writing to this field has no effect if the P_INDICATOR bit in the HCSPARAMS register is a zero. Refer to the USB Specification Revision 2.0 for a description on how these bits are to be used. This field is zero if *Port Power* is zero. Bit Value Meaning 00 Port indicators are off 01 Amber 10 Green 11 Undefined
13 PO	Port Owner-Read/Write. Default = 0. This bit unconditionally goes to a 0 when the configured bit in the CONFIGFLAG register makes a 0 to 1 transition. This bit unconditionally goes to 1 whenever the Configured bit is zero System software uses this field to release ownership of the port to a selected host controller (in the event that the attached device is not a high-speed device). Software writes a one to this bit when the attached device is not a high-speed device. A one in this bit means that an internal companion controller owns and controls the port. Port owner handoff is not supported in all controller cores, therefore this bit will always be 0.
12 PP	Port Power (PP)-Read/Write or Read Only. The function of this bit depends on the value of the Port Power Switching (PPC) field in the HCSPARAMS register. The behavior is as follows: PPC PP Operation 0 1b *Read Only - Host controller does not have port power control switches. Each port is hard-wired to power.* 1 1b/0b - *Read/Write. Host/OTG controller requires port power control switches. This bit represents the current setting of the switch (0=off, 1=on). When power is not available on a port (that is, PP equals a 0), the port is non-functional and will not report attaches, detaches, etc.* When an over-current condition is detected on a powered port and PPC is a one, the PP bit in each affected port may be transitional by the host controller driver from a one to a zero (removing power from the port). This feature is implemented in all controller cores (PPC = 1).
11–10 LS	Line Status-Read Only. These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. In host mode, the use of linestate by the host controller driver is not necessary (unlike EHCI), because the port controller state machine and the port routing manage the connection of LS and FS. In device mode, the use of linestate by the device controller driver is not necessary. The encoding of the bits are: Bits [11:10] Meaning 00 SE0 10 J-state

Table continues on the next page...

USBx_*n*PORTSC1 field descriptions (continued)

Field	Description
	01 K-state 11 Undefined
9 HSP	High-Speed Port - Read Only. Default = 0b. When the bit is one, the host/device connected to the port is in high-speed mode and if set to zero, the host/device connected to the port is not in a high-speed mode. **NOTE:** HSP is redundant with PSPD(bit 27, 26) but remained for compatibility.
8 PR	Port Reset - Read/Write or Read Only. Default = 0b. In Host Mode: Read/Write. 1=Port is in Reset. 0=Port is not in Reset. Default 0. When software writes a one to this bit the bus-reset sequence as defined in the USB Specification Revision 2.0 is started. *This bit will automatically change to zero after the reset sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the reset duration is timed in the driver.* In Device Mode: This bit is a read only status bit. Device reset from the USB bus is also indicated in the USBSTS register. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero.
7 SUSP	Suspend - Read/Write or Read Only. Default = 0b. 1=Port in suspend state. 0=Port not in suspend state. In Host Mode: Read/Write. Port Enabled Bit and Suspend bit of this register define the port states as follows: Bits [Port Enabled, Suspend] Port State 0x Disable 10 Enable 11 Suspend When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB. The host controller will unconditionally set this bit to zero when software sets the *Force Port Resume* bit to zero. The host controller ignores a write of zero to this bit. If host software sets this bit to a one when the port is not enabled (that is, *Port enabled* bit is a zero) the results are undefined. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero in host mode. In Device Mode: Read Only. In device mode this bit is a read only status bit.
6 FPR	Force Port Resume -Read/Write. 1= Resume detected/driven on port. 0=No resume (K-state) detected/driven on port. Default = 0. In Host Mode: Software sets this bit to one to drive resume signaling. The Host Controller sets this bit to one if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to a one because a J-to-K transition is detected, the *Port Change Detect* bit in the USBSTS register is also set to one. *This bit*

Table continues on the next page...

USBx_nPORTSC1 field descriptions (continued)

Field	Description
	will automatically change to zero after the resume sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the resume duration is timed in the driver. Note that when the Host controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed 'K') is driven on the port as long as this bit remains a one. This bit will remain a one until the port has switched to the high-speed idle. Writing a zero has no effect because the port controller will time the resume operation, clear the bit the port control state switches to HS or FS idle. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero in host mode. This bit is not-EHCI compatible. In Device mode: After the device has been in Suspend State for 5ms or more, software must set this bit to one to drive resume signaling before clearing. The Device Controller will set this bit to one if a J-to-K transition is detected while the port is in the Suspend state. The bit will be cleared when the device returns to normal operation. Also, when this bit wil be cleared because a K-to-J transition detected, the *Port Change Detect* bit in the USBSTS register is also set to one.
5 OCC	Over-current Change-R/WC. Default=0. This bit is set '1b' by hardware when there is a change to Over-current Active. Software can clear this bit by writing a one to this bit position.
4 OCA	Over-current Active-Read Only. Default 0. This bit will automatically transition from one to zero when the over current condition is removed. 1 This port currently has an over-current condition 0 This port does not have an over-current condition.
3 PEC	Port Enable/Disable Change-R/WC. 1=Port enabled/disabled status has changed. 0=No change. Default = 0. In Host Mode: For the root hub, this bit is set to a one only when a port is disabled due to disconnect on the port or due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification). Software clears this by writing a one to it. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero. In Device mode: The device port is always enabled, so this bit is always '0b'.
2 PE	Port Enabled/Disabled-Read/Write. 1=Enable. 0=Disable. Default 0. In Host Mode: Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a one to this field. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by the host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events. When the port is disabled, (0b) downstream propagation of data is blocked except for reset. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero in host mode. In Device Mode: The device port is always enabled, so this bit is always '1b'.

Table continues on the next page...

USBx_*n*PORTSC1 field descriptions (continued)

Field	Description
1 CSC	Connect Status Change-R/WC. 1 =Change in Current Connect Status. 0=No change. Default 0. In Host Mode: Indicates a change has occurred in the port's Current Connect Status. The host/device controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be 'setting' an already-set bit (that is, the bit will remain set). Software clears this bit by writing a one to it. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero in host mode. In Device Mode: This bit is undefined in device controller mode.
0 CCS	Current Connect Status-Read Only. In Host Mode: 1=Device is present on port. 0=No device is present. Default = 0. This value reflects the current state of the port, and may not correspond directly to the event that caused the *Connect Status Change* bit (Bit 1) to be set. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero in host mode. In Device Mode: 1=Attached. 0=Not Attached. Default=0. A one indicates that the device successfully attached and is operating in either high speed or full speed as indicated by the High Speed Port bit in this register. A zero indicates that the device did not attach successfully or was forcibly disconnected by the software writing a zero to the Run bit in the USBCMD register. It does not state the device being disconnected or suspended.

11.1.5.32 On-The-Go Status & control (USBx_*n*OTGSC)

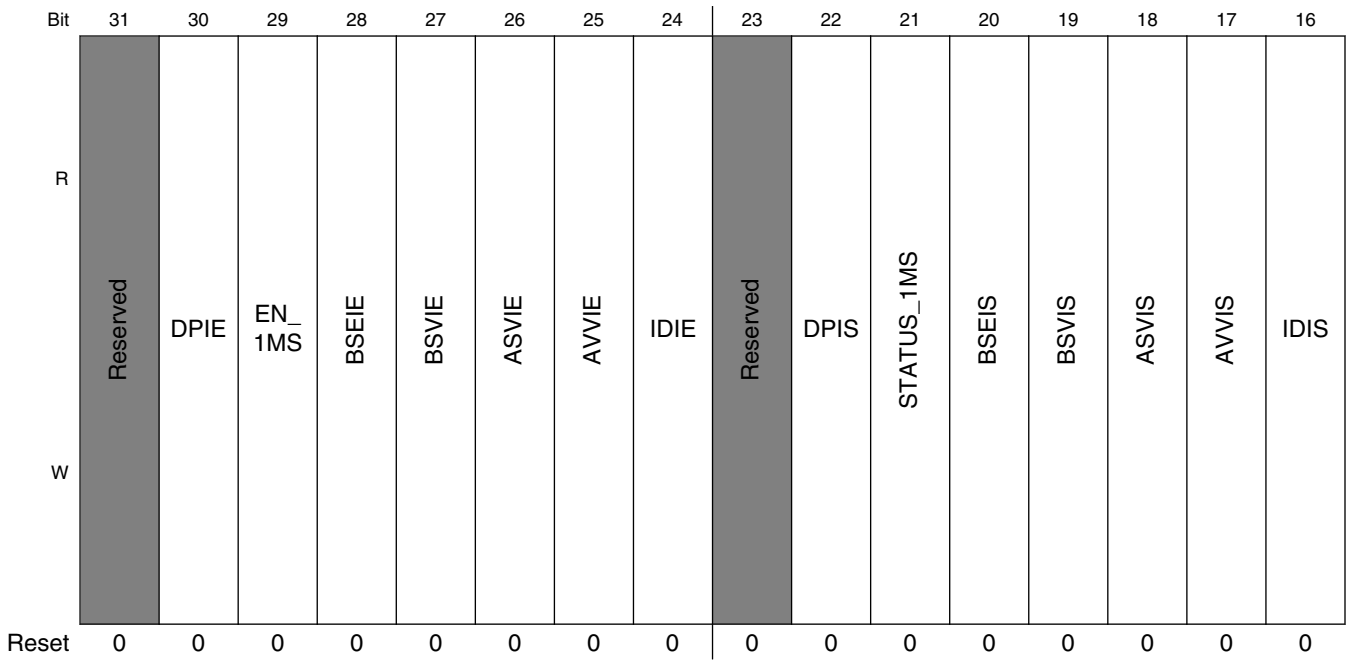
This register is availabe only in OTG controller core. It has four sections:

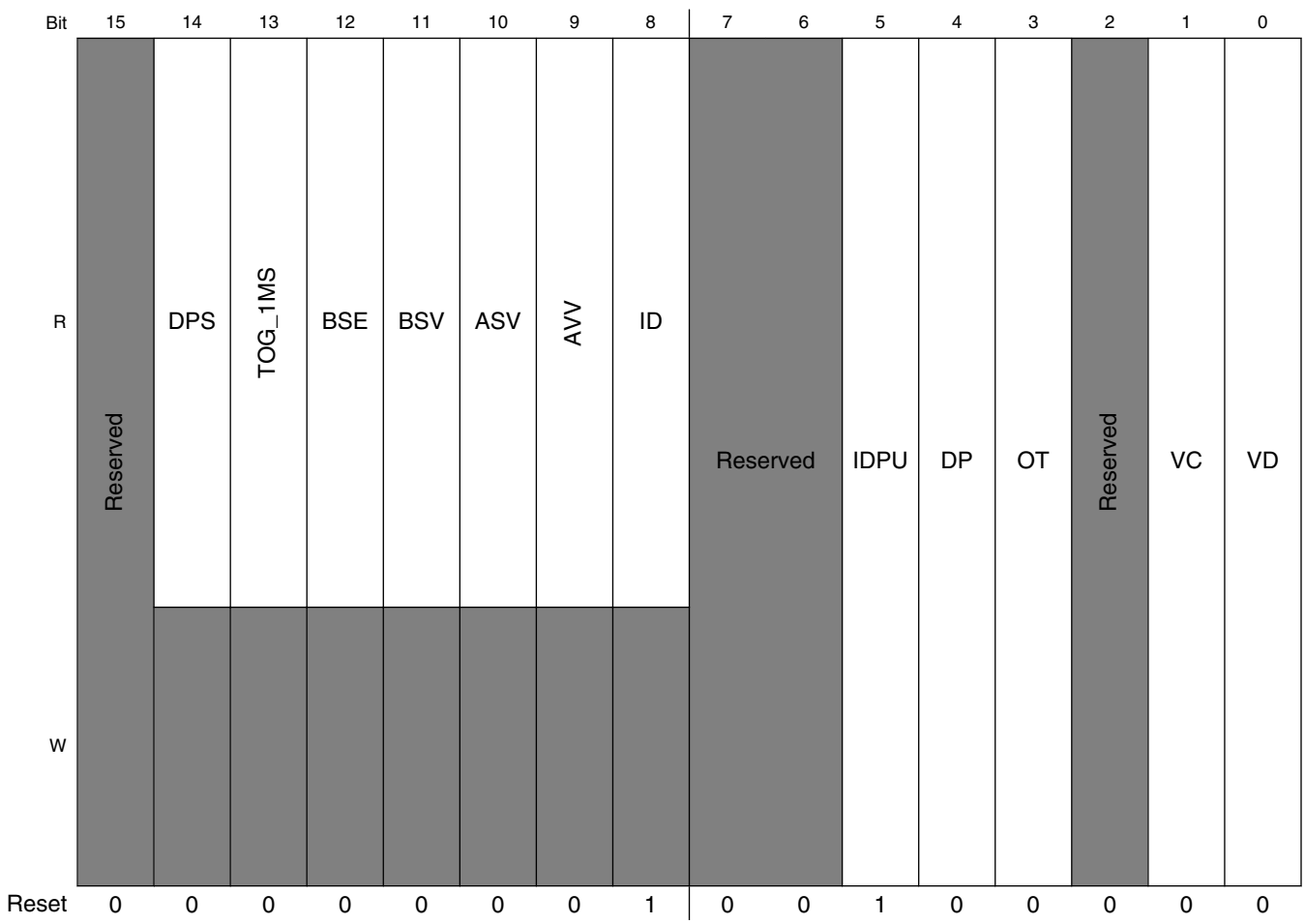
- OTG Interrupt enables (Read/Write)
- OTG Interrupt status (Read/Write to Clear)
- OTG Status inputs (Read Only)
- OTG Controls (Read/Write)

The status inputs are debounced using a 1 ms time constant. Values on the status inputs that do not persist for more than 1 ms does not cause an update of the status input register, or cause an OTG interrupt.

See also USB Device Mode (USB_*n*USBMODE) register.

Address: Base address + 1A4h offset + (65536d × i), where i=0d to 1d





USBx_nOTGSC field descriptions

Field	Description
31 -	This field is reserved. Reserved
30 DPIE	Data Pulse Interrupt Enable
29 EN_1MS	1 millisecond timer Interrupt Enable - Read/Write
28 BSEIE	B Session End Interrupt Enable - Read/Write. Setting this bit enables the B session end interrupt.
27 BSVIE	B Session Valid Interrupt Enable - Read/Write. Setting this bit enables the B session valid interrupt.
26 ASVIE	A Session Valid Interrupt Enable - Read/Write. Setting this bit enables the A session valid interrupt.
25 AVVIE	A VBus Valid Interrupt Enable - Read/Write. Setting this bit enables the A VBus valid interrupt.
24 IDIE	USB ID Interrupt Enable - Read/Write. Setting this bit enables the USB ID interrupt.
23 -	This field is reserved. Reserved
22 DPIS	Data Pulse Interrupt Status - Read/Write to Clear. This bit is set when data bus pulsing occurs on DP or DM. Data bus pulsing is only detected when USBMODE.CM = Host (11) and PORTSC1(0)[PP] = 0. Software must write a one to clear this bit.
21 STATUS_1MS	1 millisecond timer Interrupt Status - Read/Write to Clear. This bit is set once every millisecond. Software must write a one to clear this bit.
20 BSEIS	B Session End Interrupt Status - Read/Write to Clear. This bit is set when VBus has fallen below the B session end threshold. Software must write a one to clear this bit.
19 BSVIS	B Session Valid Interrupt Status - Read/Write to Clear. This bit is set when VBus has either risen above or fallen below the B session valid threshold. Software must write a one to clear this bit.
18 ASVIS	A Session Valid Interrupt Status - Read/Write to Clear. This bit is set when VBus has either risen above or fallen below the A session valid threshold. Software must write a one to clear this bit.
17 AVVIS	A VBus Valid Interrupt Status - Read/Write to Clear. This bit is set when VBus has either risen above or fallen below the VBus valid threshold on an A device. Software must write a one to clear this bit.
16 IDIS	USB ID Interrupt Status - Read/Write. This bit is set when a change on the ID input has been detected. Software must write a one to clear this bit.

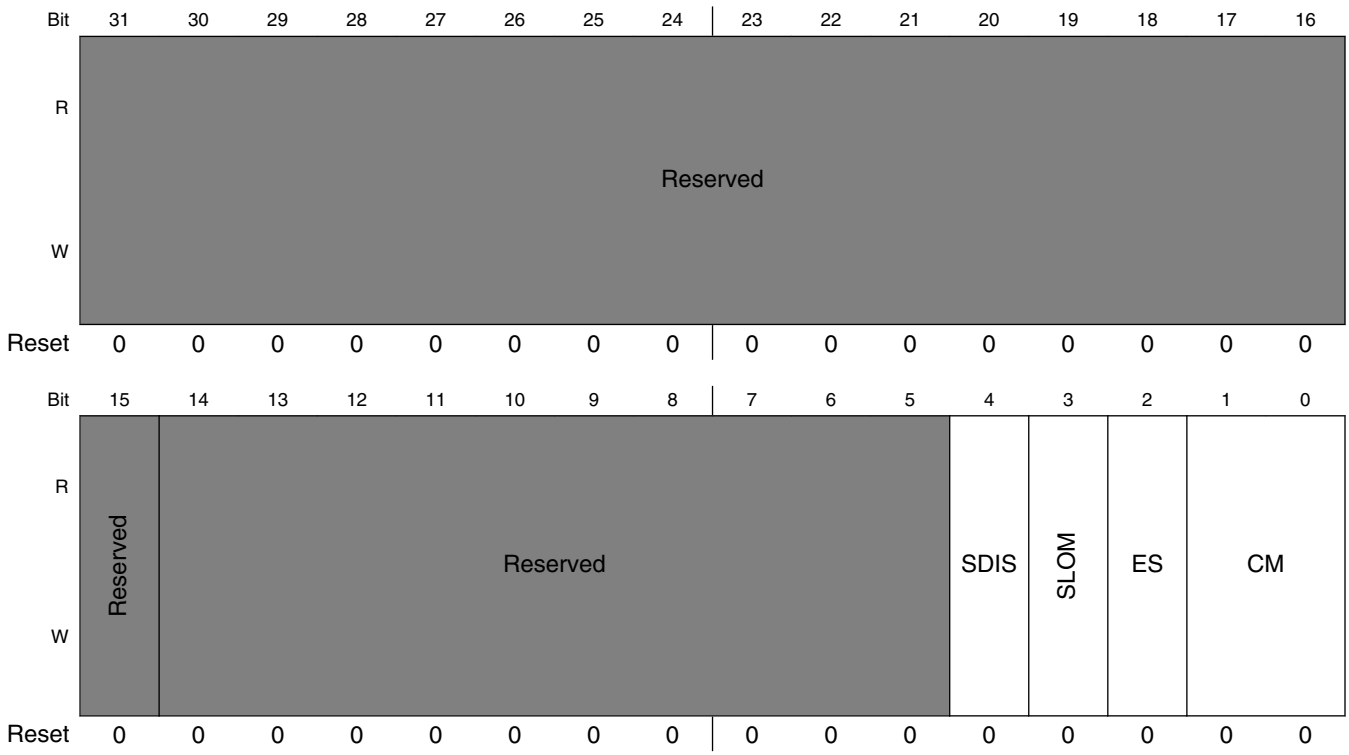
Table continues on the next page...

USB*x*_*n*OTGSC field descriptions (continued)

Field	Description
15 -	This field is reserved. Reserved
14 DPS	Data Bus Pulsing Status - Read Only. A '1' indicates data bus pulsing is being detected on the port.
13 TOG_1MS	1 millisecond timer toggle - Read Only. This bit toggles once per millisecond.
12 BSE	B Session End - Read Only. Indicates VBus is below the B session end threshold.
11 BSV	B Session Valid - Read Only. Indicates VBus is above the B session valid threshold.
10 ASV	A Session Valid - Read Only. Indicates VBus is above the A session valid threshold.
9 AVV	A VBus Valid - Read Only. Indicates VBus is above the A VBus valid threshold.
8 ID	USB ID - Read Only. 0 = A device, 1 = B device
7–6 -	This field is reserved. Reserved
5 IDPU	ID Pullup - Read/Write This bit provide control over the ID pull-up resistor; 0 = off, 1 = on [default]. When this bit is 0, the ID input will not be sampled.
4 DP	Data Pulsing - Read/Write. Setting this bit causes the pullup on DP to be asserted for data pulsing during SRP.
3 OT	OTG Termination - Read/Write. This bit must be set when the OTG device is in device mode, this controls the pulldown on DM.
2 -	This field is reserved. Reserved
1 VC	VBUS Charge - Read/Write. Setting this bit causes the VBus line to be charged. This is used for VBus pulsing during SRP.
0 VD	VBUS_Discharge - Read/Write. Setting this bit causes VBus to discharge through a resistor.

11.1.5.33 USB Device Mode (USBx_*n*USBMODE)

Address: Base address + 1A8h offset + (65536d × i), where i=0d to 1d



USBx_*n*USBMODE field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 -	This field is reserved. Reserved
14–5 -	This field is reserved. Reserved
4 SDIS	Stream Disable Mode. (0 - Inactive [default]; 1 - Active) Device Mode: Setting to a '1' disables double priming on both RX and TX for low bandwidth systems. This mode ensures that when the RX and TX buffers are sufficient to contain an entire packet that the standard double buffering scheme is disabled to prevent overruns/underruns in bandwidth limited systems. Note: In High Speed Mode, all packets received are responded to with a NYET handshake when stream disable is active. Host Mode: Setting to a '1' ensures that overruns/underruns of the latency FIFO are eliminated for low bandwidth systems where the RX and TX buffers are sufficient to contain the entire packet. Enabling stream disable also has the effect of ensuring the TX latency is filled to capacity before the packet is launched onto the USB. **NOTE:** Time duration to pre-fill the FIFO becomes significant when stream disable is active. See TX FIFO Fill Tuning (USB_*n*TXFILLTUNING) and TXTTFILLTUNING [MPH Only] to characterize the adjustments needed for the scheduler when using this feature.

Table continues on the next page...

USBx_nUSBMODE field descriptions (continued)

Field	Description
	NOTE: The use of this feature substantially limits of the overall USB performance that can be achieved.
3 SLOM	Setup Lockout Mode. In device mode, this bit controls behavior of the setup lock mechanism. See Control Endpoint Operation Model . 0 Setup Lockouts On (default); 1 Setup Lockouts Off (DCD requires use of Setup Data Buffer Tripwire in USB Command Register (USB_nUSBCMD) .
2 ES	Endian Select - Read/Write. This bit can change the byte alignment of the transfer buffers to match the host microprocessor. The bit fields in the microprocessor interface and the data structures are unaffected by the value of this bit because they are based upon the 32-bit word. Bit Meaning 0 Little Endian [Default] 1 Big Endian
CM	Controller Mode - R/WO. Controller mode is defaulted to the proper mode for host only and device only implementations. For those designs that contain both host & device capability, the controller defaults to an idle state and needs to be initialized to the desired operating mode after reset. For combination host/device controllers, this register can only be written once after reset. If it is necessary to switch modes, software must reset the controller by writing to the *RESET* bit in the USBCMD register before reprogramming this register. For OTG controller core, reset value is '00b'. For Host-only controller core, reset value is '11b'. 00 Idle [Default for combination host/device] 01 Reserved 10 Device Controller [Default for device only controller] 11 Host Controller [Default for host only controller]

11.1.5.34 Endpoint Setup Status (USBx_nENDPTSETUPSTAT)

Address: Base address + 1ACh offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																ENDPTSETUPSTAT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBx_nENDPTSETUPSTAT field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
ENDPTSETUPSTAT	Setup Endpoint Status. For every setup transaction that is received, a corresponding bit in this register is set to one. Software must clear or acknowledge the setup transfer by writing a one to a respective bit after it has read the setup data from Queue head. The response to a setup packet as in the order of operations and total response time is crucial to limit bus time outs while the setup lock our mechanism is engaged. See Managing Endpoints in the Device Operational Model. This register is only used in device mode.

11.1.5.35 Endpoint Prime (USB*x*_*n*ENDPTPRIME)

This register is only used in device mode.

When software sets the prime bit for a given endpoint, the device controller loads the transfer descriptor, pointed to by the queue head, such that the endpoint is ready to transmit or receive when the host sends a request (IN/OUT token). The endpoint will NAK all requests from the host until the endpoint is primed. The controller will automatically re-prime the endpoint with a new transfer descriptor when one is found via the next_dtd pointer of the current transfer descriptor. Hence, the prime bit must only be set by software when a descriptor is added to the queue head.

Address: Base address + 1B0h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								PETB								Reserved								PERB							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB*x*_*n*ENDPTPRIME field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 PETB	Prime Endpoint Transmit Buffer - R/WS. For each endpoint a corresponding bit is used to request that a buffer is prepared for a transmit operation in order to respond to a USB IN/INTERRUPT transaction. Software should write a one to the corresponding bit when posting a new transfer descriptor to an endpoint queue head. Hardware automatically uses this bit to begin parsing for a new transfer descriptor from the queue head and prepare a transmit buffer. Hardware clears this bit when the associated endpoint(s) is (are) successfully primed. **NOTE:** These bits are momentarily set by hardware during hardware re-priming operations when a dTD is retired, and the dQH is updated. PETB[N] - Endpoint #N, N is in 0..7
15–8 -	This field is reserved. Reserved
PERB	Prime Endpoint Receive Buffer - R/WS. For each endpoint, a corresponding bit is used to request a buffer prepare for a receive operation for when a USB host initiates a USB OUT transaction. Software should write a one to the corresponding bit whenever posting a new transfer descriptor to an endpoint queue head. Hardware automatically uses this bit to begin parsing for a new transfer descriptor from the queue head and prepare a receive buffer. Hardware clears this bit when the associated endpoint(s) is (are) successfully primed. **NOTE:** These bits are momentarily set by hardware during hardware re-priming operations when a dTD is retired, and the dQH is updated. PERB[N] - Endpoint #N, N is in 0..7

11.1.5.36 Endpoint Flush (USB*x*_*n*ENDPTFLUSH)

This register is only used in device mode.

Address: Base address + 1B4h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								FETB								Reserved								FERB							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB*x*_*n*ENDPTFLUSH field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 FETB	Flush Endpoint Transmit Buffer - R/WS. Writing one to a bit(s) in this register causes the associated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer continues until completion. Hardware clears this register after the endpoint flush operation is successful. FETB[N] - Endpoint #N, N is in 0..7
15–8 -	This field is reserved. Reserved
FERB	Flush Endpoint Receive Buffer - R/WS. Writing one to a bit(s) causes the assocUOGiated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer continues until completion. Hardware clears this register after the endpoint flush operation is successful. FERB[N] - Endpoint #N, N is in 0..7\

11.1.5.37 Endpoint Status (USB*x*_*n*ENDPTSTAT)

This register is only used in device mode.

Address: Base address + 1B8h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								ETBR								Reserved								ERBR							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB*x*_*n*ENDPTSTAT field descriptions

Field	Description
31–24 -	This field is reserved. Reserved

Table continues on the next page...

USB*x*_*n*ENDPTSTAT field descriptions (continued)

Field	Description
23–16 ETBR	Endpoint Transmit Buffer Ready -- Read Only. One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set to one by the hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. There is always a delay between setting a bit in the ENDPTPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the ENDPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTFLUSH register. **NOTE:** These bits are momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated. ETBR[N] - Endpoint #N, N is in 0..7
15–8 -	This field is reserved. Reserved
ERBR	Endpoint Receive Buffer Ready -- Read Only. One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set to a one by the hardware as a response to receiving a command from a corresponding bit in the ENDPRIME register. There is always a delay between setting a bit in the ENDPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the ENDPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTFLUSH register. **NOTE:** These bits are momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated. ERBR[N] - Endpoint #N, N is in 0..7

11.1.5.38 Endpoint Complete (USB*x*_*n*ENDPTCOMPLETE)

This register is only used in device mode.

Address: Base address + 1BCh offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								ETCE								Reserved								ERCE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB*x*_*n*ENDPTCOMPLETE field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 ETCE	Endpoint Transmit Complete Event - R/WC. Each bit indicates a transmit event (IN/INTERRUPT) occurred and software should read the corresponding endpoint queue to determine the endpoint status. If the corresponding IOC bit is set in the Transfer Descriptor, then this bit is set simultaneously with the *USBINT* . Writing one clears the corresponding bit in this register. ETCE[N] - Endpoint #N, N is in 0..7
15–8 -	This field is reserved. Reserved

Table continues on the next page...

USB*x*_*n*ENDPTCOMPLETE field descriptions (continued)

Field	Description
ERCE	Endpoint Receive Complete Event - RW/C. Each bit indicates a received event (OUT/SETUP) occurred and software should read the corresponding endpoint queue to determine the transfer status. If the corresponding IOC bit is set in the Transfer Descriptor, then this bit is set simultaneously with the *USBINT* . Writing one clears the corresponding bit in this register. ERCE[N] - Endpoint #N, N is in 0..7

11.1.5.39 Endpoint Control0 (USB*x*_*n*ENDPTCTRL0)

Every Device implements Endpoint 0 as a control endpoint.

Address: Base address + 1C0h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	Reserved			TXT		Reserved	TXS
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	Reserved			RXT		Reserved	RXS
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

USB*x*_*n*ENDPTCTRL0 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 1 Enabled Endpoint0 is always enabled.

Table continues on the next page...

USBx_nENDPTCTRL0 field descriptions (continued)

Field	Description
22–20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 - Control Endpoint0 is fixed as a Control End Point.
17 -	This field is reserved. Reserved
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK [Default] 1 End Point Stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues returning STALL until the bit is cleared by software or it is automatically cleared upon receipt of a new SETUP request. After receiving a SETUP request, this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. **NOTE:** There is a slight delay (50 clocks max.) between the endptsetupstat being cleared and hardware continuing to clear this bit. In most systems it is unlikely the DCD software will observe this delay. However, should the dcd observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a newsetup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 1 Enabled Endpoint0 is always enabled.
6–4 -	This field is reserved. Reserved
3–2 RXT	RX Endpoint Type - Read/Write 00 Control Endpoint0 is fixed as a Control End Point.
1 -	This field is reserved. Reserved
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues returning STALL until the bit is cleared by software or it is automatically cleared upon receipt of a new SETUP request. After receiving a SETUP request, this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared.

Table continues on the next page...

USBx_nENDPTCTRL0 field descriptions (continued)

Field	Description
	NOTE: There is a slight delay (50 clocks max.) between the endptsetupstat being cleared and hardware continuing to clear this bit. In most systems it is unlikely the dcd software will observe this delay. However, should the dcd observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a newsetup has been received by checking the associated endptsetupstat bit.

11.1.5.40 Endpoint Control 1 (USBx_nENDPTCTRL1)

This is endpoint control register for endpoint 1 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: Base address + 1C4h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBx_nENDPTCTRL1 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved

Table continues on the next page...

USBx_nENDPTCTRL1 field descriptions (continued)

Field	Description
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

11.1.5.41 Endpoint Control 2 (USBx_nENDPTCTRL2)

This is endpoint control register for endpoint 2 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: Base address + 1C8h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBx_nENDPTCTRL2 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.

Table continues on the next page...

USBx_nENDPTCTRL2 field descriptions (continued)

Field	Description
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence

Table continues on the next page...

USBx_nENDPTCTRL2 field descriptions (continued)

Field	Description
	Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

11.1.5.42 Endpoint Control 3 (USBx_nENDPTCTRL3)

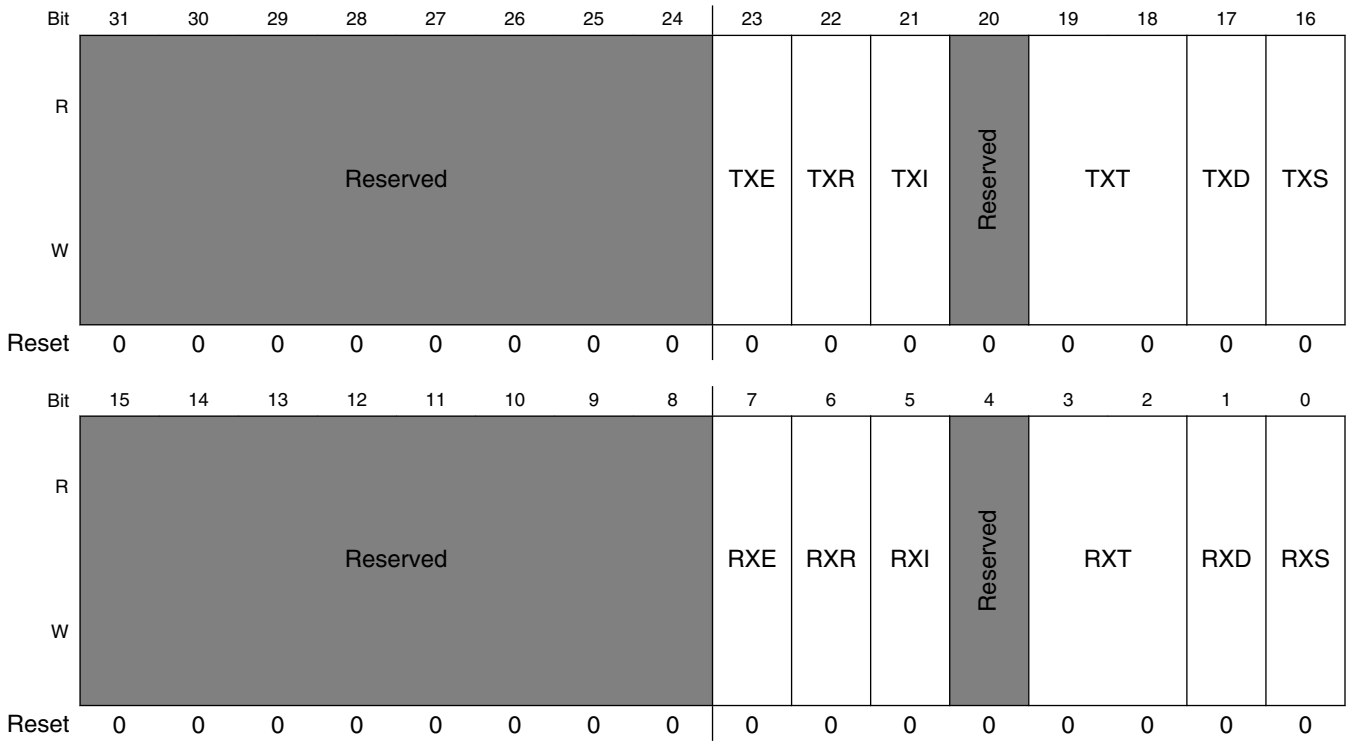
This is endpoint control register for endpoint 3 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control

causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: Base address + 1CCh offset + (65536d × i), where i=0d to 1d



USB*x*_*n*ENDPTCTRL3 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved

Table continues on the next page...

USBx_nENDPTCTRL3 field descriptions (continued)

Field	Description
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control

Table continues on the next page...

USBx_nENDPTCTRL3 field descriptions (continued)

Field	Description
	01 Isochronous 10 Bulk 11
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

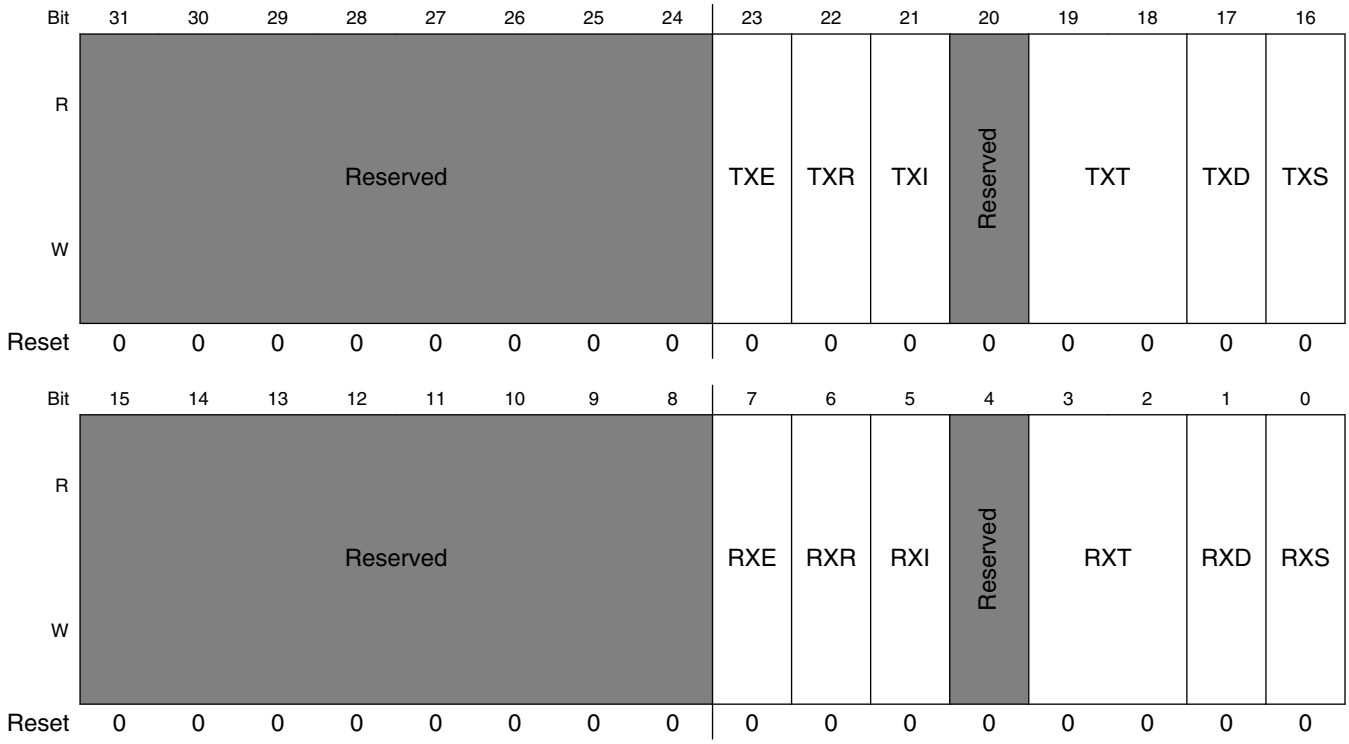
11.1.5.43 Endpoint Control 4 (USBx_nENDPTCTRL4)

This is endpoint control register for endpoint 4 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: Base address + 1D0h offset + (65536d × i), where i=0d to 1d



USBx_nENDPTCTRL4 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous

Table continues on the next page...

USBx_nENDPTCTRL4 field descriptions (continued)

Field	Description
	10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved

Table continues on the next page...

USBx_nENDPTCTRL4 field descriptions (continued)

Field	Description
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

11.1.5.44 Endpoint Control 5 (USBx_nENDPTCTRL5)

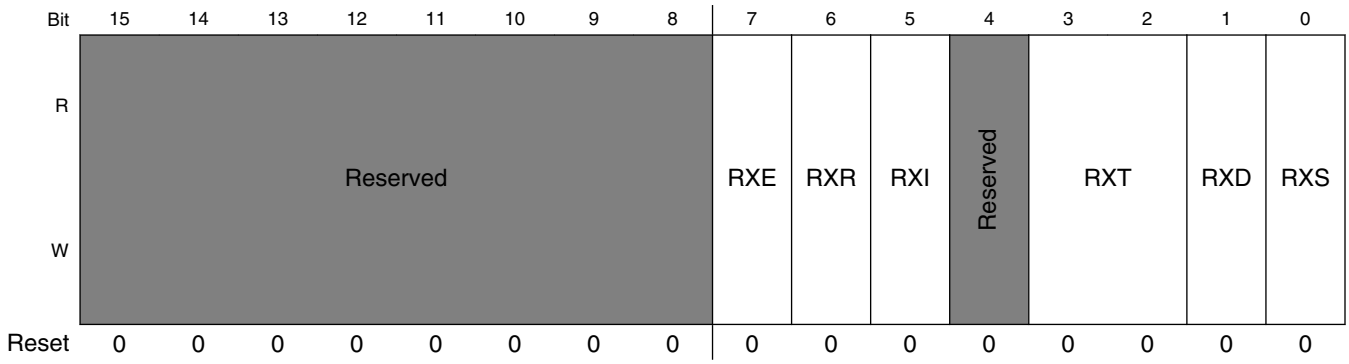
This is endpoint control register for endpoint 5 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: Base address + 1D4h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



USBx_nENDPTCTRL5 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled

Table continues on the next page...

USBx_nENDPTCTRL5 field descriptions (continued)

Field	Description
	This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled

Table continues on the next page...

USBx_nENDPTCTRL5 field descriptions (continued)

Field	Description
	This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

11.1.5.45 Endpoint Control 6 (USBx_nENDPTCTRL6)

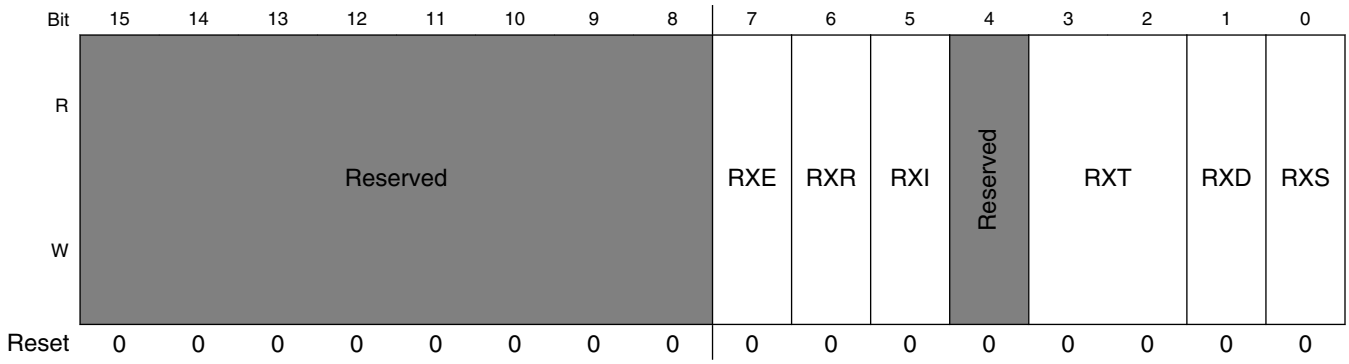
This is endpoint control register for endpoint 6 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: Base address + 1D8h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



USBx_nENDPTCTRL6 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled

Table continues on the next page...

USBx_nENDPTCTRL6 field descriptions (continued)

Field	Description
	This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled

Table continues on the next page...

USBx_nENDPTCTRL6 field descriptions (continued)

Field	Description
	This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

11.1.5.46 Endpoint Control 7 (USBx_nENDPTCTRL7)

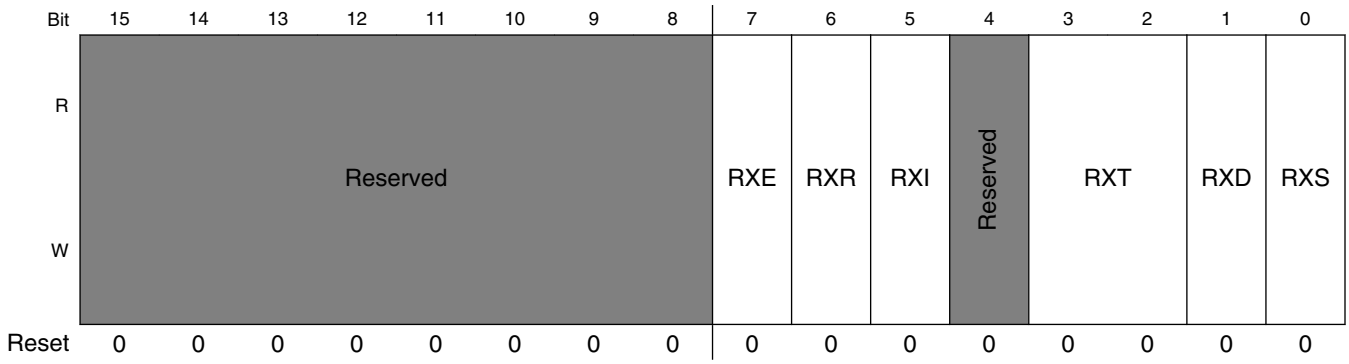
This is endpoint control register for endpoint 7 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: Base address + 1DCh offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



USBx_nENDPTCTRL7 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled

Table continues on the next page...

USBx_nENDPTCTRL7 field descriptions (continued)

Field	Description
	This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled

Table continues on the next page...

USB*x*_*n*ENDPTCTRL7 field descriptions (continued)

Field	Description
	This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

11.2 Universal Serial Bus 2.0 PHY (USB2_PHY)

11.2.1 Overview

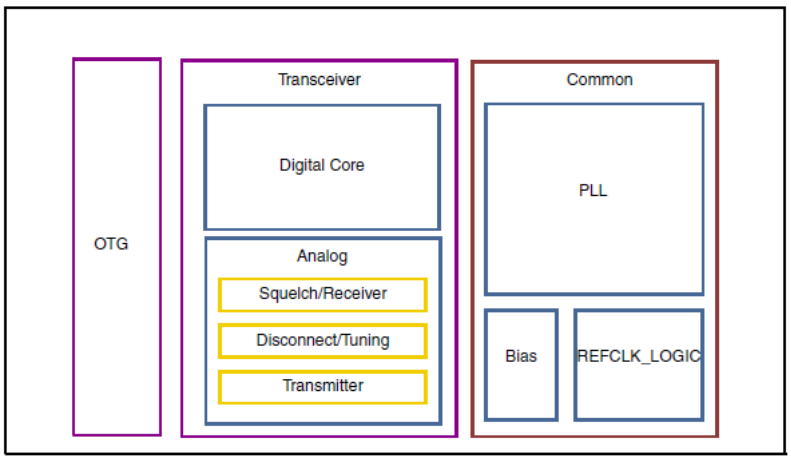
The USB 2.0 PHY, connects a USB OTG controller to a USB system. This chapter provides an introduction to the USB 2.0 PHY and its features.

NOTE

The information within this block guide is Synopsys Proprietary. Used with permission.

11.2.1.1 Block Diagram

The following figure shows the USB 2.0 PHY functional block diagram for a one-port macro.



11.2.1.2 Features

The USB 2.0 PHY provides the following features.

General Features:

- Implements low power dissipation while active, idle, or on standby
- Fully integrates high-, full-, and low-speed (Host mode only) termination and signal switching
- Implements one parallel data interface and clock for high-, full-, and low-speed (Host mode only) USB data transfers
- Provides on-chip PLL to reduce clock noise and eliminate the need for an external clock generator
- Provides Universal Asynchronous Receiver/Transmitter (UART) support to enable the USB 2.0 PHY to receive and transmit asynchronous, serial data

USB 2.0 Features:

- Supports 480-Mbps high-speed, 12-Mbps full-speed, and 1.5-Mbps low-speed (Host mode only) data transmission rates
- Supports 8/16-bit unidirectional parallel interfaces for HS, FS, and LS (Host mode only) modes of operation, in accordance with the UTMI+ specification
- Provides dual (HS/FS) mode host support
- Implements data recovery from serial data on the USB connector
- Implements SYNC/End-of-Packet (EOP) generation and checking
- Implements Non Return to Zero Invert (NRZI) encoding and decoding Implements logic to support suspend, sleep, resume, and remote wakeup operations
- Supports battery charging to enable devices to draw current in excess of the USB 2.0 specification for charging and/or powering up from dedicated charging ports or charging downstream ports

11.2.1.3 Interfaces

Each USB 2.0 PHY port has three distinct external interfaces:

- USB Data Plus (D+) and Data Minus (D–) lines: These lines are USB 1.1 and 2.0 specificationcompliant. The USB 2.0 PHY supports high-speed, 480-Mbps transfers, as well as USB 1.1 fullspeed and low-speed transfers through the USB.
- USB 2.0 Transceiver Macrocell Interface (UTMI+): The USB 2.0 PHY supports the following modes through the UTMI+:
  - High-Speed (HS)
  - Full-Speed (FS)
  - Low-Speed Preamble (LS Preamble)

  The UTMI+ contains a receive port, a transmit port, and associated control lines to interface with a USB host controller or device controller. The receive and transmit ports can be configured as 8/16-bit parallel ports for all applicable modes of operation.
- Serial interface: This is the USB 1.1 controller interface that supports full-speed (FS-Serial mode) and low-speed (LS-Serial mode) data transmission rates to and from a USB 1.1 controller.

The USB 2.0 PHY handles low-level USB protocol and signaling. The USB 2.0 PHY supports SYNC detection, data serialization and deserialization, and data recovery.

11.2.1.4 Standards Compliance

The USB 2.0 PHY complies with the following standards:

- Universal Serial Bus Specification, Revision 2.0, with a High-Speed (HS) Transmit to Receive Inter- Packet Delay requirement of 38 bit-times provided by the UTMI specification version 1.05.
- Battery Charging Specification, Revision 1.2, USB Implementers Forum, Inc.
- On-The-Go and Embedded Host Supplement to the USB Revision 2.0 Specification, Revision 2.0, USB Implementers Forum, Inc. ADP (Attach Detach Protocol) is not supported.
- UTMI+ Specification, Revision 1.0 (Level 3)

11.2.2 Architecture

This section describes the high-level architecture of the USB 2.0 PHY.

11.2.2.1 Block Descriptions

The USB 2.0 PHY provides the following features.

As shown in Block Diagram, the USB 2.0 PHY consists of three basic components: the Common block, Transceiver block, and OTG block.

- Common block: This block contains circuitry that provides biasing and reference clocks for each Transceiver block.
- Transceiver block: This block contains circuitry for data processing and transfers.
- OTG block: This block contains circuitry to initiate the Session Request Protocol (SRP) and the Host Negotiation Protocol (HNP).

11.2.2.1.1 Common Block

The Common block contains the Phase-Locked Loop (PLL), Bias, and REFCLK_LOGIC blocks.

11.2.2.1.1.1 Phase-Locked Loop

The PLL takes a reference clock from the REFCLK_LOGIC block and outputs 16 equally-spaced (in time) phases that are propagated to the digital core for use in its Delay-Locked Loop (DLL) clock and data recovery circuit.

To limit power-supply-induced jitter, internal regulators supply power to all jitter-sensitive circuitry in the PLL such as the charge pump, oscillator, and frequency divider.

11.2.2.1.1.2 Bias

The Bias block provides bias references to the analog circuitry in both the Common and Transceiver blocks.

11.2.2.1.1.3 REFCLK_LOGIC

The REFCLK_LOGIC block performs all mode control, multiplexing, and division of core clock sources. The output clocks from the REFCLK_LOGIC block are divide by 1, 2, or 5 versions of the input clock, and these output clocks are provided to the digital core and the PLL block. The REFCLK_LOGIC block accepts 9.6-, 10-, 12-,19.2-, 20-, 24-,

and 50-MHz core clock input frequencies. Dividers in the REFCLK_LOGIC block, whose division rates are set by top-level ports, convert the reference clock frequency to the PLL reference.

11.2.2.1.2 Transceiver Block

The Transceiver block comprises the Analog block and the digital core, which control the transmit, receive, disconnect detection, squelch detection, and automatic tuning functions.

11.2.2.1.2.1 Analog Block

The Analog block comprises the Squelch/Receiver, Disconnect/Tuning, and Transmitter blocks.

11.2.2.1.2.2 Squelch/Receiver Block

This block combines the squelch detection and receiver functions.

The Receiver regulator provides power to the Squelch/Receiver block's core-voltage circuitry and Disconnect/Tuning block's core-voltage circuitry.

11.2.2.1.2.3 Disconnect/Tuning Block

This block employs a low-offset comparator to perform host disconnect detection.

In addition, the comparator is used in the high-speed transmit DC level tuning algorithm. To ensure that temperature changes do not affect the tuned DC level, this tuning algorithm is run at each Start-of-Frame (SOF) transmission or reception. Furthermore, the Disconnect/Tuning block contains the control circuitry for the ANALOGTEST pin.

11.2.2.1.2.4 Transmitter Block

The Transmitter block contains drivers for all three speed modes in the USB protocol: HS mode at 480 Mbps, FS mode at 12 Mbps, and LS mode at 1.5 Mbps.

This block also includes automatically tuned 45-Ω resistors that set the HS input and output impedance, 1.5-kΩ D+ device pull-up resistor for FS operation, and 15-kΩ D+ and D– host pull-down resistors for FS and LS operations. The USB 2.0 PHY does not integrate a 1.5-kΩ D– device pull-up resistor for LS operation, because according to the USB 2.0 specification, an HS-capable device cannot support LS device operation.

11.2.2.1.3 OTG Block

The OTG block enables A-devices and B-devices to initiate the Session Request Protocol (SRP), and dualrole devices to initiate the Host Negotiation Protocol (HNP).

11.2.3 External Signals

The following table describes the external signals.

Table 11-76. External Signals

Signal	I/O	Description
FREECLK	O	Free-Running UTMI+ Clock Function: This controller signal is valid only if the USB 2.0 PHY is used as a downstream-facing port. In Suspend or Sleep mode, the state of FREECLK is indeterminate.
PHYCLOCK<#>	O	UTMI+ Clock Function: This controller signal clocks parallel receive and transmit data on the UTMI+.
CHGDET<#>	O	Battery Charger Detection Output Function: Indicates whether the voltage level on DP<#> or DM<#> is greater than VDAT_REF as defined in the Battery Charger specification.
VBUS0	IO	USB 5-V Signal Function: The PHY VBUS0 pin must not connect directly to the 5v VBUS voltage on the USB2.0 link. The PHY VBUS0 pin must be isolated by an external resistor (REXT1) so that the PHY VBUS0 pin sees a lower voltage. A charge pump external to the USB 2.0 PHY must provide power to this pin. If VBUS0 is not used, leave it floating.
ID0	IO	USB Mini-Receptacle Identifier or Alternate Test Point for DC Points Probes Inside USB 2.0 PHY Function: This signal differentiates a mini-A from a mini-B plug or between the ACA ID resistances. If this signal is not used, internal resistance pulls the signal's voltage level up to VDD18<#> when IDPULLUP0 is high.
DP<#>	IO	USB D+ Signal

Table continues on the next page...

Table 11-76. External Signals (continued)

Signal	I/O	Description
		Function: In normal operation, DP<#> carries USB data to and from the USB 2.0 PHY. In HS operation, this pin receives/transmits a maximum of 800 mV or 400 mV nominally. In FS or LS operation, this pin receives/transmits 3.3 V nominally.
DM<#>	IO	USB D– Signal Function: In normal operation, DP<#> carries USB data to and from the USB 2.0 PHY. In HS operation, this pin receives/transmits a maximum of 800 mV or 400 mV nominally. In FS or LS operation, this pin receives/transmits 3.3 V nominally.
TXRTUNE	IO	Transmitter Resistor Tune Pin Function: This analog signal connects to an external resistor (REXT2) that adjusts the USB 2.0 PHY's high-speed source impedance.
DVDD	I	Digital Power Supply Function: Digital power supply from the PHY’s I/O power pads.
VDD33<#>	I	3.3-V Analog Power Supply Function: Analog power supply.
VDD18<#>	I	1.8V Analog Power Supply Function: Analog power supply
VSSA<#>	I	Ground Supply Function: Ground supply for the Common, Transceiver, and OTG blocks.
REFCLOUT	O	Buffered Version of CLKCORE Function: This output clock is a buffered version of the reference clock (CLKCORE).
CLKCORE	I	On-Chip Reference Clock Source Function: This clock signal enables the USB 2.0 PHY to accept a corevoltage reference clock on this pin. If this signal is not used, tie it to ground.
CLK48MOHCI	O	48-MHz Reference Clock for Open Host Controller Interface (OHCI) Function: This 48-MHz reference output clock signal can be used by logic external to the USB 2.0 PHY. This signal does not control any UTMI+ input or output and is derived from the USB 2.0 PHY's internal PLL’s 480-MHz output

Table continues on the next page...

Table 11-76. External Signals (continued)

Signal	I/O	Description
		through an internal divide-by-10 circuit. This signal starts pulsing when the PLL is locked and stops pulsing when the PLL is powered down in Suspend or Sleep mode. Setting the COMMONONN input to 1'b0 prevents the PLL from being powered down in Suspend or Sleep mode. Therefore, CLK48MOHCI continues to toggle while COMMONONN is asserted, even in Suspend or Sleep mode. The peak output jitter on CLK48MOHCI is ± 200 ps. Notes: • When using this output as a clock signal for external logic, ensure that CLK48MOHCI has its own clock tree. • Setting COMMONONN to 1'b0 causes the PLL to remain powered, even in Suspend or Sleep mode. The resultant current draw of approximately 15 mA might affect suspend current limits, especially for bus-powered devices. • CLK48MOHCI is a 50% duty cycle output. • Loading effects on this pin are specified in the .lib timing models. • When the internal PLL is powered down in Suspend or Sleep mode, the CLK48MOHCI output settles down to either logic 1 or logic 0. Therefore, CLK48MOHCI is not appropriate as a clock signal for external latch-based logic.
CLK12MOHCI	O	12-MHz Reference Clock for OHCI Function: This 12-MHz reference output clock signal can be used by logic external to the USB 2.0 PHY. This signal does not control any UTMI+ input or output and is derived from the USB 2.0 PHY's internal PLL's 480-MHz output through an internal divide-by-40 circuit. This signal starts pulsing when the PLL is locked and stops pulsing when the PLL is powered down in Suspend or Sleep mode. Setting the COMMONONN input to 1'b0 prevents the PLL from being powered down in Suspend or Sleep mode. Therefore, CLK12MOHCI continues to toggle while COMMONONN

Table continues on the next page...

Table 11-76. External Signals (continued)

Signal	I/O	Description
		is asserted, even in Suspend or Sleep mode. The peak output jitter on CLK12MOHCI is ± 200 ps. Notes: • When using this output as a clock signal for external logic, ensure that CLK12MOHCI has its own clock tree. • Setting COMMONONN to 1'b0 causes the PLL to remain powered, even in Suspend or Sleep mode. The resultant current draw of approximately 15 mA might affect suspend current limits, especially for bus-powered devices. • CLK12MOHCI is a 50% duty cycle output. • Loading effects on this pin are specified in the .lib timing models. • When the internal PLL is powered down in Suspend or Sleep mode, the CLK12MOHCI output settles down to either logic 1 or logic 0. Therefore, CLK12MOHCI is not appropriate as a clock signal for external latch-based logic.
CLK480M	O	480-MHz Buffered Output Clock From PLL Function: This 480-MHz clock is a buffered output clock from the PLL that can be used by logic external to the USB 2.0 PHY. Setting the COMMONONN input to 1’b0 prevents the PLL from being powered down in Suspend and Sleep modes. Therefore, CLK480M continues to toggle while COMMONONN is asserted, even in Suspend or Sleep mode. Notes: • When using this output as a clock signal for external logic, ensure that CLK480M has its own clock tree. • Setting COMMONONN to 1’b0 causes the PLL, REFCLK_LOGIC, and Bias blocks to remain powered, even in Suspend or UART/Autoresume mode. The resultant current draw of approximately 15 mA might affect suspend current limits, especially for bus-powered devices.

Table 11-76. External Signals

Signal	I/O	Description
		• CLK480M has a 40/60 duty cycle output with 100 ps peak-to-peak jitter. • Loading effects on this pin are specified in the .lib timing models. • When the internal PLL is powered down, the state and frequency of CLK480M is indeterminate. • During a POR, the state and frequency of CLK480M is indeterminate. • When the PLL is not locked, the state and frequency of CLK480M is indeterminate. Gating circuitry for CLK480M must be external to the USB 2.0 PHY, because the PHYCLOCK<#> output starts pulsing only after the PLL is locked.

11.2.4 Clock Generation

The following figure shows the USB 2.0 PHY clock generation and distribution architecture, and the functional behavior of blocks that generate various clocks in the USB 2.0 PHY.

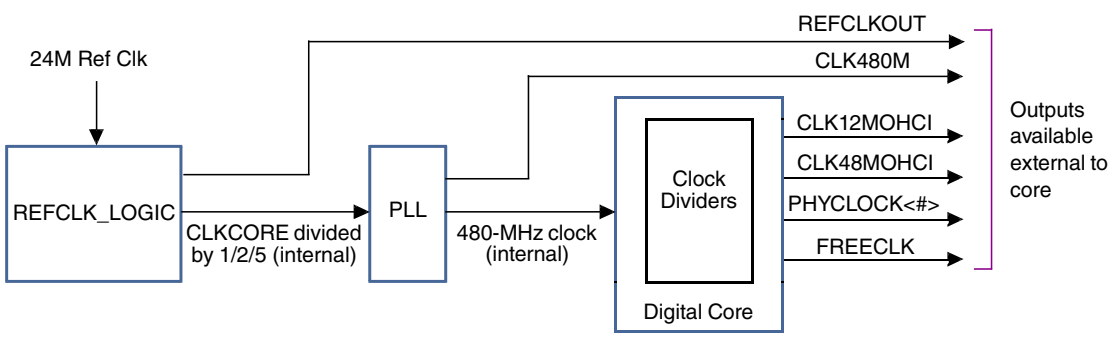


Figure 11-32. Clock Generation

11.2.5 OTG 2.0 Settings and Usage

This section describes the settings and usage of the USB 2.0 PHY's OTG 2.0 functionality.

11.2.5.1 OTG Block Overview

The OTG block enables A-devices and B-devices to initiate the Session Request Protocol (SRP), and dualrole devices to initiate the Host Negotiation Protocol (HNP).

Comparators

The OTG block includes two voltage comparators, session request protocol circuitry, and ID detection circuitry.

- VBUS Valid comparator: This comparator enables an A-device to determine if the voltage on VBUS is above the VBUS Valid threshold for valid operation.
- VBUS Detect comparator: This comparator detects when a session is being started (that is, when the voltage on VBUS is above the Session Valid threshold).

ID Detection Circuitry

The ID Detection circuitry senses the ID line's condition to indicate which type of plug is connected. The ID Accessory Charger Adapter (ACA) detection circuitry complies with the Battery Charging specification requirements. For more information about these requirements, refer to the Battery Charging specification.

11.2.6 Charge Pump Connection

An off-chip charge pump is required to provide power to the USB 2.0 PHY VBUS0 pin. Figure 6-5 shows the charge pump connection to the USB 2.0 PHY.

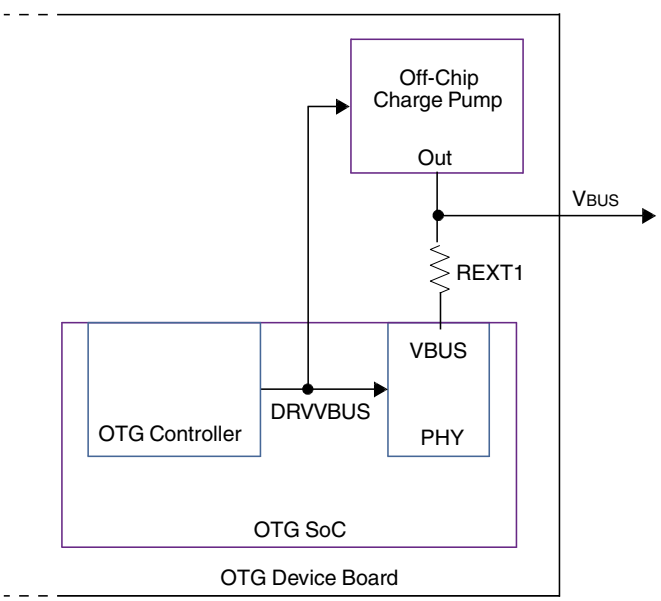


Figure 11-33. Charge Pump Connection

The charge pump's output is connected directly to VBUS on the device board. The USB 2.0 PHY's VBUS0 pin is connected to VBUS with series resistor (REXT1). The charge pump's DRVVBUS0 input is an output of the OTG HNP finite state machine and an input to the USB 2.0 PHY.

The VBUS0 pin presents a worst-case core-side load of 500 fF. This worst-case load is a small addition to the overall capacitance budget that includes the external capacitive load due to routing, pads, package, board traces, and receivers.

11.2.7 Pad/Package/Board Connections

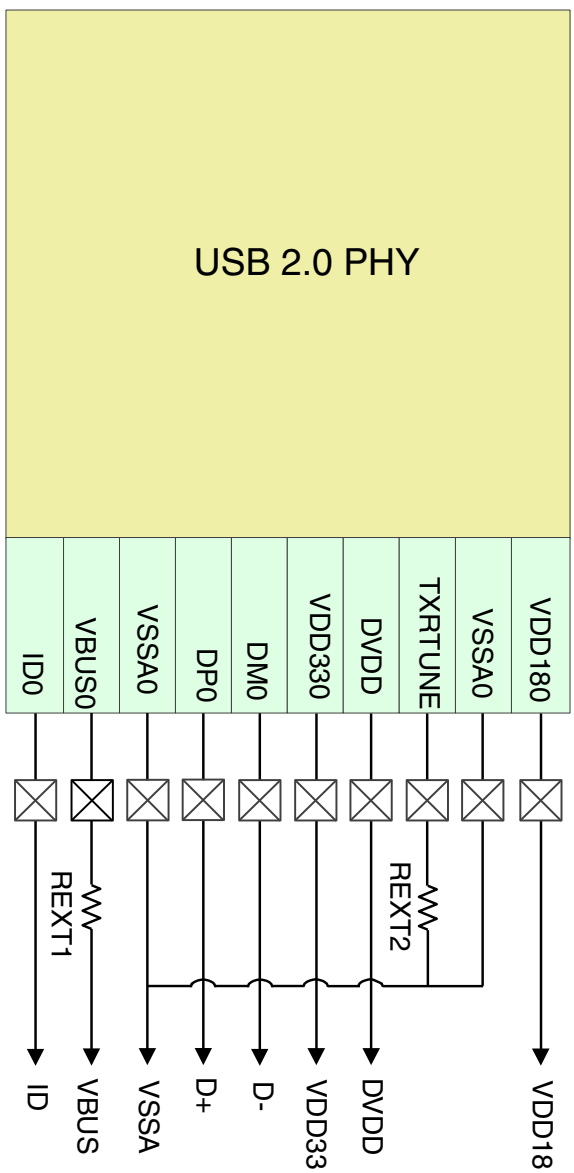


Figure 11-34. Pad/Package/Board Connections

11.2.8 Pad Implementation Requirements

Customers must ensure that all USB pads meet the criteria in the following table.

Table 11-77. Maximum Pad/Package/Board Requirements

Signal	Signal Description	Series Resistancea (Ω)[1]	Series Inductance (nH)	Capacitancea (pF)[1]	5-V Tolerance Needed	Worst-Case Interconnect Metal DC Current Handling Requirements (mA)
ID0	USB mini-receptacle identifier	1	4	5	N	1
VBUS0	USB mini-receptacle VBUS	1	4	5	N	8
VDD33<#>	Macro supply	1	4	-	N	35
VSSA<#>[2]	Macro supply	1	4	-	N	50
VDD18<#><#>	Macro supply	1	4	-	N	35
DP<#>	Analog signal	1	4	5	Y	30
DM<#>	Analog signal	1	4	5	Y	30
TXRTUNE	Analog signal	1	4	5[3]	N	15
DVDD	Digital-core supply	1	4	-	N	15
The simplest way to construct the pad ring is to use the following pad types: • VDD33: 3.3-V analog power supply pad • VDD18<#>: 1.8-VVREG analog power supply pad • VSSA<#>: ground supply pad • DVDD: Core power pad • DP, DM, VBUS: 5-V tolerant analog pads • TXRTUNE, ID: 1.8-V analog pads • VBUS0: 3.3-V analog pad						

1. Total resistance/capacitance of the ESD I/O pad, package, bondwire (if applicable), board trace, and USB connector must not exceed specified value.
2. The PHY macro has multiple VSSA<#> pins. The total impedance between all the instances of the VSSA node and the board ground should be less than the value mentioned in this table.
3. Total capacitance of the package and board trace must not exceed specified value.

11.2.9 Device Initialization

According to the USB 2.0 specification, a host must provide 100 ms between first detecting the D+ line at a high state and initiating a USB reset (SE0). The device pulling up the D+ line is expected to complete initialization within the 100-ms interval and be ready to respond to the host.

If the device requires more than 100 ms to initialize the device might connect the D+ resistance prior to initialization. After the 100-ms interval, the host initiates a USB reset. If the D+ pull-up resistance is enabled and remains enabled after the 100-ms interval, and if the device has not completed initialization, the device does not respond to the host correctly, and the transaction times out.

11.2.10 Tuning

The USB 2.0 PHY has an internal tuning circuit that ensures that the driver's impedance is within the USB 2.0 specification.

The USB 2.0 PHY performs two types of tuning:

- Incremental tuning: Incremental tuning occurs during an EOP, SOF, or SYNC pattern.
- Full tuning: Full tuning occurs during a power-on reset or after resume operation.

11.3 PCI Express (PCIe)

11.3.1 Overview

This PCI Express dual mode (DM) controller provides a solution to implement a PCI Express port for a PCI Express root complex or endpoint application.

A complete PCI Express port solution includes the controller, an analog PHY macro, and application logic to source and sink data. The physical layer is split across the PIPE and controller such that the MAC functionality (LTSSM, lane-to-lane deskew) is in the controller and the PHY functionality is implemented in the PIPE-compliant PHY. The PHY is outside of the controller, interfacing through the standard PIPE interface.

NOTE

The information within this block guide is Synopsys Proprietary. Used with permission.

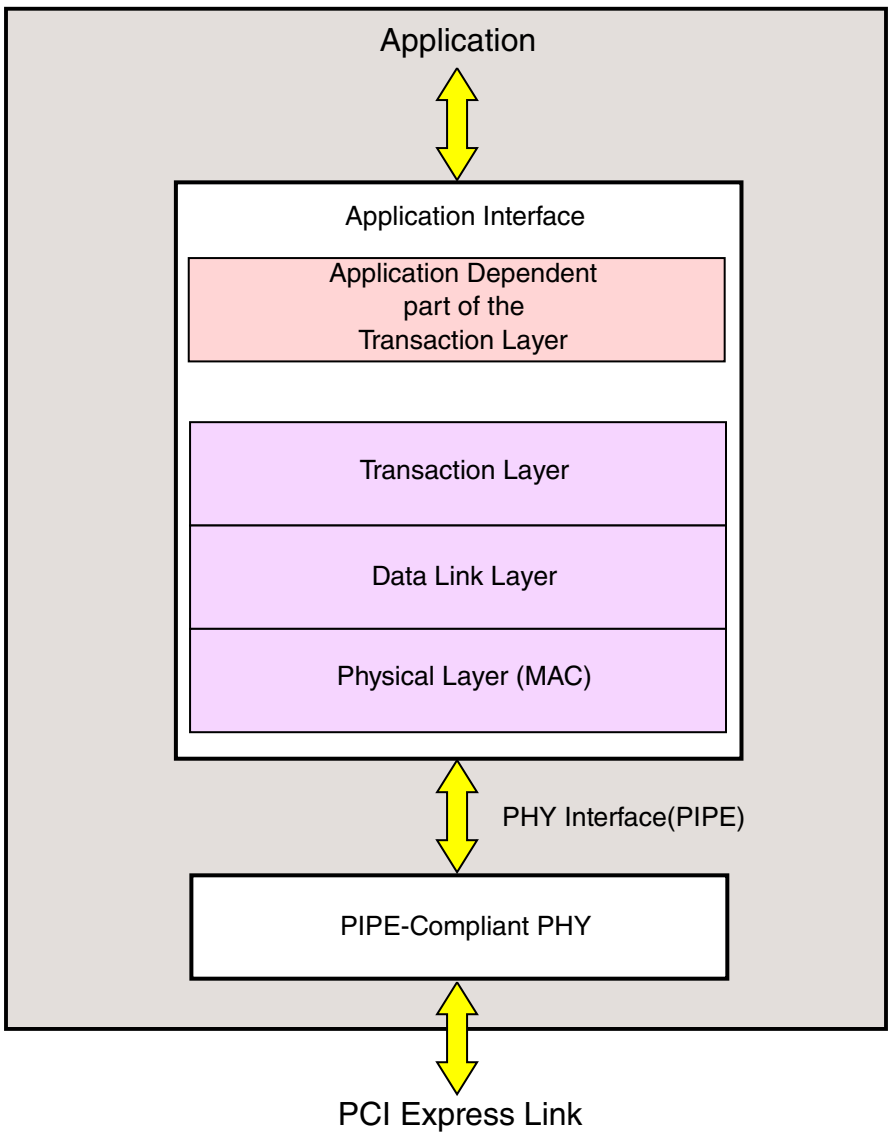


Figure 11-35. PCIe Application

11.3.1.1 Features

The PCI Express DM controller supports:

- Latency Tolerance Reporting (LTR)
- Address Translation Services (ATS) by your application
- Page Request Services (PRS) by your application
- Completion Timeout Ranges
- ID-Based Ordering (IDO) by your application
- M-PCIe
- PASID

- L1 Substates (L1SS)
- Extended Tag Support
- Resizable BAR (RBAR) with Expanded RBAR support, and VF re-sizable BAR support
- Separate Refclk with Independent Spread Spectrum Clocking (SRIS)
- Lightweight Notifications (LN)
- Readiness Notifications (RN)
- PCI Express Active State Power Management (ASPM)
- PCI Express Advanced Error Reporting (AER) with Multiple Header Logging
- PCI Express Advanced Error Reporting (AER)
- Vital Product Data (VPD)
- Access Control Services (ACS)
- x1 Gen2 lane
- Automatic Integration With Most PHY types (8-bit, 16-bit, 32-bit, and 64-bit PIPE Width per Lane)
- Power Gating (UPF) Support
- Advanced Power and Clock Management
- Internal Address Translation Unit
- Internal MSI-X Generation Module
- AXI3
- Embedded DMA
- Upconfigure Support
- Multiple Functions. Up to 32 Functions with ARI or SR-IOV, and up to 8 Functions otherwise.
- ECRC Generation and Checking
- Configuration Intercept Controller to allow your application modify CFG access from wire
- Multiple Virtual Channels (VCs)[2] and Traffic Classes (TCs)
- Bypass, Cut-through, and Store-and-forward Queue Modes for Rx TLPs
- Configurable Receive and Retry Buffer sizes
- Configurable Max_Payload_Size size (128 bytes to 4 KB)
- Configurable Filtering Rules for Posted, Non-posted, and Completion Traffic
- Configurable BAR Filtering, I/O Filtering, Configuration Filtering and Completion Lookup/Timeout
- Three Application Transmit Clients
- MSI and MSI-X with Per-Vector Masking (PVM), Extended message data for MSI
- Type 1 Configuration Space
- Application-initiated Manual Lane Reversal/flip for situations where controller does not detect Lane 0
- Type 0 Configuration space

2. The AXI bridge supports multi-VC operation but it does not support QOS.

- PHY register access (Only for PHYs supporting Control Register Parallel Interface)
- Automotive support
- Transmit Interface Progress Detection
- CDM Register Check

Attention: Some features are not available or might have limitations, when you are using the AXI bridge.

11.3.1.2 Standards

The PCIe controller implements the following standards:

- *PCI Express Base Specification, Revision 4.0, Version 0.7*
- Access to this specification requires membership in PCI-SIG.
- PIPE Specification for PCI Express, Version 4.3, Intel Corporation
- PCI Local Bus Specification, Revision 3.0
- Access to this specification requires membership in PCI-SIG.
- PCI Bus Power Management Specification, Revision 1.2 Access to this specification requires membership in PCI-SIG.
- PCI Express Card Electromechanical Specification, Revision 1.1 Access to this specification requires membership in PCI-SIG.

11.3.2 Link Establishment

The controller implements the LTSSM function according to the *PCI Express Base Specification, Revision 4.0, Version 0.7*. .

11.3.2.1 Transmit TLP Processing

The following section describe the flow of transmit TLPs through the controller.

NOTE

The controller does not check the TLP for errors.

The native PCIe controller does not check that the TLP payload size is less than the Maximum Payload Size limit. However, the AXI/AHB bridge module does guarantee that this limit is not exceeded.

The native PCIe controller does not check that the TLP payload size is less than max payload limit. Exceeding this limit overflows the retry buffer, resulting in data corruption. However the AXI/AHB bridge module does guarantee that this limit is not exceeded.

11.3.2.1.1 Transmit TLP Arbitration

TLPs and DLLPs have equal priorities during transmit arbitration as shown in the figure below. The priority of TLP types and DLLP types are shown in tables below.

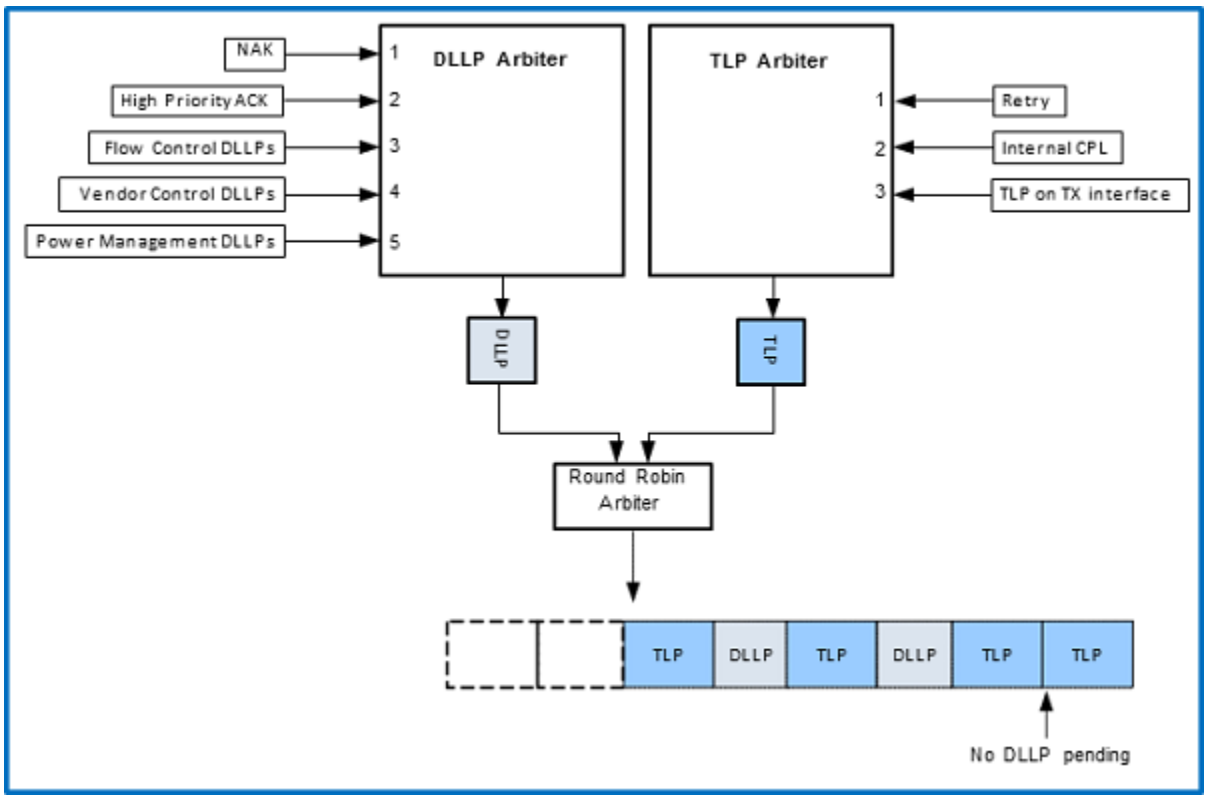


Figure 11-36. Transmit TLP/DLLP Arbitration

Table 11-78. TLP Arbitration Priority (1= Highest, 4= Lowest)

Priority	TLP Type
1	Completion of any TLP currently in progress
2	Retry buffer retransmissions
3	TLPs from the transaction layer (in the following order of priority): • Messages generated by the controller (including by your application through the MSI interface) • Upstream ports: Completions generated by the controller (including memory or I/O mapped application register space) for Type 0 configuration read and write requests, or responses to error conditions (unsupported requests) • Downstream ports: Completions generated by the controller for unsupported requests or completer aborts
4	TLPs on the transmit client interface (XALI0/1/2), or on the AXI Master/Slave interface if the AXI bridge is being used.

Table 11-79. DLLP Arbitration Priority (1= Highest, 6 = Lowest)

Priority	DLLP Type
1	Completion of any DLLP currently in progress
2	NAK DLLP
3	High Priority ACK DLLP
4	Flow Control DLLP
5	Vendor Specific Message DLLP
6	Power management or any other Low Priority DLLP

11.3.2.1.1.1 Effects of Flow Control Credits On Transmit Client Arbitration

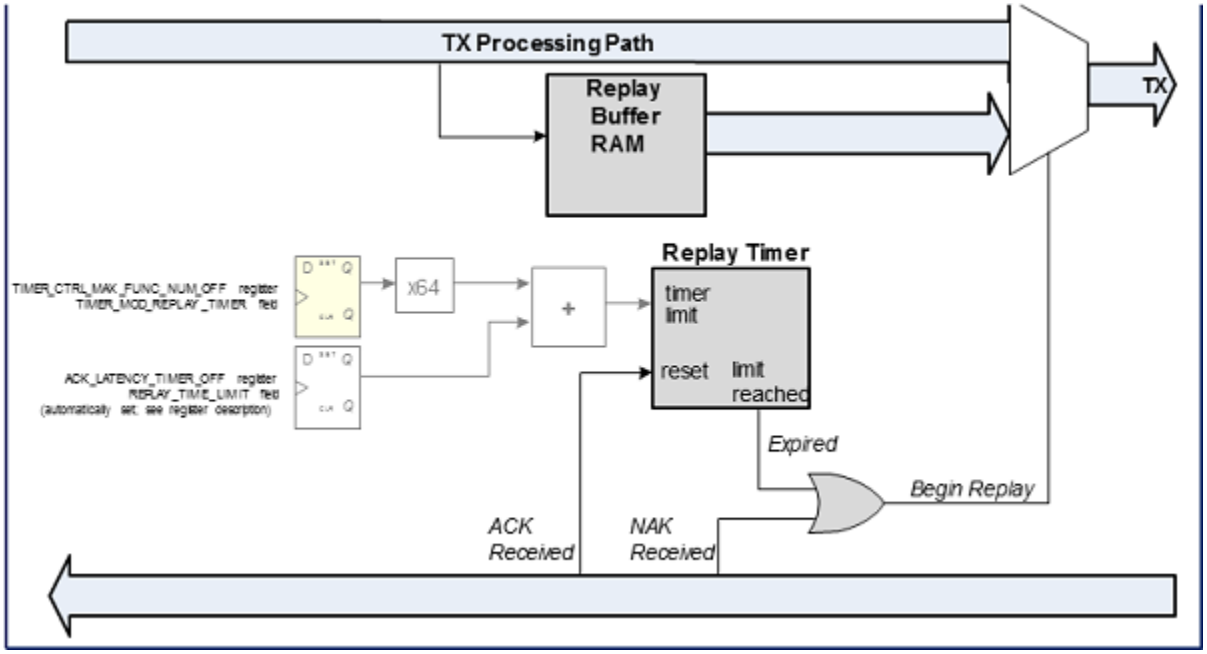
The controller checks that enough flow control (FC) credits are available in the remote device for the specific type of transaction (posted, non-posted, completion) before allowing a transmission of a TLP. TLPs that passed the credit check are arbitrated according to the supported arbitration method. Internally generated completions and messages are also gated by the arbitration logic, though at highest priority, and must also pass the FC credit test before they are accepted for transmission.

For example when using the RR scheme, when a posted (P) transaction is presented onto XALI1 followed by a completion (CPL) on XALI0, and when credits permit, then the P is transmitted onto the wire before the CPL. When the credit is not available then the CPL on XALI0 can pass the P on XALI1 and be sent onto the wire. However, any non-posted (NP) or CPL TLPs on XALI1 (behind the blocked P) are blocked by the halted P. It is the responsibility of your application to make appropriate use of the three interfaces. There is no guarantee that order is preserved among client interfaces.

11.3.2.1.2 Transmit Replay

You can modify the time-out value of the replay buffer as shown in the figure below. For a typical system, you do not have to modify the default settings, unless the remote device is executing unexpected replays.

Figure 11-37. Replay Buffer And Time (Representative)



11.3.2.2 Receive TLP Processing

This section describes the flow of received TLPs through the controller.

11.3.2.2.1 Receive Filtering

The controller contains a filter module that is responsible for the following tasks:

- Determine the status of a received TLP using filtering rules.
- Determine the destination of a received TLP based on the filtering status.
- Indicate the status of the received TLP using outputs.
- Report Errors to AER registers based on filter results. When more than one type of error is detected, Error Pollution, of the *PCI Express Base Specification, Revision 4.0, Version 0.7* is followed.
- The controller filters and routes received TLPs according to a set of rules determined by the TLP type based on the *PCI Express Base Specification, Revision 4.0, Version 0.7* and user-configurable filtering options.

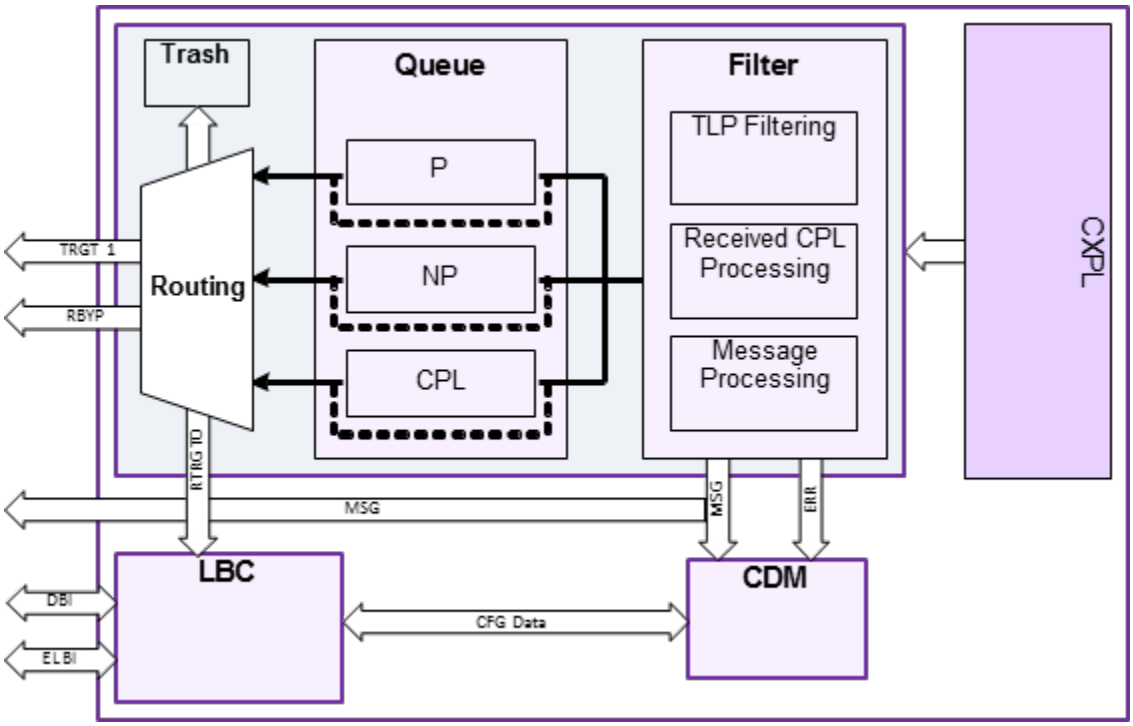


Figure 11-38. RADM Block Diagram

The following general rules apply to all incoming TLPs that are not malformed. For details on what happens to malformed TLPs, see "Error Detection for Received TLPs" By default:

- For a function in device power states D1, D2, and D3hot, the controller only accepts CFG and MSG requests TLPs for that function. All other incoming request types for that function are treated as unsupported requests (UR).
- When the controller detects an error in a received TLP, it normally performs the following:
  - Discards the TLP
  - Generates a completion (for non-posted requests) with the completion status set to CA or UR
  - Sets the status in the PCI-compatible Status register
  - Sets the status in the AER registers
  - Generates an error message (upstream port only)

All error-free MSG requests are decoded internally, signaled on the SII interface and then dropped. When you want to have the decoded message also sent to the application interface, then see “Routing of Received Messages”.

11.3.2.2.2 Receive Routing

This section discusses how the RADM routes different TLP types to different receive interfaces depending on TLP type and filter result. When you are using the AXI bridge, then TRGT1 is replaced by the AXI bridge interfaces. TRGT0 refers to the internal interface used to access the CDM registers or the ELBI in an upstream port.

11.3.2.2.2.1 EP Mode Routing Overview

The possible destinations of a posted or non-posted request TLP are TRGT1 interface, TRGT0 interface and Discard. By default:

- CFG requests are routed to TRGT0 and then to CDM through the LBC.
- BAR-matched MEM and I/O requests are routed to TRGT1.
- MSG requests are decoded internally, signaled on the SII interface, and then terminated.

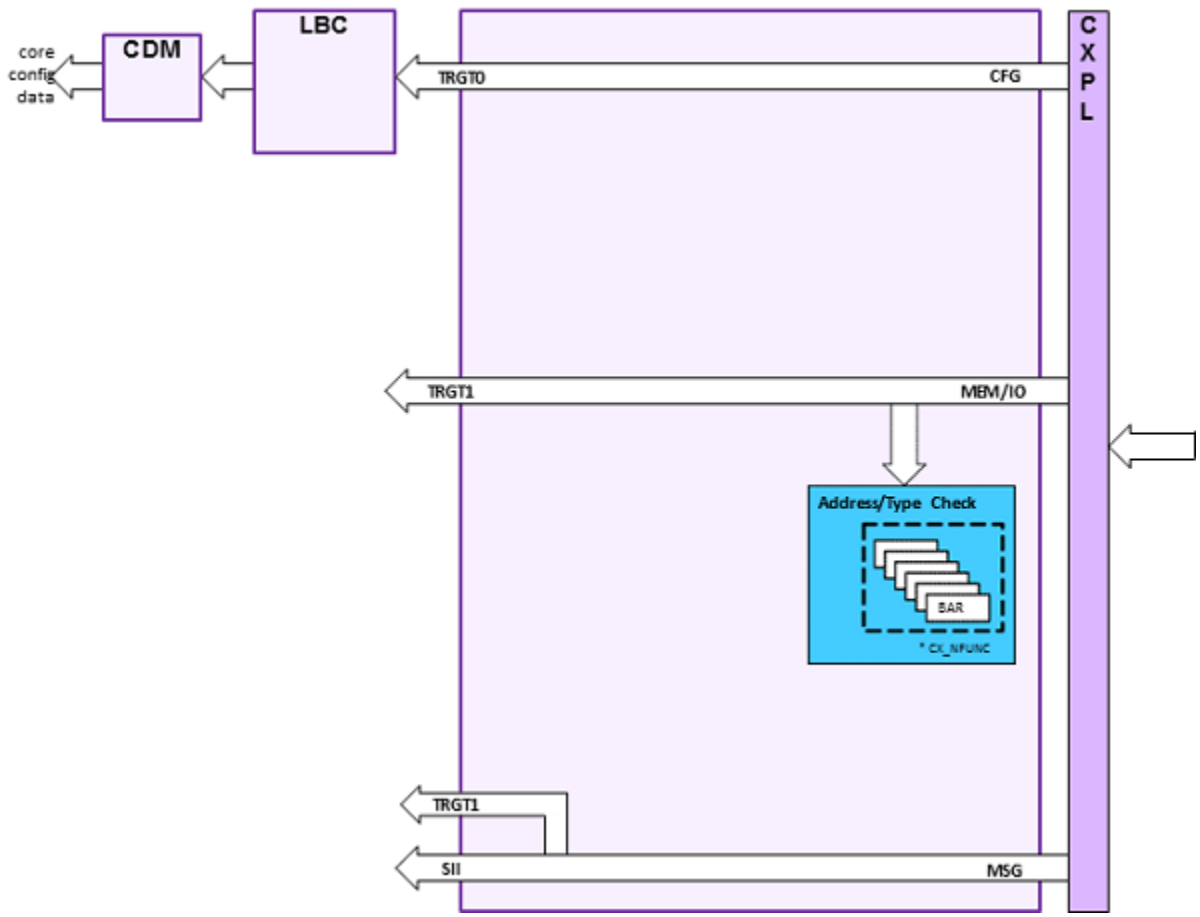


Figure 11-39. EP Mode Routing

11.3.2.2.2.2 RC Mode Routing Overview

The possible destinations of a posted or non-posted request are TRGT1 and Discard. By default:

- MEM requests outside of the memory range and prefetchable memory range as determined by the corresponding Base and Limit fields in the Type-1 header are routed to TRGT1.
- MSG requests are decoded internally, signaled on the SII interface, and then terminated.
- An RC does not expect to receive CFG or I/O requests.
- BARs should be disabled and not used.

The possible destinations of a completion TLP are TRGT1 and Discard.

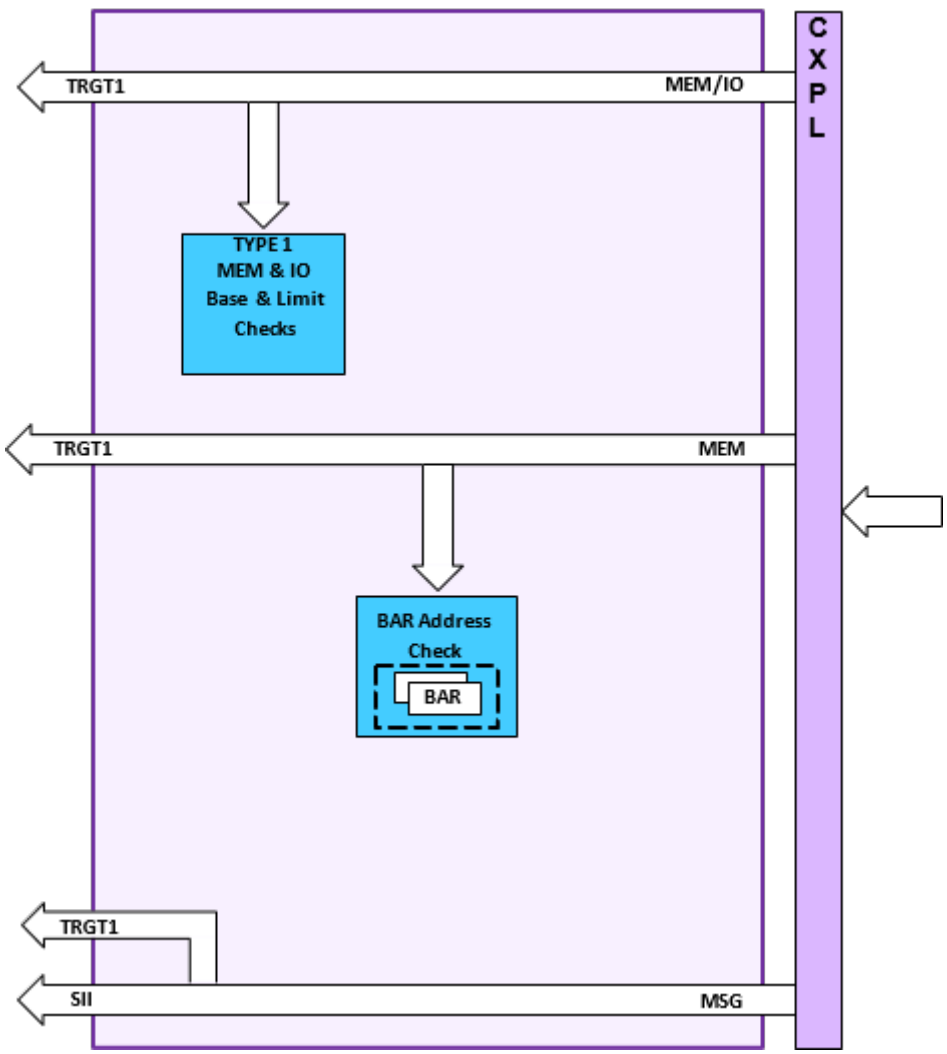


Figure 11-40. Default Request TLP Routing (Assuming no TLPs with CA/CRS/UR Error Status)

11.3.2.2.3 Error Handling

The controller supports the Baseline Capabilities, AER capabilities, and Advisory Non-Fatal Error Messaging as specified in section "Error Signaling and Logging" of the *PCI Express Base Specification, Revision 4.0, Version 0.7*. These include Correctable and Uncorrectable (Fatal and Non-Fatal) errors. For an RC port, the reporting of most errors is internal to the root port. No external error notifications are generated. One exception to this (for example) is unsupported request (UR) completion status.

11.3.2.2.3.1 Error Detection for Received TLPs

The controller performs all mandatory error detections, the error reporting mechanism based on the PCI Express Base Specification, Revision 4.0, Version 0.7. When the controller detects an error in a received TLP, it normally performs the following:

- Discards the TLP
- Generates a completion (for non-posted requests) with the completion status set to CA or UR
- Sets the status in the PCI-compatible status register
- Sets the status in the AER registers
- Generates an error MSG (upstream port only)

11.3.2.2.3.2 AER Multiple Header Logging

The controller supports the AER multiple header logging as specified in Section, "Multiple Error Handling (Advanced Error Reporting Capability)" of the *PCI Express Base Specification, Revision 4.0, Version 0.7*.

11.3.2.2.3.3 Physical Functions

By default, each physical function only logs the first header that caused an error. Per physical function, the controller implements a FIFO queue that interfaces with the AER Header Log registers (HDR_LOG_i_OFF). A valid queue entry is never overwritten. When the queue is full and your software does not read the header log registers fast enough, new header log information will be missed.

11.3.2.2.3.4 Virtual Functions

By default, each virtual function only logs the first header that caused an error. Each VF in each physical function has the exact same header log queue depth. Per VF, the controller implements a queuethat interfaces with the AER Header Log registers (VF_HDR_LOG_i_OFF). A valid queue entry is never overwritten. When the queue is full and your software does not read the header log registers fast enough, new header log information will be missed.

The depth of each shared queue is identical for all PFs . The queue is not a FIFO, it is just a resource-saving mechanism, and each VF can randomly access it’s own header logs independently of other VFs.

Table 11-80. Shared VF Header Log Full Behavior

VF First Error Pointer Status	VF Shared Header Log Status	Core Actions
Invalid	Full	• A read to the Header Log returns all 1's. • A read to the "TLP Prefix Log Present" bit indicates ‘0’. • The controller sets the Uncorrectable Error Status Register according to the First Error Pointer.
Valid	Full	• The controller sets the Uncorrectable Error Status Register according to the First Error Pointer that is associated with the rejected TLP. • The Header log can be cleared or updated to the next stored header log, if one exists, by writing to the bit of the Uncorrectable Status Error Register pointed to by the First Error Pointer. • This applies to both when a normally displayed TLP Header is present in the Header Log Register or when an overflow condition is shown on the Uncorrectable Status Error Register.

11.3.2.3 Register Module and Data Bus Interface (DBI)

11.3.2.3.1 Overview

The Local Bus Controller (LBC) module provides a mechanism for a link partner (in upstream mode) or a local CPU (through the DBI) to access:

- Internal registers (in the CDM)
- External application registers connected externally to the ELBI
- In RC mode:
- There is no PCIe wire access to the CDM registers and ELBI.
- There is full DBI access to the CDM registers and ELBI.
- The controller has 4096 bytes of register PCIe configuration and port logic space per function. The controller also has a common iATU and DMA configuration register space for all functions. The controller register space is fully accessible from the DBI without any restrictions.

Table 11-81. EP Mode Port Region Access

Register Location		CDM					ELBI
Register Type		PCI-SIG		Port Logic			User
		Normal	Shadow	Misc	IATU	DMA	User
Access Details	CDM/ELBI Selector Bit[1] Value	0	0	0	1	1	1
	CS2 Selector Bit Value	0	1	0	1	1	0
DBI/Wire Access Allowed							
DBI		Y	Y	Y	Y	Y	Y
Wire	**CFG Request**	Y	-	Y	-	-	Y
	BAR Matched MEM Request	-	-	Y[2]	Y	Y	Y[-1]

1. Selector bit locations are configuration-dependent.
2. IO request also supported.

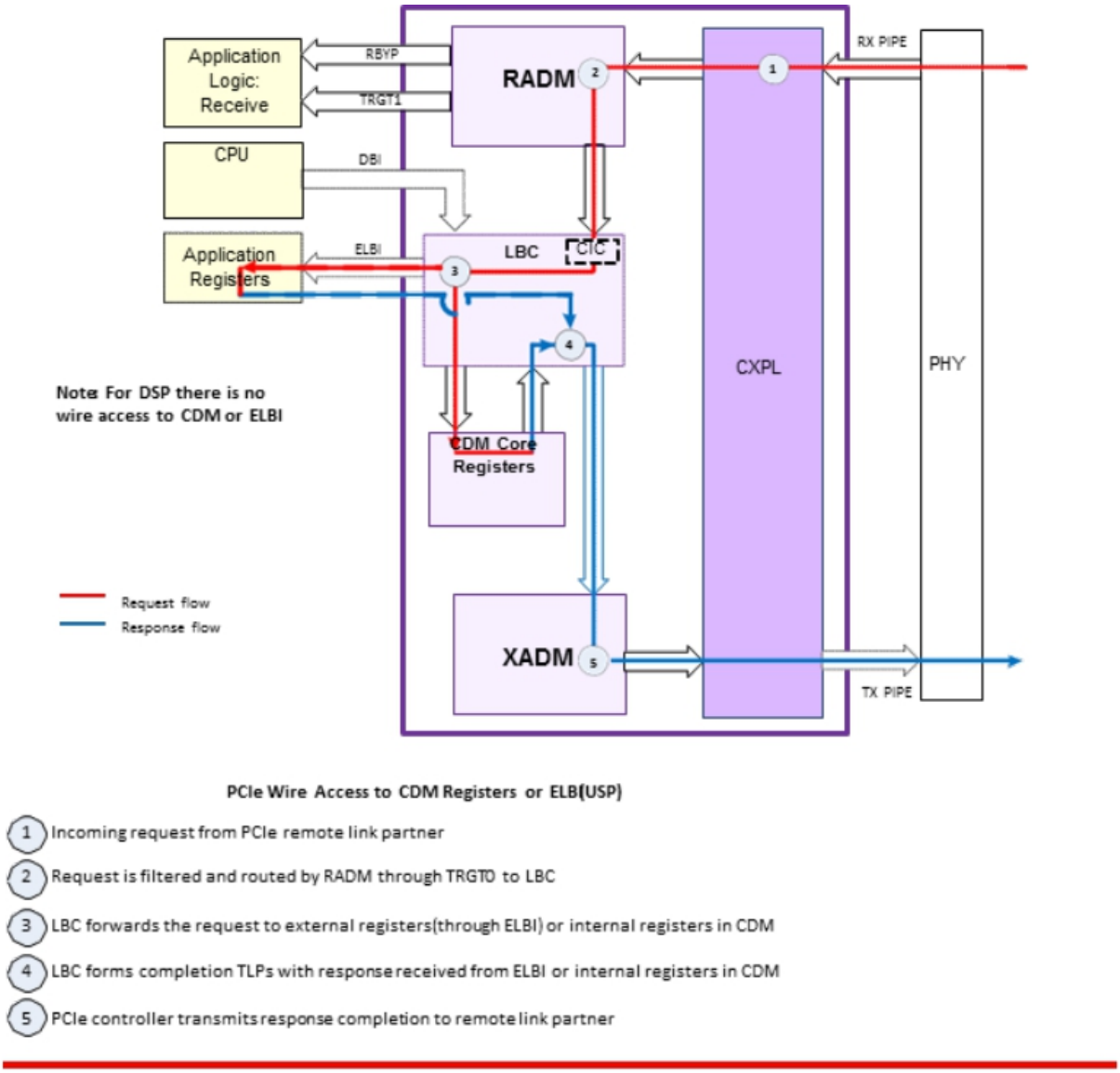


Figure 11-41. PCIe Wire Access to CDM Registers or ELB

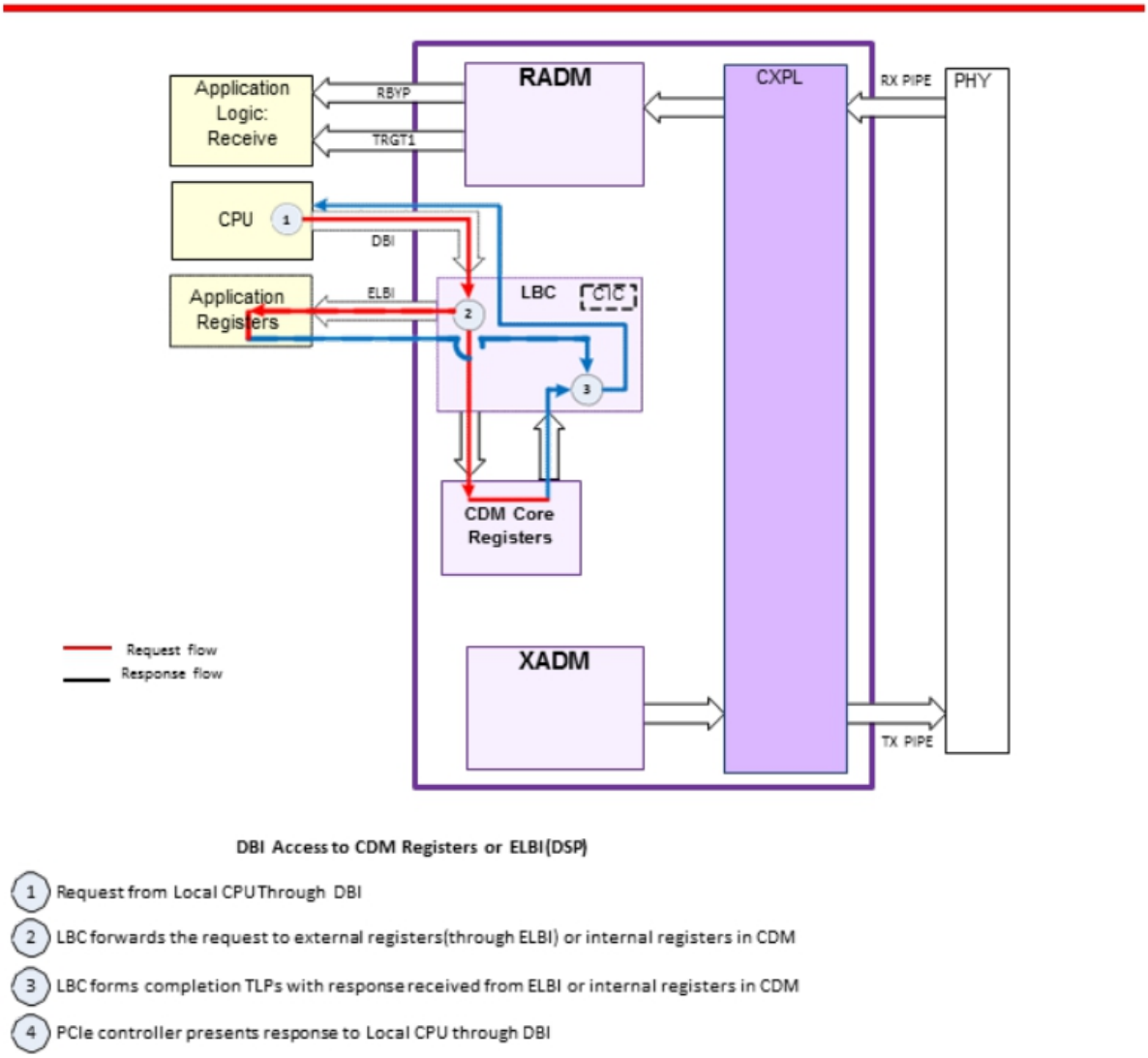


Figure 11-42. DBI Access to CDM Registers or ELBI

11.3.2.3.2 CDM Register Space Layout

11.3.2.3.2.1 Overview

The controller has 4096(0xFFF)1 bytes of register PCIe configuration and port logic space per function. The controller also has a common iATU and DMA configuration register space for all functions. The controller register space is fully accessible from the DBI without any restrictions.

Table 11-82. Port Region Access

Register Location	CDM		ELBI
Register Type	PCI-SIG	Port Logic	User

Table continues on the next page...

Table 11-82. Port Region Access (continued)

		Normal	Shadow	Misc	IATU	DMA	User
Access Details	CDM/ELBI Selector Bit[-1] Value	0	0	0	1	1	1
	CS2 Selector Bit Value	0	1	0	1	1	0
DBI/Wire Access Allowed							
DBI		Y	Y	Y	Y	Y	Y
Wire	**CFG Request**	Y	-	Y	-	-	Y
	BAR Matched MEM Request	-	-	Y[-1]	Y	Y	Y[-1]

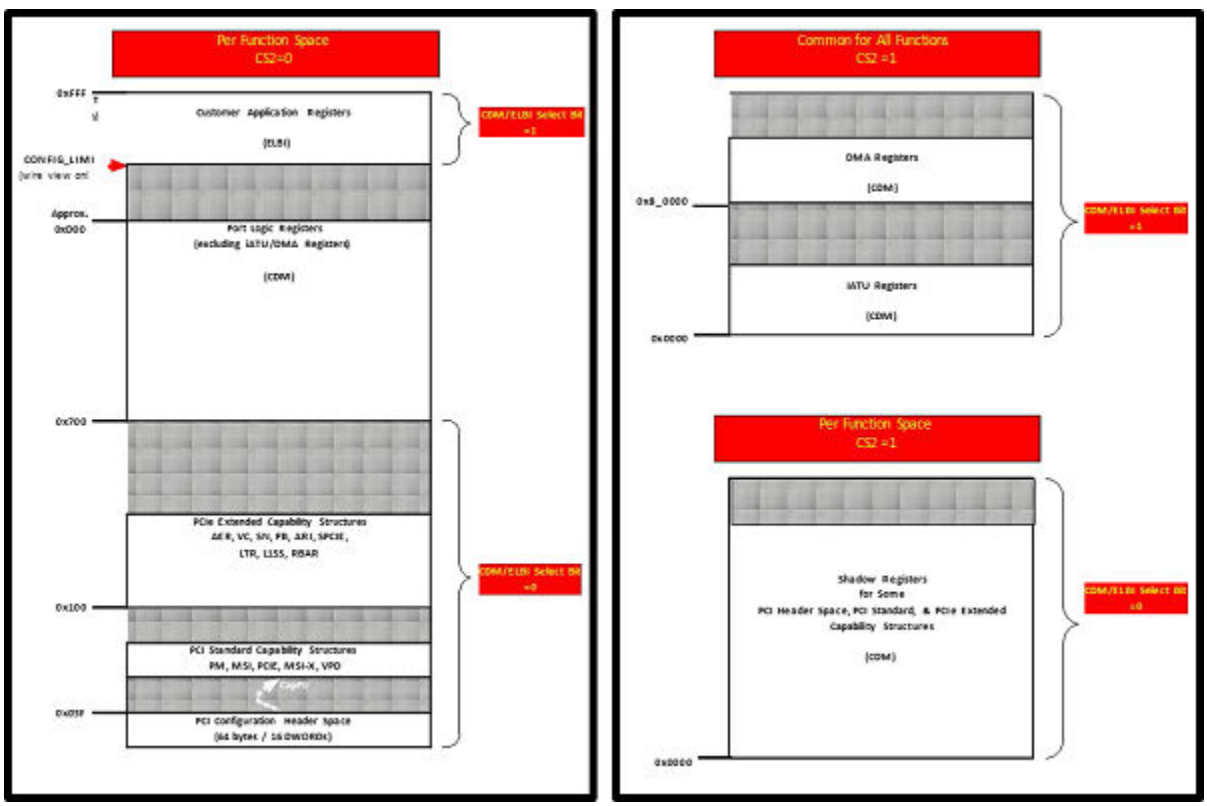


Figure 11-43. Controller Configuration Space Layout (EP Mode)

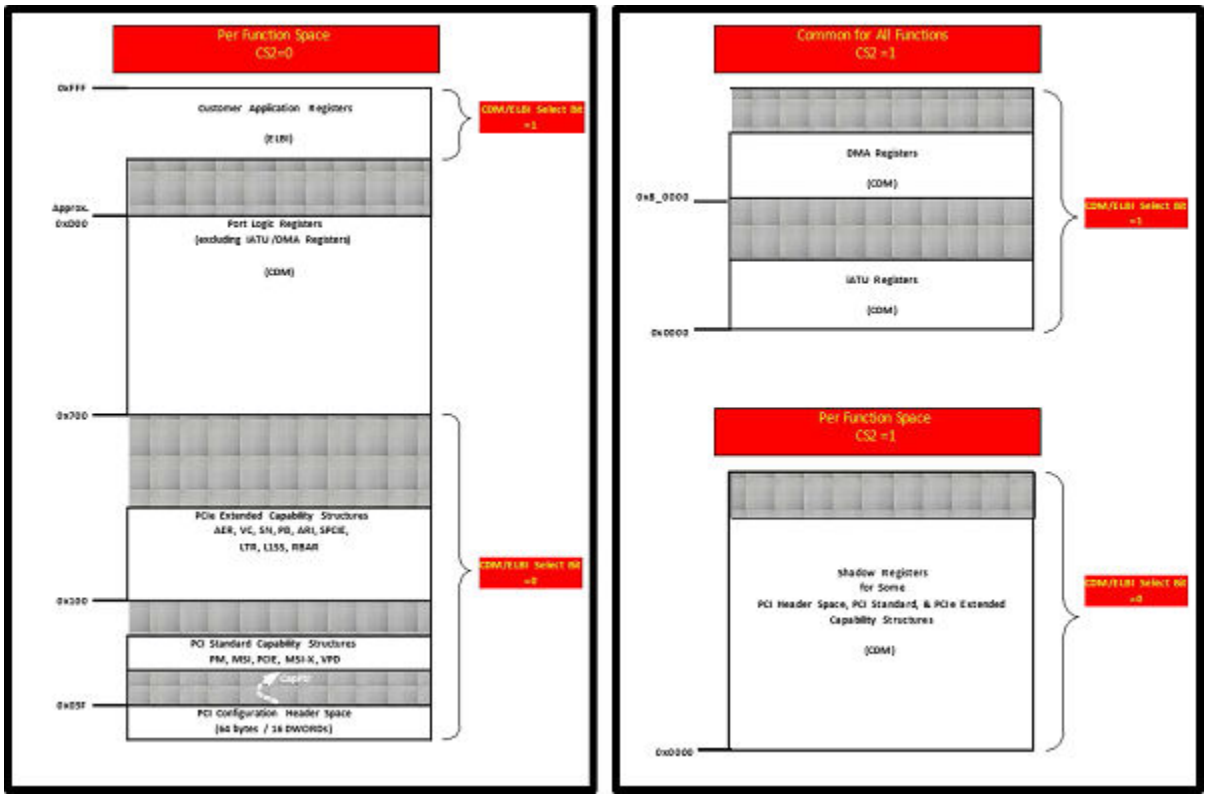


Figure 11-44. Controller Configuration Space Layout: (RC Mode)

Wire Access (Upstream Ports Only):

- Any CFG > CONFIG_LIMIT goes to TRGT1 or ELBI
- Any MEM/IO captured by a BAR whose interface target is TRGT0 goes to ELBI
- Any MEM captured by BAR*n* (*n* =UNROLL_BAR_NUM) when
- ENABLE_MEM_MAP_UNROLL_{DMA|ATU}_REG=1, whose interface target is TRGT0, and whose address falls into the range defined in "Memory Mapping iATU and DMA Port Logic Registers" goes to the IATU/DMA registers.

DBI Access (Upstream and Downstream Ports):

- No concept of MEM/IO/CFG
- Uses the specified address bit to route the access to ELBI or CDM
- No restrictions regarding what spaces can be accessed or not

11.3.2.3.2.2 Core Configuration Space Registers

- CDM Registers
  - PCI Configuration Header and Capability Registers:
    - The *PCI configuration header and capability registers* in the previous diagrams are PCIe controller configuration registers specified by the *PCI Express Base Specification, Revision 4.0, Version 0.7*. Access from the PCIe

wire is possible with CFG requests (upstream ports). These registers are fully accessible from the DBI without any restrictions.

- Port Logic Registers:
  - The port logic (PL) registers are configuration registers which are not specified by the *PCI Express Base Specification, Revision 4.0, Version 0.7*. Access from the PCIe wire is normally with CFG requests. These registers are fully accessible from the DBI without any restrictions.
- iATU and DMA Port Logic Registers:
  - This memory space is dedicated for iATU and DMA configuration registers. Access from the PCIe wire is with specific BAR matched MEM requests. These registers are fully accessible from the DBI without any restrictions.

11.3.2.3.2.3 PCIe Wire Access (EP Mode)

By default all requests to the standard PCI configuration space or to the port logic registers (except iATU and DMA registers) are routed to TRGT0 and then to the CDM through the LBC. To access the iATU/DMA registers, you must use MEM requests that target the BAR you have setup to capture these addresses. For more details, "Memory Mapping iATU and DMA Port Logic Registers" The "Configuration Intercept Controller (CIC) for USP" enables your application to detect the occurrence of; and to modify the behavior of Rx CFG requests (from the remote link partner) that are accessing the controller's internal registers.

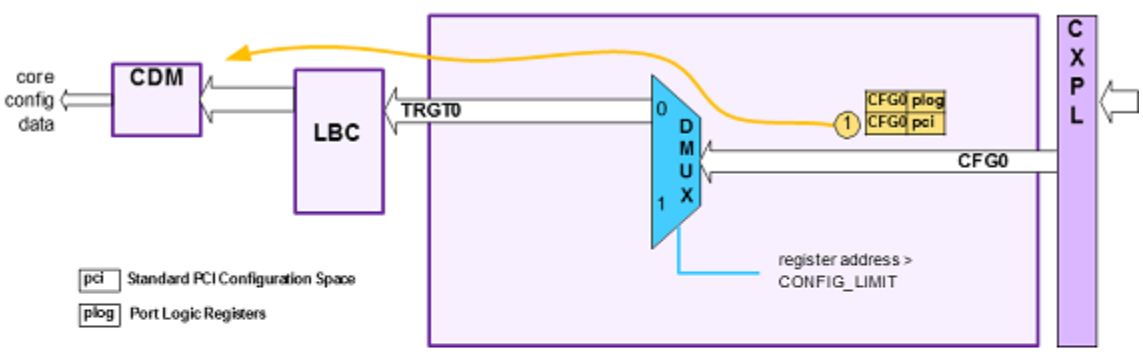


Figure 11-45. Request TLP Routing: Typical Use Model

11.3.2.3.3 Data Bus Interface (DBI)

11.3.2.3.3.1 Overview

The DBI can access all 4096 bytes (1024 DWORDs) of the PCI Express configuration space per function. DBI can also access iATU and DMA configuration space. This address space is fully accessible from the DBI without any restrictions.

11.3.2.3.3.1.1 Native Core DBI Access

When you are not using the AXI DBI slave, bit 0 of the DBI address input (dbi_addr) and chip select signal (dbi_cs2) determines the target of a DBI transaction as follows:

Table 11-83. Native Core DBI Address Bus Layout for Accessing CDM

	Register Location			CDM			ELBI
	Register Type	PCI-SIG		Port Logic			User
		Normal Shadow		Misc	IATU	DMA	User
Access Details	CDM/ELBI Selector Bit[a] Value	0	0	0	1	1	1
	CS2 Selector Bit Value	0	1	0	1	1	0

NOTE

a. Selector bit locations are configuration-dependent.

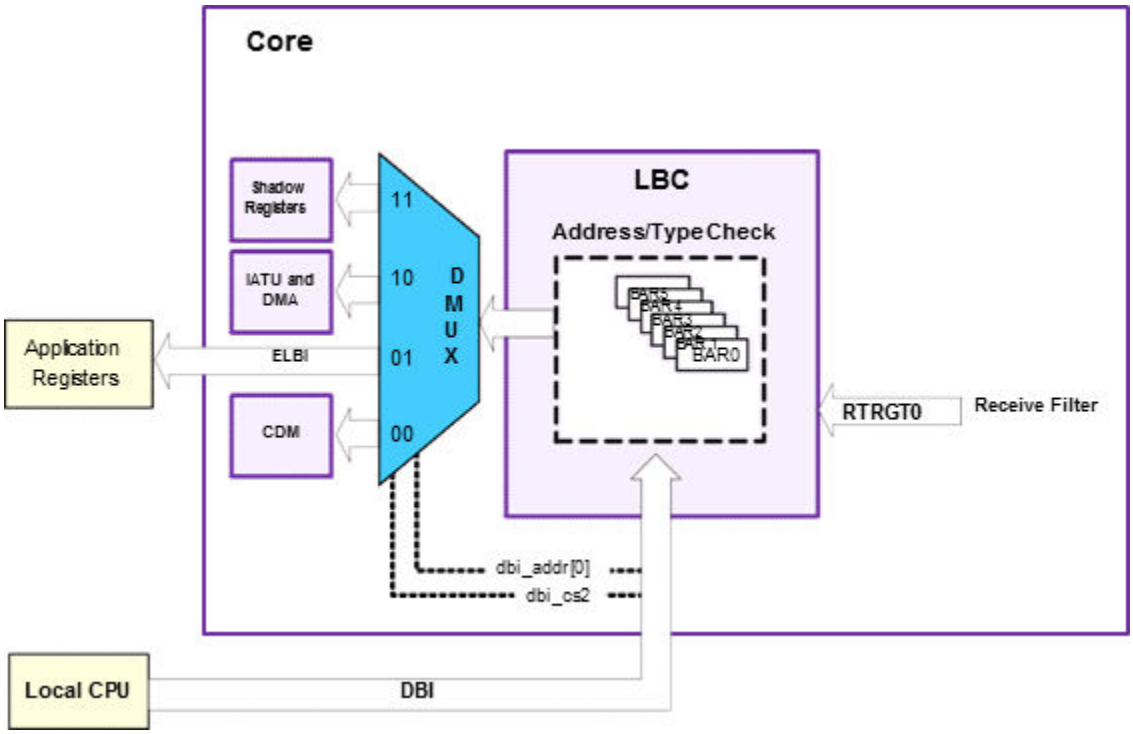


Figure 11-46. DBI Access Using Native DBI Interface (Not Using AXI DBI Slave)

11.3.2.3.3.2 AXI Dedicated DBI Access

When you are using the AXI DBI slave, then the DBI interface is not directly available to your application but is accessed through an AXI bridge slave interface. You select the target of the DBI transaction using a bit of the AXI DBI slave address signal, whose position moves around.

11.3.2.3.3.3 AXI Shared DBI Access

Select the target of the DBI transaction using a bit of the AXI DBI slave address signal whose position moves around.

When you are using the AXI bridge module, you do not have to use the AXI DBI slave interface. You can alternatively access the DBI through the native DBI interface.

11.3.2.3.3.4 Native Core DBI to CDM/ELBI Access Details

- The DBI address must be a DWORD address with dbi_addr[1]=1b0.
- Byte access within a DWORD for write requests is possible using dbi_wr[3:0].
- The dbi_wr[3:0]bus also identifies the type of access as a write or read (4h0).

Table 11-84. Native Core DBI Address Bus Layout for Accessing CDM (dbi_cs2=0)

31-19	18-16	15-12	11-2	1	0
Not used	Function #0x0	Not used	1 K-DWORD Register Address	0	0
The lower 12 bits are used to access the 4KB (1K DWORDs) of the PCI Express configuration space per function.					

Table 11-85. Native Core DBI Address Bus Layout for Accessing ELBI (dbi_cs2=0)

31-19	18-16	15-2	1	0
Not used	Function #0x0	16 K-DWORD Register Address	0	1
The lower 16 bits are used to access the 64KB (16K DWORDs) of the ELBI space per function.				

When SR-IOV is enabled, this field does not exist. For an access to a physical function, the function number is indicated on dbi_func_num. For an access to a virtual function, the function number is indicated on dbi_vfunc_num.

Table 11-86. Native Core DBI Address Bus Layout for Accessing iATU and DMA Configuration Registers(dbi_cs2=1)

Access Type	31-20	19	18-17	16-9	8	7-2	1	0
iATU	Not used	0	Reserved	Region Number Select	0: Outbound Region	Register Address	0	1
	Not used	0	Reserved	Region Number Select	1:Inbound Region	Register Address	0	1
DMA	Not used	1	Reserved	Channel Number Select	0:Write Channel	Register Address	0	1
	Not used	1	Reserved	Channel Number Select	1:Read Channel	Register Address	0	1

11.3.2.3.3.5 AXI Bridge DBI to CDM/ELBI Access Details

The DBI address must be a DWORD address (*addr[1:0]=2b00) with the CDM/ELBI selection bit of the address set to CDM(0) or ELBI(1).

The position of the CDM/ELBI selection bit within the address bus moves around

11.3.2.3.3.6 iATU and DMA Disabled (CC_DMA_ENABLE = 0 and CX_INTERNAL_ATU_ENABLE = 0)

Table 3-14Position of CDM/ELBI Select Bit when DMA and iATU Disabled

Parameters	CDM/ELBI Select Bit Position
CX_NFUNC =1 and CX_ARI_ENABLE =0 and CX_SRIOV_ENABLE =0	13
CX_NFUNCa >1 and CX_ARI_ENABLE =0 and CX_SRIOV_ENABLE =0	20
CX_NFUNCa >1 and CX_ARI_ENABLE =0 and CX_SRIOV_ENABLE =0 and DBI_MULTI_FUNC_BAR_EN=1	21
CX_ARI_ENABLE =1 or CX_SRIOV_ENABLE =1	32

a. Bit position is based on CX_NFUNC and not on device_type, so a DM controller with CX_NFUNC > 1 uses bit 20 (not bit 13) in RC mode.

Table 3-15AXI DBI Address Bus Layout (CX_NFUNC =1; CX_ARI_ENABLE =0: CX_SRIOV_ENABLE =0)

Access Type	31-19	18-16	15	14	13	12	2-Nov	1	0
CDM	0	Function	Not Used		0	CS2	1 K-dword Register Address	0	0
ELBI	0	Function	Not Used		1	0	1 K-dword Register Address	0	0

Table 3-16AXI DBI Address Bus Layout (CX_NFUNC >1; CX_SRIOV_ENABLE =0; CX_ARI_ENABLE =0)

Access Type	31-21	20	19	18-16	15-12	11-2	1	0
CDM	Not Used	0	CS2	Function	Reserved	1 K-DWORD Register Address	0	0
ELBI	Not Used	1	0	Function	Reserved	1 K-DWORD Register Address	0	0

Table 3-17AXI DBI Address Layout (CX_NFUNCa >1 and CX_ARI_ENABLE =0 and CX_SRIOV_ENABLE =0 and DBI_MULTI_FUNC_BAR_EN=1)

Access Type	31-22	21	20	19	18-16	15-12	11-2	1	0
CDM	Not Used	0	CS2	ROM	Function	Reserved	1 K-DWORD Register Address	0	0
ELBI	Not Used	1	0	ROM	Function	Reserved	1 K-DWORD Register Address	0	0

Table 3-18AXI DBI Address Bus Layout in EP Mode (CX_SRIOV_ENABLE =1 or CX_ARI_ENABLE =1)

Access Type	32	31	30	29-22	21	20-16	15-12	11-2	1	0
CDM	0	CS2	ROM	VF#	VF Active	PF#	Reserved	1 K-DWORD Register Address	0	0
ELBI	1	0	ROM	VF#	VF Active	PF#	Reserved	1 K-DWORD Register Address	0	0

Note: If DBI_MULTI_FUNC_BAR_EN=1, refer to "DBI_MULTI_FUNC_BAR_EN Parameter (EP Mode)"

11.3.2.3.4 Features and Limitations

The following limitations exist for the LBC.

- CDM Access Restrictions:
  - Maximum payload Length =1 DWORD.
  - PCIe MEM requests with Length > 1 are processed as completer abort.
  - PCIe MEM requests with Length =0 are processed as completer abort.
  - PCIe IO or CFG requests with Length !=1 are processed as malformed TLPs.
- ELBI Access Restrictions:
  - CS2 bit must be 1’b0.
  - PCIe MEM requests with Length =0 are processed as completer abort.
  - PCIe IO or CFG requests with Length !=1 are processed as malformed TLPs.
  - Does not support "Atomic Operations (AtomicOps)"
  - Does not support TPH. Core does not forward the TPH bits to the ELBI.
  - Data (Read)
- DBI Data/Address Port Widths (with no AXI bridge):

- Data: 32 bit
- Address: 32 bit

11.3.2.3.4.1 Additional AXI DBI Limitations

The following limitations exist when you configure the controller to use the AXI bridge module.

- The DBI address must be a DWORD address.
- AXI transfer widths of 8, 16, and 32 bits are supported. The AXI dedicated/shared DBI interface returns an error response to requests with width greater than 32 bits. The type of error response is programmable through AMBA_ERROR_RESPONSE_DEFAULT_OFF register1.
- For AHB you get an error response whenever the size is not equal to 32 bits.
- The CDM only supports 32-bit data access. The AXI dedicated/shared DBI interface returns an error response to multi beat requests. The type of error response is programmable through AMBA_ERROR_RESPONSE_DEFAULT_OFF register.
- Byte access within a DWORD for DBI requests is not possible for AHB.
- Byte access within a DWORD for DBI WRITE requests is possible for AXI (not for AHB)
  - Dedicated DBI: The write strobes dbi_wstrb[3:0] are sent to the LBC.
  - Shared DBI: The relevant four strobes from slv_wstrb are sent to the LBC

A zero-byte write over DBI is processed as a read and should never be issued. The AXI dedicated/shared DBI slave drops a zero-byte write before it reaches the internal (native controller) DBI and returns an error response. The type of error response is programmable through AMBA_ERROR_RESPONSE_DEFAULT_OFF register.

The type of error response returned by the AXI dedicated/shared DBI slave is the same as that programmed for UR completions in the AMBA_ERROR_RESPONSE_DEFAULT_OFF register.

11.3.2.4 Flow Control

The flow control mechanism is divided into two phases: initialization and update. The controller automatically performs both of these phases with minimal support required from your application. By default the RADM is responsible for returning flow control credits when it reads data out of the receive queues.

The controller provides optional I/O signals to enable your application to handle flow control returns in an application-specific manner. For error scenarios in bypass queue mode your application can decide whether to return credits or not based on state of the Bypass/RTRGT1 Interface signals described in the table below.

Table 11-87. Bypass/RTRGT1 Interface Signals to Consider for Credit Return (AXI Bridge not Present)

Bypass/RTRGT1 Interface Signal	State	Credit return policy
radm_bypass_tlp_abort radm_trgt1_tlp_abort	Asserted	Credit needs to be returned
radm_bypass_dllp_abort radm_trgt1_dllp_abort	Asserted	Credit need not be returned
radm_bypass_ecrc_err radm_trgt1_ecrc_err	Asserted	Credit need not be returned. But, it is strongly recommended to return the credit if the TLP is not ambiguous.

11.3.2.4.1 Calculation of Initial Credits and Receive Buffer Sizes

The default buffer sizes and credits for each TLP type are automatically calculated from the number of lanes, the controller datapath width, the maximum PCIe payload , flow control update latencies, internal delays, and the PHY latency.

You can determine the transmit posted, non-posted, and completion credits that are advertised by the receiver at the other end of the link by reading the following port logic registers:

- Transmit Posted FC Credit Status (TX_P_FC_CREDIT_STATUS)
- Transmit Non-Posted FC Credit Status (TX_NP_FC_CREDIT_STATUS)
- Transmit Completion FC Credit Status (TX_CPL_FC_CREDIT_STATUS)

NOTE

You can override the default flow control update latency timer value using the FC Latency Timer Override Value field (TIMER_MOD_FLOW_CONTROL) in the Queue Status register (Q_STATUS).

Scaled Flow Control

The controller supports Scaled Flow Controls described in *PCI Express Base Specification, Revision 4.0, Version 0.7* with the following features and limitations:

- This feature when enabled applies individually to all P, NP or CPLs.
- This feature when enabled applies to all VCs.
- The allocated initial credits will scale down to 127 header credits and 2047 data credits, if the link partner does not support scaling.

- At larger scaling factors, it is not possible to return single credits.
- When scaling, the minimum number of header and data credits must be calculated as follows:
  - Minimum Number of Header Credits = 32 * Scaling Factor, and
  - Minimum Number of Data Credits = 512 * Scaling Factor.
- When manually enabling Flow Control Scaling on a new configuration by editing the configuration file:
  - If you are modifying the Header and Data credits, the scaling factors must be specified.
  - If the Header and Data credits have default values, there is no need to add the scaling factors.

11.3.2.5 ATS, PASID and PRS

This section discusses the following optional features of the controller as per PCI Express Address Translation Services 1.1 Specification, January 26 2009:

11.3.2.5.1 Address Translation Services (ATS)

The controller implements ATS with the following features and limitations.

- To send an ATS TLP, you must supply it on an application transmit client interface (XALI0/1/2)
- You can also send an Invalidation Completion using the VMI (ven_msg_type[4:0]=0x12; ven_msg_code[7:0]=0x02; ven_msg_data[63:0]={ITAG Vector, CC, Device ID}).
- The controller does not pass Rx ATS Invalidate Requests and ATS Invalidate Completions to the SII interface because they are too big. All ATS Invalidate Requests and ATS Invalidate/Completions and pass through to TRGT1 (or AXI bridge Master) regardless of the filtering rule settings.
- The controller routes all Rx ATS TLPs to the TRGT1 receive interface, or to the AXI bridge master interface.
- The controller also routes all Rx ATS TLPs. (except ATS Invalidate Requests ATS Invalidate Completions) to the SII interface.
- When the AXI bridge is enabled:
  - Requests for multiple translations are not supported
  - Translation requests are supported for a single address only (length =two DWORDs)
- The controller does not implement the ATS cache.

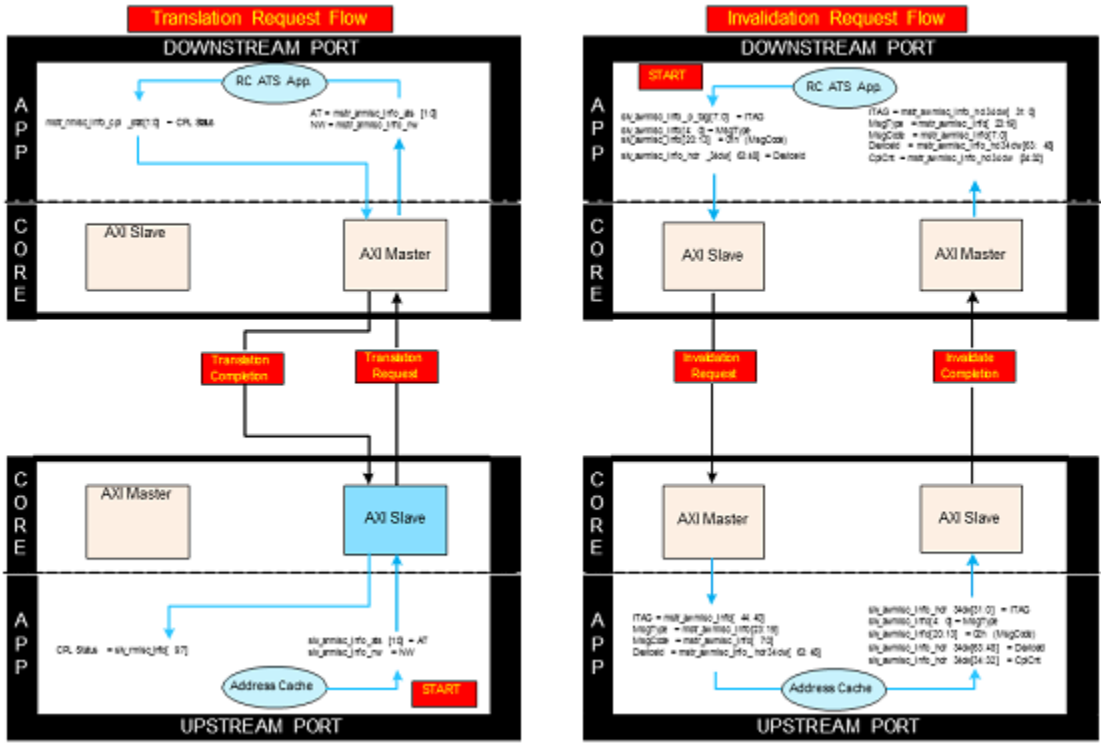


Figure 11-47. AXI ATS Sideband Signal Usage

11.3.2.5.2 Process Address Space ID (PASID)

The controller includes the PASID capability structure (in EP mode)

- The PASID TLP prefix is additionally permitted on Translation and Invalidation Requests when ATS is enabled. The controller does not implement the ATS cache
- SRIOV is supported.
- he controller does not implement the IDO ordering rules based on PASIDs.

11.3.2.5.3 Page Request Services (PRS)

The controller supports so that your EP application can use the memory page request mechanism. Your RC application can receive PRS messages.

- It supports memory requests with and without translated addresses (ATS).
- The controller does not process the PASID, Execute Requested, or Privileged Mode Requested fields of the PASID TLP prefix. Therefore, your application must determine if the rules are met for these fields. Consequently, the PCIe controller does not process the PASID TLP Prefix to determine if PASIDs are permitted on Address

Translation or Invalidation Requests, or on untranslated memory requests. If required, your application must check if the PASID Width value is valid.

- A PASID TLP prefix is an End-End TLP prefix that an endpoint can prepend to requests that it gener- ates.

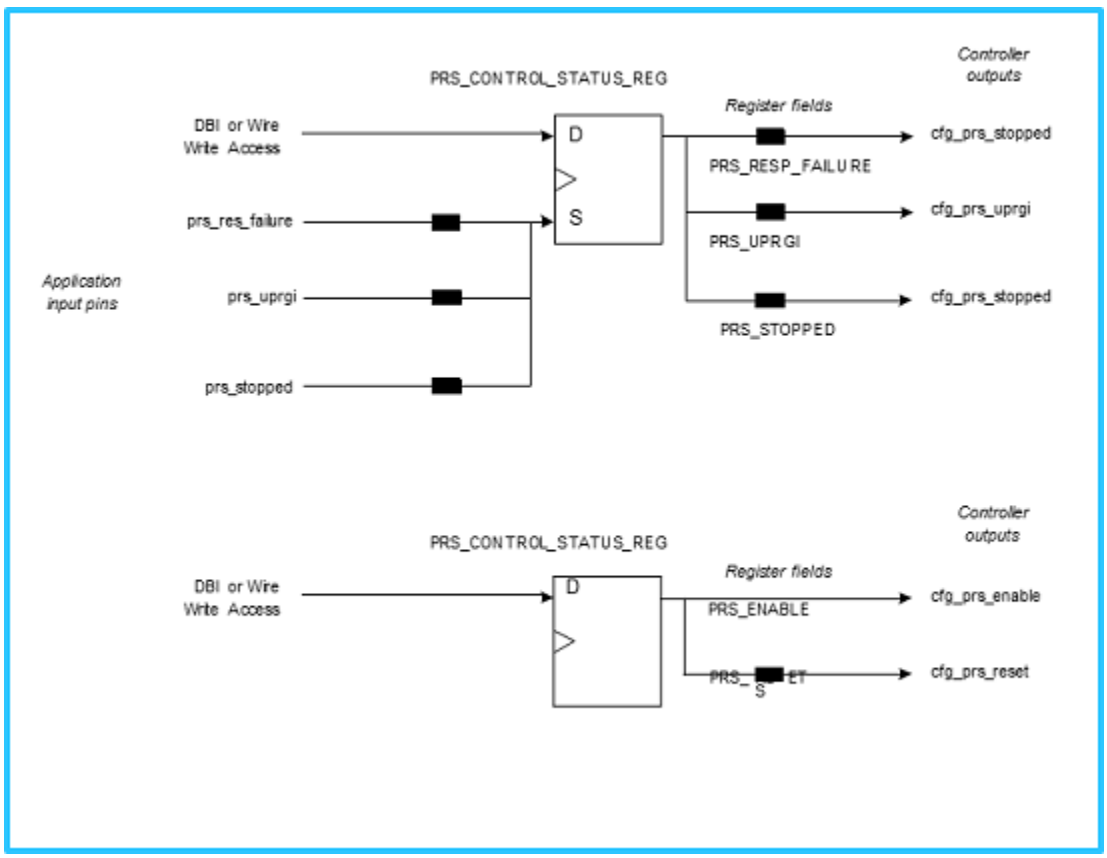


Figure 11-48. PRS Control and Status Register-to-Pin Connections

You can clear the CX_FLT_MASK_PRS_DROP bit in the MASK_RADM_2 field of FILTER_MASK_2_OFF to instruct the controller to pass1 PRS messages to the application on the Rx interfaces; otherwise they are silently dropped.

11.3.2.6 Peer-to-Peer Support (P2P)

In certain multi-port PCI Express applications such as a PCIe switch or multi-port root complex, it is useful to allow TLPs to flow from one endpoint port to another1, instead of strictly flowing up and down the PCIe hierarchy. The PCI Express specification describes this capability in terms of routing peer-to-peer (P2P) transactions between ports. Though identified by the PCIe specification, the mechanism and support of this feature is considered to be outside the scope of the base specification. The DM (RC mode) provides support for this mechanism. The P2P support in RC mode modifies the operation of the controller in the following ways:

- The controller preserves the original requester or completer IDs for P2P transmit traffic while automatically adding the appropriate ID for locally generated traffic.
- Only received completions corresponding to requests originated by the device are subject to the normal receive completion LUT processing. Completions of P2P requests do not reference the LUT.
- The controller preserves the original requester or completer IDs for P2P transmit traffic while auto- matically adding the appropriate ID for locally generated traffic.
- All TLP header fields are preserved in both transmit and receive directions. Therefore the controller provides all reserved fields at receive interfaces and expects your application to drive them at the client interfaces.
- The controller optionally appends ECRC fields to locally generated TLPs, passing P2P traffic along without modification.

The following limitations apply with respect to the P2P feature:

- Receive queue architecture must be segmented-buffer.
- No ECRC stripping.
- The controller does not split transmit packets. Your application must handle this condition when the peer-to-peer packet exceeds the max payload limitation.

11.3.2.7 Completion Timeout Ranges

Timeout ranges are supported as defined in the *PCI Express Base Specification, Revision 4.0, Version 0.7*. The controller supports all ranges. The Device Control 2 Register has a default equal to the default in the specification: 0000b Default range: 50 us to 50 ms. When the default is used, then the timeout is in Range B: 0101b: 16 ms to 55 ms. This range was chosen for the default because the *PCI Express Base Specification, Revision 4.0, Version 0.7* states, "It is strongly recommended that the completion Timeout mechanism not expire in less than 10 ms". The table below illustrates the specification values versus the PCI Express controller values for the ranges.

NOTE

As the PCIe specification states, This mechanism is intended to be activated only when there is no reasonable expectation that the completion is returned, and should never occur under normal operating conditions.

Table 11-88. Comparison of PCIe Specification and PCIe Core Completion Timeout Ranges

Range	Encoding	Spec Minimum	Spec Maximum	PCIe controller Minimum	PCIe controller Maximum

Table continues on the next page...

Table 11-88. Comparison of PCIe Specification and PCIe Core Completion Timeout Ranges (continued)

Default	0000b	50 us	50 ms	28 ms	44 ms (M-PCIe: 45)
A	0001b	50 us	100 us	65 us	99 ms (M-PCIe: 100)
A	0010b	1 ms	10 ms	4.1 ms	6.2 ms (M-PCIe: 6.4)
B	0101b	16 ms	55 ms	28 ms	44 ms (M-PCIe: 45)
B	0110b	65 ms	210 ms	86 ms	131 ms (M-PCIe: 133)
C	1001b	260 ms	900 ms	260 ms	390 ms (M-PCIe: 398)
C	1010b	1s	3.5 s	1.8 s	2.8 s (M-PCIe: 2.8)
D	1101b	4s	13 s	5.4 s	8.2 s (M-PCIe: 8.4)
D	1110b	17s	64 s	38 s	58 s (M-PCIe: 59)

11.3.2.8 Crosslink

Crosslink allows a downstream port to be connected to another downstream port or an upstream port to be connected to another upstream port. When a crosslink capable port negotiates a crosslink connection, the port changes its behavior accordingly. For this reason crosslink is supported in the DM controller because it contains the functionality for both upstream and downstream support.

Crosslink provides more than simply the ability to connect two like ports together. For instance, a DM operating in EP mode that negotiates a crosslink connection switches to RC mode and follows all of the *PCI Express Base Specification, Revision 4.0, Version 0.7* rules for an RC.

11.3.3 DMA Controller

This section describes the embedded multichannel DMA controller.

11.3.3.1 DMA Overview

The RC system CPU, or the EP application CPU, can offload the transferring of large blocks of data to the embedded DMA controller, leaving the CPU free to perform other tasks. You can configure the DMA to have one to eight read channels and one to eight write channels. It can simultaneously perform the following types of memory transactions (as shown in the figure 'System Level View of DMA'):

DMA write: Transfer (copy) of a block of data from local (application) memory to remote (link partner) memory.

DMA read: Transfer (copy) of a block of data from remote (link partner) memory to local (application) memory.

Therefore the DMA supports full duplex operation, processing read and write transfers at the same time, and in parallel with normal (non-DMA) traffic. Upon completion of a DMA transfer or an error, the DMA optionally interrupts the local CPU or sends an interrupt MWr (IMWr) to the remote CPU.

In linked list mode, the DMA fetches the transfer control information (called channel context) for each transfer (block), from a list of DMA elements that you have constructed in local memory.

The DMA supports “Native Core Operation (No AXI Bridge)”. You can also use the DMA with the AXI/AHB bridge, in which case the DMA is located between the native PCIe controller and the AXI bridge module.

11.3.3.2 DMA Architecture

This section describes the architecture of the DMA.

11.3.3.2.1 Architecture Overview

A DMA write and read channel operate independently to maximize the performance of the DMA read and write data transfers over the PCIe link. DMA with multiple read channels uses a weighted round robin (WRR) arbitration scheme to select the next read channel to be serviced. The same applies for multiple write channels.

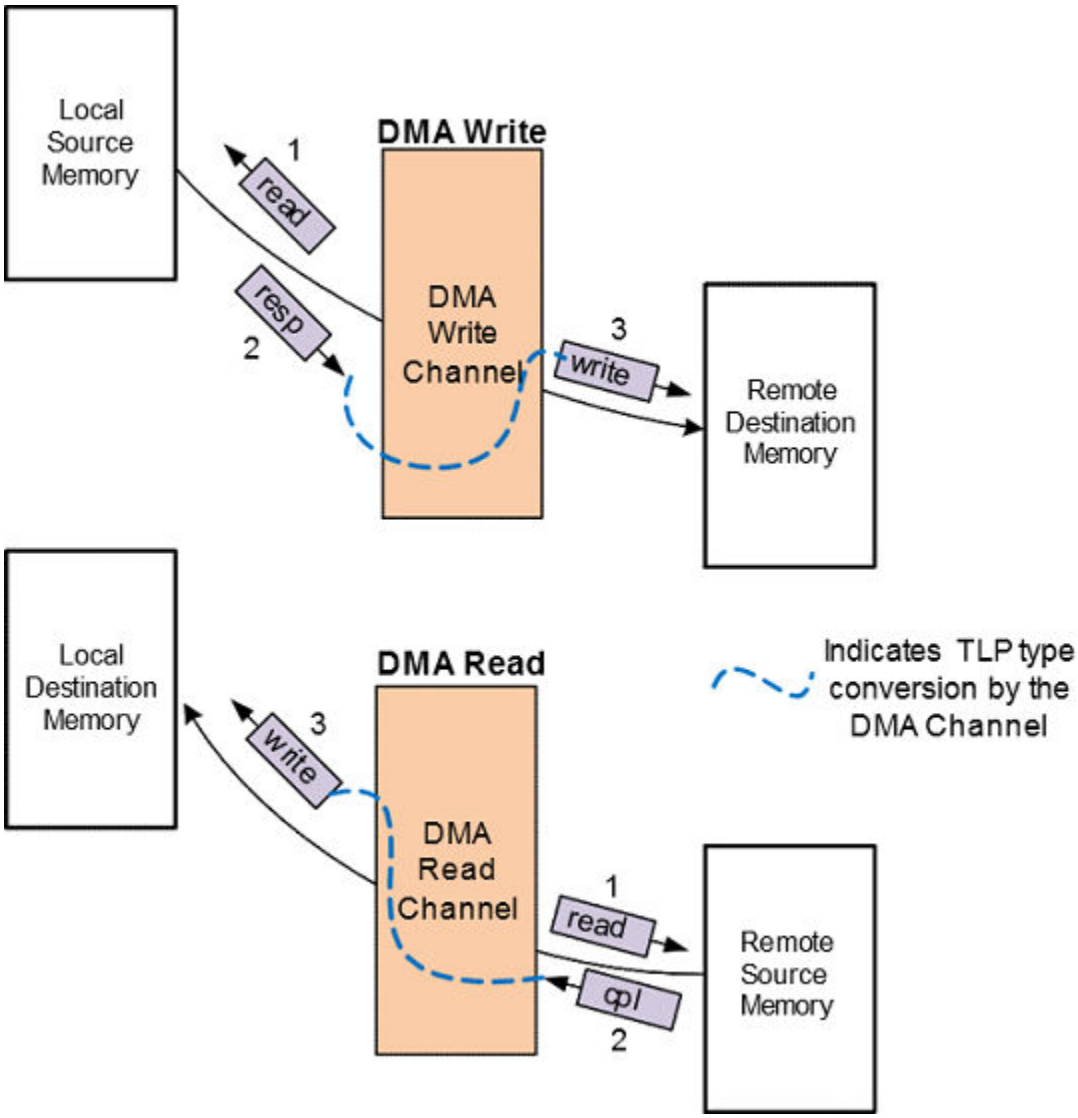


Figure 11-49. System Level View of DMA

11.3.3.2.1.1 DMA Read Transfer

The DMA injects multiple MRd requests of size less than or equal to the minimum of {Max Read Request Size, Max Payload Size} into the outbound request path, directed toward the remote link partner. The DMA converts the read responses into write requests, that it then transmits to the local application. When the DMA data transfer is complete, the CPU is notified. For a read transfer, the SAR is the address of the remote memory, and the DAR is the address of the local memory.

11.3.3.2.1.2 DMA Write Transfer

The DMA injects multiple MRd requests of size less than or equal to Max_Payload_Size into the inbound request path, directed toward the local application. The DMA converts the read responses into MWr TLPs, that it then transmits to the remote link partner. When the DMA data transfer is complete, the CPU is notified. For a write transfer, the SAR is the address of the local memory, and the DAR is the address of the remote memory.

11.3.3.2.2 Interrupts and Error Handling

11.3.3.2.2.1 Interrupt Handling

The DMA generates two interrupts per channel:

Done: The DMA successfully completes the transfer.

Abort: The DMA fails to complete the transfer, or an error occurs during the transfer.

The interrupt handling mechanism is different for linked list (LL) mode (than non LL mode), and there are also some differences between the read and write channels.

11.3.3.2.2.2 Non Linked List Mode

You enable the local and remote interrupts through the local and remote interrupt enable (LIE and RIE) bits:

DMA_CH_CONTROL1_OFF_WRCH_0.lie and
DMA_CH_CONTROL1_OFF_WRCH_0.rie.

In the write channel, there is only one error condition that results in an abort interrupt. For more details, see "Linked List Mode". You mask, clear, and read the status of each of the two interrupts (done and abort) through the DMA interrupt registers as indicated in figure 'Write Interrupt Generation - Non Linked List Mode - Shown For Write Channel #0'.

In the read channel, there are five error conditions that results in an abort interrupt. For more details, see "Linked List Mode". You mask and clear each of the two interrupts (done and abort) through the DMA interrupt registers as indicated in figure 'Read Interrupt Generation - Non Linked List Mode - Shown For Read Channel #0'. However, you can read the status of each of the five abort errors (that contribute to the abort interrupt) through DMA_READ_ERR_STATUS_LOW_OFF and DMA_READ_ERR_STATUS_HIGH_OFF.

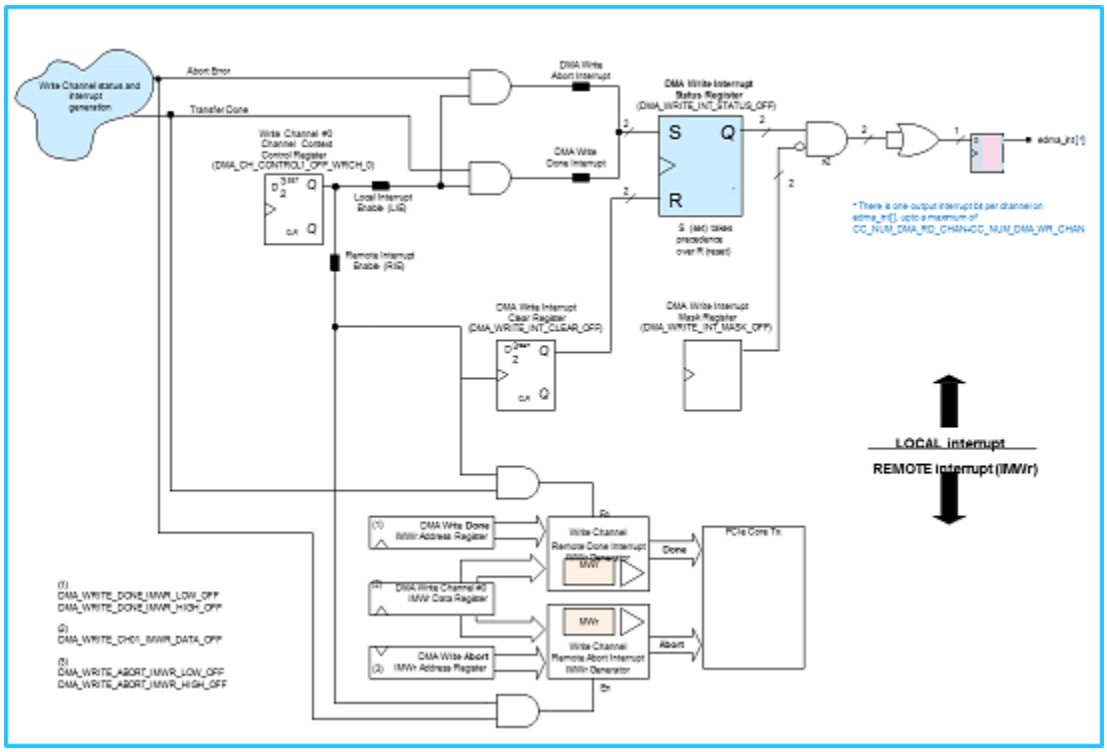


Figure 11-50. Write Interrupt Generation - Non Linked List Mode - Shown For Write Channel #0

If you want a remote interrupt and not a local interrupt then:

- Set LIE and RIE
- Mask the local interrupt using the mask register

This will allow you to poll the status register to distinguish a DONE from an ABORT

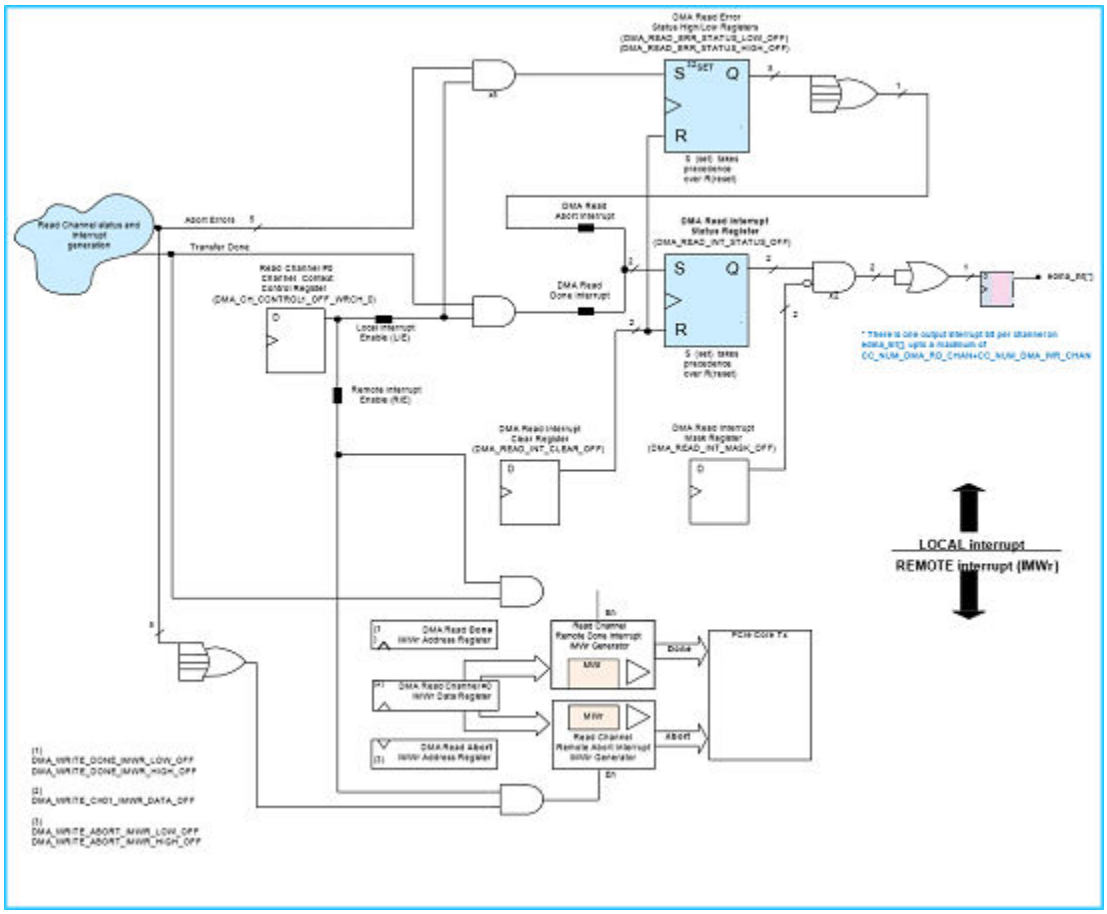


Figure 11-51. Read Interrupt Generation - Non Linked List Mode - Shown For Read Channel #0

11.3.3.2.2.3 Linked List Mode

The LIE and RIE bits in the LL element enable the channel done interrupts (local and remote). The LLLAIE and LLRAIE bits of the DMA_WRITE_LINKED_LIST_ERR_EN_OFF/ DMA_READ_LINKED_LIST_ERR_EN_OFF registers enable the channel abort interrupts (local and remote). In the write channel, there are two error conditions that results in an abort interrupt. For more details, see "Linked List Mode". You mask and clear each of the two interrupts (done and abort) through the DMA interrupt registers as indicated in the figure below. You can read the status of each of the two abort errors (that contribute to the abort interrupt) through the DMA_WRITE_ERR_STATUS_OFF register.

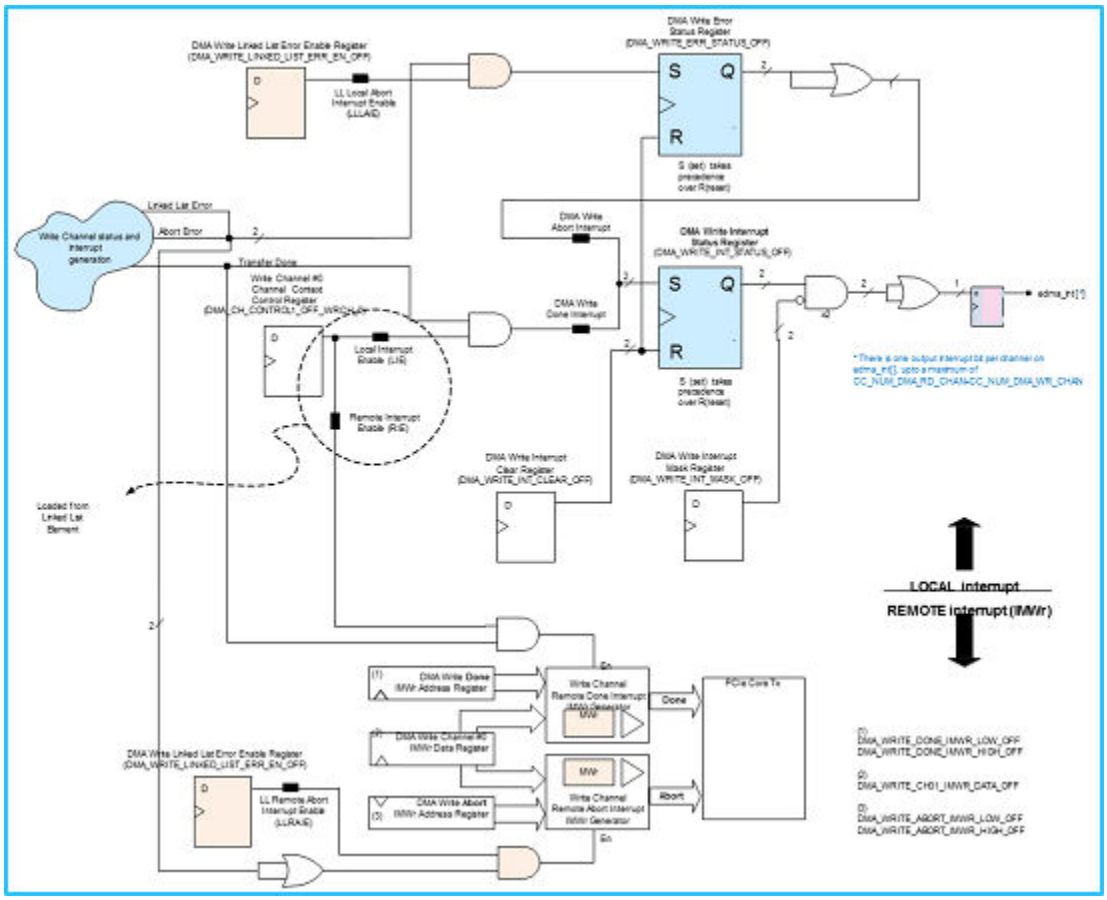


Figure 11-52. Write Interrupt Generation - Linked List Mode - Shown For Write Channel #0

In the read channel, there are six error conditions that results in an abort interrupt. For more details, see "Linked List Mode". You mask and clear each of the two interrupts (done and abort) through the DMA interrupt registers as indicated in the figure below. You can read the status of each of the six abort errors (that contribute to the abort interrupt) through the DMA_READ_ERR_STATUS_LOW_OFF and DMA_READ_ERR_STATUS_HIGH_OFF registers.

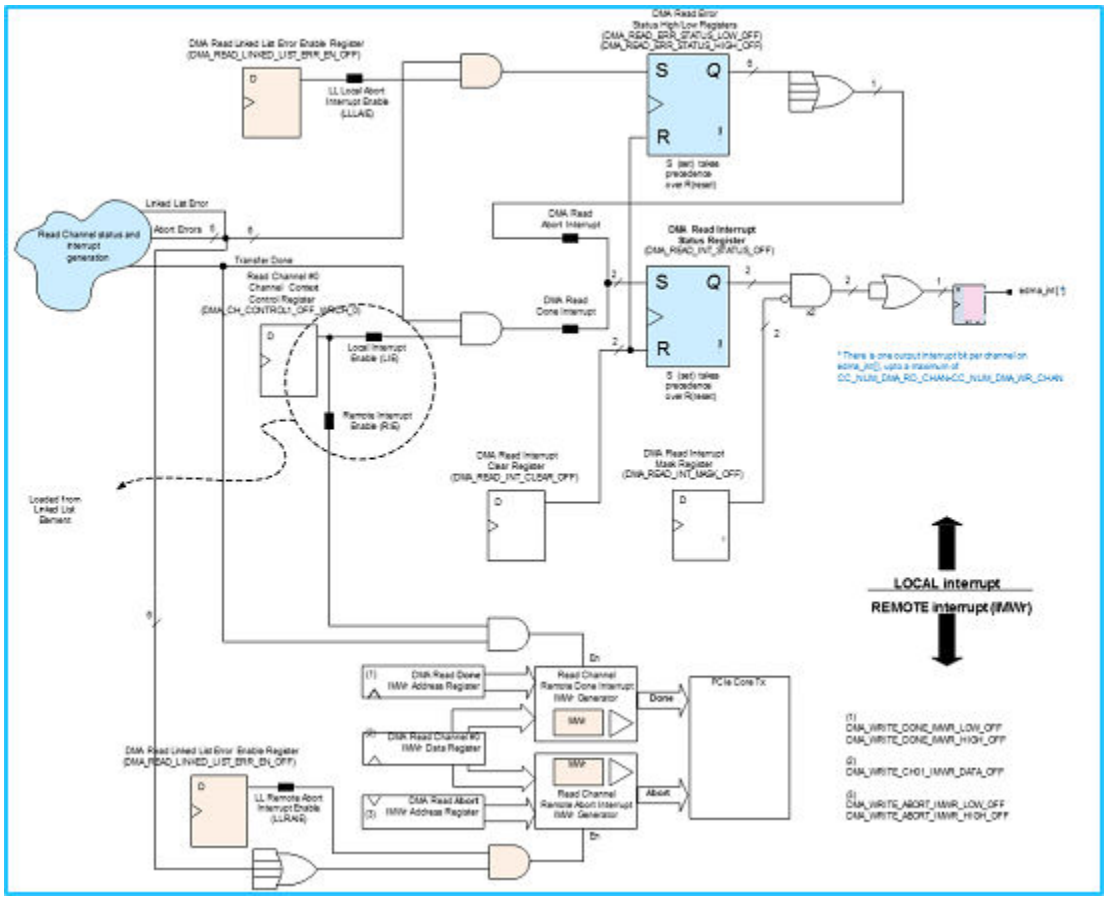


Figure 11-53. Read Interrupt Generation - Linked List Mode - Shown For Read Channel #0

NOTE

In non-linked list mode, LIE acts as a global switch. However when in linked list mode, LIE is just local to the current linked list element and the global switch is LLLAIE.

NOTE

If the DMA driver is running on the host and the interrupt service routine is reading local interrupts to determine if the transfer is successful, then you must set LIE and RIE in the same element and you should mask or ignore the local interrupt pin. Setting RIE and LIE in element A followed by RIE (only) in element B is not a verified usage scenario.

11.3.3.2.3 Linked List Mode

The DMA provides a linked list (LL) mode to efficiently move data from source to destination with minimal intervention from the local CPU. This mode provides an alternative to programming the DMA multiple times to transfer multiple blocks of data.

The programming information (address, size, and so on) for each block of memory is pre-programmed by your software into a LL element (also known as a descriptor) in local memory. Each element (called a data element) in the LL structure (called a transfer list) can transfer up to 4 GB of data. You enable LL operation for a channel, by setting the LLE field of the DMA_CH_CONTROL1_OFF_WRCH_0 (Replace WRCH with RDCH for a read channel) register to 1. You can enable LL mode independently for each channel. When you enable LL for more than one channel, then you must have a separate LL structure in local memory for each channel. Your application must produce the LL element structure in local memory as shown in the figure below. Normally, all of the elements are contiguous (one after the other) in memory, and each element has six DWORDs containing the information about the block of data to be transferred. You program the channel context registers (DMA_LLP_LOW_REG_WRCH_0 and DMA_LLP_HIGH_OFF_WRCH_0) with the location of where you have placed the LL element structure in local memory.

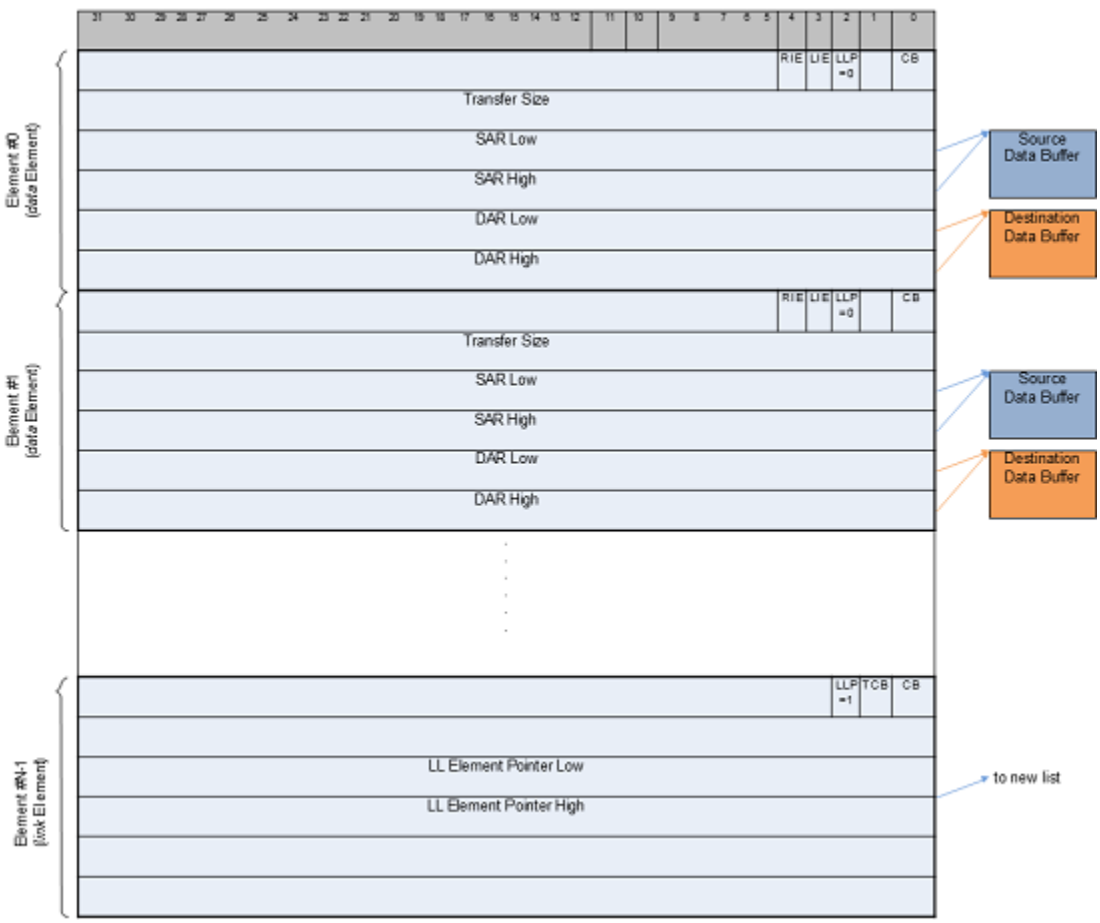


Figure 11-54. Linked List Element/Descriptor Structure in Local Memory with N Elements

When you start the DMA transfer (by writing to the DMA Write Doorbell Register DMA_WRITE_DOORBELL_OFF or DMA Read Doorbell Register DMA_READ_DOORBELL_OFF), the DMA reads (consumes) each element from local memory, and loads the information (SAR, DAR, size, and so on) from that element into the channel context registers in the DMA. These channel context registers determine the operation of the channel that the DMA controller is currently servicing. The DMA then proceeds to transfer the block of data (as defined by the element), and when it is finished, reads the next element from local memory. Normally, all of the elements are contiguous (one after the other) in memory, with the starting address defined in the channel context DMA Linked List Pointer Low Register DMA_LLP_LOW_OFF_WRCH_0. When you want to jump in local memory to another element list (or recycle the consumed elements), then you set the LLP bit in the element (for example, link element #N-1 in figure shown)), specify the location of the next element structure using the LL Element Pointer DWORDs (as indicated in the figure), and, set TCB to 1 (for recycling) or to 0 (to jump to another list).

11.3.3.2.3.1 Relationship Between Element DWORDs and Channel Context Registers

Notice the similarity between a data element and the DMA Channel Context registers for each channel. Each element has six DWORDs as in figure 'Linked List Element/ Descriptor Structure in Local Memory with N Elements'. There are eight channel context registers (DWORDs) for a channel. The DMA loads the six element DWORDs into the following channel context:

- CB, LLP, LIE, and RIE fields of the DMA Channel Control 1 register.
- DMA Transfer Size
- DMA SAR Low and DMA SAR High
- DMA DAR Low and DMA DAR High

The definitions of the element DWORD bit fields are the same as the DMA Channel Context registers described in the "DMA Software Register Map", with the exception of the LIE and RIE bits. The LIE and RIE bits in a LL element, only enable the done interrupt. In non-LL mode, the RIE and LIE bits (in the channel context registers) enable the done and abort interrupts.

When the LLP field of the first DWORD in the element is set to 1, the element is a link element. The DMA only loads the following information into the channel context registers:

- CB, TCB, and LLP bits of the first DWORD. The LIE and RIE bits are not defined in a link element.
- LL Element Pointer (3rd and 4th DWORDs) into the DMA Linked List Pointer registers (DMA_LL- P_LOW_REG_WRCH_0 and DMA_LLP_HIGH_OFF_WRCH_0).

Linked List Operation

- When you enable linked list operation (DMA_CH_CONTROL1_OFF_WRCH_0.lle =1b1), then the DMA overwrites the following DMA channel context registers with the following information from linked list data elements:
  - The CB, LLP, LIE, and RIE fields of the DMA Channel Control 1 register.
  - DMA Transfer Size
  - DMA SAR Low & High
  - DMA DAR Low & High
- The structure of a link element is different to that of the data element. A data element has no TCB field. A link element has no LIE or RIE fields. It has no SAR, DAR, or Transfer Size DWORDs, but has LL Element Pointer DWORDs instead of the SAR DWORDs.
- The LIE and RIE bits in a LL element, only enable the done interrupt. In non-LL mode, the RIE and LIE bits enable the done and abort interrupts. For more information, see "Interrupts and Error Handling".

11.3.3.3 Using the DMA

This section describes how to use the DMA. It can simultaneously perform the following types of memory transactions:

- DMA write : Transfer (copy) of a block of data from local memory to remote memory
- DMA read :Transfer (copy) of a block of data from remote memory to local memory

Therefore the DMA supports full duplex operation, processing read and write transfers at the same time, and in parallel with normal (non-DMA) traffic. Upon completion of a DMA transfer or an error, the DMA sends an interrupt MWr (IMWr).

The channels are named read channel 7..0 and write channel 7..0. Each channel is programmed using a number of registers.

11.3.3.3.1 Source and Destination Address Registers (SAR, DAR)

The DMA channel context SAR and DAR registers (DMA_SAR_LOW_OFF_WRCH_01 / DMA_SAR_HIGH_OFF_WRCH_0 / DMA_DAR_LOW_OFF_WRCH_0 / DMA_DAR_HIGH_OFF_WRCH_0) provide support for remote-to-local, and local-to-remote PCIe address mapping. You program the start of the local and remote data buffers using these registers, and the DMA increments the SAR and DAR as the DMA transfer progresses. For a write transfer, the SAR is the address of the local memory, and the DAR is the address of the remote memory, as shown in Figure 'Write Transfer: SAR and DAR for Write Channel 0'.

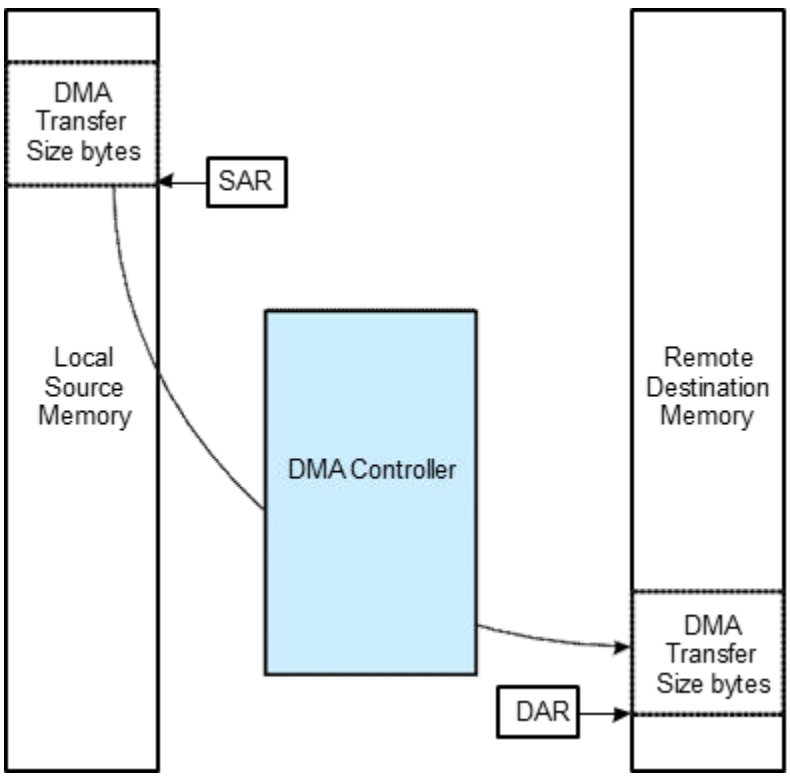


Figure 11-55. Write Transfer: SAR and DAR for Write Channel 0

For a read transfer, the SAR is the address of the remote memory, and the DAR is the address of the local memory, as shown in the figure below.

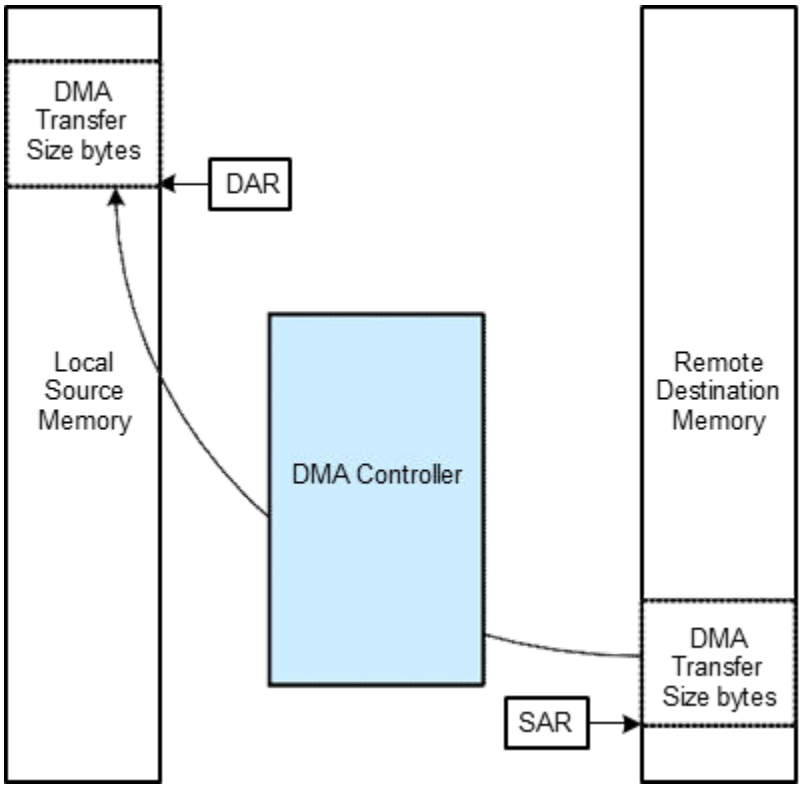


Figure 11-56. Read Transfer: SAR and DAR for Read Channel 0

The presence of the DMA controller does not affect:

- Normal filtering rules for inbound TLPs (for example, BAR checking in an Endpoint, as described in as described in “Receive Filtering”
- The operation of any internal or external address translation as described in “Internal Address Translation Unit (iATU)”).

NOTE

When you do not want the iATU to translate outbound requests that are generated by the internal DMA module, then you must implement one of the following approaches:

- Ensure that the combination of DMA channel programming and iATU control register programming, causes no translation of DMA traffic to be done in the iATU.
- Activate the DMA bypass mode to allow request TLPs which are initiated by the DMA controller to pass through the iATU untranslated. You can activate DMA bypass mode by setting the DMA Bypass field of the iATU Control 2 Register (IATU_REGION_C-TRL_OFF_2_OUTBOUND_0).

11.3.3.3.2 DMA Transfer Size Register

You program the DMA transfer size using the DMA Transfer Size Register (DMA_TRANSFER_SIZE_OFF_WRCH_0). The DMA decrements the value in this register as the DMA transfer progresses. You can read this register at any time to determine the number of bytes remaining to be transferred. When all bytes are successfully transferred, the result is zero. The DMA transfer size can be of any value from one byte, up to a maximum of 4 GB.

11.3.3.3.3 Starting The DMA Transfer

After you program the DMA controller registers (including writing to the DMA Read Engine Enable or DMA Write Engine Enable register), you start a DMA transfer by writing the channel number to the Doorbell Number field of the DMA Write Doorbell Register (DMA_WRITE_DOORBELL_OFF) or DMA Read Doorbell Register (DMA_READ_DOORBELL_OFF). You can program and start both a read and a write transfer at the same time. The DMA supports full duplex operation, processing read and write transfers at the same time and in parallel with normal (non-DMA) traffic. You can program and start any number of channels in the DMA controller sequentially or simultaneously.

Sequentially: configure and start a channel, and then configure and start another channel

Simultaneously: configure multiple channels, and then start multiple channels

NOTE

You must not write to any of the context registers for a particular channel after you start the channel by writing the channel number to the Doorbell Number field of the DMA Write Doorbell Register (DMA_WRITE_DOORBELL_OFF) or DMA Read Doorbell Register (DMA_READ_DOORBELL_OFF).

11.3.3.3.4 Detecting the End of The DMA Transfer

If a DMA transfer proceeds without any errors, it stops automatically when finished.

11.3.3.3.4.1 Detecting End of Transfer without Errors

The normal end of a DMA transfer is detected by any of the following methods:

- Done local interrupt (pin) asserted.
- Done remote interrupt (IMWr) received.

- Channel status field of the Channel Control 1 register is Stopped, and the DMA Transfer Size register is 0x0.
- Polling of the DMA Write Interrupt Status Register (DMA_WRITE_INT_STATUS_OFF) or DMA Read Interrupt Status Register (DMA_READ_INT_STATUS_OFF).

11.3.3.3.4.2 Detecting End of Transfer with Errors

The abnormal end of a DMA transfer is detected by any of the following methods:

- Abort local interrupt (pin) asserted.
- Abort remote interrupt (IMWr) received.
- Polling of the DMA Write Interrupt Status Register (DMA_WRITE_INT_STATUS_OFF) or DMA Read Interrupt Status Register (DMA_READ_INT_STATUS_OFF).
- Channel Status field of the Channel Control 1 register is halted. When the DMA controller detects an error, it forces the DMA to stop issuing requests for the channel. It also sets the channel status field of the Channel Control 1 register to halted, generates an abort interrupt (if enabled), and sends an abort IMWr (if enabled). The DMA Transfer Size register indicates the remaining number bytes to be transferred, except when there is an AXI write error during a DMA read transfer. For more details, see table 'Possible Sources of an Error during a DMA Read Transfer'.
- Channel Status field of the Channel Control 1 register is Stopped, and the DMA Transfer Size register is not 0x0. You have prematurely stopped this channel as described in “Stopping the DMA Transfer (Software Stop)”.

11.3.3.3.5 Stopping the DMA Transfer (Software Stop)

You can manually abort (stop) the DMA transfer by writing the channel number to the Doorbell Number field and writing 1 to the Stop field in DMA_WRITE_DOORBELL_OFF. This causes the DMA to:

- Place the channel in a Stopped state.
- The channel Status field in DMA_CH_CONTROL1_OFF_WRCH_0 is Stopped and the value in DMA_TRANS- FER_SIZE_OFF_WRCH_0 will not be 0x0.
- Wait for all outstanding pending transactions.
- Assert the abort interrupt (if it is enabled) in DMA_WRITE_INT_STATUS_OFF.

You might do this as part of error handling which is described in “Error Handling Assistance by Remote Software”, which is only necessary during software development if you incorrectly program the DMA write channel DAR. You might also do this as part of a function level reset (FLR). FLR does not directly affect the DMA transfer so you must manually stop the DMA transfer before initiating an FLR.

Use This Feature With Caution in Linked List Mode

You should not use the STOP feature in operational mode. It is only a debug feature and has associated hazards.

- Before setting the Stop bit, you must read the channel status field (CS) of the DMA Channel Control 1 register to ensure that the corresponding channel is Running (transferring data). To eliminate the possibility of a race condition between these two actions (read and write), you should confirm the presence of the abort bit in the DMA Write Interrupt Status Register (DMA_WRITE_INT_STATUS_OFF) or DMA Read Interrupt Status Register (DMA_READ_INT_STATUS_OFF) and check the status of any fatal error if some other event has occurred.
- After a Stop event, you cannot seamlessly resume the transfer again because the DMA will not carry on exactly from the point that it was stopped at. Therefore you must setup and start the complete channel context again, including the
  - linked list pointer
  - linked list structure in memory
  - related PCS and CCS values

11.3.3.3.6 TLP Generator Header Control

You must program bits [31:12] of the DMA Channel Control 1 Register (DMA_CH_CONTROL1_REG_ WRCH_0/DMA_CH_CONTROL1_OFF_RDCH_0), so that the DMA TLP generator correctly sets the TC, RO, NS, AT, and FN TLP header fields for any TLPs that it generates. When SRIOV is enabled, then you must also program DMA_CH_CONTROL2_OFF_WRCH_0/ DMA_CH_CONTROL2_OFF_RDCH_0.

11.3.3.3.7 DMA Software Register Map

This section discusses the grouping and access methods of the DMA registers. The following registers are present when the DMA is enabled.

The DMA Global Registers control the global settings

Table 11-89. DMA Global Registers

Name	Description

Table continues on the next page...

Table 11-89. DMA Global Registers (continued)

0x8_0000 +0x000	DMA Arbitration Scheme for TRGT1 Interface
+0x008	Number of DMA Channels
+0x00C	DMA Write Engine Enable
+0x010	DMA Write Doorbell
+0x018	DMA Write Engine Channel Arbitration Weight Low
+0x01C	DMA Write Engine Channel Arbitration Weight High
+0x02C	DMA Read Engine Enable
+0x030	DMA Read Doorbell
+0x034	DMA Global Read Control
+0x038	DMA Read Engine Channel Arbitration Weight Low
+0x03C	DMA Read Engine Channel Arbitration Weight High

The DMA Interrupt Registers control the generation of local interrupts and remote IMWr. The IMWr addresses and data are stored in RAM.

Table 11-90. DMA Interrupt Registers

Byte Offset	Description
0x8_0000 +0x4C	DMA Write Interrupt Status
+0x50	Not used (RsvdP).
+0x54	DMA Write Interrupt Mask
+0x58	DMA Write Interrupt Clear
+0x5C	DMA Write Error Status
+0x60	DMA Write Done IMWr Address Low
+0x64	DMA Write Done IMWr Address High
+0x68	DMA Write Abort IMWr Address Low
+0x6C	DMA Write Abort IMWr Address High
+0x70	DMA Write Channels 1 and 0 IMWr Data
+0x84	DMA Write Channels 3 and 2 IMWr Data
+0x88	DMA Write Channels 5 and 4 IMWr Data
+0x8C	DMA Write Channels 7 and 6 IMWr Data
+0x90	DMA Write Linked List Error Enable
up to +0x9C	Not used (RsvdP).
+0xA0	DMA Read Interrupt Status
+0xA4	Not used (RsvdP).
+0xA8	DMA Read Interrupt Mask
+0xAC	DMA Read Interrupt Clear
+0xB0	Not used (RsvdP).
+0xB4	DMA Read Error Status Low
+0xB8	DMA Read Error Status High
up to +0xC0	Not used (RsvdP).

Table continues on the next page...

Table 11-90. DMA Interrupt Registers (continued)

+0xC4	DMA Read Linked List Error Enable
+0xC8	Not used (RsvdP).
+0xCC	DMA Read Done IMWr Address Low
+0xD0	DMA Read Done IMWr Address High
+0xD4	DMA Read Abort IMWr Address Low
+0xD8	DMA Read Abort IMWr Address High
+0xDC	DMA Read Channels 1 and 0 IMWr Data
+0xE0	DMA Read Channels 3 and 2 IMWr Data
+0xE4	DMA Read Channels 5 and 4 IMWr Data
+0xE8	DMA Read Channels 7 and 6 IMWr Data

DMA Channel Context Registers: these registers (which are stored in RAM) are for the control and status information that is specific to each channel. There is a set of registers for each channel.

Table 11-91. DMA Channel Context Registers for Each Channel

Byte Offset	Description
DMA Channel Context	
0x8_0000 + 0x200*ia + 0x100*(WR|RD)b	DMA Channel Control 1 Register
+0x004*i + 0x200*i + 0x100 *(WR|RD)	DMA Channel Control 2 Register
+0x008*i + 0x200*i + 0x100 *(WR|RD)	DMA Transfer Size
+0x00C*i +0x200*i + 0x100 *(WR|RD)	DMA SAR Low
+0x010*i + 0x200*i + 0x100 *(WR|RD)	DMA SAR High
+0x014*i + 0x200*i + 0x100 *(WR|RD)	DMA DAR Low
+0x018*i + 0x200*i + 0x100 *(WR|RD)	DMA DAR High
+0x01C*i + 0x200*i + 0x100 *(WR|RD)	DMA Linked List Pointer Low
+0x020*i + 0x200*i + 0x100 *(WR|RD)	DMA Linked List Pointer High

a. i is the number of channels

b. WR=0 | RD=1

11.3.3.3.7.1 Implementation of Channel Context Registers

- The channel context registers are stored in RAM whose contents are automatically initialized after power-on.
  - Therefore, you must program every “register” value as the default is undefined

  - You must program all fields marked Reserved to 1’b0
  - These registers are not affected by any of the reset signals
- You must not write to any of the context registers for a particular channel after you start the channel by writing the channel number to the Doorbell Number field of the DMA Write Door- bell Register (DMA_WRITE_DOORBELL_OFF) or DMA Read Doorbell Register (DMA_READ_DOORBELL_OFF).

Linked List Operation

When you enable linked list operation (Linked List Enable field of DMA Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_0) t o 1b1), the DMA overwrites the following DMA channel context registers with the following information from linked list data elements:

- The CB, LLP, LIE, and RIE fields of the DMA Channel Control 1 register.
- DMA Transfer Size
- DMA SAR Low
- DMA SAR High
- DMA DAR Low
- DMA DAR High

NOTE

- The structure of a link element is different to that of the data element. A data element has no TCB field. A link element has no LIE or RIE fields. It has no SAR, DAR, or Transfer Size DWORDs, but has LL Element Pointer DWORDs instead of the SAR DWORDs. For more details, see Relationship Between Element DWORDs and Channel Context Registers.
- The LIE and RIE bits in a LL element, only enable the Done interrupt. In non-LL mode, the RIE and LIE bits enable the Done and Abort interrupts. For more information, see Interrupts and Error Handling.

11.3.3.3.8 Programming Examples

This section provides a programming and operation flow for each of the following scenarios:

- 1 MB Write Transfer On Write Channel #0 Initiated by Local CPU (Non LL mode)
- 1 MB Read Transfer On Read Channel #0 Initiated by Local CPU (Non LL mode)
- 1 MB Write Transfer On Write Channel #5 Initiated by Remote CPU (Non LL mode)
- 1 MB Read Transfer On Read Channel #5 Initiated by Remote CPU (Non LL mode)
- Linked List Mode Programming Example (Write Channel #0)

NOTE

The order in which the registers in the following examples are programmed is not important, except for the Doorbell register, which initiates the DMA transfer. Therefore, to start both a "1 MB Write Transfer On Write Channel #0 Initiated by Local CPU (Non LL mode)" and a "1 MB Read Transfer On Read Channel #0 Initiated by Local CPU (Non LL mode)" by executing the commands on the table. If, as mentioned in "Starting The DMA Transfer" , you want both channels to start at (approximately) the same time, then do not execute the last command (at 0x280) in the table 'DMA Register Setup' until after you have executed all the commands in table.

NOTE

You must enable the DMA before you start a DMA channel. Therefore you must write ‘1’ to DMA Read Engine Enable or DMA Write Engine Enable register before you write ‘1’ to the DMA Read Doorbell or DMA Write Doorbell register.

11.3.3.3.8.1 1 MB Write Transfer On Write Channel #0 Initiated by Local CPU (Non LL mode)

In this example, the IMWr generation is disabled, as the local CPU initiates the DMA transfer. The local CPU is interrupted using the edma_int bus. The SAR is the address of the local memory, and the DAR is the address of the remote memory, as shown in the figure below. The table below provides the programming details for this example transfer.

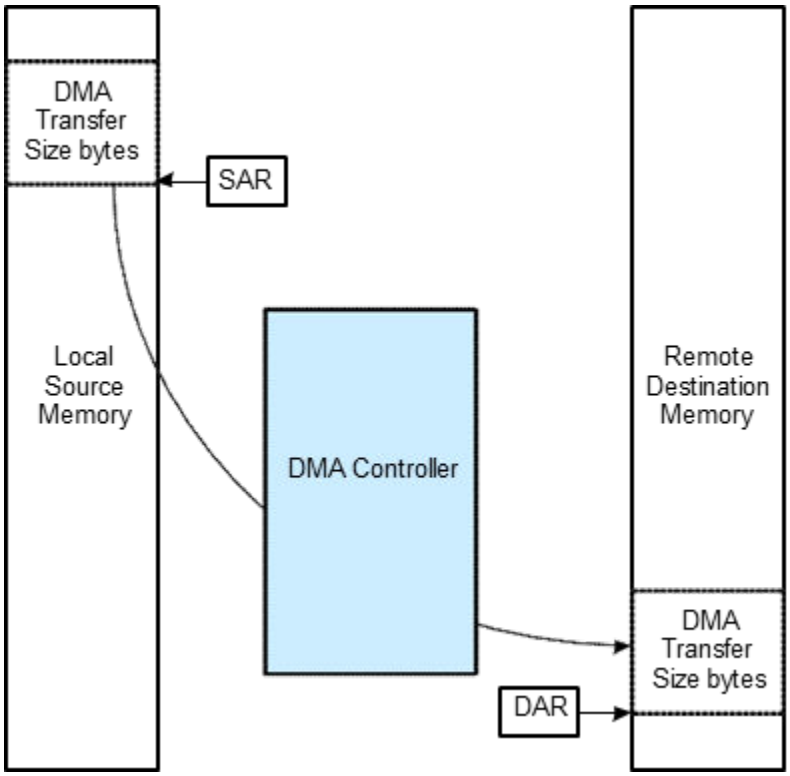


Figure 11-57. 1 MB Write DMA Transfer from 0xBEEF_BEE0 to 0xCAFE_CAF0

Table 11-92. 1 MB Write DMA Transfer from 0xBEEF_BEE0 to 0xCAFE_CAF0

Address (0x8_0000+)	Name	Value
0x00C	DMA Write Engine Enable You must not write 0 to this register. Even temporarily writing 0 to this register resets the DMA logic. For more details, see DMA Write Engine Enable Register (DMA_WRITE_ENGINE_EN_OFF).	0x1
0x054	DMA Write Interrupt Mask	0x0
0x200	DMA Channel Control 1 register • Local Interrupt Enable (LIE) =1 • Remote Interrupt Enable (RIE) =0 • AT, • RO, NS, TC, Function Number =0	0x04000008
0x208	DMA Transfer Size	0x00100000
0x20C	DMA SAR Low	0xBEEF_BEE0
0x210	DMA SAR High	0x0000_0000
0x214	DMA DAR Low	0xCAFE_CAF0
0x218	DMA DAR High	0x0000_0000
0x010	DMA Write Doorbell	0x0

11.3.3.3.8.2 1 MB Read Transfer On Read Channel #0 Initiated by Local CPU (Non LL mode)

In this example, the IMWr generation is disabled, as the local CPU initiates the DMA transfer. The local CPU is interrupted using the edma_int bus. The SAR is the address of the remote memory, and the DAR is the address of the local memory, as shown in the figure below. The table below provides the programming details for this example transfer.

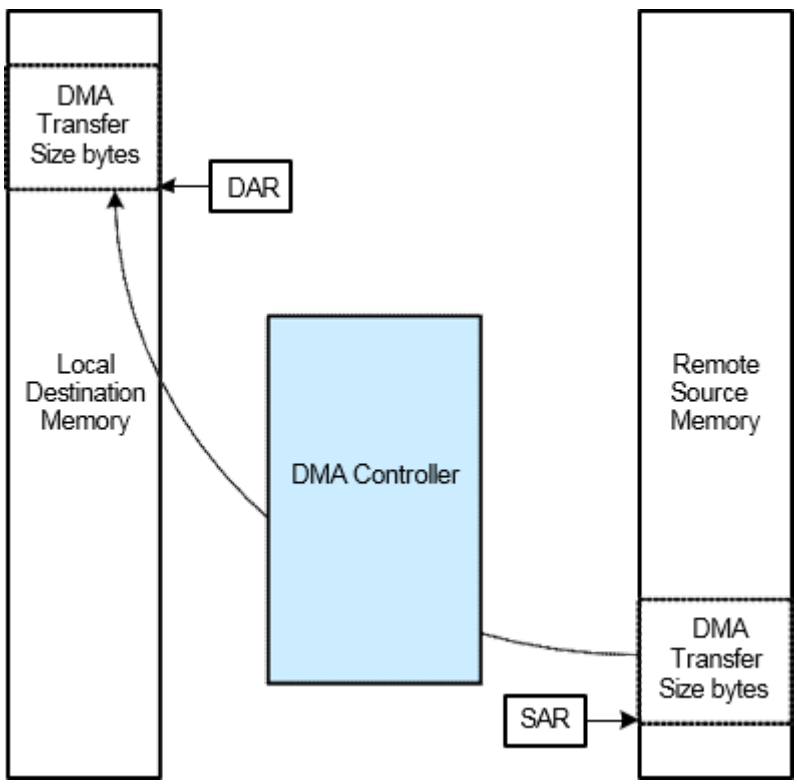


Figure 11-58. 1 MB Read DMA Transfer from 0xBEEF_BEE0 to 0xCAFE_CAF0

Table 11-93. DMA Register Setup

Address (0x8_0000+)	Name	Value
0x02C	DMA Read Engine Enable You must not write 0 to this register. Even temporarily writing 0 to this register resets the DMA logic. For more details, see DMA Read Engine Enable Register (DMA_READ_ENGINE_EN_OFF).	0x1
0x0A8	DMA Read Interrupt Mask	0x0
0x300	DMA Channel Control 1 register	0x04000008

Table continues on the next page...

Table 11-93. DMA Register Setup (continued)

	• Local Interrupt Enable (LIE) =1 • Remote Interrupt Enable (RIE) =0 • AT, • RO, NS, TC, Function Number =0	
0x308	DMA Transfer Size	0x00100000
0x30C	DMA SAR Low	0xBEEF_BEE0
0x310	DMA SAR High	0x0000_0000
0x314	DMA DAR Low	0xCAFE_CAF0
0x318	DMA DAR High	0x0000_0000
0x030	DMA Read Doorbell	0x0

11.3.3.3.8.3 1 MB Write Transfer On Write Channel #5 Initiated by Remote CPU (Non LL mode)

In this example, the local interrupt generation is disabled, as the remote CPU initiates the DMA transfer. The remote CPU is interrupted using an IMWr. The SAR is the address of the local memory, and the DAR is the address of the remote memory, as shown in the figure below. The table below provides the programming details for this example transfer.

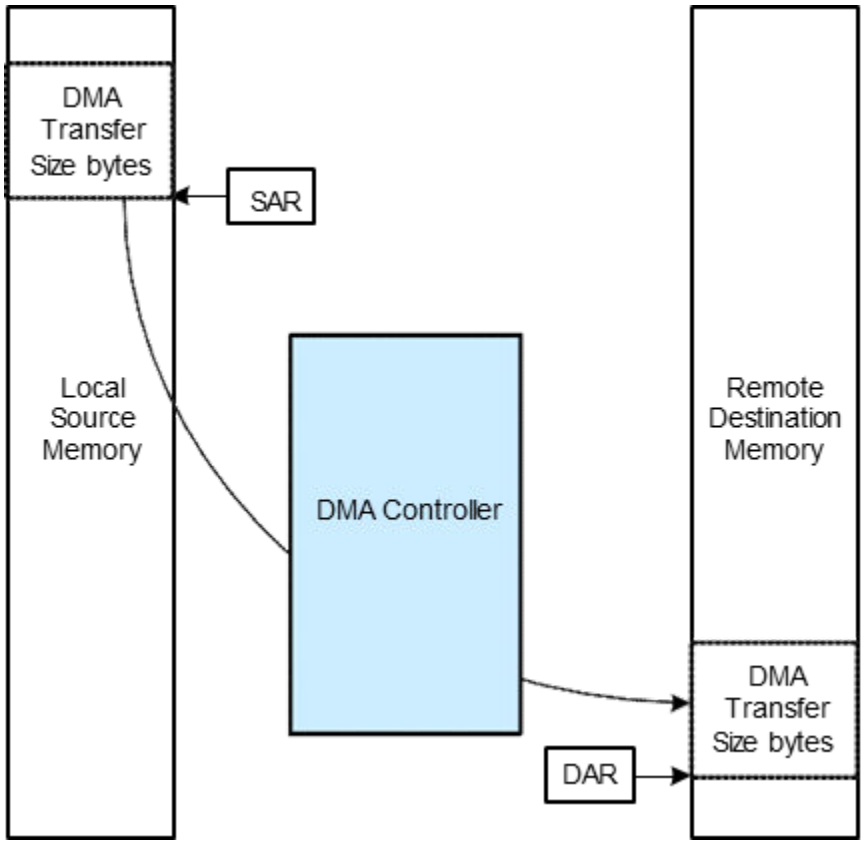


Figure 11-59. 1 MB Write DMA Transfer from 0xBEEF_BEE0 to 0xCAFE_CAF0

Table 11-94. DMA Register Setup

Address (0x8_0000+)	Name	Value
0x00C	DMA Write Engine Enable You must not write 0 to this register. Even temporarily writing 0 to this register resets the DMA logic. For more details, see DMA Write Engine Enable Register (DMA_WRITE_ENGINE_EN_OFF).	0x1
0x060 / 0x064	DMA Write Done IMWr Address Low and High	your IMWr Address #1
0x068 / 0x06C	DMA Write Abort IMWr Address Low and High	your IMWr Address #2
0x06C	DMA Write Channel 0 IMWr Data	your IMWr Data
0xA00	DMA Channel Control 1 register • Local Interrupt Enable (LIE) =0 • Remote Interrupt Enable (RIE) =1 • AT, • RO, NS, TC, Function Number =0	0x04000010
0xA08	DMA Transfer Size	0x00100000

Table continues on the next page...

Table 11-94. DMA Register Setup (continued)

0xA0C	DMA SAR Low	0xBEEF_BEE0
0xA10	DMA SAR High	0x0000_0000
0xA14	DMA DAR Low	0xCAFE_CAF0
0xA18	DMA DAR High	0x0000_0000

11.3.3.3.8.4 1 MB Read Transfer On Read Channel #5 Initiated by Remote CPU (Non LL mode)

In this example, the local interrupt generation is disabled, as the remote CPU initiates the DMA transfer. The remote CPU is interrupted using an IMWr. The SAR is the address of the remote memory, and the DAR is the address of the local memory, as shown in Figure below. The table below provides the programming details for this example transfer.

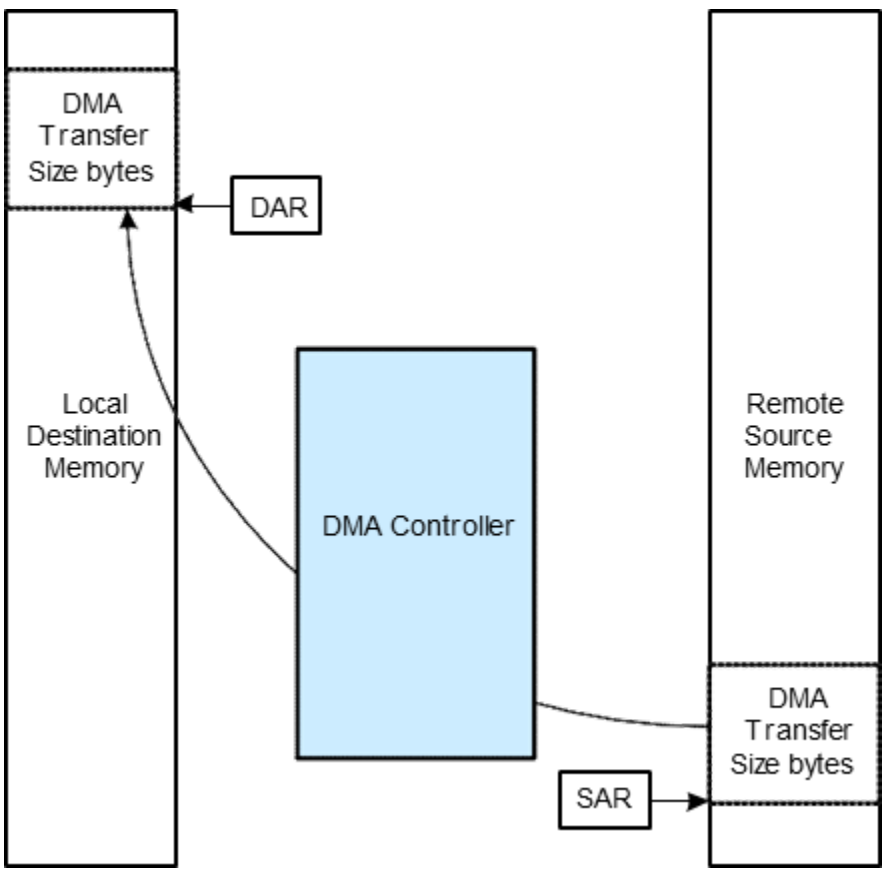


Figure 11-60. 1 MB Read DMA Transfer from 0xBEEF_BEE0 to 0xCAFE_CAF0

Table 11-95. DMA Register Setup

Address (0x700+)	Name	Value

Table continues on the next page...

Table 11-95. DMA Register Setup (continued)

0x29C	DMA Read Engine Enable You must not write “0” to this register. Even temporarily writing 0 to this register resets the DMA logic. For more details, see “DMA Read Engine Enable Register” (DMA_READ_ENGINE_EN).	0x1
0x33C / 0x340	DMA Read Done IMWr Address Low and high	your IMWr Address #1
0x344 / 0x348	DMA Read Abort IMWr Address Low and High	your IMWr Address #2
0x34C	DMA Read Channel 0 IMWr Data	your IMWr Data
0x36C	DMA Channel Context Index	0x8000_0005
0x370	DMA Channel Control 1 register • Local Interrupt Enable (LIE) =0 • Remote Interrupt Enable (RIE) =1 • AT, RO, NS, TC, Function Number =0 • TD =1	0x04000010
0x378	DMA Transfer Size	0x00100000
0x37C	DMA SAR Low	0xBEEF_BEE0
0x380	DMA SAR High	0x0000_0000
0x384	DMA DAR Low	0xCAFE_CAF0
0x388	DMA DAR High	0x0000_0000
0x2A0	DMA Read Doorbell	0x5

11.3.3.3.8.5 Linked List Mode Programming Example (Write Channel #0)

This write channel example corresponds to figure 'Example Producer (Software) - Consumer (DMA) Synchronization Flow', with the following features:

- Recycled linked list (LL) of nine data elements1and one link element.
- Transfer is initiated by local CPU.
- Each data element corresponds to a 1 MB (0x0010_0000) write transfer.
- The SARs for each element are separated by 2 MB (0x0020_0000). The DAR for each element are sepa- rated by 16 MB (0x0100_0000). The SAR is the address of the local memory, and the DAR is the address of the remote memory, as shown in the figure below.
- The LL transfer list is located at address 0x0000_0200 in local memory.
- No remote interrupt is generated so the IMWr generation is disabled, as the local CPU initiates the DMA transfer. The local CPU is interrupted using the edma_int bus.
- There is a Watermark interrupt at element #5, and an Empty interrupt at element #8. Both of these interrupts are done interrupts (as opposed to abort interrupts).

You should familiarize yourself with the Linked List Flow for Producer and Consumer as shown in Figure 'Linked List Flow for Producer and Consumer'. These are the main steps for this process in this example:

Step 1: Create TL in local memory:

- Create the 10 elements as per Table 'Linked List Element Initial Setup'.
  - For elements 0, 1, 2, 3, 4, 6, 7: RIE, LIE, LLP, CB =0,0,0,1
  - For element 5: RIE, LIE, LLP, CB =0,1,0,1 ❑For element 8: RIE, LIE, LLP, CB =0,1,0,1 ❑For element 9: LLP, TCB, CB =1,1,0
- For more details, see “PCS-CCS-CB-TCB Producer-Consumer Synchronization” and “Linked List Mode”

Step 2: Program and Start DMA:

- Configure and start the DMA registers as per Table 'DMA Register Setup'.

Step 4: Wait for done interrupt and recycle TL

- For more details, see “Using Interrupts for Linked List Producer-Consumer Synchronization”
- When recycling elements, consider “Element Recycling”

NOTE

If the DMA driver is running on the host and the interrupt service routine is reading local interrupts to determine if the transfer is successful, then you must set LIE and RIE in the same element and you should mask or ignore the local interrupt pin. Setting RIE and LIE in element A followed by RIE (only) in element B is not a verified usage scenario.

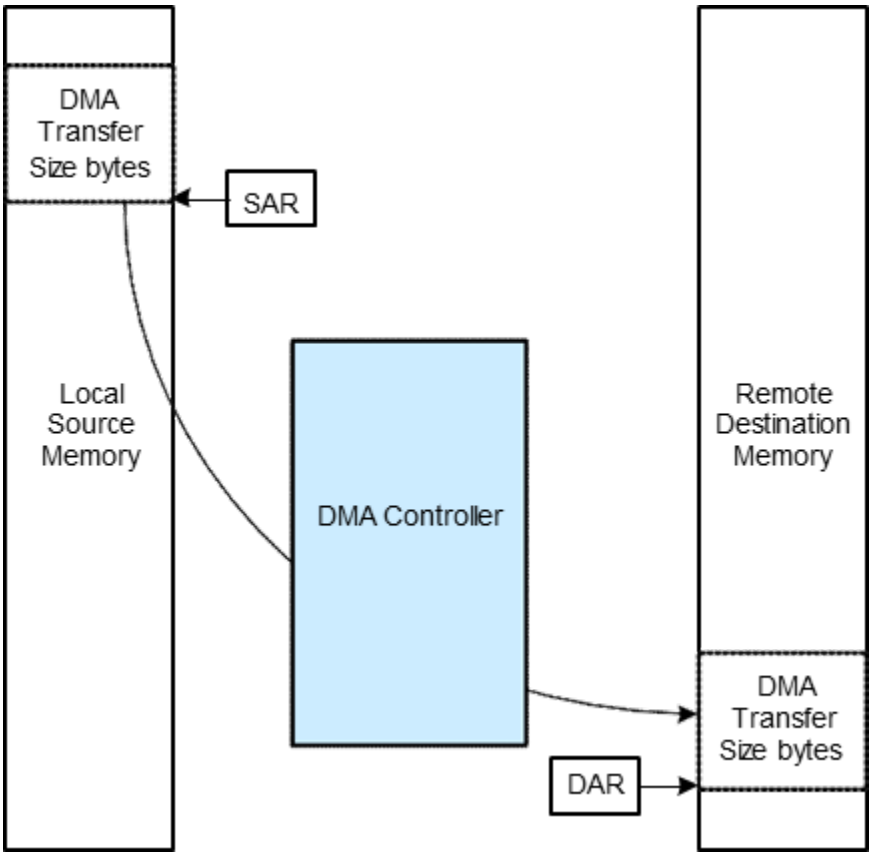


Figure 11-61. 1 MB Write DMA Transfer from SAR to DAR

Table 11-96. Linked List Element Initial Setup

Element #	Address	Data	Label
0	0x0000_0200	0x00000001	Channel Control
0	0x0000_0204	0x00100000	Transfer Size
0	0x0000_0208	0xBEEF_BEE0	SAR Low
0	0x0000_020C	0x0	SAR High
0	0x0000_0210	0xCAFE_CAF0	DAR Low
0	0x0000_0214	0x0	DAR High
1	0x0000_0218	0x00000001	Channel Control
1	0x0000_021C	0x00100000	Transfer Size
1	0x0000_0220	0xBF0F_BEE0	SAR Low
1	0x0000_0224	0x0	SAR High
1	0x0000_0228	0xCBFE_CAF0	DAR Low
1	0x0000_022C	0x0	DAR High
2	0x0000_0230	0x00000001	Channel Control
2	0x0000_0234	0x00100000	Transfer Size
2	0x0000_0238	0xBF2F_BEE0	SAR Low
2	0x0000_023C	0x0	SAR High
2	0x0000_0240	0xCCFE_CAF0	DAR Low

Table continues on the next page...

Table 11-96. Linked List Element Initial Setup (continued)

2	0x0000_0244	0x0	DAR High
3	0x0000_0248	0x00000001	Channel Control
3	0x0000_024C	0x00100000	Transfer Size
3	0x0000_0250	0xBF4F_BEE0	SAR Low
3	0x0000_0254	0x0	SAR High
3	0x0000_0258	0xCDFE_CAF0	DAR Low
3	0x0000_025C	0x0	DAR High
4	0x0000_0260	0x00000001	Channel Control
4	0x0000_0264	0x00100000	Transfer Size
4	0x0000_0268	0xBF6F_BEE0	SAR Low
4	0x0000_026C	0x0	SAR High
4	0x0000_0270	0xCEFE_CAF0	DAR Low
4	0x0000_0274	0x0	DAR High
5	0x0000_0278	0x00000009	Channel Control
5	0x0000_027C	0x00100000	Transfer Size
5	0x0000_0280	0xBF8F_BEE0	SAR Low
5	0x0000_0284	0x0	SAR High
5	0x0000_0288	0xCFFE_CAF0	DAR Low
5	0x0000_028C	0x0	DAR High
6	0x0000_0290	0x00000001	Channel Control
6	0x0000_0294	0x00100000	Transfer Size
6	0x0000_0298	0xBFAF_BEE0	SAR Low
6	0x0000_029C	0x0	SAR High
6	0x0000_0290	0xD0FE_CAF0	DAR Low
6	0x0000_0294	0x0	DAR High
7	0x0000_0298	0x00000001	Channel Control
7	0x0000_029C	0x00100000	Transfer Size
7	0x0000_02A0	0xBFCF_BEE0	SAR Low
7	0x0000_02A4	0x0	SAR High
7	0x0000_02A8	0xD1FE_CAF0	DAR Low
7	0x0000_02AC	0x0	DAR High
8	0x0000_02B0	0x00000009	Channel Control
8	0x0000_02B4	0x00100000	Transfer Size
8	0x0000_02B8	0xBFEF_BEE0	SAR Low
8	0x0000_02BC	0x0	SAR High
8	0x0000_02C0	0xD2FE_CAF0	DAR Low
8	0x0000_02C4	0x0	DAR High
9	0x0000_02C8	0x00000006	Channel Control
9	0x0000_02CC	0x0	Reserved
9	0x0000_02D0	0x0000_0200	Linked List Element Pointer Low

Table continues on the next page...

Table 11-96. Linked List Element Initial Setup (continued)

9	0x0000_02D4	0x0	Linked List Element Pointer High

Table 11-97. DMA Register Setup

Address (0x8_0000+)	Name	Value
0x030	DMA Write Engine Enable You must not write 0 to this register. Even temporarily writing 0 to this register resets the DMA logic. For more details, see DMA Write Engine Enable Register (DMA_WRITE_ENGINE_EN_OFF).	0x1
0x054	DMA Write Interrupt Mask	0x0
0x090	DMA Write Linked List Error Enable • LLLAIE =1	0x1
0x200	DMA Channel Control 1 register • Linked List Enable (LLE) =1 • Consumer Cycle Status (CCS) =PCS =1 • AT, • RO, NS, TC, Function Number =0	0x04000300
0x21C	DMA Linked List Pointer Low	0x0000_0200
0x220	DMA Linked List Pointer High	0x0000_0000
0x010	DMA Write Doorbell	0x0

11.3.3.4 Advanced DMA Information and Operation

This section discusses advanced features and operation of the DMA.

11.3.3.4.1 Overview

The DMA write and read channels operate independently to maximize the performance of the DMA read and write data transfers over the PCIe link.

The DMA can simultaneously perform the following types of memory transactions:

- DMA write: transfer (copy) of a block of data from local memory to remote memory
- DMA read: transfer (copy) of a block of data from remote memory to local memory

After you have programmed and started a DMA transfer (see "Using the DMA"), the DMA transfers the data as described in "**DMA Write Transfer"** and "**DMA Read Transfer"**

When the PCIe controller is configured to use the AXI bridge, then the DMA is located between the native PCIe controller and the AXI bridge module as shown in Figure 'Location of the DMA Controller when the PCIe includes the AXI Bridge'. It must share, with your normal non-DMA traffic, access to the AXI bridge, and PCIe controller and link. This sharing process is described in "**DMA Write Transfer"** and "**DMA Read Transfer"**. The internal TRGT1, RBYP, and XALI0/1 interfaces have been renamed to iTRGT1, iRBYP, and iXALI0/1.

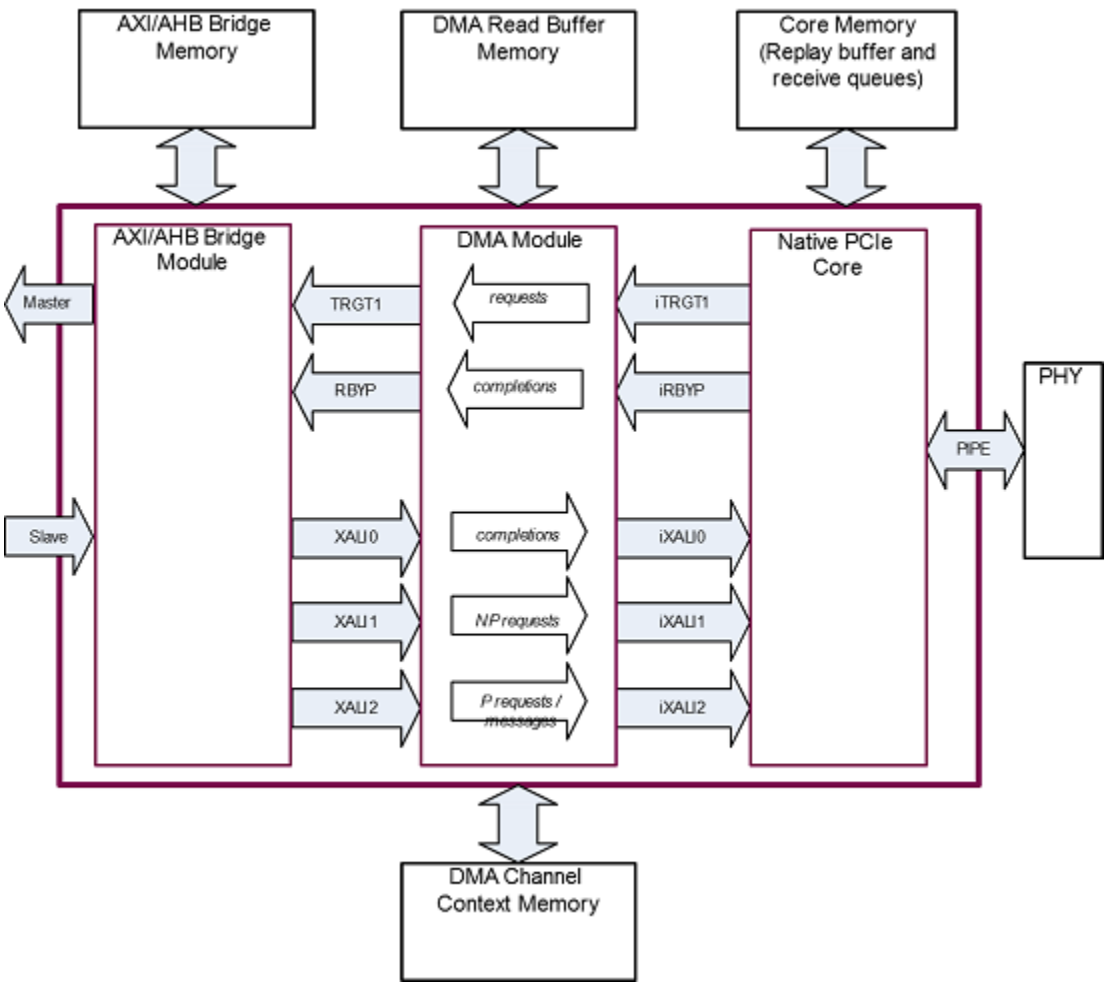


Figure 11-62. Location of the DMA Controller when the PCIe includes the AXI Bridge

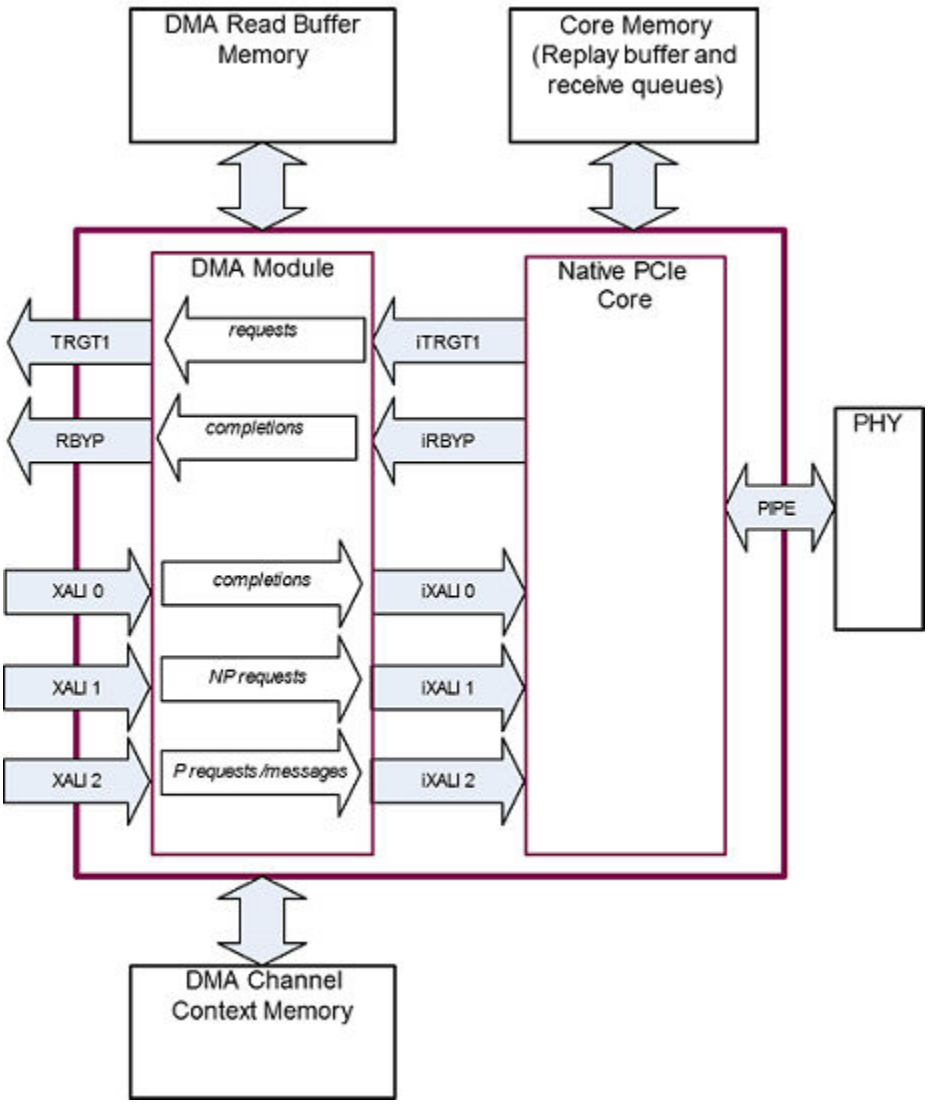


Figure 11-63. Location of the DMA Controller when the PCIe does not include the AXI Bridge

11.3.3.4.2 Internal Architecture

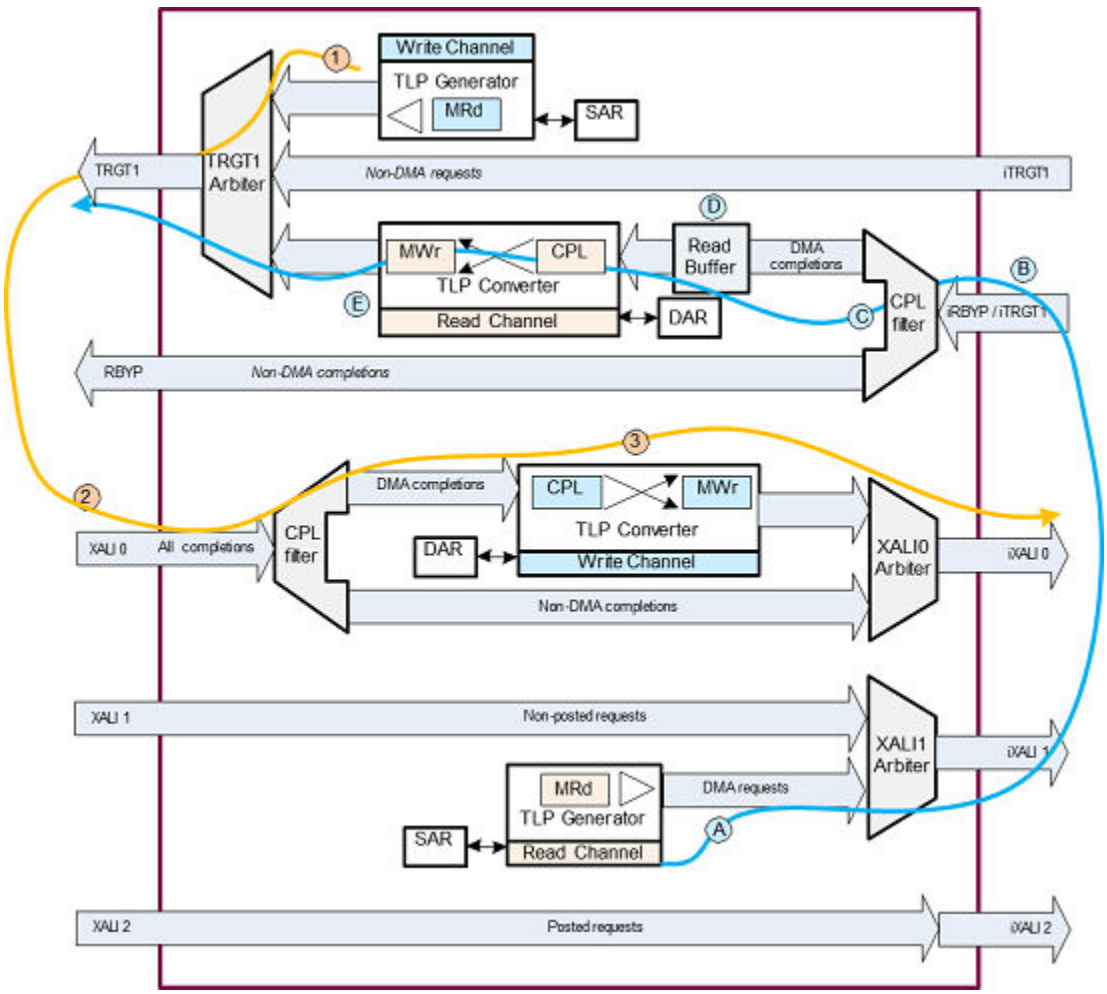


Figure 11-64. Operation of DMA Read (blue) and Write (orange) Channels

11.3.3.4.2.1 DMA Write Transfer

The DMA injects multiple MRd requests of size less than or equal to Max_Payload_Size into the inbound request path. The TLP addresses are derived from the "Source and Destination Address Registers (SAR, DAR)". When the DMA data transfer is complete, the CPU is notified. For more details, see "Interrupts and Error Handling". Referring to the orange paths in Figure 'Operation of DMA Read (blue) and Write (orange) Channels':

1. The DMA generates and sends READ requests. Because the TRGT_CPL_LUT is always enabled, it uses one of the shared DMA/non-DMA TRGT_CPL_LUT IDs instead of a PCIe TAG.

2. Data is returned on the XALI0 interface (or AXI bridge master) in response to these requests. Data must be returned in strict order within a single channel.

The DMA converts the CPL TLPs to WRITE requests and sends these to the PCIe native controller for transmission to the remote link partner.

11.3.3.4.2.2 DMA Read Transfer

The DMA injects multiple MRd requests of size less than or equal to the minimum of {Max Read Request Size, Max Payload Size} into the outbound request path. The TLP addresses are derived from the "Source and Destination Address Registers (SAR, DAR)". When the DMA data transfer is complete, the CPU is notified. For more details, see "Interrupts and Error Handling".

1. The DMA generates and sends READ requests to the remote link partner. It uses a DMA tag.

2. The link partner returns data in response to these requests on the iRBYP or iTRGT1 interface.

3. The DMA module filters this data from non-DMA traffic on iRBYP/iTRGT1 and sends it to the Read Buffer. The DMA module identifies DMA traffic by looking for PCIe completion TLPs that have a DMA tag.

4. The Read Buffer reorders completions that have been returned out-of-order. This is to ensure that the DMA writes data to the application in address-order.

5. The DMA converts the completion TLPs to WRITE requests and sends these to your application on TRGT1 (or AXI bridge master).

In Figure 'Operation of DMA Read (blue) and Write (orange) Channels', the XALI1 arbitration priority is Read Channel MSI (highest), Write Channel MSI, Non-DMA requests, Read Channel MRd. The TRGT1 arbitration priority is programmable (using the DMA_CTRL_DATA_ARB_PRIOR_OFF register) and defaults to Non-DMA requests (highest), Write Channel MRd (DMA data requests and LL element/descriptor access), Read Channel MRd (LL element/descriptor access), Read Channel MWr.

11.3.3.4.3 DMA Ordering

The DMA assumes that all DMA read channel data must be written to the local application memory in strict address order. For a DMA read transfer, each original outgoing request has a unique PCIe tag. The remote link partner or intermediate switch does not have to preserve the order of CPLs corresponding to such individual requests. When the DMA receives the CPLs out of order, it reorders that data in the "Read Buffer" as described in "**DMA Read Transfer"** . DMA write transfer data is sourced from the local application memory and sent to the remote memory in address order.

11.3.3.4.4 Error Handling

The DMA supports full AXI and PCIe error handling in conjunction with the PCIe controller. The tables below list the possible sources of an error for DMA read and write transfers.

Table 11-98. Possible Sources of an Error during a DMA Read Transfer

Source	Type	Description
Application Write Error Detected	Fatal	The DMA read channel has received an error response from the AXI bus at the bridge master interface (or TRGT1 interface when the AXI bridge is not used) while writing posted data to it. This error is fatal. • You must restart the transfer from the beginning because the channel context is corrupted. • The MWr is not rolled back. • The AXI bridge master does not report this error to software through error logging or MSG. **NOTE:** During linked list mode, the Application Write Error only corrupts the current element in the linked list. This is because the DMA does not fetch the next element from local memory until it has completely finished transferring the current block of data (as defined by the current linked list element). Finished implies that your application has successfully acknowledged (without ERROR) all of the MWr requests.
Unsupported Request (UR)	Non-fatal	The DMA read channel has received a PCIe UR CPL status from the remote device in response to the MRd request.
Completer Abort (CA)	Non-fatal	The DMA read channel has received a PCIe CA CPL status from the remote device in response to the MRd request.
CPL Time Out	Non-fatal	The DMA read channel has timed-out while waiting for the remote device to respond to the MRd request, or a malformed CplD has been received.
Data Poisoning	Fatal	The DMA read channel has detected data poisoning in the CPL from the remote device in response to the MRd request. The DMA read channel will drop the completion and then be halted.

Table continues on the next page...

Table 11-98. Possible Sources of an Error during a DMA Read Transfer (continued)

Linked List Element Fetch Error Detected	Fatal	The DMA read channel has received an error response from the AXI bus (or TRGT1 interface when the AXI bridge is not used) while reading a linked list element from local memory.

Table 11-99. Possible Sources of an Error during a DMA Write Transfer

Source	Type	Description
Application Read Error Detected	Fatal	The DMA write channel has received an error response from the AXI bus at the bridge master interface (or TRGT1 interface when the AXI bridge is not used) while reading data from it. • The response TLP is discarded by the DMA and is not converted to an outbound MWr TLP. • The UR or CA CPL status TLP generated by the bridge is discarded by the DMA. • The AXI bridge master does not report this error to software through error logging or MSG.
Linked List Element Fetch Error Detected	Fatal	The DMA write channel has received an error response from the AXI bus at the bridge master interface (or TRGT1 interface when the AXI bridge is not used) while reading a linked list element from local memory. • The UR or CA CPL status TLP generated by the bridge is discarded by the DMA. • The AXI bridge master1 does not report this (or for non-DMA traffic) error to software through error logging or MSG.

When the DMA detects an error, it performs the following actions (for the channel with the error):

- Stops issuing new requests to the remote link partner.
- Stops issuing new requests.
  - All data in the “Read Buffer”, that was received before the error was detected, is valid. In the case of a non-fatal error, this is reflected in the DMA Transfer Size Register DMA_TRANSFER_SIZE_OFF_WRCH_0.
  - All data in the read buffer (for that channel), that was received after the error was detected, is discarded.

- Places the DMA channel in a halted state (channel status field of the DMA Channel Control 1 Register is halted).
- Waits for all outstanding requests and CPLs to complete.
- Generates the abort interrupt or IMWr, if enabled, as outlined in “Interrupt Handling”.

For a DMA read transfer, you can read the DMA Read Error Status High Register (DMA_READ_ERR_ STATUS_HIGH_OFF) and DMA Read Error Status Low Register (DMA_READ_ERR_STATUS_LOW_OFF) to identify the source of the error. For a DMA write transfer (in linked list mode only), you can read the DMA Write Error Status Register (DMA_WRITE_ERR_STATUS_OFF) to identify the source of the error. After your software has finished error and interrupt routine handling for Non-Fatal errors, you can request the DMA to continue processing (for the channel with the error) by:

1. Reprogramming the SAR based on the number of bytes remaining in the DMA Transfer Size register
2. Writing the channel number to the Doorbell Number field of the DMA Write Doorbell Register (DMA_WRITE_DOORBELL_OFF) or DMA Read Doorbell Register (DMA_READ_DOORBELL_REG).

NOTE

You cannot continue after a Fatal Error is detected, as the channel context is corrupted. Any actions that were in progress are not rolled back. To exit from this condition, you need to Soft Reset the complete DMA controller, as described in DMA Write Engine Enable Register (DMA_WRITE_ENGINE_EN_OFF) and DMA Read Engine Enable Register (DMA_READ_ENGINE_EN_OFF).

When an error occurs in linked list mode, the next LL element is not fetched from local memory. DMA Linked List Pointer Low Register (DMA_LLP_LOW_OFF_WRCH_0) is not incremented by the DMA, but remains pointing to the element that caused the error.

11.3.3.4.4.1 Error Handling Assistance by Remote Software

During a DMA write transfer, the DMA is not aware of MWr errors that are detected in the remote device, because an MWr is a posted request and has no corresponding completion TLP. In these cases, your remote CPU must manually abort the DMA transfer by writing the channel number to the Doorbell Number field, and 1 to the Stop field of the DMA Write Doorbell Register (DMA_WRITE_DOORBELL_OFF). This causes the DMA to:

- Place the channel in a stopped state. The channel Status field of the DMA Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_0) is stopped and the DMA Transfer Size register is not 0x0.
- Wait for all outstanding pending transactions.
- Assert the abort interrupt if it is enabled

11.3.3.4.5 Linked List Operation

This section describes the detailed operation of the DMA in linked list (LL) mode. It also discusses how the software should produce LL elements1, and how the DMA and software maintain synchronization.

11.3.3.4.5.1 LL Operation Overview

You should first read "Linked List Mode" for an overview of LL mode. In this section, a normal LL operation called recycling2 is described in detail. The process is described for a write channel, but it is possible to have duplicate processes running in parallel for other write or read channels. In this process interrupts are used to trigger the producer (software) to recycle elements. Typically a Watermark interrupt (positioned near the middle of the transfer list (TL)) and an Empty interrupt (in the last data element) are used by setting LIE in these two elements to ‘1’.

The steps in this process are:

1. Software creates a LL element structure called a TL in local memory consisting of N-1 data elements and one link element.

   The link elements "LL Element Pointer Low/High DWORDs"(as shown in Linked List Element figure in Linked List Mode topic) are programmed to point back to the beginning of the first data element.

   One of the data elements (near the middle of the TL) is programmed to generate a "Watermark" interrupt.

   The last data element is programmed to generate an "Empty" interrupt.

2. Software programs the DMA with the location of the TL. This is done by writing to the DMA Linked List Pointer Low Register (DMA_LLP_LOW_OFF_WRCH_0) and DMA Linked List Pointer High Register (DMA_LLP_HIGH_OFF_WRCH_0).
3. Software starts the DMA process by ringing the write channel Doorbell. This is done by writing to the DMA Write Doorbell Register (DMA_WRITE_DOORBELL_OFF).

   The solid purple loop L1 in Figure 11-65 corresponds to steps 1-3.

4. The DMA reads (**consumes**) each data element from local memory, and if the CB and CCS bits match, loads the information (SAR, DAR, size, and so on) from that data element into the channel context registers. These context registers determine the operation of the channel that the DMA is currently servicing. The DMA executes the check for the CB/CCS match after it has evaluated TCB and toggled CCS. The DMA always loads link elements into the channel context.

NOTE

If the DMA stops on a link element, then your software must set the DMA Linked List Pointer registers, and restart the DMA process by ringing the doorbell.

With recycling, the same LL structure is used over and over again. Another mode of operation would be to jump to a new LL element structure when the current one is consumed.

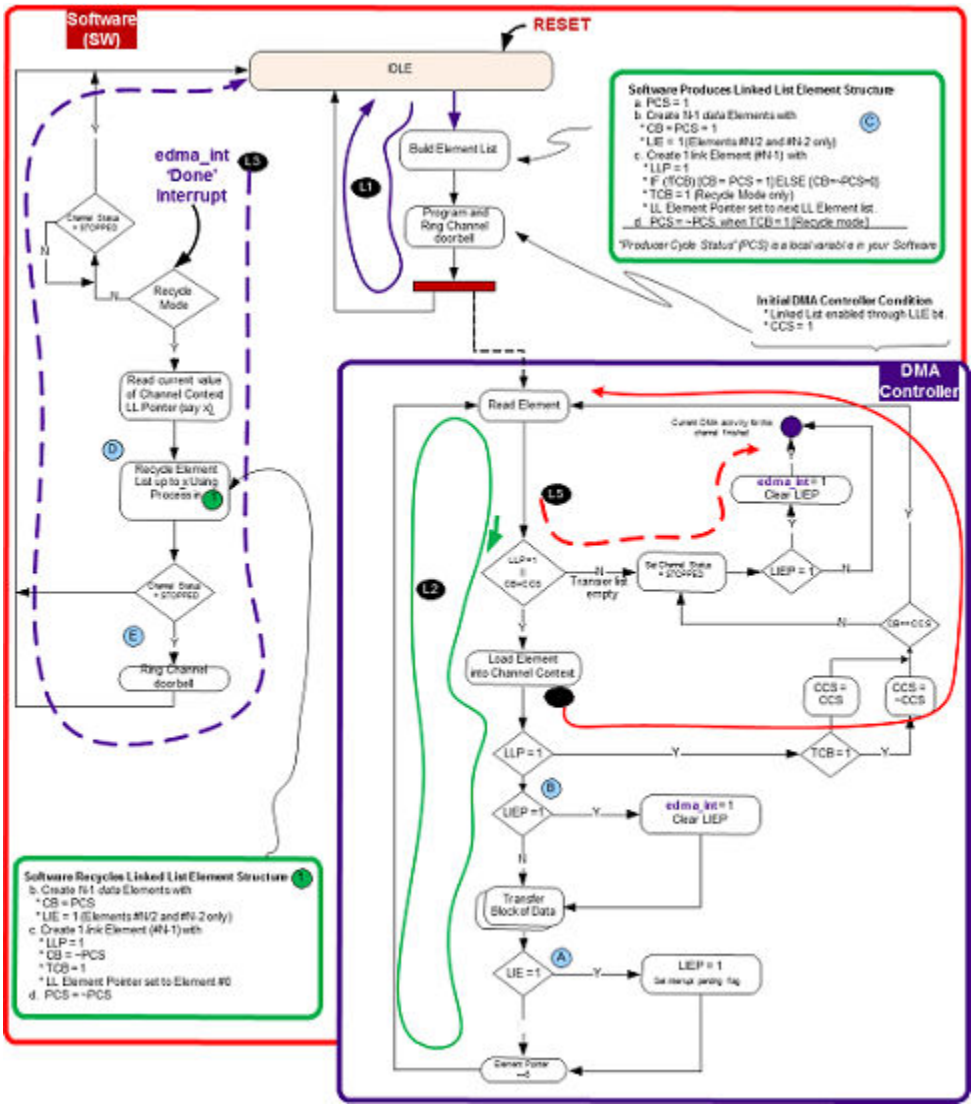


Figure 11-65. Linked List Flow for Producer and Consumer

5. The DMA then proceeds to transfer the block of data (as defined by the element), and when it is finished, reads the next element from local memory.

   The solid green loop L2 in Figure 11-65 corresponds to steps 4-5.

6. The last element in the list (called a link element, and indicated by having its LLP field set to 1) is not used to transfer a block of data.

   It causes the DMA to effectively repeat the task in step 4.

   Its TCB is typically set to 1. This causes the DMA to toggle its CCS bit, and the software to toggle its PCS bit. The software and DMA use the "PCS-CCS-CB-TCB" producer-consumer synchronization mechanism to ensure that:

   Software does not recycle elements that have not yet being consumed by the DMA

DMA correctly recognizes and consumes recycled elements. The solid red loop L4 in Figure 11-65 corresponds to step 6.

7. Upon reception of the interrupt (mentioned in step 1), the software starts to recycle the TL.

   Software reprograms each data element with new DMA transfer information.

   The software and DMA use the "PCS-CCS-CB-TCB" producer-consumer synchronization mecha- nism described later in this section.

   The dashed purple loop L3 in Figure 11-65 corresponds to step 7.

8. As some point, the software wants to terminate the complete DMA process, by not recycling any more elements. The DMA recognizes this condition when CB !=CCS, and sets the channel status to stopped

   The dashed red loop L5 in Figure 11-65 corresponds to step 8.

11.3.3.4.5.2 Using Interrupts for Linked List Producer-Consumer Synchronization

When the DMA is finished with an element, the DMA checks the LIE bit, before reading the next LL element. For more details, see circled step A in Figure 11-65. When the LIE is set, then the DMA does not assert the "done" interrupt immediately, but sets an internal interrupt pending flag (LIEP). It asserts the actual done interrupt after the next data element has been read from system memory. For more details, see circled step B in Figure 11-65. This automatic internal process of delaying the interrupt avoids a race condition between the DMA and software when the LIE bit is set in an element.

Your software should set the "Watermark" interrupt early to schedule the recycling of the consumed elements. Placing the Watermark interrupt too far down in the list, combined with a slow element recycling process in your application; ensures that the DMA returns to the start of the LL before your software has recycled it. The DMA channel STOPs, and you have to restart it by writing to its Doorbell. Typically a "Watermark interrupt" (positioned near the middle of the TL) and an Empty interrupt (in the last data element) are used by setting LIE in these two elements to 1b1. Upon reception of the interrupt, the software starts to recycle the TL.

Operational interrupts are always "done" interrupts. For more details, see "Interrupts and Error Handling"

NOTE

The DMA processes the remote interrupt enable (RIE) bit in the LL element in the same way as the LIE. Therefore, everywhere

that LIE and edma_int are mentioned in this section can be interpreted to mean LIE and edma_int, or RIE and IMWr.

11.3.3.4.5.3 PCS-CCS-CB-TCB Producer-Consumer Synchronization

The software and DMA use the "PCS-CCS-CB-TCB" producer-consumer synchronization mechanism to ensure that software does not recycle elements that have not yet being consumed by the DMA, and that the DMA correctly recognizes and consumes recycled elements. This process, which is shared between the DMA and the software, is illustrated through an example producer-consumer flow in the figure below.

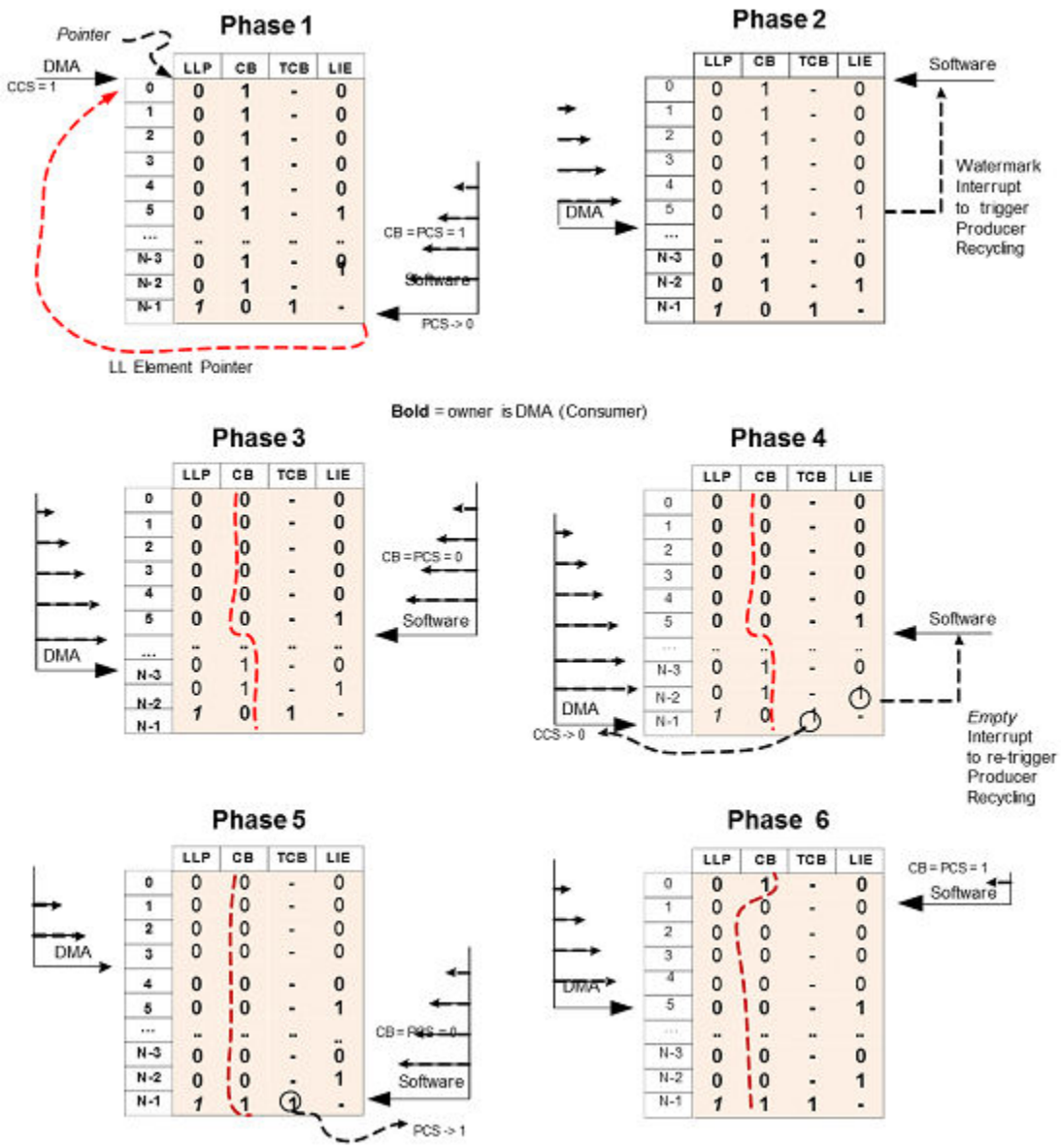


Figure 11-66. Example Producer (Software) - Consumer (DMA) Synchronization Flow

Before looking at the example in detail, it is useful to note that the DMA performs the following two tests as part of this process:

- Consumer-Owned Element, or Transfer List (TL) Empty Test

  The solid red loop L4 in the Figure 11-65 corresponds to this test. The Cycle Bit (CB) of DMA Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_0) (which was loaded from the LL element) is tested against the DMA Consumer Cycle State (CCS) bit of the same register. When CB =CCS, the element is owned by the consumer (DMA) and the data transfer is executed. When CB !=CCS, the TL is empty and the DMA sets channel Status (CS) in DMA Channel Control 1 Register (DMA_CH_CON-TROL1_OFF_WRCH_0) to "Stopped".

- Toggle Cycle Bit (TCB) Test

  The Dashed Red Loop Marked L5 in Figure 11-65 corresponds to this test. When the Toggle Cycle Bit (TCB) of the DMA Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_0) (which was loaded from the LL element) is set to 1, the DMA toggles the CCS bit of the DMA Channel Control 1 Register (DMA_CH_CONTROL1_OFF_WRCH_0) to be ~CB. The producer (software) also performs a similar action on its local variable "Producer Cycle Status" (PCS), as it sets PCS =~PCS. The TCB field is only valid in the last LL element (the link element). Toggling the PCS and CCS flags in this manner synchronizes the consumer to the producer.

The example in Figure 11-66 has the following six phases:

1. The software builds the transfer list for the write (or read) channel, according to the steps outlined in the circled step C in Figure 11-65. For each element, CB =PCS =CCS =1. The exception to this is the link element where CB is set to the next iteration value because the link element is always loaded into the channel context.
   - The link element points to the start of the TL and its TCB bit is 1. Software toggles its PCS variable.
   - Software writes the location of the TL into the channel context LL Pointer register.
   - Software rings the write channel Doorbell.
2. The DMA starts processing elements. After processing element 5 (and fetching the next element), the DMA asserts a Watermark “done” interrupt to signal the software to start recycling elements.
3. The software is recycling elements. It reads the channel context DMA Linked List Pointer Low Register (DMA_LLP_LOW_OFF_WRCH_0) and DMA Linked List Pointer High Register (DMA_LL- P_HIGH_OFF_WRCH_0), and recycles elements up to this location1in the TL. It sets CB for each element to be PCS. The Dashed

Purple Loop Marked L3 in Figure 11-65 corresponds to this activity. The DMA continues processing (consuming) elements as before.

4. The DMA then asserts an Empty “done” interrupt (from the LIE bit in the last data element at the end of TL that was set in the previous phase) to indicate to the software, to start recycling elements again. The DMA reaches the end of the TL, and as the TCB is set to 1, it toggles its CCS bit to 0 (~CB). For more details, see circled step D in Figure 11-65.
5. The software reads the channel context DMA Linked List Pointer Low Register (DMA_LLP_LOW_REG_WRCH_0) and DMA Linked List Pointer High Register (DMA_LLP_HIGH_OFF_WRCH_0), and recycles elements up to this location in the TL. It sets CB for each element to be PCS. As it recycles the link element, it detects that TCB is set to 1. It toggles its internal variable PCS to 0, and the initial TL has been fully recycled. The DMA starts processing the recycled elements.
6. The DMA continues processing (consuming) elements as before. Software did not have to ring the Doorbell, as the first element was already recycled before the DMA started processing it. The software might continue recycling from the top of the list, if the DMA had already returned to the top of the list when the software read the channel context DMA Linked List Pointer Low Register (DMA_LLP_LOW_OFF_WRCH_0) and DMA Linked List Pointer High Register (DMA_LL-P_HIGH_OFF_WRCH_0), in step 5. If not, it starts recycling again when the DMA processes element 4 and the DMA asserts a Watermark “done” interrupt.

11.3.3.4.5.4 Element Recycling

When your software is recycling elements, it should program the first DWORD (control bits) in the Figure 11-54 only after the other DWORDs have been programmed. In addition, it should set the CB bit (to ~PCS) within this DWORD only after the other bits in this DWORD have been set. This programming sequence avoids the DMA incorrectly fetching an element that software is currently recycling.

When the software is finished recycling it must check the channel context status (CS). When CS is "Stopped", the software must ring the channel Doorbell again. For more details, see circled step E in Figure 11-50. If your element recycling process was fast enough, then the DMA will seamlessly move from processing the last old element to the first new element, without the need for the software to ring the channel Doorbell again. To eliminate the possibility of a race condition between these two actions (read and write), you should first confirm the presence of the "done" interrupt bit in the DMA Write Interrupt Status Register (DMA_WRITE_INT_STATUS_OFF) or DMA Read Interrupt Status Register (DMA_READ_INT_ STATUS_OFF) and also check the DMA Linked List Pointer Low Register (DMA_LLP_LOW_REG_WRCH_0) to confirm that it is pointing to the correct location.

11.3.3.4.5.5 Link Down Recovery

During normal operation when the link goes down, the controller generates a "link reset" request. A

high-to-low transition on link_req_rst_not indicates that the PCIe controller is requesting external logic to reset the PCIe controller because the PHY link is down. For more details, see "Hot Reset". To determine the progress of the linked list operation when the link went down, you can read the contents of the channel context DMA Linked List Pointer Low Register (DMA_LLP_LOW_OFF_WRCH_0) and DMA Linked List Pointer High Register (DMA_LLP_HIGH_OFF_WRCH_0).

11.3.3.4.6 Multichannel Arbitration

A DMA write and read channel operate independently in full duplex mode. This maximizes the performance of the DMA read and write data transfers over the PCIe link. When DMA is configured with multiple read channels, it uses a weighted round robin (WRR) arbitration scheme to select the next read channel to be serviced. The same applies when you have multiple write channels. The DMA handles the multiple channels in a time-multiplexed way. From a software point of view every channel operates independently, so each one has its own instance of linked list and context. The bandwidth for all channels is shared in time and is controlled by a weighted round-robin arbiter.

The arbitration weight for each channel can be set using the DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF, DMA_WRITE_CHANNEL_ARB_WEIGHT_HIGH_OFF), DMA_READ_CHANNEL_ARB_WEIGHT_LOW_REG, and DMA_READ_CHANNEL_ARB_WEIGHT_HIGH_OFF registers. Each register has a 5-bit field (representing a weight from 1 to 32) for up to four channels. The weights must be set before "ringing" the doorbell.

The channel weight specifies the number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached, the WWR arbiter selects the next channel to be processed. For example, consider two DMA write channels (write channel #0 and write channel #1) where you want to assign the bandwidth for write channel #1 to be two times that of write channel #3. Therefore set:

- Write channel #0 weight to 32
- Write channel #1 weight to 16

Depending on the transfer size and the Maximum Payload Size, the DMA will issue 32 MRd requests for write channel #0, followed by 16 MRd requests for write channel #1, followed by 32 MRd requests for write channel #0 and so on, until the current transfer (or element when using LL mode) is finished. Therefore in this example, the weights reserve 33% bandwidth for write channel #1 and 67% bandwidth for write channel #0.

NOTE

- Max tag constraints the maximum number of outstanding DMA write engine MRd requests to the application, so the sum of weights should not exceed this value.
- If these limits are exceeded, then the respective DMA engine requester thread is back back pressured, defeating purpose of the round-robin bandwidth sharing mechanism.
- When the ratio (transfer-size / TLP-size) is smaller than the round-robin arbiter setting (say, DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF), the DMA issues less requests than the value in DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF. The TLP size is determined by the DMA using each function's Maximum Payload Size and Maximum Read Request Size rules. This means that the arbitration value in DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF determines the number of TLPs send and not the absolute data bandwidth.

11.3.3.4.6.1 Request, Response, and Descriptor-Fetch Threading In LL Mode For A Read Channel

For each read channel there are two threads (1) the request thread and (2) the response thread which are running in opposite directions. The DMA processes these threads independently. The request thread does not wait for the individual responses to arrive before issuing the next request. The request thread for the current channel i continues to issue requests until one of the following is true:

- The DMA exhausts its pool of available tags and the logic in the “Read Buffer” back-pres-sures the DMA read engine controller.
- The DMA completes its current DMA transfer block for the current channel. It has requested all of the data specified by the currently-loaded LL element.
- The current channel has issued Wi MRd requests, where Wi is the round robin weight associated with that channel.

A channel arbitration event now occurs and the DMA loads the context for the winning channel j. When the DMA is loading the new context, the response path for the previous channel i will be still utilizing the wire because the response thread lags the request thread. As soon as the DMA has loaded the context (and LL element if in LL mode) for the current channel j, it starts to issue requests. Therefore, in multichannel operation, when the block for channel i completes and a new LL element is been fetched, the DMA can be issuing requests belonging to other DMA channels. The currently inactive requester thread for channel i needs to synchronize with the response thread (that is, wait for all completions) before fetching its next LL element from local memory.

11.3.4 M-PCIe

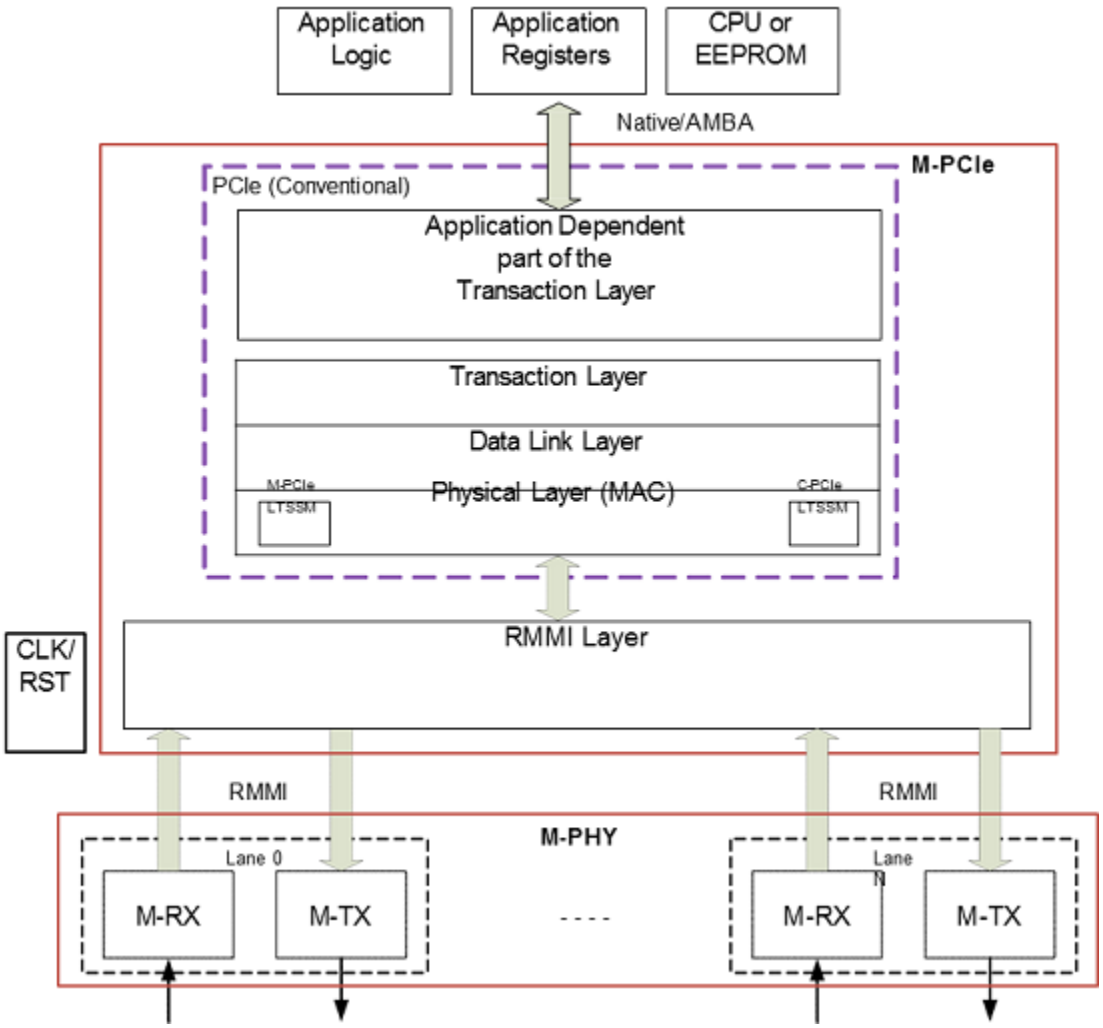
This section describes the optional M-PCIe feature.

11.3.4.1 Feature Description

The new M-PCIe specification enables the PCIe controller to operate over the low-power MIPI M-PHY physical layer technology, extending the benefits of PCIe to mobile devices including thin laptops, tablets and smartphones. The M-PCIe controller is compliant with the M-PCIe ECN1 specification. The interface between the controller and the PHY is compliant with the RMMI (Reference M-PHY MODULE Interface) specified in Annex A of the M-PHY Specification.

A complete M-PCIe port solution includes the controller, an analog PHY macro, and application logic to source and sink data. The physical layer is split across the RMMI such that the MAC functionality (LTSSM, lane-to-lane deskew) is in the controller and the PHY functionality is implemented in the RMMI-compliant PHY. The PHY module resides outside of the controller, interfacing through the standard RMMI. For more details, see Integrating with the PHY in the User Guide.

Figure 11-1 Complete M-PCIe Port Solution



a. HS=high-speed. LS=low-speed.

11.3.4.1.1 Features

This release supports the following features: **Table 11-1M- PCIe Features**

M-PCIe Feature	Supported
HSa-Gear Link Speeds	HS-Gear1 HS-Gear2 HS-Gear3
HS-Gear1 Base RATE-A Frequencies	31.2 MHz 62.4 MHz 124.8 MHz
Elastic Buffer Maximum Frequency Difference Compensation Between Link Partners • HS-Mode: • LS-Mode:	+/- 2000 ppm +/- 300 %

Table continues on the next page...

Dynamic Frequency Gear Speed Changing Mode	✓
Dynamic Width Gear Speed Changing Mode (RMMI Standard does not Support this)	✓
Controller and PHY can have Different Gear Speed Changing Modes	✓
x1, x2, x4, x8, x16 Link Width Support	✓
10-bit RMMI Width	✓
20-bit RMMI Width	✓
40-bit RMMI Width	✓
Controller and PHY can have Different RMMI Widths	✓
Runtime Selectable Mode of Operation Between M-PCIe and Conventional PCIe (Selectable PHY)	✓
Automatic RRAP Function (Off-loads Attribute Discovery/ Configuration From Your CPU In Downstream Port)	✓
LS-Gear Link Speeds Note: All bytes on the RMMI are valid in LS mode.	LS-Gear1
RATE-A / RATE-B Series Support	✓
Dynamic Asymmetric Link Width Adjustment	✓
Static (Compile Time) Asymmetric Link Width Adjustment	✗
Dynamic Link Reconfiguration Mechanism (Up Configure)	✓
Digital Remote Loopback	✓
Digital Local (RMMI) Loopback	✓

M-PCIe Feature	**Supported**
M-PHY Interoperability Support	✓
Third Party M-PHY Interoperability Support Through Custom PHY Flow	✓
Hardware Compliance Pattern Generator	✓
Internal Databus Width	32, 64, 128
Optional M-PCIe Features	
Manual Lane Flip	✓
Manual Polarity Change	✓
WAKE# Sideband Signal Support	✓
Optional System Features	
Advanced Power Management, Power Gating, and UPF Support	✓

Table continues on the next page...

Multifunction Support	✓
DMA Support	✓
ARI Support	✓
SR-IOV Support	✓
Latency Tolerance Reporting (LTR) Support	✓
OBFF Support	✓
Address Translation Services (ATS) Support	✓
Atomic Ops Support	✓
TLP Processing Hints (TPH) Support	✓
TLP Prefix Support (Including PASID)	✓
ID Based Ordering (IDO) Support	✓
Conventional PCIe Features Not Applicable in M-PCIe Mode	
CLKREQ# Sideband Signal and L1 Clock Power Management (CPM) Support	✗
Separate Refclk with Independent Spread Spectrum Clocking (SRIS)	✗
L1 Substates Support	✗
Dynamic Lane Reversal	✗
Dynamic Polarity Change	✗

a. HS=high-speed. LS=low-speed.

11.3.4.1.2 Limitations

Table 11-2 identifies the temporary limitations when using the M-PCIe module.

Table 11-2 M-PCIe Limitations

Limitation	Note
Limited Analog Loopback Support	Analog loopback requires the M-PHY to autonomously loop and re-transmit the Rx data. However, the RMMI specification does not define the entry mechanism for analog loopback. • If you are using the M-PHY (which has no input to accept the cores pa_rx_analog_loopback output), the analog loopback path must

Table continues on the next page...

	pass through the controller involving the elastic buffer. The Tx lanes in the controller use the Tx (and not the Rx) clock. The elastic buffer cannot operate indefinitely under this condition. If there is a frequency difference between the Rx and Tx clocks, there will be some data overflow or underflow. • If you are using a third party M-PHY, the M-PHY enters analog loop- back mode when the controller asserts pa_rx_analog_loopback. If the M-PHY does not support such an input, you can instruct the M- PCIe controller to do loopback from Rx to Tx through the elastic buffer.
Verilog Testbench (VTB) VIP not Included	M-PCIe configurations currently have no M-PCIe VIP and connect instead to an RTL DUT of the opposite port type.
No Midstream De-assertion of TX_PhyDIRDY	The controller does not support midstream de-assertion of phy_mac_tx_phydirdy during burst mode. This is not an M-PCIe or RMMI requirement. However, the tx_symbolclk is derived from the same clock that the M-PHY TX path uses, and there is no reason for the M-PHY to insert wait cycles.
Midstream De-assertion of RX_PhyDORDY	The rx_symbolclk is the recovered clock and there is no elastic buffer in the M-PHY. Therefore, there is no reason for the M-PHY to insert de-assertion of mac_phy_rx_phydordy during burst mode. If your M-PHY de-asserts mac_phy_rx_phydordy during burst mode, there is a risk of underflowing the elastic buffer in the controller.
TLP in 2K+ PPM System with Payload Greater than 2 KB not Supported.	The controller can handle a clock difference of 2K PPM only. When clock compensation greater than 2K PPM is required, the elastic buffer overflows or underflows on receiving a TLP with payload greater than 2 KB.
M-PHY RX Registers Cannot be set when LTSSM Exits L1 or L2.	The controller cannot set M-PHY RX registers when: • LTSSM exits L1 in a downstream or upstream port • LTSSM exits L2 in a downstream port.

11.3.4.1.3 Frequency, Speed, and Width Support

To set the M-PCIe frequency, datapath width, and speed modes of the controller, use the following configuration parameters in the coreConsultant GUI:

■ Controller Base Frequency (CM_FREQ)

■ PHY Base Frequency (CM_PHY_FREQ)

■ Controller Gear2 and Gear3 Modes (CM_GEAR2_MODE and CM_GEAR3_MODE)

■ PHY Gear2 and Gear3 Modes (CM_PHY_GEAR2_MODE and CM_PHY_GEAR3_MODE)

■ Maximum Tx and Rx Link Width (CM_TXNL_GUI and CM_RXNL_GUI)

❑CX_NL = max {CM_TXNL_GUI, CM_RXNL_GUI}

❑ CX_NB = (CM_FREQ == 124.8) ? 1 : (CM_FREQ == 62.4) ? 2 : 4

■CM_FREQ_STEP_DOWN_EN. This is the enable for the “Frequency Step Module”

11.3.4.1.4 Gear2/Gear3 Modes

For Dynamic Frequency (DF) configurations, the number of active symbols on the RMMI is constant; the frequency of the controller doubles each time as the controller transitions from Gear1 -> Gear2 -> Gear3 rates. For Dynamic Width (DW) configurations, the frequency of the controller is constant; the number of active symbols on the RMMI doubles each time as the controller transitions from Gear1 -> Gear2 -> Gear3 rates. The controller cannot have different Gear2 and Gear3 modes. For example, you cannot configure the controller for Dynamic Width in Gear2 mode, and at the same time configure it for Dynamic Frequency in Gear3 mode. For more details, see Figure 11-3.

Figure 11-3 Gear2/Gear3 Speed Changing Paths

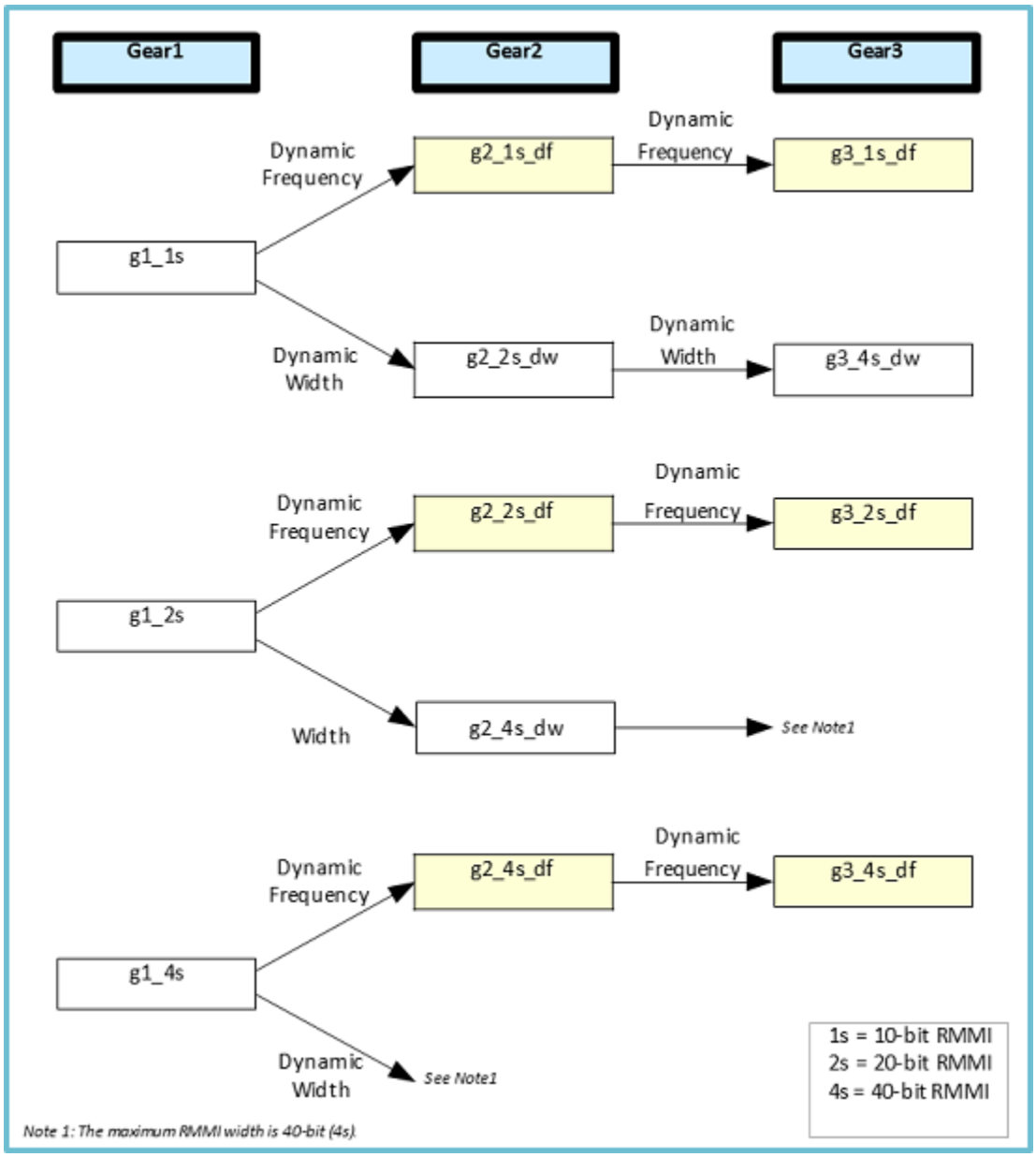


Table 11-3 Supported Controller Configurations in Standard M-PCIe Mode

				Maxa Link Width							
Gear1 RATE A Frequencyb (CM_FREQ)	**Gear2 Mode**	**Gear3 Mode**	**RMMI Widthc**	**For 32-bit datapath**	**For 64-bit datapathd**	**For 128-bit datapath**	**For 256-bit datapath**	**Maximum core_clk Frequency (x CM_FREQ)**	**Gear1 Speed Modee**	**Gear2 Speed Mode**	**Gear3 Speed Mode**
124.8	Disable	Disable	10-bit	x4	x8	x16	-	x1	g1_1s		
62.4			20-bit	x2	x4	x8	x16	x1	g1_2s		

Table continues on the next page...

31.2			40-bit	x1	x2	x4	x8	x1	g1_4s		
124.8	DW	Disable	20-bit	x2	x4	x8	x16	x1	g1_1s	g2_2s_dw	
62.4	DW		40-bit	x1	x2	x4	x8	x1	g1_2s	g2_4s_dw	
124.8	DF		10-bit	x4	x8	x16	-	x2	g1_1s	g2_1s_df	
62.4	DF		20-bit	x2	x4	x8	x16	x2	g1_2s	g2_2s_df	
31.2	DF		40-bit	x1	x2	x4	x8	x2	g1_4s	g2_4s_df	
124.8	DW	DW	40-bit	x1	x2	x4	x8	x1	g1_1s	g2_2s_dw	g3_4s_dw
124.8	DF	DF	10-bit	x4	x8	x16	-	x4	g1_1s	g2_1s_df	g3_1s_df
62.4	DF	DF	20-bit	x2	x4	x8	x16	x4	g1_2s	g2_2s_df	g3_2s_df
31.2	DF	DF	40-bit	x1	x2	x4	x8	x4	g1_4s	g2_4s_df	g3_4s_df

a. The value in each cell indicates the maximum link width supported. You can configure the controller with a link width up to this value, and coreConsul- tant automatically calculates the datapath width.

b. Corresponding RATE B frequencies are 145.76, 72.88, and 36.44 MHz.

c. The actual physical RMMI interface I/O width is 10*CM_NB bits per lane. The coreConsultant tool automatically calculates this width from CM_GEAR2_MODE, CM_GEAR3_MODE, and CM_FREQ. In a DW configuration, the number of active bits in the RMMI interface increases as the controller transitions to a higher Gear speed.

d. A 64-bit datapath requires a 32-bit/64-bit license. You cannot create a 32-bit or 64-bit configuration with a 128-bit/256-bit license

e. The ‘g’ value indicates the speed mode. The ‘s’ value indicates the number of 10-bit symbols processed per clock cycle per lane by the controller RMMI I/F, in the indicated speed mode. The ‘dw/df’ value indicates how the speed change to that mode is achieved

11.3.4.1.5 Controller PHY Compatibility

The following tables indicate which combinations of controller configurations from Table 11-3 and PHYs are supported. In Table 11-4, the triplet of values for each row indicates how the controller moves from Gear1 to Gear2 speed mode and from Gear2 to Gear3 speed mode. The triplet of values also identifies the particular controller configuration in Table 11-3. The controller always uses the same mechanism (DF or DW) to go from Gear1 to Gear2 and from Gear2 to Gear3.

Table 11-4 Gear3 Controller-PHY Combinations Supported

	PHY

Table continues on the next page...

		g1_1s g2_2s_dw g3_4s_dw	g1_1s g2_1s_df g3_1s_df	g1_2s g2_2s_df g3_2s_df	g1_4s g2_4s_df g3_4s_df
CONTROLLER	g1_1s g2_2s_dw g3_4s_dw	✓	✗	✗	✗
	g1_1s g2_1s_df g3_1s_df	✗	✓	✗	✗
	g1_2s g2_2s_df g3_2s_df	✗	✓	✓	✗
	g1_4s g2_4s_df g3_4s_df	✗	✓	✓	✓

Table 11-5 Gear2 Controller-PHY Combinations Supported

		PHY				
		g1_1s g2_2s_dw	**g1_2s g2_4s_dw**	**g1_1s g2_1s_df**	**g1_2s g2_2s_df**	**g1_4s g2_4s_df**
CONTROLL ER	g1_1s g2_2s_dw	✓	✗	✓	✗	✗
	g1_2s g2_4s_dw	✓	✓	✓	✓	✗
	g1_1s g2_1s_df	✗	✗	✓	✗	✗
	g1_2s g2_2s_df	✗	✗	✓	✓	✗
	g1_4s g2_4s_df	✗	✗	✓	✓	✓

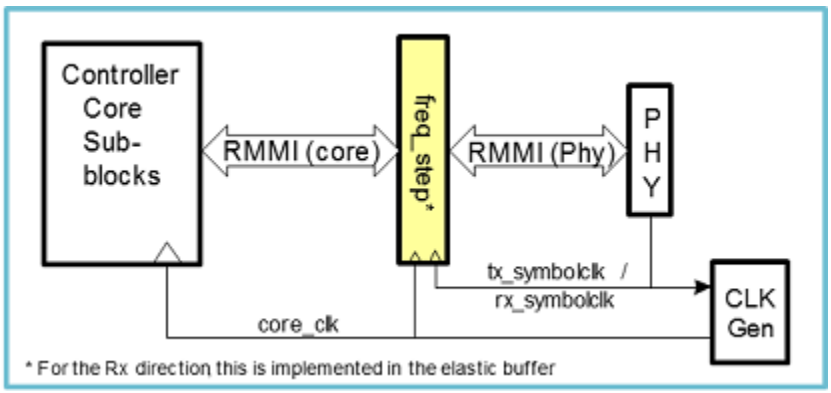
Table 11-6 Gear1 Controller-PHY Combinations Supported

		PHY		
		g1_1s	**g1_2s**	**g1_4s**
CONTROLLER	g1_1s	✓	✗	✗
	g1_2s	✓	✓	✗
	g1_4s	✓	✓	✓

11.3.4.1.6 Frequency Step Module

When the controller RMMI lane width (CM_NB) and M-PHY RMMI lane width (CM_PHY_NB) are different, or when the controller and the PHY have different Gear2 speed changing modes, the automatically-derived CM_FREQ_STEP_DOWN_EN parameter is defined, and a RMMI adapter module1 is added between the controller RMMI and the PHY RMMI.

Figure 11-4 Location of freq_step RMMI Adapter Module



This module steps up/down the signals to/from the RMMI interface and adapts (x2,x4, x0.5, x0.25) the controller RMMI width to the PHY RMMI width when the controller clock frequency is slower or faster than the RMMI clock frequency.

11.3.5 Memory Map and Register Definition

This section includes the PCIe module memory map and detailed descriptions of all registers.

NOTE

Synopsys Proprietary. Used with permission.

11.3.5.1 Register Descriptions

11.3.5.1.1 PCIe Memory Map

pcie base address: 3380_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Device ID and Vendor ID Register. (TYPE1_DEV_ID_VEND_ID_REG)	32	RW	ABCD_16C3h
4h	Status and Command Register. (TYPE1_STATUS_COMMAND_REG)	32	RW	0010_0000h
8h	Class Code and Revision ID Register. (TYPE1_CLASS_CODE_REV_ID_REG)	32	RW	0000_0001h
Ch	BIST, Header Type, Latency Timer, and Cache Line Size Register. (TYPE1_BIST_HDR_TYPE_LAT_CACHE_LINE_SIZE_REG)	32	RW	0001_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
18h	Secondary Latency Timer, Subordinate Bus Number, Secondary Bus Number, and Primary Bus Number Register. (SEC_LAT_TIMER_SUB_BUS_SEC_BUS_PRI_BUS_REG)	32	RW	0000_0000h
1Ch	Secondary Status, and I/O Limit and Base Register. (SEC_STAT_IO_LIMIT_IO_BASE_REG)	32	RW	0000_0000h
20h	Memory Limit and Base Register. (MEM_LIMIT_MEM_BASE_REG)	32	RW	0000_0000h
24h	Prefetchable Memory Limit and Base Register. (PREF_MEM_LIMIT_PREF_MEM_BASE_REG)	32	RW	0000_0000h
28h	Prefetchable Base Upper 32 Bits Register. (PREF_BASE_UPPER_REG)	32	RO	0000_0000h
2Ch	Prefetchable Limit Upper 32 Bits Register. (PREF_LIMIT_UPPER_REG)	32	RO	0000_0000h
30h	I/O Limit and Base Upper 16 Bits Register. (IO_LIMIT_UPPER_IO_BASE_UPPER_REG)	32	RO	0000_0000h
34h	Capabilities Pointer Register. (TYPE1_CAP_PTR_REG)	32	RW	0000_0040h
38h	Expansion ROM Base Address Register. (TYPE1_EXP_ROM_BASE_REG)	32	RW	0000_0000h
3Ch	Bridge Control, Interrupt Pin, and Interrupt Line Register. (BRIDGE_CTRL_INT_PIN_INT_LINE_REG)	32	RW	0000_01FFh
40h	Power Management Capabilities Register. (CAP_ID_NXT_PTR_REG)	32	RW	DBC3_5001h
44h	Power Management Control and Status Register. (CON_STATUS_REG)	32	RW	0000_0000h
50h	MSI Capability ID, Next Pointer, Capability/Control Registers. (PCI_MSI_CAP_ID_NEXT_CTRL_REG)	32	RW	0180_7005h
54h	MSI Message Lower Address Register. (MSI_CAP_OFF_04H_REG)	32	RW	0000_0000h
58h	For a 32 bit MSI Message, this register contains Data. (MSI_CAP_OFF_08H_REG)	32	RW	0000_0000h
5Ch	For a 64 bit MSI Message, this register contains Data. (MSI_CAP_OFF_0CH_REG)	32	RW	0000_0000h
60h	Used for MSI when Vector Masking Capable. (MSI_CAP_OFF_10H_REG)	32	RW	0000_0000h
64h	Used for MSI 64 bit messaging when Vector Masking Capable. (MSI_CAP_OFF_14H_REG)	32	RO	0000_0000h
70h	PCI Express Capabilities, ID, Next Pointer Register. (PCIE_CAP_ID_PCIE_NEXT_CAP_PTR_PCIE_CAP_REG)	32	RW	0042_0010h
74h	Device Capabilities Register. (DEVICE_CAPABILITIES_REG)	32	RW	0000_8000h
78h	Device Control and Status Register. (DEVICE_CONTROL_DEVICE_STATUS)	32	RW	0000_2010h
7Ch	Link Capabilities Register. (LINK_CAPABILITIES_REG)	32	RW	0073_CC12h
80h	Link Control and Status Register. (LINK_CONTROL_LINK_STATUS_REG)	32	RW	1011_0000h
84h	Slot Capabilities Register. (SLOT_CAPABILITIES_REG)	32	RW	0000_0000h
88h	Slot Control and Status Register. (SLOT_CONTROL_SLOT_STATUS)	32	RW	0040_03C0h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
8Ch	Root Control and Capabilities Register. (ROOT_CONTROL_ROOT_CAPABILITIES_REG)	32	RW	0001_0000h
90h	Root Status Register. (ROOT_STATUS_REG)	32	W1C	0000_0000h
94h	Device Capabilities 2 Register. (DEVICE_CAPABILITIES2_REG)	32	RO	0000_041Fh
98h	Device Control 2 and Status 2 Register. (DEVICE_CONTROL2_DEVICE_STATUS2_REG)	32	RW	0000_0000h
9Ch	Link Capabilities 2 Register. (LINK_CAPABILITIES2_REG)	32	RO	0000_0006h
A0h	Link Control 2 and Status 2 Register. (LINK_CONTROL2_LINK_STATUS2_REG)	32	RW	0001_0002h
100h	Advanced Error Reporting Extended Capability Header. (AER_EXT_CAP_HDR_OFF)	32	RW	1482_0001h
104h	Uncorrectable Error Status Register. (UNCORR_ERR_STATUS_OFF)	32	W1C	0000_0000h
108h	Uncorrectable Error Mask Register. (UNCORR_ERR_MASK_OFF)	32	RW	0040_0000h
10Ch	Uncorrectable Error Severity Register. (UNCORR_ERR_SEV_OFF)	32	RW	0046_2030h
110h	Correctable Error Status Register. (CORR_ERR_STATUS_OFF)	32	W1C	0000_0000h
114h	Correctable Error Mask Register. (CORR_ERR_MASK_OFF)	32	RW	0000_E000h
118h	Advanced Error Capabilities and Control Register. (ADV_ERR_CAP_CTRL_OFF)	32	RW	0000_00A0h
11Ch	Header Log Register 0. (HDR_LOG_0_OFF)	32	RO	0000_0000h
120h	Header Log Register 1. (HDR_LOG_1_OFF)	32	RO	0000_0000h
124h	Header Log Register 2. (HDR_LOG_2_OFF)	32	RO	0000_0000h
128h	Header Log Register 3. (HDR_LOG_3_OFF)	32	RO	0000_0000h
12Ch	Root Error Command Register. (ROOT_ERR_CMD_OFF)	32	RW	0000_0000h
130h	Root Error Status Register. (ROOT_ERR_STATUS_OFF)	32	RW	0000_0000h
134h	Error Source Identification Register. (ERR_SRC_ID_OFF)	32	RO	0000_0000h
138h	TLP Prefix Log Register 1. (TLP_PREFIX_LOG_1_OFF)	32	RO	0000_0000h
13Ch	TLP Prefix Log Register 2. (TLP_PREFIX_LOG_2_OFF)	32	RO	0000_0000h
140h	TLP Prefix Log Register 3. (TLP_PREFIX_LOG_3_OFF)	32	RO	0000_0000h
144h	TLP Prefix Log Register 4. (TLP_PREFIX_LOG_4_OFF)	32	RO	0000_0000h
148h	L1 Substates Extended Capability Header. (L1SUB_CAP_HEADER_REG)	32	RW	0001_001Eh
14Ch	L1 Substates Capability Register. (L1SUB_CAPABILITY_REG)	32	RW	0028_0A1Bh
150h	L1 Substates Control 1 Register. (L1SUB_CONTROL1_REG)	32	RW	0000_0A00h
154h	L1 Substates Control 2 Register. (L1SUB_CONTROL2_REG)	32	RW	0000_0028h
700h	Ack Latency Timer and Replay Timer Register. (ACK_LATENCY_TIMER_OFF)	32	RW	1846_0817h
704h	Vendor Specific DLLP Register. (VENDOR_SPEC_DLLP_OFF)	32	RW	FFFF_FFFFh
708h	Port Force Link Register. (PORT_FORCE_OFF)	32	RW	0000_0004h
70Ch	Ack Frequency and L0-L1 ASPM Control Register. (ACK_F_ASPM_CTRL_OFF)	32	RW	1B2C_2C00h
710h	Port Link Control Register. (PORT_LINK_CTRL_OFF)	32	RW	0001_0120h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
714h	Lane Skew Register. (LANE_SKEW_OFF)	32	RW	0000_0000h
718h	Timer Control and Max Function Number Register. (TIMER_CTRL_MAX_FUNC_NUM_OFF)	32	RW	0000_C000h
71Ch	Symbol Timer Register and Filter Mask 1 Register. (SYMBOL_TIMER_FILTER_1_OFF)	32	RW	0000_0280h
720h	Filter Mask 2 Register. (FILTER_MASK_2_OFF)	32	RW	0000_0000h
724h	AMBA Multiple Outbound Decomposed NP SubRequests Control Register. (AMBA_MUL_OB_DECOMP_NP_SUB_REQ_CTRL_OFF)	32	RW	0000_0001h
728h	Debug Register 0 (PL_DEBUG0_OFF)	32	RO	See description.
72Ch	Debug Register 1 (PL_DEBUG1_OFF)	32	RO	See description.
730h	Transmit Posted FC Credit Status (TX_P_FC_CREDIT_STATUS_OFF)	32	RO	0000_0000h
734h	Transmit Non-Posted FC Credit Status (TX_NP_FC_CREDIT_STATUS_OFF)	32	RO	0000_0000h
738h	Transmit Completion FC Credit Status (TX_CPL_FC_CREDIT_STATUS_OFF)	32	RO	0000_0000h
73Ch	Queue Status (QUEUE_STATUS_OFF)	32	RW	0000_0000h
740h	VC Transmit Arbitration Register 1 (VC_TX_ARBI_1_OFF)	32	RO	0000_000Fh
744h	VC Transmit Arbitration Register 2 (VC_TX_ARBI_2_OFF)	32	RO	0000_0000h
748h	Segmented-Buffer VC0 Posted Receive Queue Control. (VC0_P_RX_Q_CTRL_OFF)	32	RW	4520_C019h
74Ch	Segmented-Buffer VC0 Non-Posted Receive Queue Control. (VC0_NP_RX_Q_CTRL_OFF)	32	RW	0520_C003h
750h	Segmented-Buffer VC0 Completion Receive Queue Control. (VC0_CPL_RX_Q_CTRL_OFF)	32	RW	0580_0000h
80Ch	Link Width and Speed Change Control Register. (GEN2_CTRL_OFF)	32	RW	0002_012Ch
810h	PHY Status Register. (PHY_STATUS_OFF)	32	RO	See description.
814h	PHY Control Register. (PHY_CONTROL_OFF)	32	RW	0000_0000h
81Ch	Programmable Target Map Control Register. (TRGT_MAP_CTRL_OFF)	32	RW	0000_007Fh
820h	Integrated MSI Reception Module (iMRM) Address Register. (MSI_CTRL_ADDR_OFF)	32	RW	0000_0000h
824h	Integrated MSI Reception Module Upper Address Register. (MSI_CTRL_UPPER_ADDR_OFF)	32	RW	0000_0000h
828h	Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_0_EN_OFF)	32	RW	0000_0000h
82Ch	Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_0_MASK_OFF)	32	RW	0000_0000h
830h	Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_0_STATUS_OFF)	32	W1C	0000_0000h
834h	Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_1_EN_OFF)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
838h	Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_1_MASK_OFF)	32	RW	0000_0000h
83Ch	Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_1_STATUS_OFF)	32	W1C	0000_0000h
840h	Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_2_EN_OFF)	32	RW	0000_0000h
844h	Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_2_MASK_OFF)	32	RW	0000_0000h
848h	Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_2_STATUS_OFF)	32	W1C	0000_0000h
84Ch	Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_3_EN_OFF)	32	RW	0000_0000h
850h	Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_3_MASK_OFF)	32	RW	0000_0000h
854h	Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_3_STATUS_OFF)	32	W1C	0000_0000h
858h	Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_4_EN_OFF)	32	RW	0000_0000h
85Ch	Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_4_MASK_OFF)	32	RW	0000_0000h
860h	Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_4_STATUS_OFF)	32	W1C	0000_0000h
864h	Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_5_EN_OFF)	32	RW	0000_0000h
868h	Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_5_MASK_OFF)	32	RW	0000_0000h
86Ch	Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_5_STATUS_OFF)	32	W1C	0000_0000h
870h	Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_6_EN_OFF)	32	RW	0000_0000h
874h	Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_6_MASK_OFF)	32	RW	0000_0000h
878h	Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_6_STATUS_OFF)	32	W1C	0000_0000h
87Ch	Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_7_EN_OFF)	32	RW	0000_0000h
880h	Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_7_MASK_OFF)	32	RW	0000_0000h
884h	Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_7_STATUS_OFF)	32	W1C	0000_0000h
888h	Integrated MSI Reception Module General Purpose IO Register. (MSI_GPIO_IO_OFF)	32	RW	0000_0000h
88Ch	RADM clock gating enable control register. (CLOCK_GATING_CTRL_OFF)	32	RW	0000_0001h
8B4h	Order Rule Control Register. (ORDER_RULE_CTRL_OFF)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
8B8h	PIPE Loopback Control Register. (PIPE_LOOPBACK_CONTROL_OFF)	32	RW	0000_0001h
8BCh	DBI Read-Only Write Enable Register. (MISC_CONTROL_1_OFF)	32	RW	0000_0001h
8C0h	UpConfigure Multi-lane Control Register. (MULTI_LANE_CONTROL_OFF)	32	RW	0000_0000h
8C4h	PHY Interoperability Control Register. (PHY_INTEROP_CTRL_OFF)	32	RW	0000_0237h
8C8h	TRGT_CPL_LUT Delete Entry Control register. (TRGT_CPL_LUT_DELETE_ENTRY_OFF)	32	RW	0000_0000h
8CCh	Link Reset Request Flush Control Register. (LINK_FLUSH_CONTROL_OFF)	32	RW	FF00_0001h
8D0h	AXI Bridge Slave Error Response Register. (AMBA_ERROR_RESPONSE_DEFAULT_OFF)	32	RW	0000_9C00h
8D4h	Link Down AXI Bridge Slave Timeout Register. (AMBA_LINK_TIMEOUT_OFF)	32	RW	0000_0032h
8D8h	AMBA Ordering Control. (AMBA_ORDERING_CTRL_OFF)	32	RW	0000_0000h
8E0h	ACE Cache Coherency Control Register 1 (COHERENCY_CONTROL_1_OFF)	32	RW	0000_0000h
8E4h	ACE Cache Coherency Control Register 2 (COHERENCY_CONTROL_2_OFF)	32	RW	0000_0000h
8E8h	ACE Cache Coherency Control Register 3 (COHERENCY_CONTROL_3_OFF)	32	RW	0000_0000h
8F0h	Lower 20 bits of the programmable AXI address where Messages coming from wire are mapped to. (AXI_MSTR_MSG_ADDR_LOW_OFF)	32	RW	0000_0000h
8F4h	Upper 32 bits of the programmable AXI address where Messages coming from wire are mapped to. (AXI_MSTR_MSG_ADDR_HIGH_OFF)	32	RW	0000_0000h
8F8h	PCIe Controller IIP Release Version Number. (PCIE_VERSION_NUMBER_OFF)	32	RO	3530_302Ah
8FCh	PCIe Controller IIP Release Version Type. (PCIE_VERSION_TYPE_OFF)	32	RO	6761_2A2Ah
B40h	Auxiliary Clock Frequency Control Register. (AUX_CLK_FREQ_OFF)	32	RW	0000_000Ah
B44h	L1 Substates Timing Register. (L1_SUBSTATES_OFF)	32	RW	0000_00D2h
8000_0000h	iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_OUTBOUND_0)	32	RW	0000_0000h
8000_0004h	iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_OUTBOUND_0)	32	RW	0000_0000h
8000_0008h	iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_OUTBOUND_0)	32	RW	0000_0000h
8000_000Ch	iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_0)	32	RW	0000_0000h
8000_0010h	iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_OUTBOUND_0)	32	RW	0000_FFFFh
8000_0014h	iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_0)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
8000_0018h	iATU Upper Target Address Register. (IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_0)	32	RW	0000_0000h
8000_0100h	iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_INBOUND_0)	32	RW	0000_0000h
8000_0104h	iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_INBOUND_0)	32	RW	0000_0000h
8000_0108h	iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_INBOUND_0)	32	RW	0000_0000h
8000_010Ch	iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_INBOUND_0)	32	RW	0000_0000h
8000_0110h	iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_INBOUND_0)	32	RW	0000_FFFFh
8000_0114h	iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_INBOUND_0)	32	RW	0000_0000h
8000_0200h	iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_OUTBOUND_1)	32	RW	0000_0000h
8000_0204h	iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_OUTBOUND_1)	32	RW	0000_0000h
8000_0208h	iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_OUTBOUND_1)	32	RW	0000_0000h
8000_020Ch	iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_1)	32	RW	0000_0000h
8000_0210h	iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_OUTBOUND_1)	32	RW	0000_FFFFh
8000_0214h	iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_1)	32	RW	0000_0000h
8000_0218h	iATU Upper Target Address Register. (IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_1)	32	RW	0000_0000h
8000_0300h	iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_INBOUND_1)	32	RW	0000_0000h
8000_0304h	iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_INBOUND_1)	32	RW	0000_0000h
8000_0308h	iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_INBOUND_1)	32	RW	0000_0000h
8000_030Ch	iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_INBOUND_1)	32	RW	0000_0000h
8000_0310h	iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_INBOUND_1)	32	RW	0000_FFFFh
8000_0314h	iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_INBOUND_1)	32	RW	0000_0000h
8000_0400h	iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_OUTBOUND_2)	32	RW	0000_0000h
8000_0404h	iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_OUTBOUND_2)	32	RW	0000_0000h
8000_0408h	iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_OUTBOUND_2)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
8000_040Ch	iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_2)	32	RW	0000_0000h
8000_0410h	iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_OUTBOUND_2)	32	RW	0000_FFFFh
8000_0414h	iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_2)	32	RW	0000_0000h
8000_0418h	iATU Upper Target Address Register. (IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_2)	32	RW	0000_0000h
8000_0500h	iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_INBOUND_2)	32	RW	0000_0000h
8000_0504h	iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_INBOUND_2)	32	RW	0000_0000h
8000_0508h	iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_INBOUND_2)	32	RW	0000_0000h
8000_050Ch	iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_INBOUND_2)	32	RW	0000_0000h
8000_0510h	iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_INBOUND_2)	32	RW	0000_FFFFh
8000_0514h	iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_INBOUND_2)	32	RW	0000_0000h
8000_0600h	iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_OUTBOUND_3)	32	RW	0000_0000h
8000_0604h	iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_OUTBOUND_3)	32	RW	0000_0000h
8000_0608h	iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_OUTBOUND_3)	32	RW	0000_0000h
8000_060Ch	iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_3)	32	RW	0000_0000h
8000_0610h	iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_OUTBOUND_3)	32	RW	0000_FFFFh
8000_0614h	iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_3)	32	RW	0000_0000h
8000_0618h	iATU Upper Target Address Register. (IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_3)	32	RW	0000_0000h
8000_0700h	iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_INBOUND_3)	32	RW	0000_0000h
8000_0704h	iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_INBOUND_3)	32	RW	0000_0000h
8000_0708h	iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_INBOUND_3)	32	RW	0000_0000h
8000_070Ch	iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_INBOUND_3)	32	RW	0000_0000h
8000_0710h	iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_INBOUND_3)	32	RW	0000_FFFFh
8000_0714h	iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_INBOUND_3)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
8008_0000h	DMA Arbitration Scheme for TRGT1 Interface. (DMA_CTRL_DATA_ARB_PRIOR_OFF)	32	RW	0000_0688h
8008_0008h	DMA Number of Channels Register. (DMA_CTRL_OFF)	32	RW	0001_0001h
8008_000Ch	DMA Write Engine Enable Register. (DMA_WRITE_ENGINE_EN_OFF)	32	RW	0000_0000h
8008_0010h	DMA Write Doorbell Register. (DMA_WRITE_DOORBELL_OFF)	32	RW	0000_0000h
8008_0018h	DMA Write Engine Channel Arbitration Weight Low Register. (DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF)	32	RW	0000_8421h
8008_001Ch	DMA Write Engine Channel Arbitration Weight High Register. (DMA_WRITE_CHANNEL_ARB_WEIGHT_HIGH_OFF)	32	RW	0000_8421h
8008_002Ch	DMA Read Engine Enable Register. (DMA_READ_ENGINE_EN_OFF)	32	RW	0000_0000h
8008_0030h	DMA Read Doorbell Register. (DMA_READ_DOORBELL_OFF)	32	RW	0000_0000h
8008_0038h	DMA Read Engine Channel Arbitration Weight Low Register. (DMA_READ_CHANNEL_ARB_WEIGHT_LOW_OFF)	32	RW	0000_8421h
8008_003Ch	DMA Read Engine Channel Arbitration Weight High Register. (DMA_READ_CHANNEL_ARB_WEIGHT_HIGH_OFF)	32	RW	0000_8421h
8008_004Ch	DMA Write Interrupt Status Register. (DMA_WRITE_INT_STATUS_OFF)	32	RW	0000_0000h
8008_0054h	DMA Write Interrupt Mask Register. (DMA_WRITE_INT_MASK_OFF)	32	RW	0001_0001h
8008_0058h	DMA Write Interrupt Clear Register. (DMA_WRITE_INT_CLEAR_OFF)	32	W1C	0000_0000h
8008_005Ch	DMA Write Error Status Register (DMA_WRITE_ERR_STATUS_OFF)	32	RO	0000_0000h
8008_0060h	DMA Write Done IMWr Address Low Register. (DMA_WRITE_DONE_IMWR_LOW_OFF)	32	RW	0000_0000h
8008_0064h	DMA Write Done IMWr Interrupt Address High Register. (DMA_WRITE_DONE_IMWR_HIGH_OFF)	32	RW	0000_0000h
8008_0068h	DMA Write Abort IMWr Address Low Register. (DMA_WRITE_ABORT_IMWR_LOW_OFF)	32	RW	0000_0000h
8008_006Ch	DMA Write Abort IMWr Address High Register. (DMA_WRITE_ABORT_IMWR_HIGH_OFF)	32	RW	0000_0000h
8008_0070h	DMA Write Channel 1 and 0 IMWr Data Register. (DMA_WRITE_CH01_IMWR_DATA_OFF)	32	RW	0000_0000h
8008_0074h	DMA Write Channel 3 and 2 IMWr Data Register. (DMA_WRITE_CH23_IMWR_DATA_OFF)	32	RW	0000_0000h
8008_0078h	DMA Write Channel 5 and 4 IMWr Data Register. (DMA_WRITE_CH45_IMWR_DATA_OFF)	32	RW	0000_0000h
8008_007Ch	DMA Write Channel 7 and 6 IMWr Data Register. (DMA_WRITE_CH67_IMWR_DATA_OFF)	32	RW	0000_0000h
8008_0090h	DMA Write Linked List Error Enable Register. (DMA_WRITE_LINKED_LIST_ERR_EN_OFF)	32	RW	0000_0000h
8008_00A0h	DMA Read Interrupt Status Register. (DMA_READ_INT_STATUS_OFF)	32	RW	0000_0000h
8008_00A8h	DMA Read Interrupt Mask Register. (DMA_READ_INT_MASK_OFF)	32	RW	0001_0001h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
8008_00ACh	DMA Read Interrupt Clear Register. (DMA_READ_INT_CLEAR_OFF)	32	WO	0000_0000h
8008_00B4h	DMA Read Error Status Low Register. (DMA_READ_ERR_STATUS_LOW_OFF)	32	RO	0000_0000h
8008_00B8h	DMA Read Error Status High Register. (DMA_READ_ERR_STATUS_HIGH_OFF)	32	RO	0000_0000h
8008_00C4h	DMA Read Linked List Error Enable Register. (DMA_READ_LINKED_LIST_ERR_EN_OFF)	32	RW	0000_0000h
8008_00CCh	DMA Read Done IMWr Address Low Register. (DMA_READ_DONE_IMWR_LOW_OFF)	32	RW	0000_0000h
8008_00D0h	DMA Read Done IMWr Address High Register. (DMA_READ_DONE_IMWR_HIGH_OFF)	32	RW	0000_0000h
8008_00D4h	DMA Read Abort IMWr Address Low Register. (DMA_READ_ABORT_IMWR_LOW_OFF)	32	RW	0000_0000h
8008_00D8h	DMA Read Abort IMWr Address High Register. (DMA_READ_ABORT_IMWR_HIGH_OFF)	32	RW	0000_0000h
8008_00DCh	DMA Read Channel 1 and 0 IMWr Data Register. (DMA_READ_CH01_IMWR_DATA_OFF)	32	RW	0000_0000h
8008_00E0h	DMA Read Channel 3 and 2 IMWr Data Register. (DMA_READ_CH23_IMWR_DATA_OFF)	32	RW	0000_0000h
8008_00E4h	DMA Read Channel 5 and 4 IMWr Data Register. (DMA_READ_CH45_IMWR_DATA_OFF)	32	RW	0000_0000h
8008_00E8h	DMA Read Channel 7 and 6 IMWr Data Register. (DMA_READ_CH67_IMWR_DATA_OFF)	32	RW	0000_0000h
8008_0200h	DMA Write Channel Control 1 Register. (DMA_CH_CONTROL1_OFF_WRCH_0)	32	RW	0000_0000h
8008_0208h	DMA Write Transfer Size Register. (DMA_TRANSFER_SIZE_OFF_WRCH_0)	32	RW	0000_0000h
8008_020Ch	DMA Write SAR Low Register. (DMA_SAR_LOW_OFF_WRCH_0)	32	RW	0000_0000h
8008_0210h	DMA Write SAR High Register. (DMA_SAR_HIGH_OFF_WRCH_0)	32	RW	0000_0000h
8008_0214h	DMA Write DAR Low Register. (DMA_DAR_LOW_OFF_WRCH_0)	32	RW	0000_0000h
8008_0218h	DMA Write DAR High Register. (DMA_DAR_HIGH_OFF_WRCH_0)	32	RW	0000_0000h
8008_021Ch	DMA Write Linked List Pointer Low Register. (DMA_LLP_LOW_OFF_WRCH_0)	32	RW	0000_0000h
8008_0220h	DMA Write Linked List Pointer High Register. (DMA_LLP_HIGH_OFF_WRCH_0)	32	RW	0000_0000h
8008_0300h	DMA Read Channel Control 1 Register. (DMA_CH_CONTROL1_OFF_RDCH_0)	32	RW	0000_0000h
8008_0308h	DMA Read Transfer Size Register. (DMA_TRANSFER_SIZE_OFF_RDCH_0)	32	RW	0000_0000h
8008_030Ch	DMA Read SAR Low Register. (DMA_SAR_LOW_OFF_RDCH_0)	32	RW	0000_0000h
8008_0310h	DMA Read SAR High Register. (DMA_SAR_HIGH_OFF_RDCH_0)	32	RW	0000_0000h
8008_0314h	DMA Read DAR Low Register. (DMA_DAR_LOW_OFF_RDCH_0)	32	RW	0000_0000h
8008_0318h	DMA Read DAR High Register. (DMA_DAR_HIGH_OFF_RDCH_0)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
8008_031Ch	DMA Read Linked List Pointer Low Register. (DMA_LLP_LOW_OFF_RDCH_0)	32	RW	0000_0000h
8008_0320h	DMA Read Linked List Pointer High Register. (DMA_LLP_HIGH_OFF_RDCH_0)	32	RW	0000_0000h

11.3.5.1.2 Device ID and Vendor ID Register. (TYPE1_DEV_ID_VEND_ID_REG)

11.3.5.1.2.1 Offset

Register	Offset
TYPE1_DEV_ID_VEND_ID_REG	0h

11.3.5.1.2.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DEVICE_ID															
Reset	1	0	1	0	1	0	1	1	1	1	0	0	1	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VENDOR_ID															
Reset	0	0	0	1	0	1	1	0	1	1	0	0	0	0	1	1

11.3.5.1.2.3 Fields

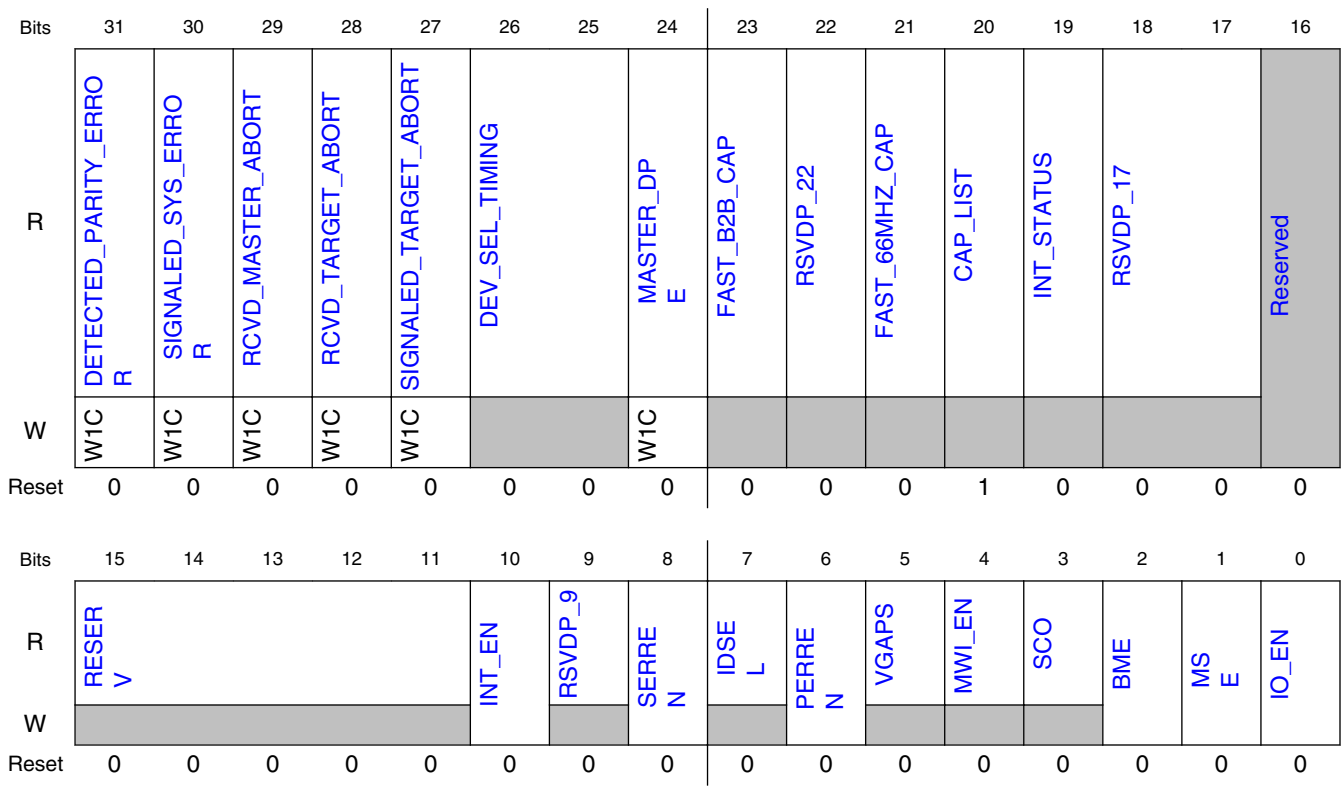
Field	Function
31-16 DEVICE_ID	Device ID. The Device ID register identifies the particular Function. This identifier is allocated by the vendor. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
15-0 VENDOR_ID	Vendor ID. The Vendor ID register identifies the manufacturer of the Function. Valid vendor identifiers are allocated by the PCI-SIG to ensure uniqueness. It is not permitted to populate this register with a value of FFFFh, which is an invalid value for Vendor ID. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

11.3.5.1.3 Status and Command Register. (TYPE1_STATUS_COMMAND_REG)

11.3.5.1.3.1 Offset

Register	Offset
TYPE1_STATUS_COMMAND_REG	4h

11.3.5.1.3.2 Diagram



11.3.5.1.3.3 Fields

Field	Function
31 DETECTED_PARITY_ERROR	Detected Parity Error. This bit is set by a Function whenever it receives a Poisoned TLP, regardless of the state the Parity Error Response bit in the Command register. The bit is set when the Poisoned TLP is received by a Function's primary side.
30 SIGNALED_SYS_ERROR	Signaled System Error. This bit is set when a Function sends an ERR_FATAL or ERR_NONFATAL Message, and the SERR# Enable bit in the Command register is 1b.

Table continues on the next page...

Field	Function
29 RCVD_MASTER_ABORT	Received Master Abort. This bit is set when a Requester receives a Completion with Unsupported Request Completion status. The bit is set when the Unsupported Request is received by a Function's primary side.
28 RCVD_TARGET_ABORT	Received Target Abort. This bit is set when a Requester receives a Completion with Completer Abort Completion status. The bit is set when the Completer Abort is received by a Function's primary side.
27 SIGNALED_TARGET_ABORT	Signaled Target Abort. This bit is set when a Function completes a Posted or Non-Posted Request as a Completer Abort error. This applies to a Function when the Completer Abort was generated by its primary side.
26-25 DEV_SEL_TIMING	DEVSEL Timing. This field was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires it to 00b.
24 MASTER_DPE	Master Data Parity Error. This bit is set by a Function if the Parity Error Response bit in the Command register is 1b and either of the following two conditions occurs: - Port receives a Poisoned Completion going downstream - Port transmits a Poisoned Request upstream If the Parity Error Response bit is 0b, this bit is never set.
23 FAST_B2B_CAP	Fast Back-to-Back Transactions Capable. This bit was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.
22 RSVDP_22	Reserved for future use.
21 FAST_66MHZ_CAP	66 MHz Capable. This bit was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.
20 CAP_LIST	Capabilities List. Indicates the presence of an Extended Capability list item. Since all PCI Express device Functions are required to implement the PCI Express Capability structure, the controller hardwires this bit to 1b.
19 INT_STATUS	Interrupt Status. When set, indicates that an INTx emulation interrupt is pending internally in the Function. INTx emulation interrupts forwarded by Functions from the secondary side are not reflected in this bit. Setting the Interrupt Disable bit has no effect on the state of this bit. For Functions that do not generate INTx interrupts, the controller hardwires this bit to 0b.
18-17 RSVDP_17	Reserved for future use.
16 —	Reserved.
15-11 RESERV	Reserved.
10 INT_EN	Interrupt Disable. Controls the ability of a Function to generate INTx emulation interrupts. When set, Functions are prevented from asserting INTx interrupts. Note: - Any INTx emulation interrupts already asserted by the Function must be deasserted when this bit is set. INTx interrupts use virtual wires that must, if asserted, be deasserted using the appropriate Deassert_INTx message(s) when this bit is set. - Only the INTx virtual wire interrupt(s) associated with the Function(s) for which this bit is set are affected. - For Functions that generate INTx interrupts on their own behalf, this bit is required. This bit has no effect on interrupts forwarded from the secondary side. For Functions that do not generate INTx interrupts on their own behalf this bit is optional. If this bit is not implemented, the controller hardwires it to 0b.
9	Reserved for future use.

Table continues on the next page...

Field	Function
RSVDP_9	
8 SERREN	SERR# Enable. When set, this bit enables reporting upstream of Non-fatal and Fatal errors detected by the Function. Note: The errors are reported if enabled either through this bit or through the PCI Express specific bits in the Device Control register. For more details see the "Error Registers" section of the PCI Express Specification. In addition, this bit controls transmission by the primary interface of ERR_NONFATAL and ERR_FATAL error Messages forwarded from the secondary interface. This bit does not affect the transmission of forwarded ERR_COR messages.
7 IDSEL	IDSEL Stepping/Wait Cycle Control. This bit was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.
6 PERREN	Parity Error Response. This bit controls the logging of poisoned TLPs in the Master Data Parity Error bit in the Status register. For more details see the "Error Registers" section of the PCI Express Specification.
5 VGAPS	VGA Palette Snoop. This bit was originally described in the PCI Local Bus Specification and the PCI-to-PCI Bridge Architecture Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.
4 MWI_EN	Memory Write and Invalidate. This bit was originally described in the PCI Local Bus Specification and the PCI-to-PCI Bridge Architecture Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b. For PCI Express to PCI/PCI-X Bridges, refer to the PCI Express to PCI/PCI-X Bridge Specification for requirements for this register.
3 SCO	Special Cycle Enable. This bit was originally described in the PCI Local Bus Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b.
2 BME	Bus Master Enable. This bit controls forwarding of Memory or I/O requests by a port in the Upstream direction. When this bit is 0b, Memory and I/O Requests received at a Root Port must be handled as Unsupported Requests (UR) For Non-Posted Requests a Completion with UR completion status must be returned. This bit does not affect forwarding of Completions in either the Upstream or Downstream direction. The forwarding of Requests other than Memory or I/O Requests is not controlled by this bit.
1 MSE	Memory Space Enable. This bit controls a Function's response to Memory Space accesses received on its primary side. - When set, the Function is enabled to decode the address and further process Memory Space accesses. - When clear, all received Memory Space accesses are caused to be handled as Unsupported Requests. You cannot write to this register if your configuration has no MEM bars; that is, the internal signal has_mem_bar =0. Note: The access attributes of this field are as follows: - Dbi: ! has_mem_bar ? RO : RW
0 IO_EN	IO Space Enable. This bit controls a Function's response to I/O Space accesses received on its primary side. - When set, the Function is enabled to decode the address and further process I/O Space accesses. - When clear, all received I/O accesses are caused to be handled as Unsupported Requests. You cannot write to this register if your configuration has no IO bars; that is, the internal signal has_io_bar =0. Note: The access attributes of this field are as follows: - Dbi: !has_io_bar ? RO : RW

11.3.5.1.4 Class Code and Revision ID Register. (TYPE1_CLASS_CODE_REV_ID_REG)

11.3.5.1.4.1 Offset

Register	Offset
TYPE1_CLASS_CODE_REV_ID_REG	8h

11.3.5.1.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BASE_CLASS_CODE								SUBCLASS_CODE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PROGRAM_INTERFACE								REVISION_ID							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

11.3.5.1.4.3 Fields

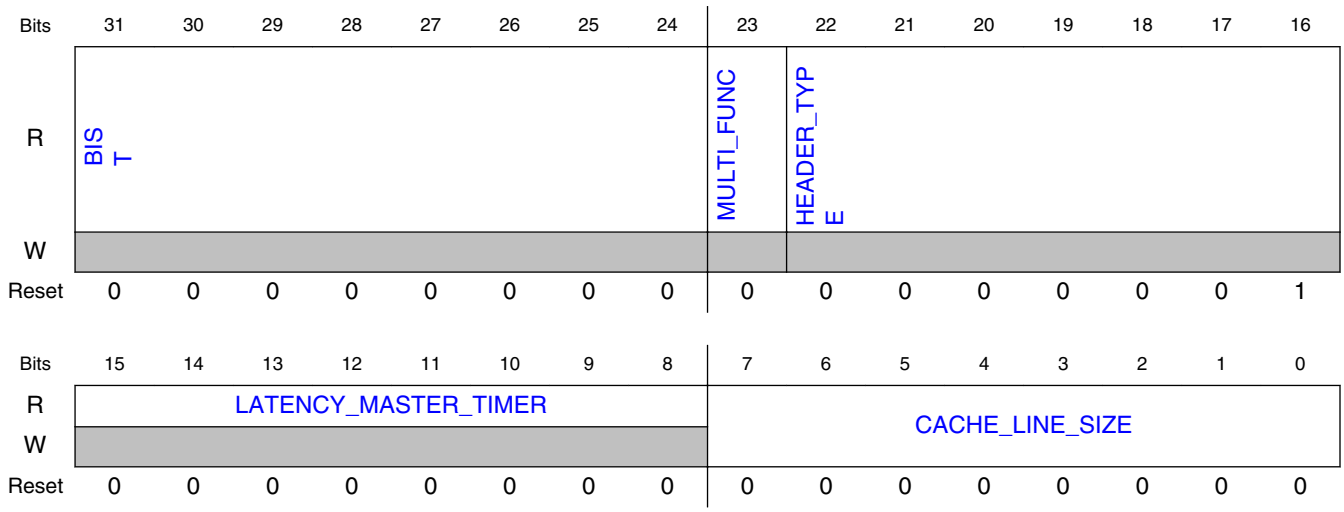
Field	Function
31-24 BASE_CLASS_CODE	Base Class Code. A code that broadly classifies the type of operation the Function performs. Encodings for base class, are provided in the PCI Code and ID Assignment Specification. All unspecified encodings are reserved. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
23-16 SUBCLASS_CODE	Sub-Class Code. Specifies a base class sub-class, which identifies more specifically the operation of the Function. Encodings for sub-class are provided in the PCI Code and ID Assignment Specification. All unspecified encodings are reserved. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
15-8 PROGRAM_INTERFACE	Programming Interface. This field identifies a specific register level programming interface (if any) so that device independent software can interact with the Function. Encodings for interface are provided in the PCI Code and ID Assignment Specification. All unspecified encodings are reserved. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
7-0 REVISION_ID	Revision ID. The value of this field specifies a Function specific revision identifier. The value is chosen by the vendor. Zero is an acceptable value. The Revision ID should be viewed as a vendor defined extension to the Device ID. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

11.3.5.1.5 BIST, Header Type, Latency Timer, and Cache Line Size Register. (TYPE1_BIST_HDR_TYPE_LAT_CACHE_LINE_SIZE_REG)

11.3.5.1.5.1 Offset

Register	Offset
TYPE1_BIST_HDR_TYPE_LAT_CACHE_LINE_SIZE_REG	Ch

11.3.5.1.5.2 Diagram



11.3.5.1.5.3 Fields

Field	Function
31-24 BIST	BIST. This register is used for control and status of BIST. Functions that do not support BIST must hardwire the register to 00h. A Function whose BIST is invoked must not prevent normal operation of the PCI Express Link. Bit descriptions: - [31]: BIST Capable. When set, this bit indicates that the Function supports BIST. When Clear, the Function does not support BIST. - [30]: Start BIST. If BIST Capable is set, set this bit to invoke BIST. The Function resets the bit when BIST is complete. Software is permitted to fail the device if this bit is not Clear (BIST is not complete) 2 seconds after it had been set. Writing this bit to 0b has no effect. The controller hardwires this bit to 0b if BIST Capable is clear. - [29:28]: Reserved. - [27:24]: Completion Code. This field encodes the status of the most recent test. A value of 0000b means that the Function has passed its test. Non-zero values mean the Function failed. Function-specific failure codes can be encoded in the non-zero values. This field's value is only meaningful when BIST Capable is set and Start BIST is Clear. This field must be hardwired to 0000b if BIST Capable is clear.
23 MULTI_FUNC	Multi-Function Device. - When set, indicates that the device may contain multiple Functions, but not necessarily. Software is permitted to probe for Functions other than Function 0. - When clear, software must not probe for Functions other than Function 0 unless explicitly indicated by another mechanism, such as an ARI or SR-IOV Capability structure. Except where stated otherwise, it is recommended that this bit be set if there are multiple Functions, and clear if there is only one Function. Note: This register field is sticky.
22-16 HEADER_TYPE	Header Layout. This field identifies the layout of the second part of the predefined header. The controller uses 000 0001b encoding. The encoding 000 0010b is reserved. This encoding was originally described in the PC Card Standard Electrical Specification and is used in previous versions of the programming model. Careful consideration should be given to any attempt to repurpose it.
15-8 LATENCY_MASTER_TIMER	Latency Timer. This register is also referred to as Primary Latency Timer. The Latency Timer was originally described in the PCI Local Bus Specification and the PCI-to-PCI Bridge Architecture Specification. Its functionality does not apply to PCI Express. The controller hardwires this register to 00h.
7-0 CACHE_LINE_SIZE	Cache Line Size. The Cache Line Size register is programmed by the system firmware or the operating system to system cache line size. However, legacy conventional PCI software may not always be able to program this register correctly especially in the case of Hot-Plug devices. This read-write register is implemented for legacy compatibility purposes but has no effect on any PCI Express device behavior.

11.3.5.1.6 Secondary Latency Timer, Subordinate Bus Number, Secondary Bus Number, and Primary Bus Number Register. (SEC_LAT_TIMER_SUB_BUS_SEC_BUS_PRI_BUS_REG)

11.3.5.1.6.1 Offset

Register	Offset
SEC_LAT_TIMER_SUB_BUS_SEC_BUS_PRI_BUS_REG	18h

11.3.5.1.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEC_LAT_TIMER								SUB_BUS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SEC_BUS								PRIM_BUS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.6.3 Fields

Field	Function
31-24 SEC_LAT_TIMER	Secondary Latency Timer. This register does not apply to PCI Express. The controller hardwires it to 00h.
23-16 SUB_BUS	Subordinate Bus Number. The Subordinate Bus Number register is used to record the bus number of the highest numbered PCI bus segment which is behind (or subordinate to) the bridge. Configuration software programs the value in this register. The bridge uses this register in conjunction with the Secondary Bus Number register to determine when to respond to and pass on a Type 1 configuration transaction on the primary interface to the secondary interface.
15-8 SEC_BUS	Secondary Bus Number. The Secondary Bus Number register is used to record the bus number of the PCI bus segment to which the secondary interface of the bridge is connected. Configuration software programs the value in this register. The bridge uses this register to determine when to respond to and convert a Type 1 configuration transaction on the primary interface into a Type 0 transaction on the secondary interface.
7-0 PRIM_BUS	Primary Bus Number. This register is not used by PCI Express Functions. It is implemented for compatibility with legacy software.

11.3.5.1.7 Secondary Status, and I/O Limit and Base Register. (SEC_STAT _IO_LIMIT_IO_BASE_REG)

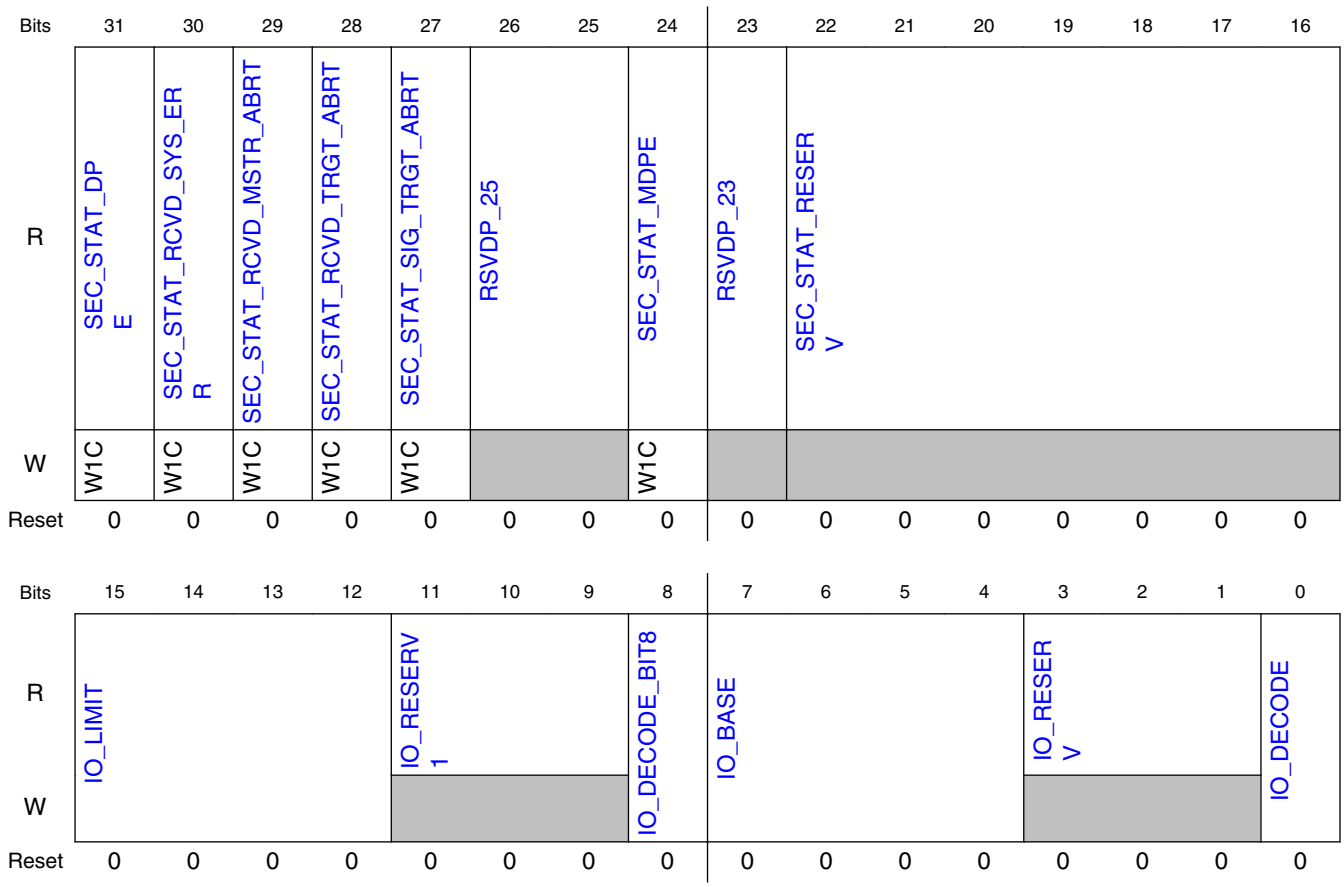
11.3.5.1.7.1 Offset

Register	Offset
SEC_STAT_IO_LIMIT_ IO_BASE_REG	1Ch

11.3.5.1.7.2 Function

Secondary Status, and I/O Limit and Base Register. The I/O Limit and I/O Base registers are optional and define an address range that is used by the bridge to determine when to forward I/O transactions from one interface to the other. If a bridge does not implement an I/O address range, then both the I/O Limit and I/O Base registers must be implemented as read-only registers that return zero when read. If a bridge supports an I/O address range, then these registers must be initialized by configuration software so default states are not specified.

11.3.5.1.7.3 Diagram



11.3.5.1.7.4 Fields

Field	Function
31 SEC_STAT_DPE	Detected Parity Error. This bit is set by a Function when a Poisoned TLP is received by its secondary side, regardless of the state the Parity Error Response Enable bit in the Bridge Control register.
30 SEC_STAT_RCVD_SYS_ERR	Received System Error. This bit is set when the secondary side of a Function receives an ERR_FATAL or ERR_NONFATAL message.
29 SEC_STAT_RCVD_MSTR_ABRT	Received Master Abort. This bit is set when the secondary side of a Function (for requests initiated by the Type 1 header Function itself) receives a Completion with Unsupported Request Completion status.
28 SEC_STAT_RCVD_TRGT_ABRT	Received Target Abort. This bit is set when the secondary side of a Function (for requests initiated by the Type 1 header Function itself) receives a Completion with Completer Abort Completion status.

Table continues on the next page...

Field	Function
27 SEC_STAT_SIG_TRGT_ABRT	Signaled Target Abort. This bit is set when the secondary side of the Function (for Requests completed by the Type 1 header Function itself) completes a Posted or Non-Posted request as a Completer Abort error.
26-25 RSVDP_25	Reserved for future use.
24 SEC_STAT_MDPE	Master Data Parity Error. This bit is set by a Function if the Parity Error Response Enable bit in the Bridge Control register is set, and either of the following two conditions occurs: - Port receives a Poisoned Completion coming Upstream - Port transmits a Poisoned Request Downstream If the Parity Error Response Enable bit is clear, this bit is never set.
23 RSVDP_23	Reserved for future use.
22-16 SEC_STAT_RESERV	Reserved.
15-12 IO_LIMIT	I/O Limit Address. These bits correspond to the address[15:12] of IO address range. For the purpose of address decoding, the bridge assumes that the lower 12 address bits, address[11:0], of the I/O limit address (not implemented in the I/O Limit register) are FFFh. The I/O Limit register can be programmed to a smaller value than the I/O Base register, if there are no I/O addresses on the secondary side of the bridge. In this case, the bridge will not forward any I/O transactions from the primary bus to the secondary and will forward all I/O transactions from the secondary bus to the primary bus.
11-9 IO_RESERV1	Reserved.
8 IO_DECODE_BIT8	I/O Addressing Encode (IO Limit Address). This bit encodes the IO addressing capability of the bridge. IO_DECODE_BIT8 indicates the following: - 0h: The bridge supports only 16-bit I/O addressing (for ISA compatibility). For the purpose of address decoding, the bridge assumes that the upper 16 address bits, Address[31:16], of the I/O limit address (not implemented in I/O Limit register) are zero. Note: The bridge must still perform a full 32-bit decode of the I/O address (that is, check that Address[31:16] are 0000h). In this case, the I/O address range supported by the bridge will be restricted to the first 64 KB of I/O Space (0000 0000h to 0000 FFFFh). - 01h: The bridge supports 32-bit I/O address decoding, and the I/O Limit Upper 16 Bits hold the upper 16 bits, corresponding to Address[31:16], of the 32-bit Limit address. In this case, system configuration software is permitted to locate the I/O address range supported by the bridge anywhere in the 4-GB I/O Space. Note: The 4-KB alignment and granularity restrictions still apply when the bridge supports 32-bit I/O addressing. Note: The access attributes of this field are as follows: - Dbi: R
7-4 IO_BASE	I/O Base Address. These bits correspond to the address[15:12] of IO address range. For the purpose of address decoding, the bridge assumes that the lower 12 address bits, address[11:0], of the I/O base address (not implemented in the I/O Base register) are zero.
3-1 IO_RESERV	Reserved.
0 IO_DECODE	I/O Addressing Encode (IO Base Address) This bit encodes the IO addressing capability of the bridge. IO_DECODE indicates the following: - 0h: The bridge supports only 16-bit I/O addressing (for ISA compatibility). For the purpose of address decoding, the bridge assumes that the upper 16 address bits, Address[31:16], of the I/O base address (not implemented in I/O base register) are zero. Note: The bridge must still perform a full 32-bit decode of the I/O address (that is, check that Address[31:16] are 0000h). In this case, the I/O address range supported by the bridge will be restricted to the first 64 KB of I/O Space (0000 0000h to 0000 FFFFh). - 01h: The bridge supports 32-bit I/O address decoding, and the I/O Base Upper 16 Bits hold the upper 16 bits, corresponding to Address[31:16], of the 32-bit Base address. In this case, system configuration software is permitted to locate the I/O address range supported by the bridge anywhere in the 4-GB I/O Space. Note: The 4-KB alignment and granularity restrictions still apply when the bridge supports 32-bit I/O addressing. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R

11.3.5.1.8 Memory Limit and Base Register. (MEM_LIMIT_MEM_BASE_REG)

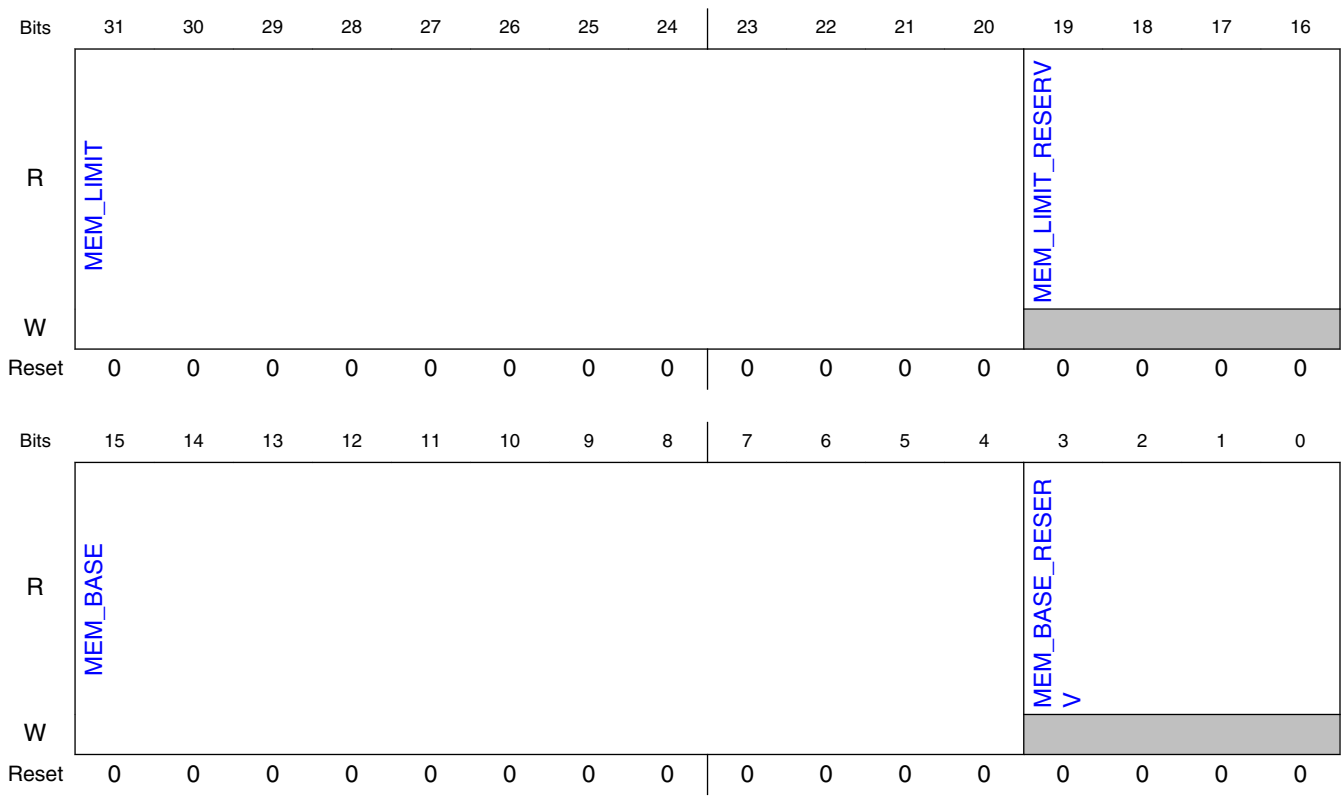
11.3.5.1.8.1 Offset

Register	Offset
MEM_LIMIT_MEM_BASE_REG	20h

11.3.5.1.8.2 Function

Memory Limit and Base Register. The Memory Limit and Memory Base registers define a memory mapped address range which is used by the bridge to determine when to forward memory transactions from one interface to the other. If there is no prefetchable memory space, and there is no memory-mapped space on the secondary side of the bridge, then the bridge will not forward any memory transactions from the primary bus to the secondary bus and will forward all memory transactions from the secondary bus to the primary bus.

11.3.5.1.8.3 Diagram



11.3.5.1.8.4 Fields

Field	Function
31-20 MEM_LIMIT	Memory Limit Address. These bits correspond to the upper 12 address bits, Address[31:20], of 32-bit addresses. For the purpose of address decoding, the bridge assumes that the lower 20 address bits, Address[19:0], of the memory limit address (not implemented in the Memory Limit register) are F FFFFh. The Memory Limit register must be programmed to a smaller value than the Memory Base register if there is no memory-mapped address space on the secondary side of the bridge.
19-16 MEM_LIMIT_RESERV	Reserved.
15-4 MEM_BASE	Memory Base Address. These bits correspond to the upper 12 address bits, Address[31:20], of 32-bit addresses. For the purpose of address decoding, the bridge assumes that the lower 20 address bits, Address[19:0], of the memory base address (not implemented in the Memory Base register) are zero.
3-0 MEM_BASE_RESERV	Reserved.

11.3.5.1.9 Prefetchable Memory Limit and Base Register. (PREF_MEM_LIMIT_PREF_MEM_BASE_REG)

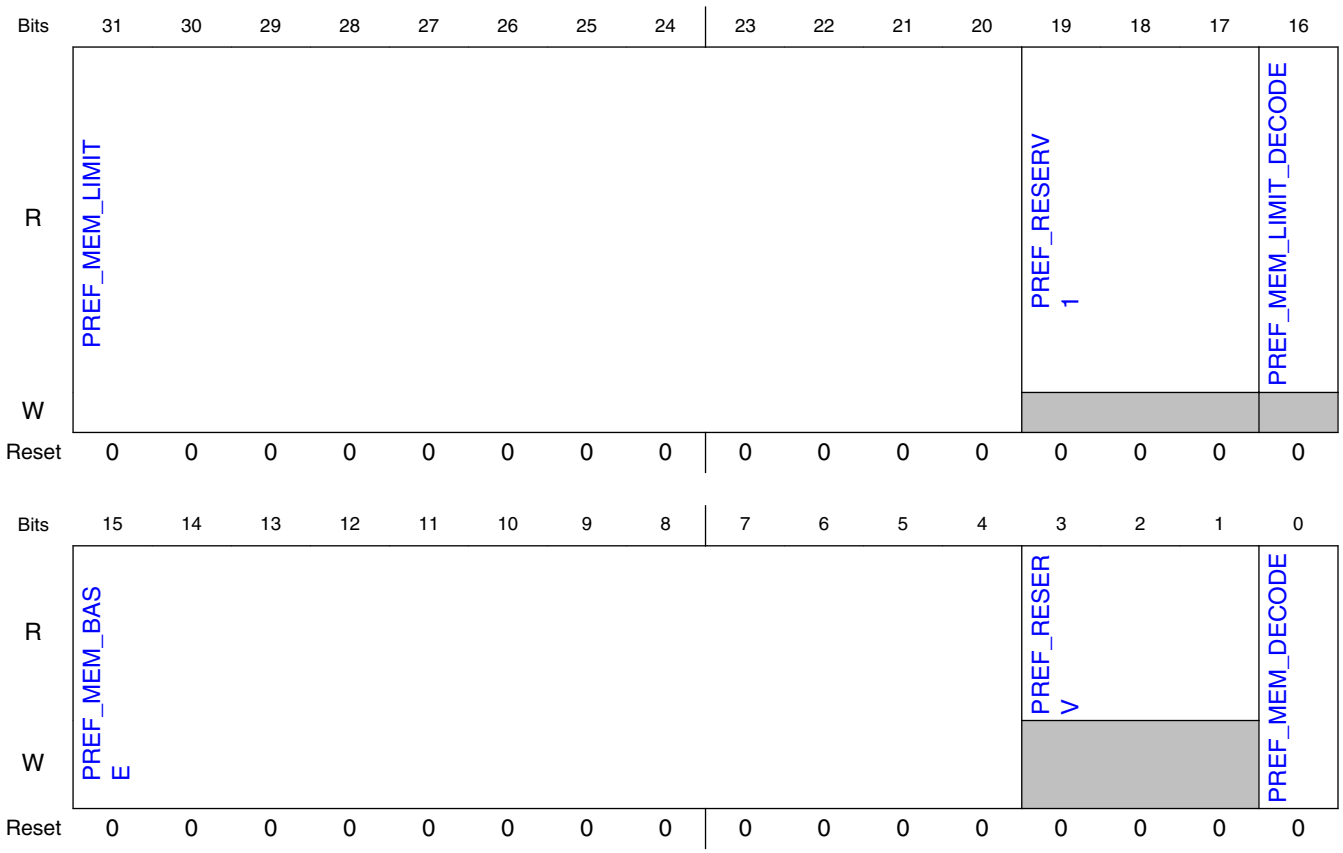
11.3.5.1.9.1 Offset

Register	Offset
PREF_MEM_LIMIT_PREF_MEM_BASE_REG	24h

11.3.5.1.9.2 Function

Prefetchable Memory Limit and Base Register. The Prefetchable Memory Limit and Prefetchable Memory Base registers must indicate that 64-bit addresses are supported, as defined in PCI-to-PCI Bridge Architecture Specification. The Prefetchable Memory Limit and Prefetchable Memory Base registers are optional. They define a prefetchable memory address range which is used by the bridge to determine when to forward memory transactions from one interface to the other (see the PCI-to-PCI Bridge Architecture Specification for additional details).

11.3.5.1.9.3 Diagram



11.3.5.1.9.4 Fields

Field	Function
31-20 PREF_MEM_LIMIT	Prefetchable Memory Limit Address. If the Prefetchable Memory Limit register indicates support for 32-bit addressing, then the Prefetchable Limit Upper 32 Bits register is implemented as a read-only register that returns zero when read. If the Prefetchable Memory Limit registers indicate support for 64-bit addressing, then the Prefetchable Limit Upper 32 Bits register is implemented as a read/write register which must be initialized by configuration software. If a 64-bit prefetchable memory address range is supported, the Prefetchable Limit Upper 32 Bits register specifies the upper 32 bits, corresponding to Address[63:32], of the 64-bit limit addresses which specify the prefetchable memory address range.
19-17 PREF_RESERV 1	Reserved.
16 PREF_MEM_LIMIT_DECODE	Prefetchable Memory Limit Decode. This bit encodes whether or not the bridge supports 64-bit addresses. The value of PREF_MEM_LIMIT_DECODE indicates the following: - 0b: Indicates that the bridge supports only 32 bit addresses - 1b: Indicates that the bridge supports 64 bit addresses. Prefetchable Limit Upper 32 Bits registers holds the rest of the 64-bit prefetchable limit address.
15-4 PREF_MEM_BASE	Prefetchable Memory Base Address. If the Prefetchable Memory Base register indicates support for 32-bit addressing, then the Prefetchable Base Upper 32 Bits register is implemented as a read-only register that returns zero when read. If the Prefetchable Memory Base register indicates support for 64-bit addressing, then the Prefetchable Limit Upper 32 Bits register is implemented as a read/write register

Table continues on the next page...

Field	Function
	which must be initialized by configuration software. If a 64-bit prefetchable memory address range is supported, the Prefetchable Base Upper 32 Bits register specifies the upper 32 bits, corresponding to Address[63:32], of the 64-bit base addresses which specify the prefetchable memory address range.
3-1 PREF_RESERV	Reserved.
0 PREF_MEM_DECODE	Prefetchable Memory Base Decode. This bit encodes whether or not the bridge supports 64-bit addresses. The value of PREF_MEM_DECODE indicates the following: - 0b: Indicates that the bridge supports only 32 bit addresses. - 1b: Indicates that the bridge supports 64 bit addresses. Prefetchable Base Upper 32 Bits registers holds the rest of the 64-bit prefetchable base address. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.

11.3.5.1.10 Prefetchable Base Upper 32 Bits Register. (PREF_BASE_UPPER_REG)

11.3.5.1.10.1 Offset

Register	Offset
PREF_BASE_UPPER_REG	28h

11.3.5.1.10.2 Function

Prefetchable Base Upper 32 Bits Register. The Prefetchable Base Upper 32 Bits register is an optional extension to the Prefetchable Memory Base register.

11.3.5.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PREF_MEM_BASE_UPPER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PREF_MEM_BASE_UPPER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.10.4 Fields

Field	Function
31-0 PREF_MEM_BASE_UPPER	Prefetchable Base Upper 32 Bit. If the Prefetchable Memory Base register indicates support for 32-bit addressing, then this register is implemented as read-only register that returns zero when read. If the Prefetchable Memory Base register indicate support for 64-bit addressing, then this register is implemented as read/write register which must be initialized by configuration software. This register specifies the upper 32 bits, corresponding to Address[63:32], of the 64-bit base addresses which specify the prefetchable memory address range. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R

11.3.5.1.11 Prefetchable Limit Upper 32 Bits Register. (PREF_LIMIT_UPPER_REG)

11.3.5.1.11.1 Offset

Register	Offset
PREF_LIMIT_UPPER_REG	2Ch

11.3.5.1.11.2 Function

Prefetchable Limit Upper 32 Bits Register. The Prefetchable Limit Upper 32 Bits register is an optional extension to the Prefetchable Memory Limit register.

11.3.5.1.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PREF_MEM_LIMIT_UPPER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PREF_MEM_LIMIT_UPPER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.11.4 Fields

Field	Function
31-0 PREF_MEM_LIMIT_UPPER	Prefetchable Limit Upper 32 Bit. If the Prefetchable Memory Limit register indicate support for 64-bit addressing, then this register is implemented as read/write register which must be initialized by configuration software. This register specifies the upper 32 bits, corresponding to Address[63:32], of the 64-bit base addresses which specify the prefetchable memory address range. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R

11.3.5.1.12 I/O Limit and Base Upper 16 Bits Register. (IO_LIMIT_UPPER_IO_BASE_UPPER_REG)

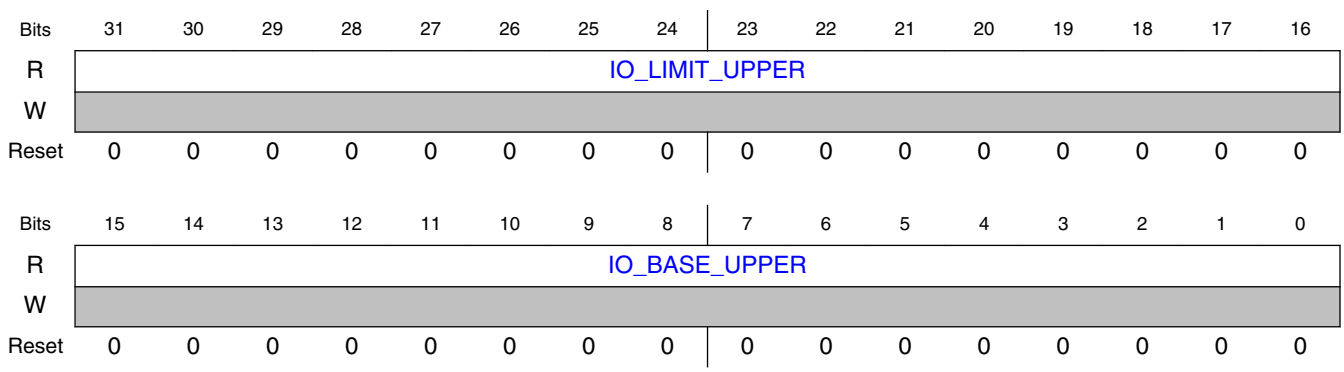
11.3.5.1.12.1 Offset

Register	Offset
IO_LIMIT_UPPER_IO_BASE_UPPER_REG	30h

11.3.5.1.12.2 Function

I/O Limit and Base Upper 16 Bits Register. The I/O Limit Upper 16 Bits and I/O Base Upper 16 Bits registers are optional extensions to the I/O Limit and I/O Base registers.

11.3.5.1.12.3 Diagram



11.3.5.1.12.4 Fields

Field	Function
31-16 IO_LIMIT_UPPER	I/O Limit Upper 16 Bits. If the I/O Limit register indicates support for 16-bit I/O address decoding, then this register is implemented as a read-only register which return zero when read. If the I/O Limit register indicates support for 32-bit I/O addressing, then this register must be initialized by configuration software.

Table continues on the next page...

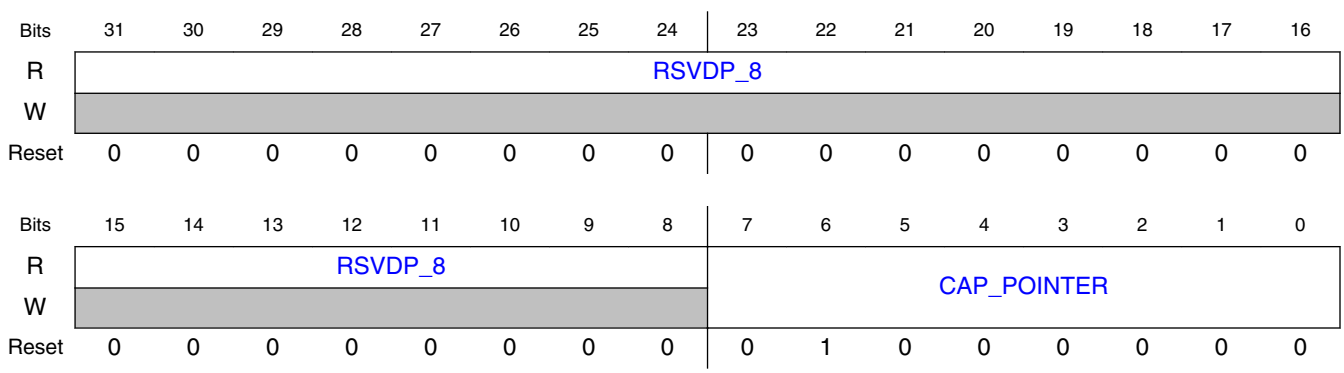
Field	Function
	If 32-bit I/O address decoding is supported, this register specifies the upper 16 bits, corresponding to Address[31:16], of the 32-bit limit address, that specify the I/O address range. See the PCI-to-PCI Bridge Architecture Specification for additional details). Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
15-0 IO_BASE_UPPER	I/O Base Upper 16 Bits. If the I/O Base register indicates support for 16-bit I/O address decoding, then this register is implemented as a read-only register which return zero when read. If the I/O base register indicates support for 32-bit I/O addressing, then this register must be initialized by configuration software. If 32-bit I/O address decoding is supported, this register specifies the upper 16 bits, corresponding to Address[31:16], of the 32-bit base address, that specify the I/O address range. See the PCI-to-PCI Bridge Architecture Specification for additional details. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R

11.3.5.1.13 Capabilities Pointer Register. (TYPE1_CAP_PTR_REG)

11.3.5.1.13.1 Offset

Register	Offset
TYPE1_CAP_PTR_REG	34h

11.3.5.1.13.2 Diagram



11.3.5.1.13.3 Fields

Field	Function
31-8 RSVDP_8	Reserved for future use.
7-0 CAP_POINTER	Capabilities Pointer. This register is used to point to a linked list of capabilities implemented by this Function. Since all PCI Express Functions are required to implement the PCI Express Capability structure, this register must point to a valid capability structure and either this structure is the PCI Express Capability structure, or a subsequent list item points to the PCI Express Capability structure. The bottom

Field	Function
	two bits are Reserved and must be set to 00b. Software must mask these bits off before using this register as a pointer in Configuration Space to the first entry of a linked list of new capabilities. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

11.3.5.1.14 Expansion ROM Base Address Register. (TYPE1_EXP_ROM_BASE_REG)

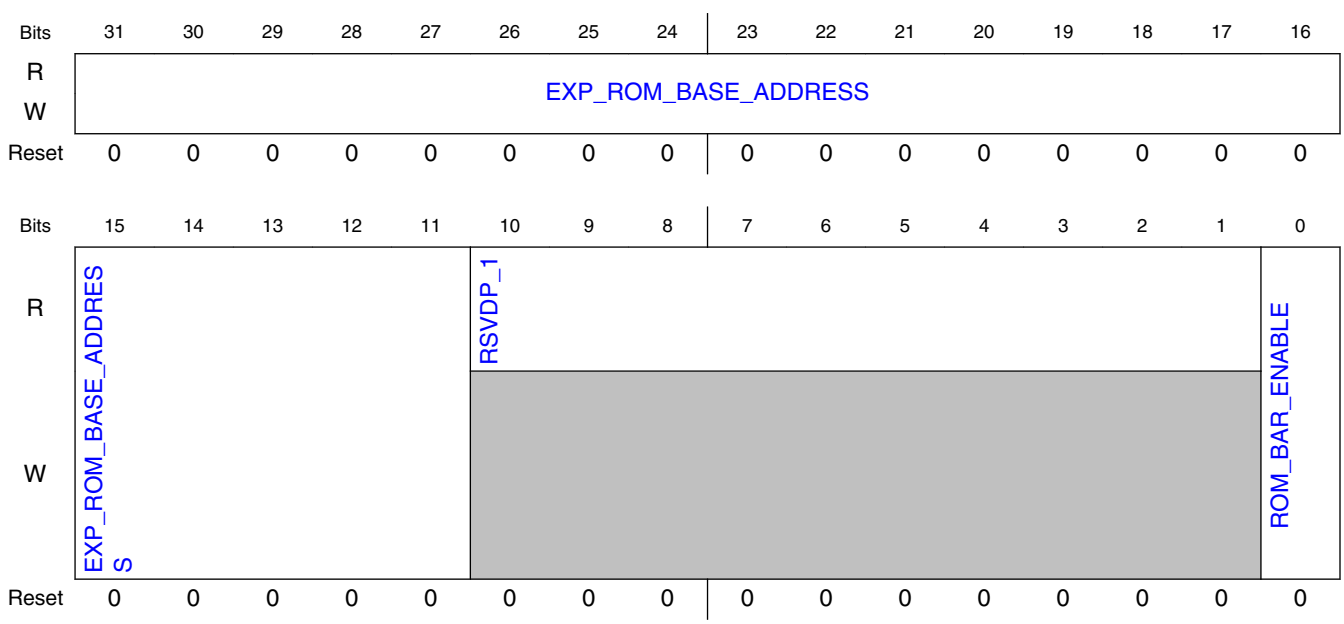
11.3.5.1.14.1 Offset

Register	Offset
TYPE1_EXP_ROM_BASE_REG	38h

11.3.5.1.14.2 Function

Expansion ROM Base Address Register. This register is defined to handle the base address and size information for this expansion ROM. The mask for this ROM BAR exists (if implemented) as a shadow register at this address. The assertion of CS2 (that is, assert the dbi_cs2 input, or the CS2 address bit for the AXI bridge) is required to write to the second register at this address.

11.3.5.1.14.3 Diagram



11.3.5.1.14.4 Fields

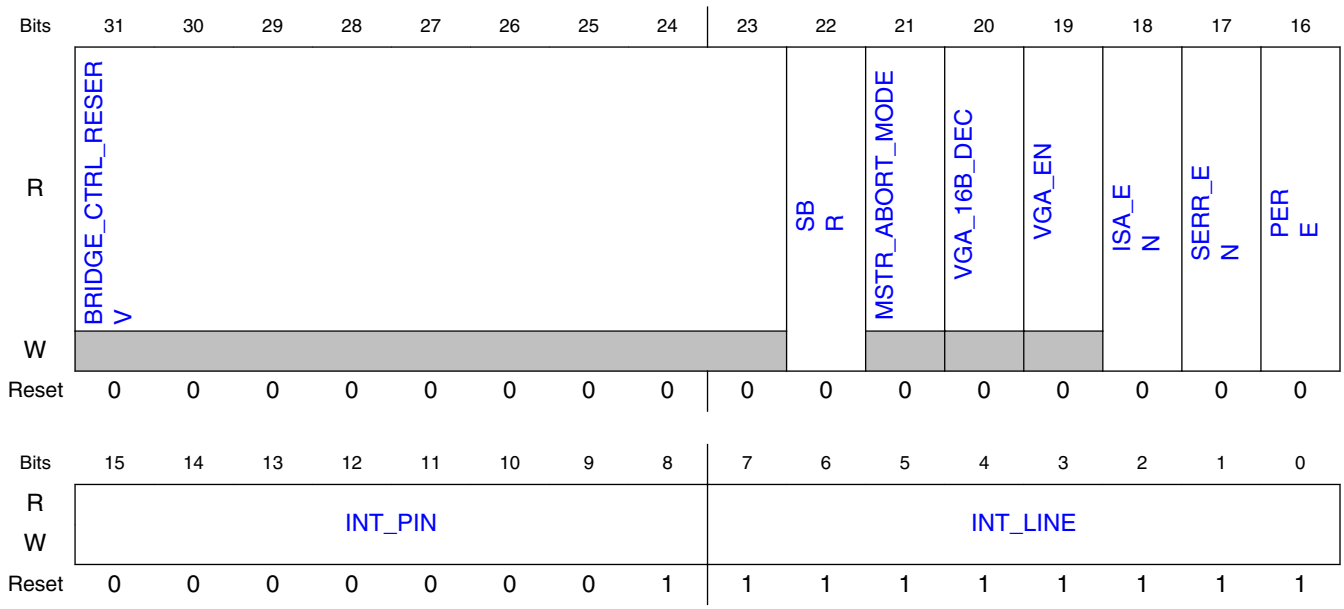
Field	Function
31-11 EXP_ROM_BASE_ADDRESS	Expansion ROM Base Address. Upper 21 bits of the Expansion ROM base address. The number of bits (out of these 21) that a Function actually implements depends on how much address space the Function requires. The mask for this ROM BAR exists (if implemented) as a shadow register at this address. The assertion of CS2 (that is, assert the dbi_cs2 input, or the CS2 address bit for the AXI bridge) is required to write to the second register at this address. Note: The access attributes of this field are as follows: - Dbi: R/W
10-1 RSVDP_1	Reserved for future use.
0 ROM_BAR_ENABLE	Expansion ROM Enable. This bit controls whether or not the Function accepts accesses to its expansion ROM. When this bit is 0b, the Function's expansion ROM address space is disabled. When the bit is 1b, address decoding is enabled using the parameters in the other part of the Expansion ROM Base Address register. The Memory Space Enable bit in the Command register has precedence over the Expansion ROM Enable bit. A Function must claim accesses to its expansion ROM only if both the Memory Space Enable bit and the Expansion ROM Enable bit are set. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.15 Bridge Control, Interrupt Pin, and Interrupt Line Register. (BRIDGE_CTRL_INT_PIN_INT_LINE_REG)

11.3.5.1.15.1 Offset

Register	Offset
BRIDGE_CTRL_INT_PIN_INT_LINE_REG	3Ch

11.3.5.1.15.2 Diagram



11.3.5.1.15.3 Fields

Field	Function
31-23 BRIDGE_CTRL_RESERV	Reserved.
22 SBR	Secondary Bus Reset. Setting this bit triggers a hot reset on the corresponding PCI Express Port. Software must ensure a minimum reset duration (Trst) as defined in the PCI Local Bus Specification. Software and systems must honor first-access-following-reset timing requirements, unless the Readiness Notifications mechanism is used or if the Immediate Readiness bit in the relevant Function's Status Register register is set. Port configuration registers must not be changed, except as required to update Port status.
21 MSTR_ABORT_MODE	Master Abort Mode. This bit was originally described in the PCI-to-PCI Bridge Architecture Specification. Its functionality does not apply to PCI Express. The controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Dbi: R/W
20 VGA_16B_DEC	VGA 16 bit decode. This bit only has meaning if VGA Enable bit is set. This bit enables system configuration software to select between 10-bit and 16-bit I/O address decoding for all VGA I/O register accesses that are forwarded from primary to secondary. The following actions are taken based on the value of the VGA_16B_DEC bit: - 0b: Execute 10-bit address decodes on VGA I/O accesses - 1b: Execute 16-bit address decodes on VGA I/O accesses For Functions that do not support VGA, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Dbi: R
19 VGA_EN	VGA Enable. Modifies the response by the bridge to VGA compatible addresses. If the VGA Enable bit is set, the bridge will positively decode and forward the following accesses on the primary interface to the secondary interface (and, conversely, block the forwarding of these addresses from the secondary to primary interface): - Memory accesses in the range 000A 0000h to 000B FFFFh - I/O addresses in the first 64 KB of the I/O address space (Address[31:16] are 0000h) where Address[9:0] are in the ranges 3B0h to 3BBh and 3C0h to 3DFh (inclusive of ISA address aliases determined by the setting of VGA 16-bit Decode) If the VGA Enable bit is set, forwarding of these accesses is independent of the I/O address range and memory address ranges defined by the I/O Base and Limit registers, the Memory Base and

Table continues on the next page...

Field	Function
	Limit registers, and the Prefetchable Memory Base and Limit registers of the bridge. (Forwarding of these accesses is also independent of the setting of the ISA Enable bit (in the Bridge Control register) when the VGA Enable bit is set. Forwarding of these accesses is qualified by the I/O Space Enable and Memory Space Enable bits in the Command register.) The following actions are taken based on the value of the VGA_EN bit: - 0b: Do not forward VGA compatible memory and I/O addresses from the primary to the secondary interface (addresses defined above) unless they are enabled for forwarding by the defined I/O and memory address ranges - 1b: Forward VGA compatible memory and I/O addresses (addresses defined above) from the primary interface to the secondary interface (if the I/O Space Enable and Memory Space Enable bits are set) independent of the I/O and memory address ranges and independent of the ISA Enable bit For Functions that do not support VGA, the controller hardwires this bit to 0b. Note: The access attributes of this field are as follows: - Dbi: R
18 ISA_EN	ISA Enable. Modifies the response by the bridge to ISA I/O addresses. This applies only to I/O addresses that are enabled by the I/O Base and I/O Limit registers and are in the first 64 KB of I/O address space (0000 0000h to 0000 FFFFh). If this bit is set, the bridge will block any forwarding from primary to secondary of I/O transactions addressing the last 768 bytes in each 1-KB block. In the opposite direction (secondary to primary), I/O transactions will be forwarded if they address the last 768 bytes in each 1-KB block. The following actions are taken based on the value of the ISA_EN bit: - 0b: Forward downstream all I/O addresses in the address range defined by the I/O Base and I/O Limit registers - 1b: Forward upstream ISA I/O addresses in the address range defined by the I/O Base and I/O Limit registers that are in the first 64 KB of PCI I/O address space (top 768 bytes of each 1-KB block.
17 SERR_EN	SERR# Enable. This bit controls forwarding of ERR_COR, ERR_NONFATAL and ERR_FATAL from secondary to primary.
16 PERE	Parity Error Response Enable. This bit controls the logging of poisoned TLPs in the Master Data Parity Error bit in the Secondary Status register.
15-8 INT_PIN	Interrupt PIN. The Interrupt Pin register register that identifies the legacy interrupt Message(s) the Function uses. Valid values are: - 01h, 02h, 03h, and 04h: map to legacy interrupt Messages for INTA, INTB, INTC, and INTD respectively. - 00h: indicates that the Function uses no legacy interrupt Message(s). - 05h through FFh: Reserved. PCI Express defines one legacy interrupt Message for a single Function device and up to four legacy interrupt Messages for a multi-Function device. For a single Function device, only INTA may be used. Any Function on a multi-Function device can use any of the INTx Messages. If a device implements a single legacy interrupt Message, it must be INTA; if it implements two legacy interrupt Messages, they must be INTA and INTB; and so forth. For a multi-Function device, all Functions may use the same INTx Message or each may have its own (up to a maximum of four Functions) or any combination thereof. A single Function can never generate an interrupt request on more than one INTx Message. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
7-0 INT_LINE	Interrupt Line. The Interrupt Line register communicates interrupt line routing information. The register must be implemented by any Function that uses an interrupt pin. Values in this register are programmed by system software and are system architecture specific. The Function itself does not use this value; rather the value in this register is used by device drivers and operating systems.

11.3.5.1.16 Power Management Capabilities Register. (CAP_ID_NXT_PTR_REG)

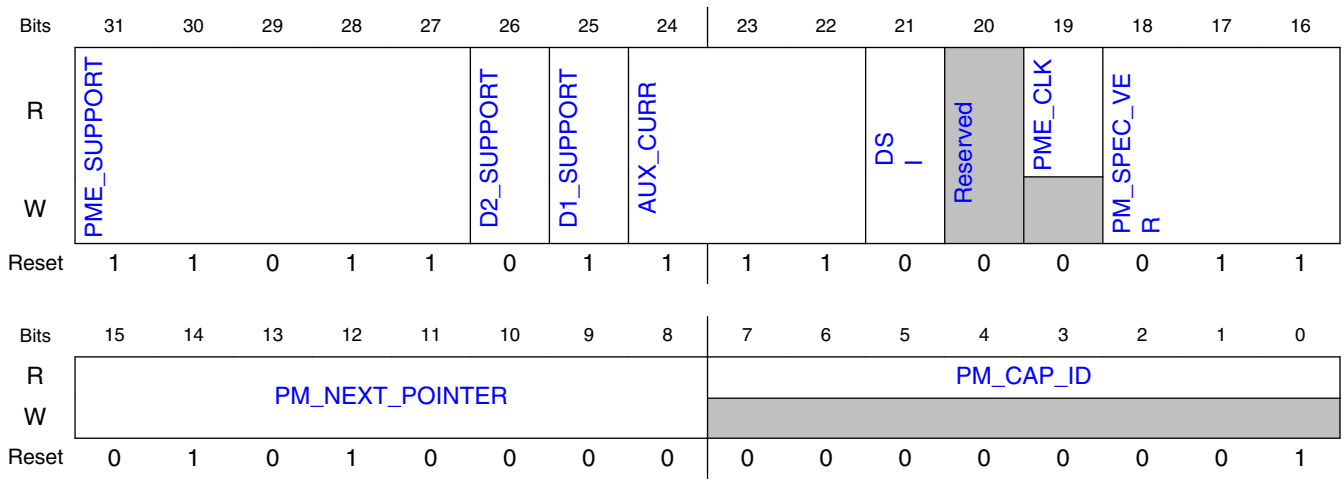
11.3.5.1.16.1 Offset

Register	Offset
CAP_ID_NXT_PTR_REG	40h

11.3.5.1.16.2 Function

Power Management Capabilities Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.16.3 Diagram



11.3.5.1.16.4 Fields

Field	Function
31-27 PME_SUPPORT	Power Management Event Support. For a description of this standard PCIe register field, see the PCI Express Specification. The read value from this field is the write value && (sys_aux_pwr_det, 1'b1, D2_SUPPORT, D1_SUPPORT, 1'b1), where D1_SUPPORT and D2_SUPPORT are fields in this register. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
26 D2_SUPPORT	D2 State Support. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
25 D1_SUPPORT	D1 State Support. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
24-22 AUX_CURR	Auxiliary Current Requirements. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
21 DSI	Device Specific Initialization Bit. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
20 —	Reserved.
19 PME_CLK	PCI Clock Requirement. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

Table continues on the next page...

Field	Function
18-16 PM_SPEC_VER	Power Management Spec Version. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
15-8 PM_NEXT_POINTER	Next Capability Pointer. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
7-0 PM_CAP_ID	Power Management Capability ID. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.17 Power Management Control and Status Register. (CON_STATUS_REG)

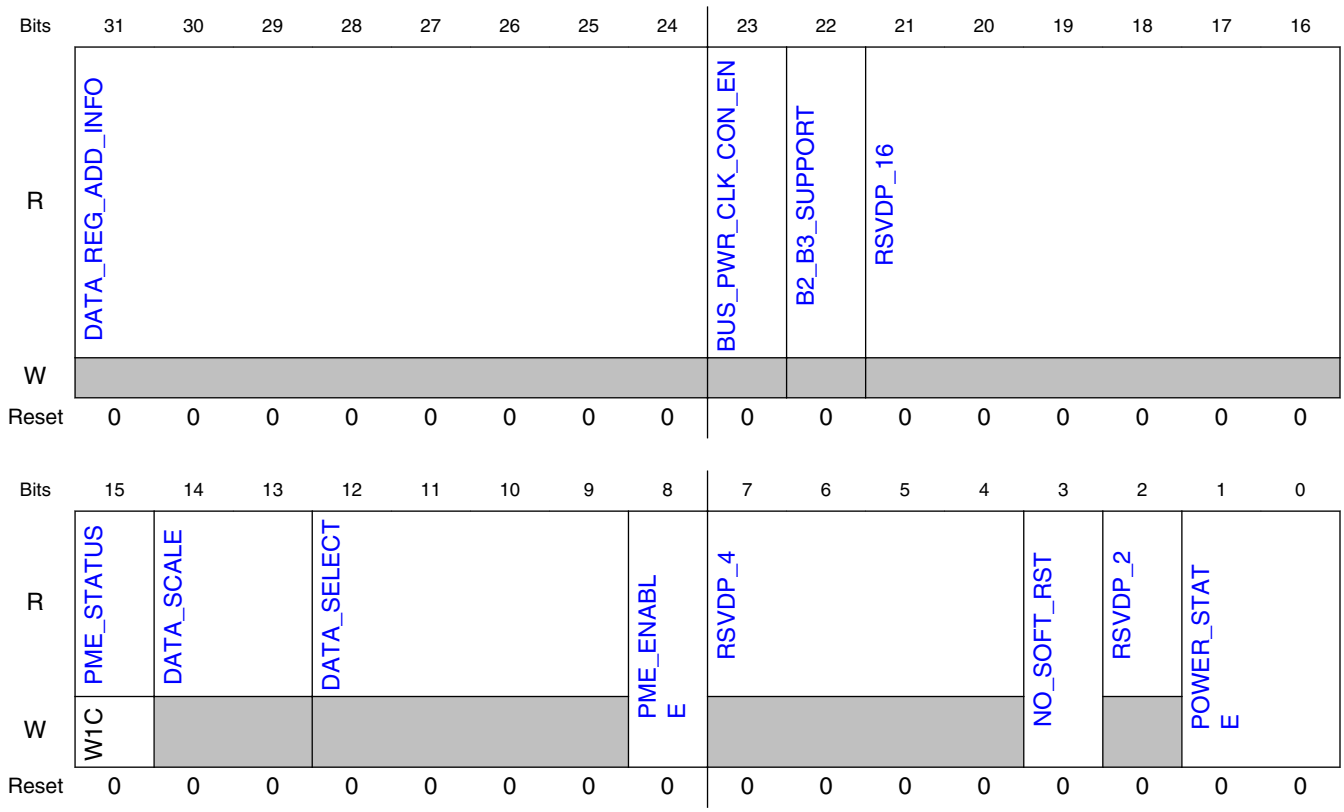
11.3.5.1.17.1 Offset

Register	Offset
CON_STATUS_REG	44h

11.3.5.1.17.2 Function

Power Management Control and Status Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.17.3 Diagram



11.3.5.1.17.4 Fields

Field	Function
31-24 DATA_REG_ADD_INFO	Power Data Information Register. For a description of this standard PCIe register field, see the PCI Express Specification.
23 BUS_PWR_CLK_CON_EN	Bus Power/Clock Control Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
22 B2_B3_SUPPORT	B2B3 Support for D3hot. For a description of this standard PCIe register field, see the PCI Express Specification.
21-16 RSVDP_16	Reserved for future use.
15 PME_STATUS	PME Status. For a description of this standard PCIe register field, see the PCI Express Specification.
14-13 DATA_SCALE	Data Scaling Factor. For a description of this standard PCIe register field, see the PCI Express Specification.
12-9	Data Select. For a description of this standard PCIe register field, see the PCI Express Specification.

Table continues on the next page...

Field	Function
DATA_SELECT	
8 PME_ENABLE	PME Enable. For a description of this standard PCIe register field, see the PCI Express Specification. The PMC registers this value under aux power. Sometimes it might remember the old value, even if you try to clear it by writing '0'. Note: This register field is sticky.
7-4 RSVDP_4	Reserved for future use.
3 NO_SOFT_RST	No soft Reset. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
2 RSVDP_2	Reserved for future use.
1-0 POWER_STATE	Power State. For a description of this standard PCIe register field, see the PCI Express Specification. You can write to this register. However, the read-back value is the actual power state, not the write value. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.18 MSI Capability ID, Next Pointer, Capability/Control Registers. (PCI_MSI_CAP_ID_NEXT_CTRL_REG)

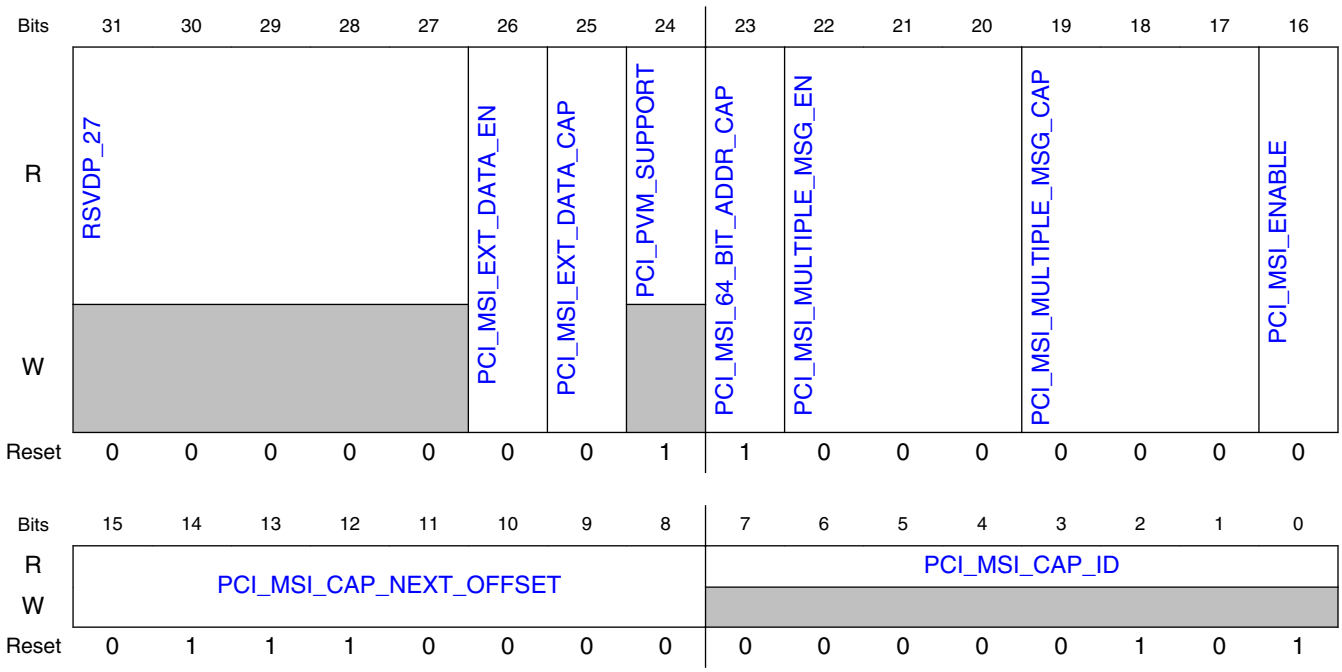
11.3.5.1.18.1 Offset

Register	Offset
PCI_MSI_CAP_ID_NEXT_CTRL_REG	50h

11.3.5.1.18.2 Function

MSI Capability ID, Next Pointer, Capability/Control Registers. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.18.3 Diagram



11.3.5.1.18.4 Fields

Field	Function
31-27 RSVDP_27	Reserved for future use.
26 PCI_MSI_EXT_DATA_EN	Extended Message Data Enable. For a description of this standard PCIe register, see the PCI-SIG ECN for Extended MSI Data, Feb 24, 2016, affecting PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: PCI_MSI_CAP_ID_NEXT_CTRL_REG.PCI_MSI_EXT_DATA_CAP ? RW : RO
25 PCI_MSI_EXT_DATA_CAP	Extended Message Data Capable. For a description of this standard PCIe register, see the PCI-SIG ECN for Extended MSI Data, Feb 24, 2016, affecting PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
24 PCI_PVM_SUPPORT	MSI Per Vector Masking Capable. For a description of this standard PCIe register field, see the PCI Express Specification.
23 PCI_MSI_64_BIT_ADDR_CAP	MSI 64-bit Address Capable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
22-20 PCI_MSI_MULTIPLE_MSG_EN	MSI Multiple Message Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
19-17	MSI Multiple Message Capable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.

Table continues on the next page...

Field	Function
PCI_MSI_MULTIPLE_MSG_CAP	
16 PCI_MSI_ENABLE	MSI Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
15-8 PCI_MSI_CAP_NEXT_OFFSET	MSI Capability Next Pointer. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
7-0 PCI_MSI_CAP_ID	MSI Capability ID. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.19 MSI Message Lower Address Register. (MSI_CAP_OFF_04H_REG)

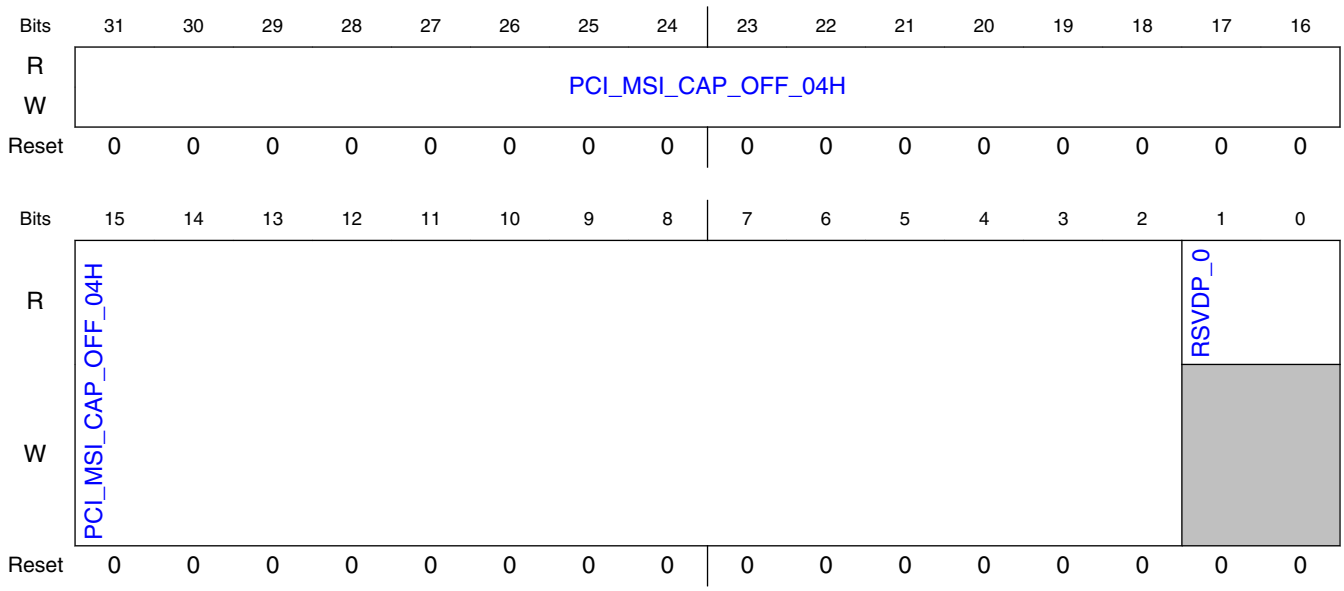
11.3.5.1.19.1 Offset

Register	Offset
MSI_CAP_OFF_04H_REG	54h

11.3.5.1.19.2 Function

MSI Message Lower Address Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.19.3 Diagram



11.3.5.1.19.4 Fields

Field	Function
31-2 PCI_MSI_CAP_OFF_04H	MSI Message Lower Address Field. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W
1-0 RSVDP_0	Reserved for future use.

11.3.5.1.20 For a 32 bit MSI Message, this register contains Data. (MSI_CAP_OFF_08H_REG)

11.3.5.1.20.1 Offset

Register	Offset
MSI_CAP_OFF_08H_REG	58h

11.3.5.1.20.2 Function

For a 32 bit MSI Message, this register contains Data. For 64 bit it contains the Upper Address. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.20.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PCI_MSI_CAP_OFF_0AH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PCI_MSI_CAP_OFF_08H															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.20.4 Fields

Field	Function
31-16 PCI_MSI_CAP_OFF_0AH	For a 32 bit MSI Message, this field contains Ext MSI Data. For 64-bit it contains upper 16 bits of the Upper Address. For a description of this standard PCIe register field, see the PCI Express Specification Note: The access attributes of this field are as follows: - Dbi: PCI_MSI_64_BIT_ADDR_CAP || `DEFAULT_EXT_MSI_DATA_CAPABLE ? R/W : R
15-0 PCI_MSI_CAP_OFF_08H	For a 32-bit MSI Message, this field contains Data. For 64-bit it contains lower 16 bits of the Upper Address. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: PCI_MSI_64_BIT_ADDR_CAP ? R/W : R

11.3.5.1.21 For a 64 bit MSI Message, this register contains Data. (MSI_CAP_OFF_0CH_REG)

11.3.5.1.21.1 Offset

Register	Offset
MSI_CAP_OFF_0CH_REG	5Ch

11.3.5.1.21.2 Function

For a 64 bit MSI Message, this register contains Data. For 32 bit, it contains Mask Bits if PVM enabled. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.21.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PCI_MSI_CAP_OFF_0EH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PCI_MSI_CAP_OFF_0CH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.21.4 Fields

Field	Function
31-16 PCI_MSI_CAP_OFF_0EH	For a 64-bit MSI Message, this field contains Data. For 32-bit, it contains the upper Mask Bits if PVM is enabled. For a description of this standard PCIe register field, see the PCI Express Specification Note: The access attributes of this field are as follows: - Dbi: (!MSI_64_EN && MSI_PVM_EN_VALUE) ? RW: MSI_64_EN && DEFAULT_EXT_MSI_DATA_CAPABLE ? RW : RO
15-0 PCI_MSI_CAP_OFF_0CH	For a 64-bit MSI Message, this field contains Data. For 32-bit, it contains the lower Mask Bits if PVM is enabled. For a description of this standard PCIe register field, see the PCI Express Specification Note: The access attributes of this field are as follows: - Dbi: PCI_MSI_64_BIT_ADDR_CAP || MSI_PVM_EN ? R/W : R

11.3.5.1.22 Used for MSI when Vector Masking Capable. (MSI_CAP_OFF_10H_REG)

11.3.5.1.22.1 Offset

Register	Offset
MSI_CAP_OFF_10H_REG	60h

11.3.5.1.22.2 Function

Used for MSI when Vector Masking Capable. For 32 bit contains Pending Bits. For 64 bit, contains Mask Bits. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.22.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PCI_MSI_CAP_OFF_10H															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PCI_MSI_CAP_OFF_10H															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.22.4 Fields

Field	Function
31-0 PCI_MSI_CAP_OFF_10H	Used for MSI when Vector Masking Capable. For 32-bit contains Pending Bits. For 64-bit, contains Mask Bits. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: PCI_MSI_64_BIT_ADDR_CAP && MSI_PVM_EN ? R/W : R

11.3.5.1.23 Used for MSI 64 bit messaging when Vector Masking Capable. (MSI_CAP_OFF_14H_REG)

11.3.5.1.23.1 Offset

Register	Offset
MSI_CAP_OFF_14H_REG	64h

11.3.5.1.23.2 Function

Used for MSI 64 bit messaging when Vector Masking Capable. Contains Pending Bits. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.23.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PCI_MSI_CAP_OFF_14H															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCI_MSI_CAP_OFF_14H															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.23.4 Fields

Field	Function
31-0 PCI_MSI_CAP_OFF_14H	Used for MSI 64-bit messaging when Vector Masking Capable. Contains Pending Bits. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.24 PCI Express Capabilities, ID, Next Pointer Register. (PCIE_CAP_ID_PCIE_NEXT_CAP_PTR_PCIE_CAP_REG)

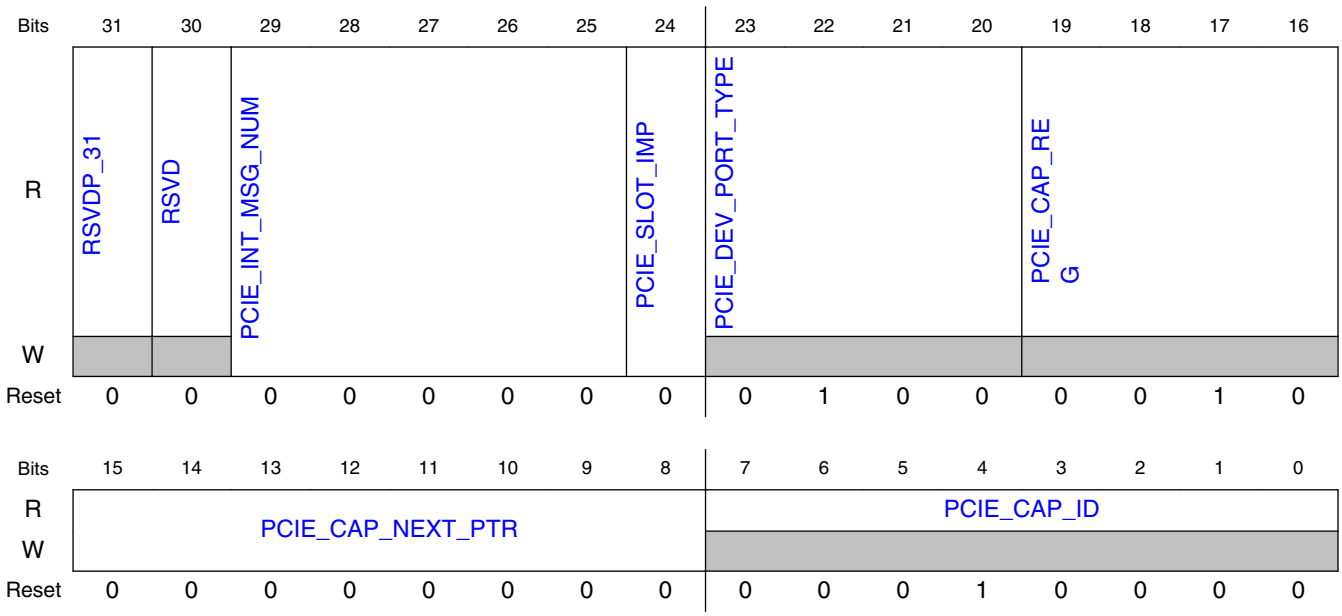
11.3.5.1.24.1 Offset

Register	Offset
PCIE_CAP_ID_PCIE_NEXT_CAP_PTR_PCIE_CAP_REG	70h

11.3.5.1.24.2 Function

PCI Express Capabilities, ID, Next Pointer Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.24.3 Diagram



11.3.5.1.24.4 Fields

Field	Function
31 RSVDP_31	Reserved for future use.
30 RSVD	Reserved. For a description of this standard PCIe register field, see the PCI Express Specification.
29-25 PCIE_INT_MSG_NUM	PCIE Interrupt Message Number. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
24 PCIE_SLOT_IMP	PCIe Slot Implemented Valid. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
23-20 PCIE_DEV_PORT_TYPE	PCIE Device/PortType. For a description of this standard PCIe register field, see the PCI Express Specification.
19-16 PCIE_CAP_REG	PCIE Capability Version Number. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
15-8 PCIE_CAP_NEXT_PTR	PCIE Next Capability Pointer. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
7-0 PCIE_CAP_ID	PCIE Capability ID. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.25 Device Capabilities Register. (DEVICE_CAPABILITIES_REG)

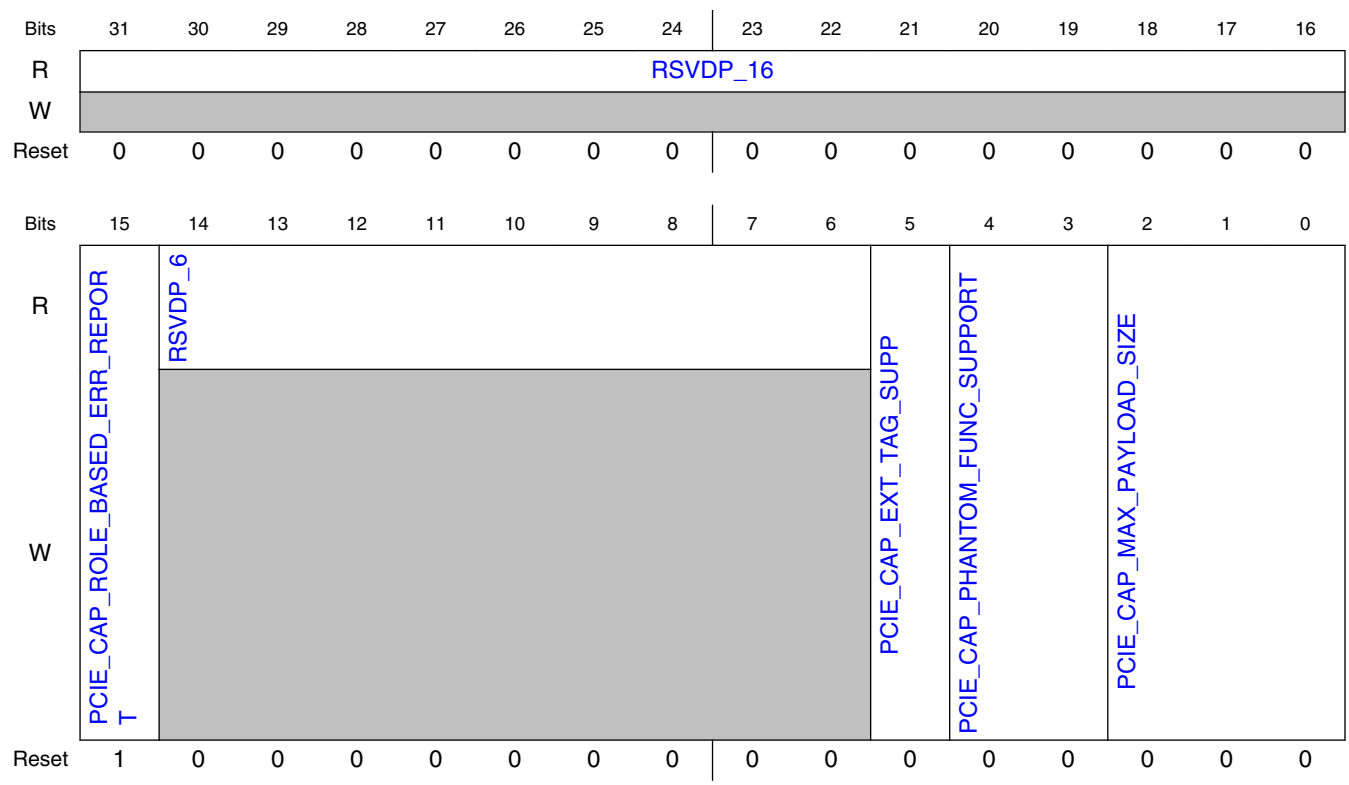
11.3.5.1.25.1 Offset

Register	Offset
DEVICE_CAPABILITIES_REG	74h

11.3.5.1.25.2 Function

Device Capabilities Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.25.3 Diagram



11.3.5.1.25.4 Fields

Field	Function
31-16	Reserved for future use.

Table continues on the next page...

Field	Function
RSVDP_16	
15 PCIE_CAP_ROLE_BASED_ERR_REPORT	Role-based Error Reporting Implemented. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
14-6 RSVDP_6	Reserved for future use.
5 PCIE_CAP_EXT_TAG_SUPP	Extended Tag Field Supported. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
4-3 PCIE_CAP_PHANTOM_FUNC_SUPPORT	Phantom Functions Supported. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
2-0 PCIE_CAP_MAX_PAYLOAD_SIZE	Max Payload Size Supported. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.

11.3.5.1.26 Device Control and Status Register. (DEVICE_CONTROL_DEVICE_STATUS)

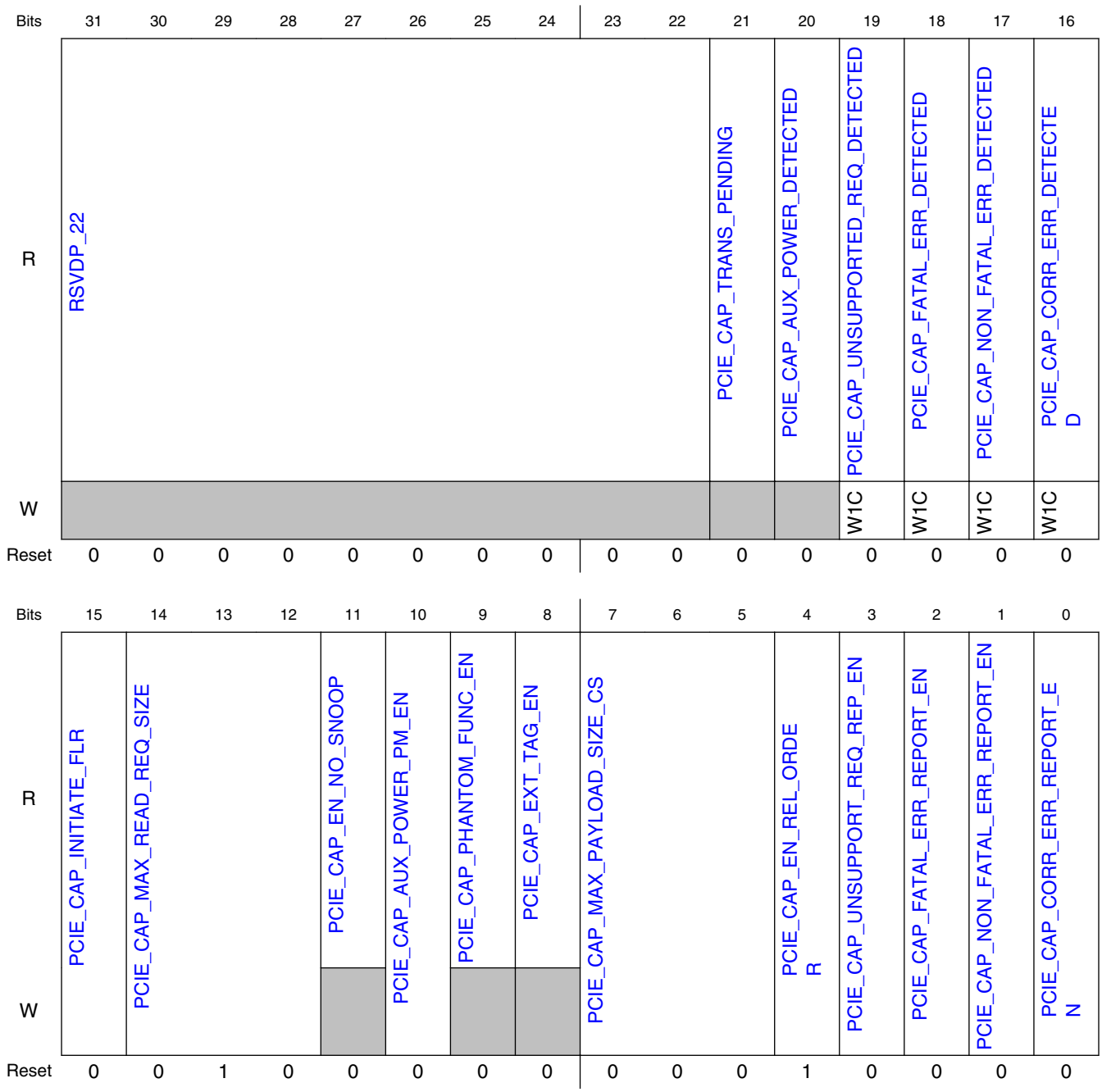
11.3.5.1.26.1 Offset

Register	Offset
DEVICE_CONTROL_DEVICE_STATUS	78h

11.3.5.1.26.2 Function

Device Control and Status Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.26.3 Diagram



11.3.5.1.26.4 Fields

Field	Function
31-22 RSVDP_22	Reserved for future use.

Table continues on the next page...

Field	Function
21 PCIE_CAP_TRANS_PENDING	Transactions Pending Status. For a description of this standard PCIe register field, see the PCI Express Specification.
20 PCIE_CAP_AUX_POWER_DETECTED	Aux Power Detected Status. For a description of this standard PCIe register field, see the PCI Express Specification. This bit is derived by sampling the sys_aux_pwr_det input.
19 PCIE_CAP_UNSUPPORTED_REQ_DETECTED	Unsupported Request Detected Status. For a description of this standard PCIe register field, see the PCI Express Specification.
18 PCIE_CAP_FATAL_ERR_DETECTED	Fatal Error Detected Status. For a description of this standard PCIe register field, see the PCI Express Specification.
17 PCIE_CAP_NON_FATAL_ERR_DETECTED	Non-Fatal Error Detected Status. For a description of this standard PCIe register field, see the PCI Express Specification.
16 PCIE_CAP_CORR_ERR_DETECTED	Correctable Error Detected Status. For a description of this standard PCIe register field, see the PCI Express Specification.
15 PCIE_CAP_INITIATE_FLR	Initiate Function Level Reset (for endpoints). For a description of this standard PCIe register field, see the PCI Express Specification.
14-12 PCIE_CAP_MAX_READ_REQ_SIZE	Max Read Request Size. For a description of this standard PCIe register field, see the PCI Express Specification.
11 PCIE_CAP_EN_NO_SNOOP	Enable No Snoop. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R
10 PCIE_CAP_AUX_POWER_PM_EN	Aux Power PM Enable. For a description of this standard PCIe register field, see the PCI Express Specification. This bit is derived by sampling the sys_aux_pwr_det input. Note: This register field is sticky.
9 PCIE_CAP_PHANTOM_FUNC_EN	Phantom Functions Enable. For a description of this standard PCIe register field, see the PCI Express Specification. The write value is gated with the PCIE_CAP_PHANTOM_FUNC_SUPPORT field of DEVICE_CAPABILITIES_REG. Note: The access attributes of this field are as follows: - Dbi: DEVICE_CAPABILITIES_REG.PCIE_CAP_PHANTOM_FUNC_SUPPORT ? RW : RO
8 PCIE_CAP_EXT_TAG_EN	Extended Tag Field Enable. For a description of this standard PCIe register field, see the PCI Express Specification. The write value is gated with the PCIE_CAP_EXT_TAG_SUPP field of DEVICE_CAPABILITIES_REG. Note: The access attributes of this field are as follows: - Dbi: DEVICE_CAPABILITIES_REG.PCIE_CAP_EXT_TAG_SUPP ? RW : RO

Table continues on the next page...

Field	Function
7-5 PCIE_CAP_MAX_PAYLOAD_SIZE_CS	Max Payload Size. Max_Payload_Size . This field sets maximum TLP payload size for the Function. Permissible values that can be programmed are indicated by the Max_Payload_Size Supported field (PCIE_CAP_MAX_PAYLOAD_SIZE) in the Device Capabilities register (DEVICE_CAPABILITIES_REG).
4 PCIE_CAP_EN_REL_ORDER	Enable Relaxed Ordering. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W
3 PCIE_CAP_UNSUPPORT_REQ_REP_EN	Unsupported Request Reporting Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
2 PCIE_CAP_FATAL_ERR_REPORT_EN	Fatal Error Reporting Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
1 PCIE_CAP_NON_FATAL_ERR_REPORT_EN	Non-fatal Error Reporting Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
0 PCIE_CAP_CORR_ERR_REPORT_EN	Correctable Error Reporting Enable. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.27 Link Capabilities Register. (LINK_CAPABILITIES_REG)

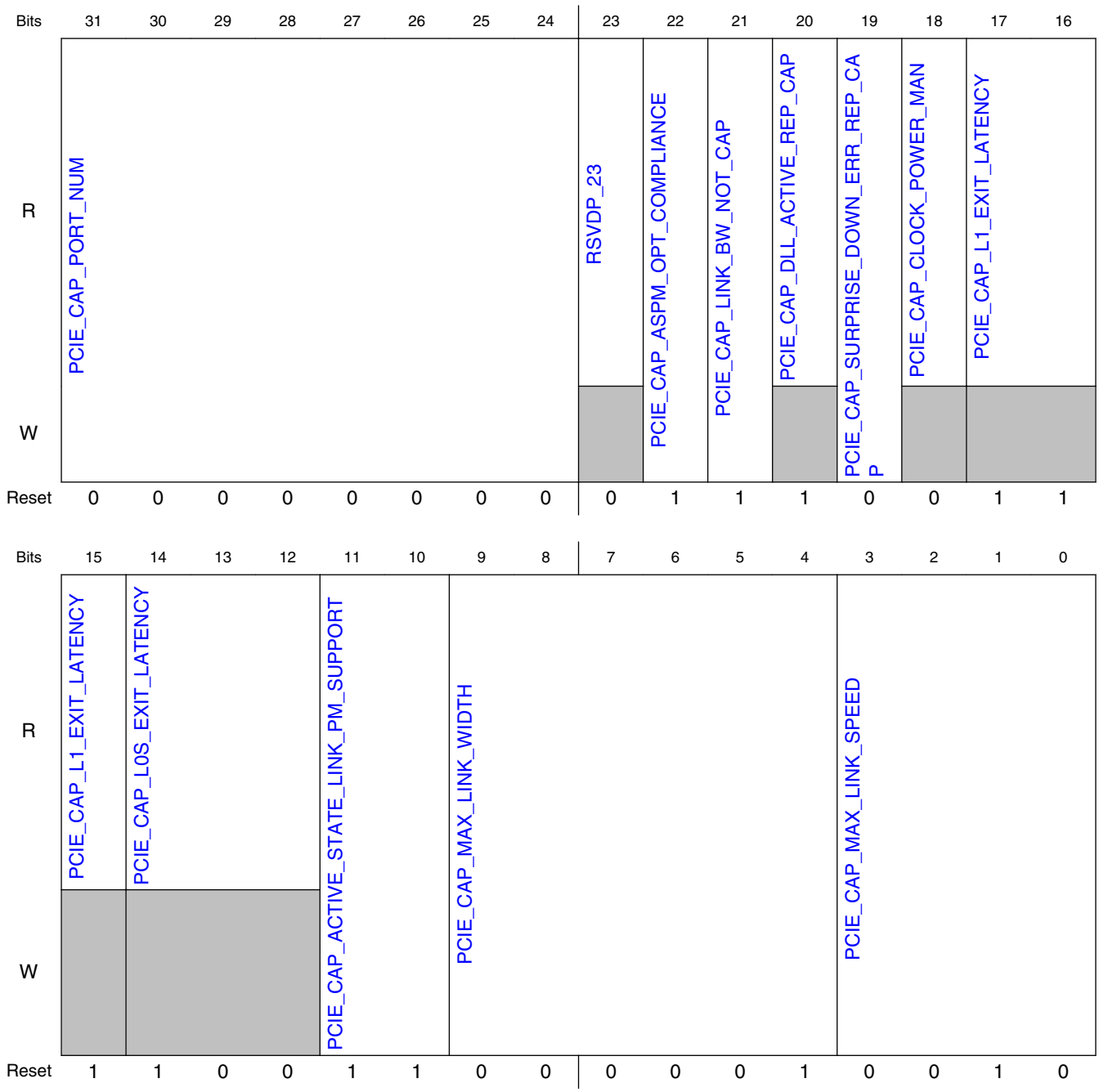
11.3.5.1.27.1 Offset

Register	Offset
LINK_CAPABILITIES_REG	7Ch

11.3.5.1.27.2 Function

Link Capabilities Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.27.3 Diagram



11.3.5.1.27.4 Fields

Field	Function
31-24 PCIE_CAP_PORT_NUM	Port Number. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R

Table continues on the next page...

Field	Function
23 RSVDP_23	Reserved for future use.
22 PCIE_CAP_ASPM_OPT_COMPLIANCE	ASPM Optionality Compliance. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
21 PCIE_CAP_LINK_BW_NOT_CAP	Link Bandwidth Notification Capable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
20 PCIE_CAP_DLL_ACTIVE_REP_CAP	Data Link Layer Link Active Reporting Capable. For a description of this standard PCIe register field, see the PCI Express Specification.
19 PCIE_CAP_SURPRISE_DOWN_ERR_REP_CAP	Surprise Down Error Reporting Capable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
18 PCIE_CAP_CLOCK_POWER_MAN	Clock Power Management. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
17-15 PCIE_CAP_L1_EXIT_LATENCY	L1 Exit Latency. For a description of this standard PCIe register field, see the PCI Express Specification. There are two each of these register fields, this one and a shadow one at the same address. The Common Clock bit (PCIE_CAP_COMMON_CLK_CONFIG) of the Link Control Register (LINK_CONTROL_LINK_STATUS_REG) determines which one is used by the controller and which one is accessed by a read request. Common Clock operation is enabled in the controller when you set the Common Clock bit (PCIE_CAP_COMMON_CLK_CONFIG) of the Link Control Register (LINK_CONTROL_LINK_STATUS_REG). The assertion of CS2 (that is, assert the dbi_cs2 input, or the CS2 address bit for the AXI bridge) is required to write to the shadow field at this location. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
14-12 PCIE_CAP_L0S_EXIT_LATENCY	LOs Exit Latency. For a description of this standard PCIe register field, see the PCI Express Specification. There are two each of these register fields, this one and a shadow one at the same address. The Common Clock bit (PCIE_CAP_COMMON_CLK_CONFIG) of the Link Control Register (LINK_CONTROL_LINK_STATUS_REG) determines which one is used by the controller and which one is accessed by a read request. Common Clock operation is enabled in the controller when you set the Common Clock bit (PCIE_CAP_COMMON_CLK_CONFIG) of the Link Control Register (LINK_CONTROL_LINK_STATUS_REG). The assertion of CS2 (that is, assert the dbi_cs2 input, or the CS2 address bit for the AXI bridge) is required to write to the shadow field at this location. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
11-10 PCIE_CAP_ACTIVE_STATE_LINK_PM_SUPPORT	Level of ASPM (Active State Power Management) Support. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
9-4	Maximum Link Width. For a description of this standard PCIe register field, see the PCI Express Specification. In M-PCIe mode, the reset and dynamic values of this field are calculated by the controller.

Table continues on the next page...

Field	Function
PCIE_CAP_MAX_LINK_WIDTH	Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.
3-0 PCIE_CAP_MAX_LINK_SPEED	Maximum Link Speed. For a description of this standard PCIe register field, see the PCI Express Specification. In M-PCIe mode, the reset and dynamic values of this field are calculated by the controller. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R Note: This register field is sticky.

11.3.5.1.28 Link Control and Status Register. (LINK_CONTROL_LINK_STATUS_REG)

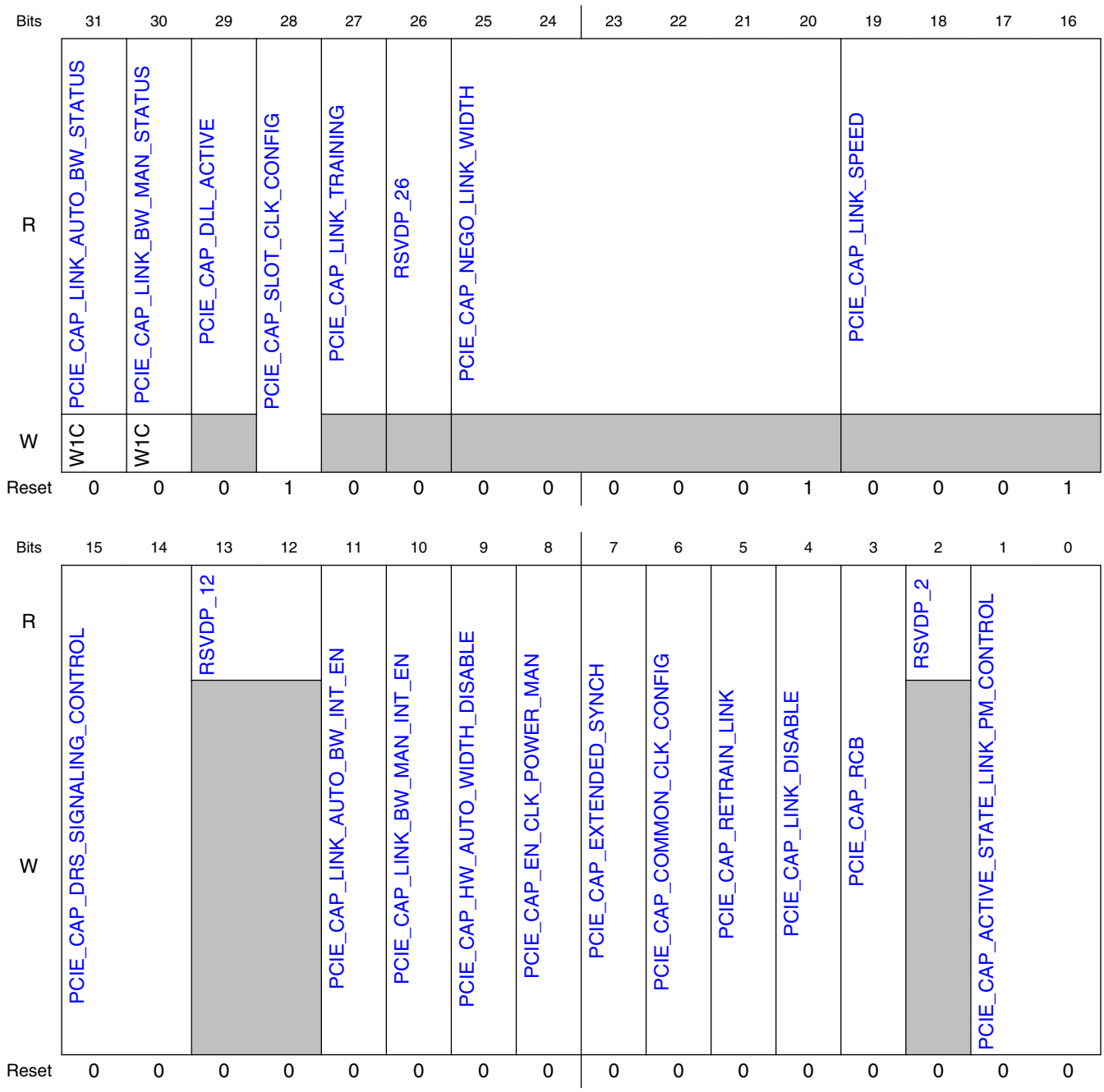
11.3.5.1.28.1 Offset

Register	Offset
LINK_CONTROL_LINK_STATUS_REG	80h

11.3.5.1.28.2 Function

Link Control and Status Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.28.3 Diagram



11.3.5.1.28.4 Fields

Field	Function
31	Link Autonomous Bandwidth Status. For a description of this standard PCIe register field, see the PCI Express Specification. The write value is gated with the PCIE_CAP_LINK_BW_NOT_CAP field in LINK_CAPABILITIES_REG.

Table continues on the next page...

Field	Function
PCIE_CAP_LINK_AUTO_BW_STATUS	
30 PCIE_CAP_LINK_BW_MAN_STATUS	Link Bandwidth Management Status. For a description of this standard PCIe register field, see the PCI Express Specification. The write value is gated with the PCIE_CAP_LINK_BW_NOT_CAP field in LINK_CAPABILITIES_REG.
29 PCIE_CAP_DLL_ACTIVE	Data Link Layer Active. For a description of this standard PCIe register field, see the PCI Express Specification.
28 PCIE_CAP_SLOT_CLK_CONFIG	Slot Clock Configuration. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
27 PCIE_CAP_LINK_TRAINING	LTSSM is in Configuration or Recovery State. For a description of this standard PCIe register field, see the PCI Express Specification.
26 RSVDP_26	Reserved for future use.
25-20 PCIE_CAP_NEGO_LINK_WIDTH	Negotiated Link Width. For a description of this standard PCIe register field, see the PCI Express Specification.
19-16 PCIE_CAP_LINK_SPEED	Current Link Speed. For a description of this standard PCIe register field, see the PCI Express Specification.
15-14 PCIE_CAP_DRS_SIGNALING_CONTROL	DRS Signaling Control. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: LINK_CAPABILITIES2_REG.DRS_SUPPORTED ? RW : RO
13-12 RSVDP_12	Reserved for future use.
11 PCIE_CAP_LINK_AUTO_BW_INT_EN	Link Autonomous Bandwidth Management Interrupt Enable. For a description of this standard PCIe register field, see the PCI Express Specification. The write value is gated with the PCIE_CAP_LINK_BW_NOT_CAP field in LINK_CAPABILITIES_REG. Note: The access attributes of this field are as follows: - Dbi: LINK_CAPABILITIES_REG.PCIE_CAP_LINK_BW_NOT_CAP ? RW : RO
10 PCIE_CAP_LINK_BW_MAN_INT_EN	Link Bandwidth Management Interrupt Enable. For a description of this standard PCIe register field, see the PCI Express Specification. The write value is gated with the PCIE_CAP_LINK_BW_NOT_CAP field in LINK_CAPABILITIES_REG. Note: The access attributes of this field are as follows: - Dbi: LINK_CAPABILITIES_REG.PCIE_CAP_LINK_BW_NOT_CAP ? RW : RO
9 PCIE_CAP_HW_AUTO_WIDTH_DISABLE	Hardware Autonomous Width Disable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W
8	Enable Clock Power Management. For a description of this standard PCIe register field, see the PCI Express Specification. The write value is gated with the PCIE_CAP_CLOCK_POWER_MAN field in

Table continues on the next page...

Field	Function
PCIE_CAP_EN_CLK_POWER_MAN	LINK_CAPABILITIES_REG. Note: The access attributes of this field are as follows: - Dbi: LINK_CAPABILITIES_REG.PCIE_CAP_CLOCK_POWER_MAN ? RWS : ROS Note: This register field is sticky.
7 PCIE_CAP_EXTENDED_SYNCH	Extended Synch. For a description of this standard PCIe register field, see the PCI Express Specification.
6 PCIE_CAP_COMMON_CLK_CONFIG	Common Clock Configuration. For a description of this standard PCIe register field, see the PCI Express Specification.
5 PCIE_CAP_RETRAIN_LINK	Initiate Link Retrain. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: see description
4 PCIE_CAP_LINK_DISABLE	Initiate Link Disable. For a description of this standard PCIe register field, see the PCI Express Specification. In a DSP that supports crosslink, the controller gates the write value with the CROSS_LINK_EN field in PORT_LINK_CTRL_OFF.
3 PCIE_CAP_RCB	Read Completion Boundary (RCB). Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
2 RSVDP_2	Reserved for future use.
1-0 PCIE_CAP_ACTIVE_STATE_LINK_PM_CONTROL	Active State Power Management (ASPM) Control. Software must not enable L0s in either direction on a given Link unless components on both sides of the Link each support L0s; otherwise, the result is undefined. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.29 Slot Capabilities Register. (SLOT_CAPABILITIES_REG)

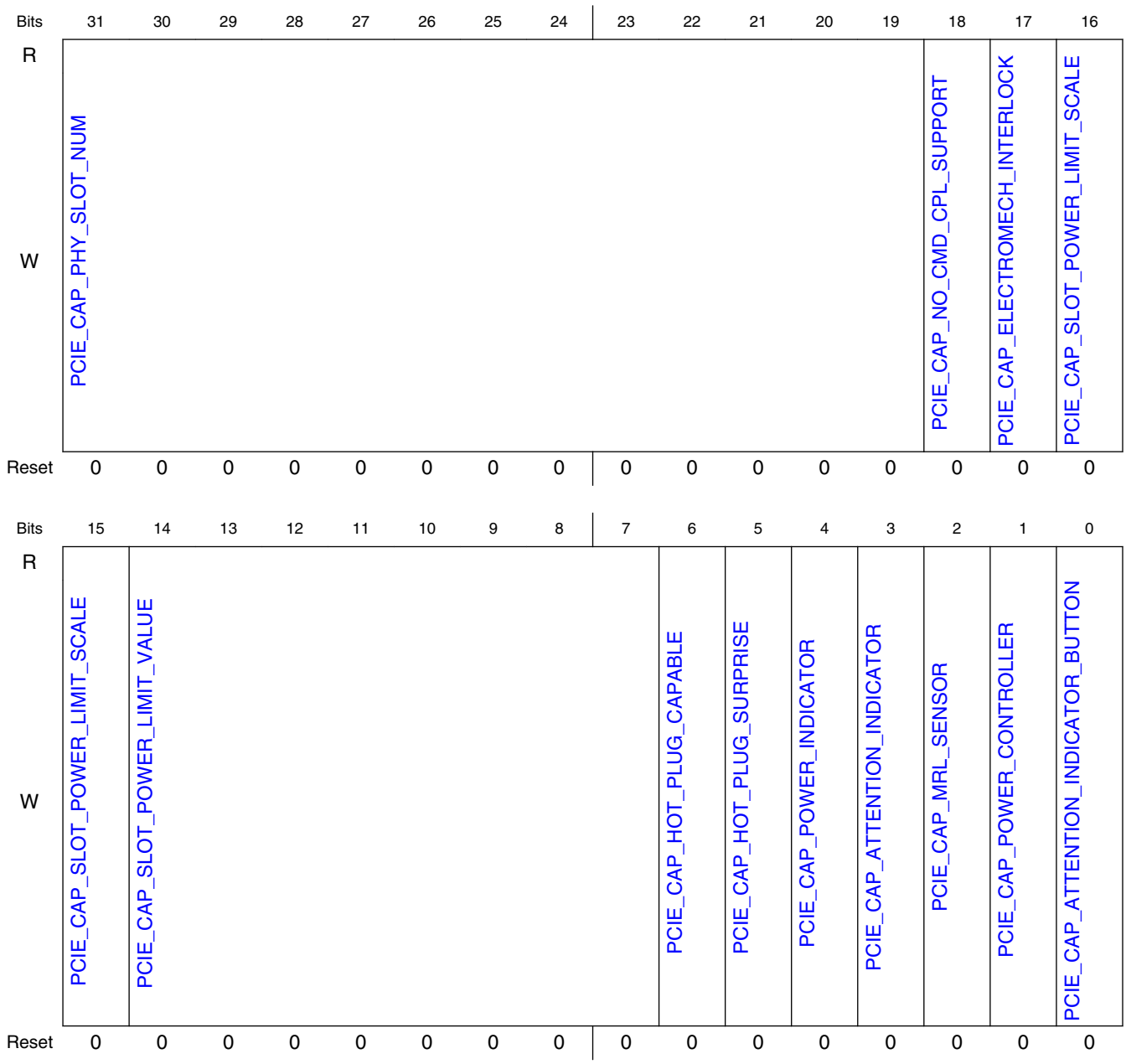
11.3.5.1.29.1 Offset

Register	Offset
SLOT_CAPABILITIES_REG	84h

11.3.5.1.29.2 Function

Slot Capabilities Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.29.3 Diagram



11.3.5.1.29.4 Fields

Field	Function
31-19 PCIE_CAP_PHY_SLOT_NUM	Physical Slot Number. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R

Table continues on the next page...

Field	Function
18 PCIE_CAP_NO_CMD_CPL_SUPPORT	No Command Completed Support. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
17 PCIE_CAP_ELECTROMECH_INTERLOCK	Electromechanical Interlock Present. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
16-15 PCIE_CAP_SLOT_POWER_LIMIT_SCALE	Slot Power Limit Scale. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
14-7 PCIE_CAP_SLOT_POWER_LIMIT_VALUE	Slot Power Limit Value. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
6 PCIE_CAP_HOT_PLUG_CAPABLE	Hot Plug Capable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
5 PCIE_CAP_HOT_PLUG_SURPRISE	Hot Plug Surprise possible. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
4 PCIE_CAP_POWER_INDICATOR	Power Indicator Present. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
3 PCIE_CAP_ATTENTION_INDICATOR	Attention Indicator Present. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
2 PCIE_CAP_MRL_SENSOR	MRL Present. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
1 PCIE_CAP_POWER_CONTROLLER	Power Controller Present. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R
0 PCIE_CAP_ATTENTION_INDICATOR_BUTTON	Attention Button Present. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W else R

11.3.5.1.30 Slot Control and Status Register. (SLOT_CONTROL_SLOT_STATUS)

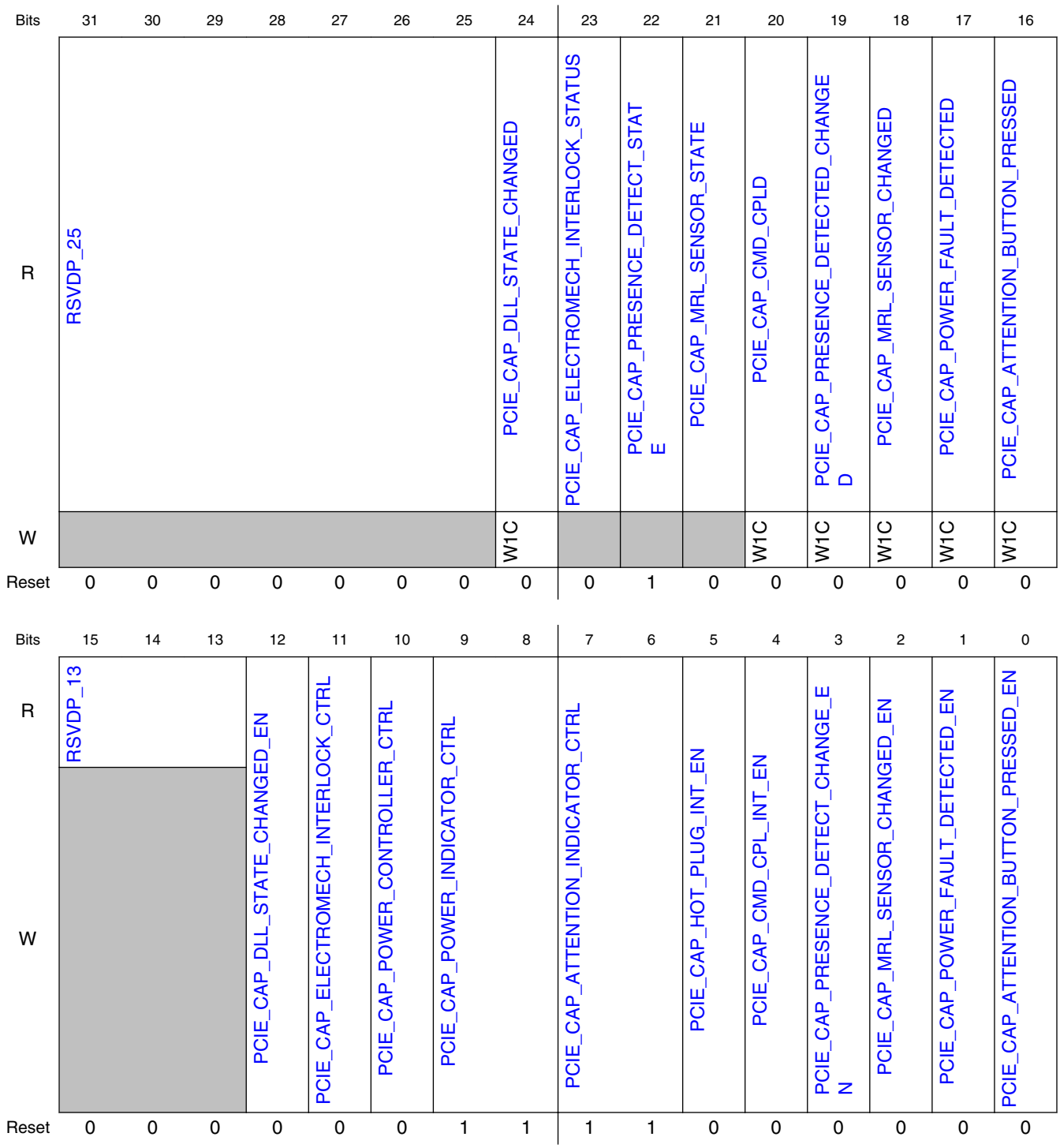
11.3.5.1.30.1 Offset

Register	Offset
SLOT_CONTROL_SLOT_STATUS	88h

11.3.5.1.30.2 Function

Slot Control and Status Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.30.3 Diagram



11.3.5.1.30.4 Fields

Field	Function
31-25	Reserved for future use.

Table continues on the next page...

Field	Function
RSVDP_25	
24 PCIE_CAP_DLL_STATE_CHANGED	Data Link Layer State Changed. For a description of this standard PCIe register field, see the PCI Express Specification.
23 PCIE_CAP_ELECTROMECH_INTERLOCK_STATUS	Electromechanical Interlock Status. For a description of this standard PCIe register field, see the PCI Express Specification.
22 PCIE_CAP_PRESENCE_DETECT_STATE	Presence Detect State. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R
21 PCIE_CAP_MRL_SENSOR_STATE	MRL Sensor State. For a description of this standard PCIe register field, see the PCI Express Specification.
20 PCIE_CAP_CMD_CPLD	Command Completed. For a description of this standard PCIe register field, see the PCI Express Specification.
19 PCIE_CAP_PRESENCE_DETECTED_CHANGED	Presence Detect Changed. For a description of this standard PCIe register field, see the PCI Express Specification.
18 PCIE_CAP_MRL_SENSOR_CHANGED	MRL Sensor Changed. For a description of this standard PCIe register field, see the PCI Express Specification.
17 PCIE_CAP_POWER_FAULT_DETECTED	Power Fault Detected. For a description of this standard PCIe register field, see the PCI Express Specification.
16 PCIE_CAP_ATTENTION_BUTTON_PRESSED	Attention Button Pressed. For a description of this standard PCIe register field, see the PCI Express Specification.
15-13 RSVDP_13	Reserved for future use.
12 PCIE_CAP_DLL_STATE_CHANGED_EN	Data Link Layer State Changed Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
11	Electromechanical Interlock Control. For a description of this standard PCIe register field, see the PCI Express Specification.

Table continues on the next page...

Field	Function
PCIE_CAP_ELECTROMECH_INTERLOCK_CTRL	
10 PCIE_CAP_POWER_CONTROLLER_CTRL	Power Controller Control. For a description of this standard PCIe register field, see the PCI Express Specification.
9-8 PCIE_CAP_POWER_INDICATOR_CTRL	Power Indicator Control. For a description of this standard PCIe register field, see the PCI Express Specification.
7-6 PCIE_CAP_ATTENTION_INDICATOR_CTRL	Attention Indicator Control. For a description of this standard PCIe register field, see the PCI Express Specification.
5 PCIE_CAP_HOT_PLUG_INT_EN	Hot Plug Interrupt Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
4 PCIE_CAP_CMD_CPL_INT_EN	Command Completed Interrupt Enable. For a description of this standard PCIe register field, see the PCI Express Specification. Write value is gated with PCIE_CAP_NO_CMD_CPL_SUPPORT field in SLOT_CAPABILITIES_REG. Note: The access attributes of this field are as follows: - Dbi: SLOT_CAPABILITIES_REG.PCIE_CAP_NO_CMD_CPL_SUPPORT ? RO : RW
3 PCIE_CAP_PRESENCE_DETECT_CHANGE_EN	Presence Detect Changed Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
2 PCIE_CAP_MRL_SENSOR_CHANGED_EN	MRL Sensor Changed Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
1 PCIE_CAP_POWER_FAULT_DETECTED_EN	Power Fault Detected Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
0 PCIE_CAP_ATTENTION_BUTTON_PRESSED_EN	Attention Button Pressed Enable. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.31 Root Control and Capabilities Register. (ROOT_CONTROL_ROOT_CAPABILITIES_REG)

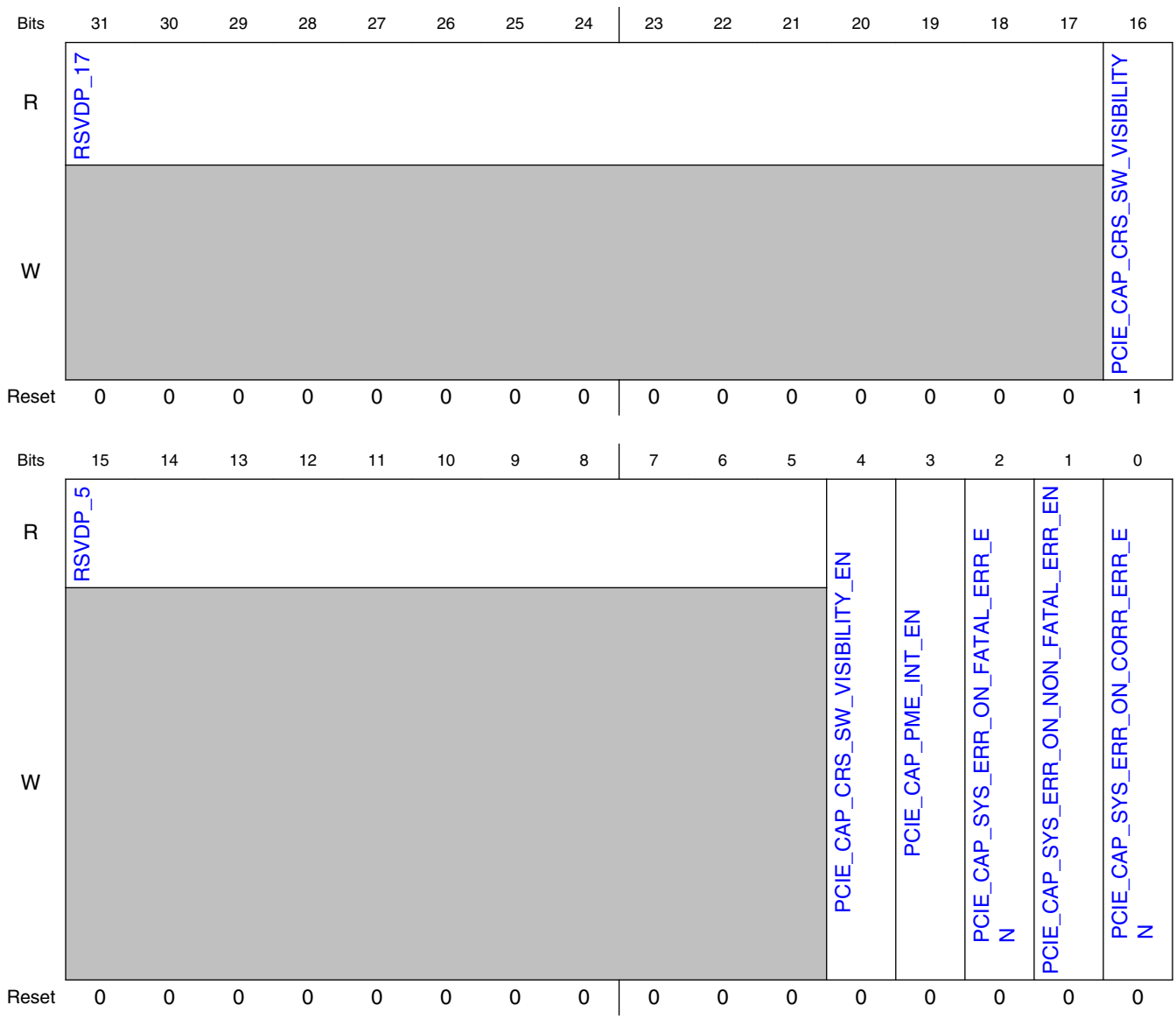
11.3.5.1.31.1 Offset

Register	Offset
ROOT_CONTROL_ROOT_CAPABILITIES_REG	8Ch

11.3.5.1.31.2 Function

Root Control and Capabilities Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.31.3 Diagram



11.3.5.1.31.4 Fields

Field	Function
31-17 RSVDP_17	Reserved for future use.
16 PCIE_CAP_CRS_SW_VISIBILITY	CRS Software Visibility Capable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W (Sticky) else R (Sticky) Note: This register field is sticky.
15-5 RSVDP_5	Reserved for future use.
4 PCIE_CAP_CRS_SW_VISIBILITY_EN	Configuration Request Retry Status (CRS) Software Visibility Enable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: ROOT_CONTROL_ROOT_CAPABILITIES_REG.PCIE_CAP_CRS_SW_VISIBILITY ? RW : RO
3 PCIE_CAP_PME_INT_EN	PME Interrupt Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
2 PCIE_CAP_SYS_ERR_ON_FATAL_ERR_EN	System Error on Fatal Error Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
1 PCIE_CAP_SYS_ERR_ON_NON_FATAL_ERR_EN	System Error on Non-fatal Error Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
0 PCIE_CAP_SYS_ERR_ON_CORR_ERR_EN	System Error on Correctable Error Enable. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.32 Root Status Register. (ROOT_STATUS_REG)

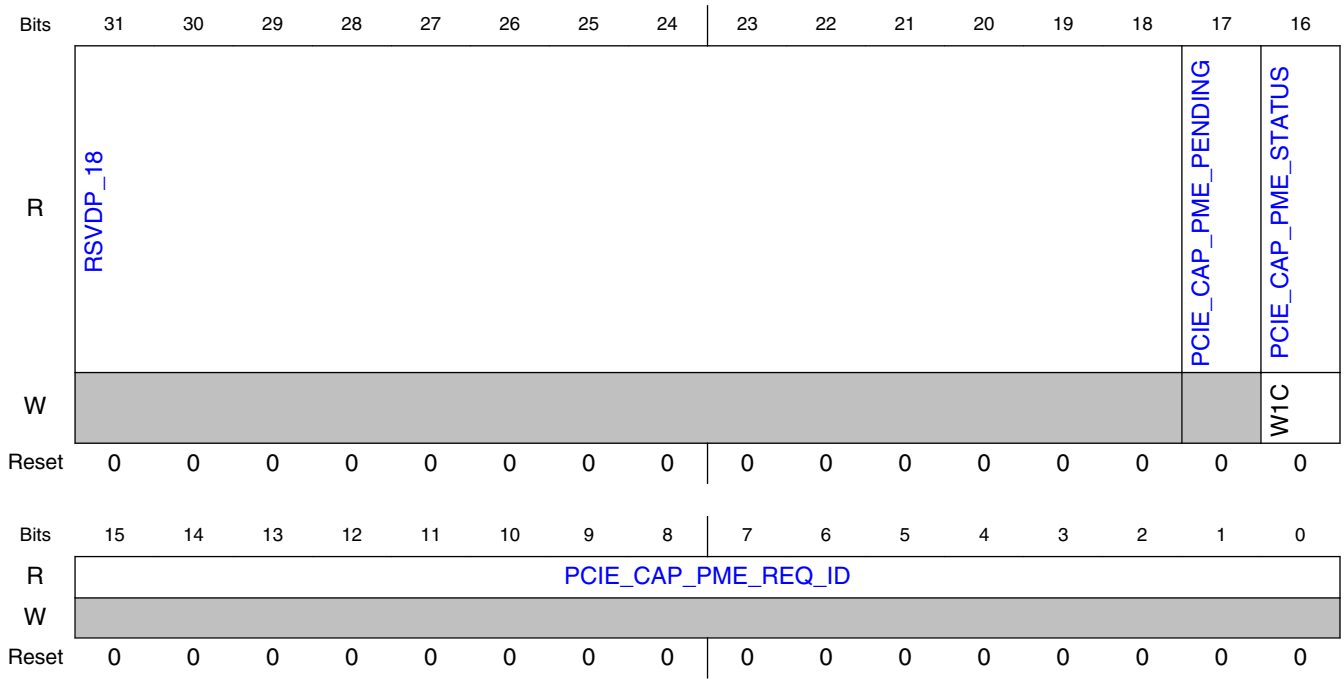
11.3.5.1.32.1 Offset

Register	Offset
ROOT_STATUS_REG	90h

11.3.5.1.32.2 Function

Root Status Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.32.3 Diagram



11.3.5.1.32.4 Fields

Field	Function
31-18 RSVDP_18	Reserved for future use.
17 PCIE_CAP_PME_PENDING	PME Pending. For a description of this standard PCIe register field, see the PCI Express Specification.
16 PCIE_CAP_PME_STATUS	PME Status. For a description of this standard PCIe register field, see the PCI Express Specification.
15-0 PCIE_CAP_PME_REQ_ID	PME Requester ID. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.33 Device Capabilities 2 Register. (DEVICE_CAPABILITIES2_REG)

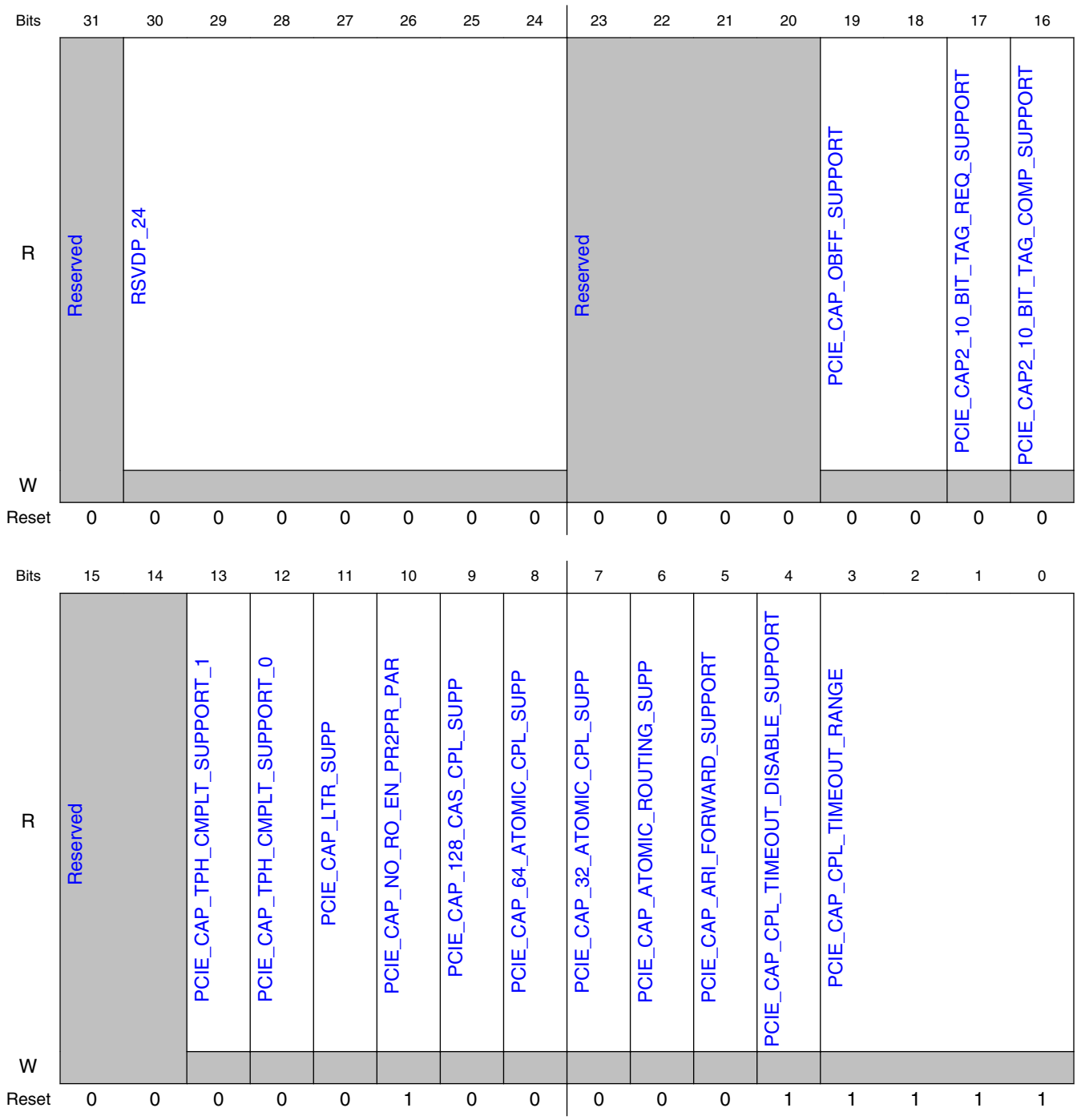
11.3.5.1.33.1 Offset

Register	Offset
DEVICE_CAPABILITIES2_REG	94h

11.3.5.1.33.2 Function

Device Capabilities 2 Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.33.3 Diagram



11.3.5.1.33.4 Fields

Field	Function
31	Reserved.

Table continues on the next page...

Field	Function
—	
30-24 RSVDP_24	Reserved for future use.
23-20 —	Reserved.
19-18 PCIE_CAP_OBFF_SUPPORT	(OBFF) Optimized Buffer Flush/fill Supported. For a description of this standard PCIe register field, see the PCI Express Specification.
17 PCIE_CAP2_10_BIT_TAG_REQ_SUPPORT	10-Bit Tag Requester Supported. For a description of this standard PCIe register field, see the PCI Express Base Specification 4.0.
16 PCIE_CAP2_10_BIT_TAG_COMP_SUPPORT	10-Bit Tag Completer Supported. For a description of this standard PCIe register field, see the PCI Express Base Specification 4.0.
15-14 —	Reserved.
13 PCIE_CAP_TPH_CMPLT_SUPPORT_1	TPH Completer Supported Bit 1. For a description of this standard PCIe register field, see the PCI Express Specification.
12 PCIE_CAP_TPH_CMPLT_SUPPORT_0	TPH Completer Supported Bit 0. For a description of this standard PCIe register field, see the PCI Express Specification.
11 PCIE_CAP_LTR_SUPP	LTR Mechanism Supported. For a description of this standard PCIe register field, see the PCI Express Specification.
10 PCIE_CAP_NO_RO_EN_PR2PR_PAR	No Relaxed Ordering Enabled PR-PR Passing. For a description of this standard PCIe register field, see the PCI Express Specification.
9 PCIE_CAP_128_CAS_CPL_SUPP	128 Bit CAS Completer Supported. For a description of this standard PCIe register field, see the PCI Express Specification.
8 PCIE_CAP_64_ATOMIC_CPL_SUPP	64 Bit AtomicOp Completer Supported. For a description of this standard PCIe register field, see the PCI Express Specification.
7 PCIE_CAP_32_ATOMIC_CPL_SUPP	32 Bit AtomicOp Completer Supported. For a description of this standard PCIe register field, see the PCI Express Specification.

Table continues on the next page...

Field	Function
6 PCIE_CAP_ATOMIC_ROUTING_SUPP	Atomic Operation Routing Supported. For a description of this standard PCIe register field, see the PCI Express Specification.
5 PCIE_CAP_ARI_FORWARD_SUPPORT	ARI Forwarding Supported. For a description of this standard PCIe register field, see the PCI Express Specification.
4 PCIE_CAP_CPL_TIMEOUT_DISABLE_SUPPORT	Completion Timeout Disable Supported. For a description of this standard PCIe register field, see the PCI Express Specification.
3-0 PCIE_CAP_CPL_TIMEOUT_RANGE	Completion Timeout Ranges Supported. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.34 Device Control 2 and Status 2 Register. (DEVICE_CONTROL2_DEVICE_STATUS2_REG)

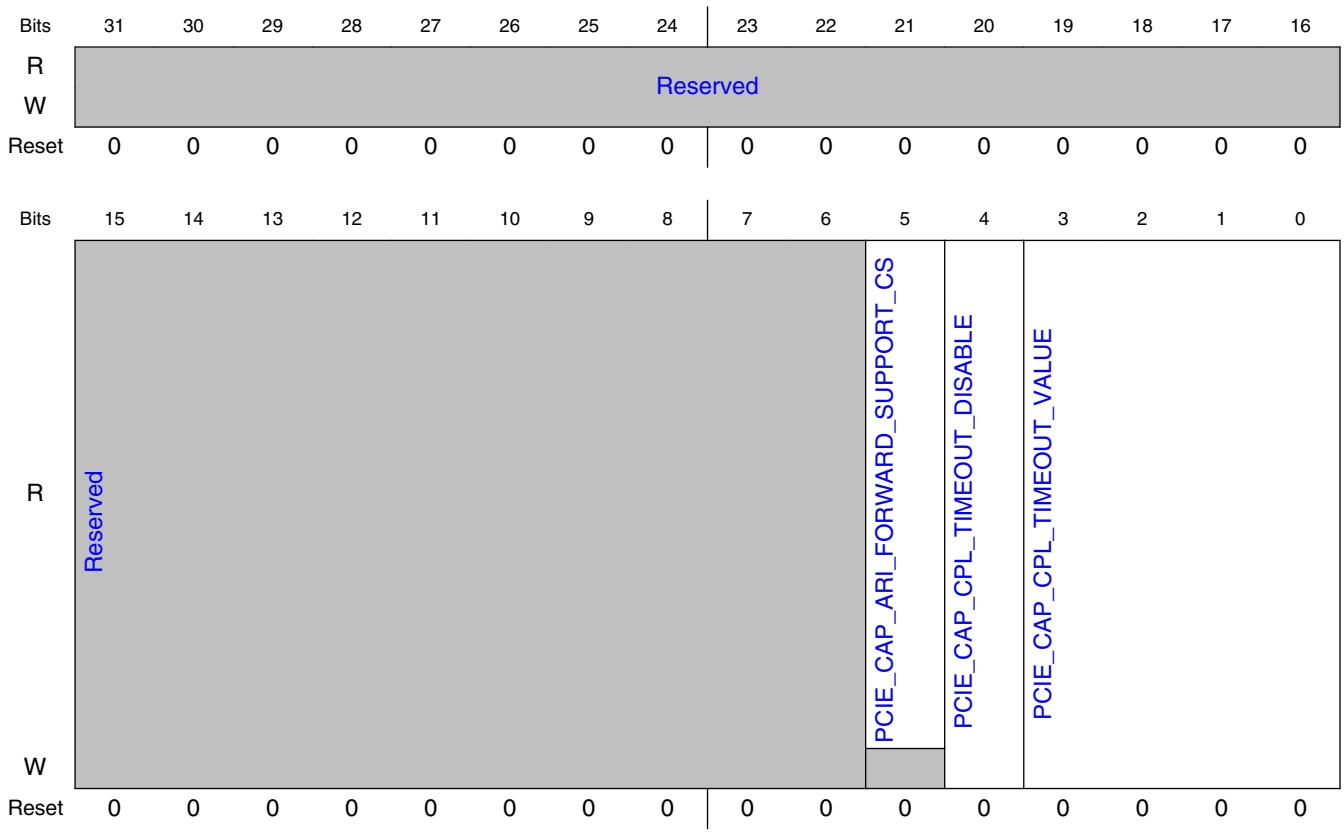
11.3.5.1.34.1 Offset

Register	Offset
DEVICE_CONTROL2_DEVICE_STATUS2_REG	98h

11.3.5.1.34.2 Function

Device Control 2 and Status 2 Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.34.3 Diagram



11.3.5.1.34.4 Fields

Field	Function
31-6 —	Reserved.
5 PCIE_CAP_ARI_FORWARD_SUPPORT_CS	ARI Forwarding Enable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W
4 PCIE_CAP_CPL_TIMEOUT_DISABLE	Completion Timeout Disable. For a description of this standard PCIe register field, see the PCI Express Specification.
3-0 PCIE_CAP_CPL_TIMEOUT_VALUE	Completion Timeout Value. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.35 Link Capabilities 2 Register. (LINK_CAPABILITIES2_REG)

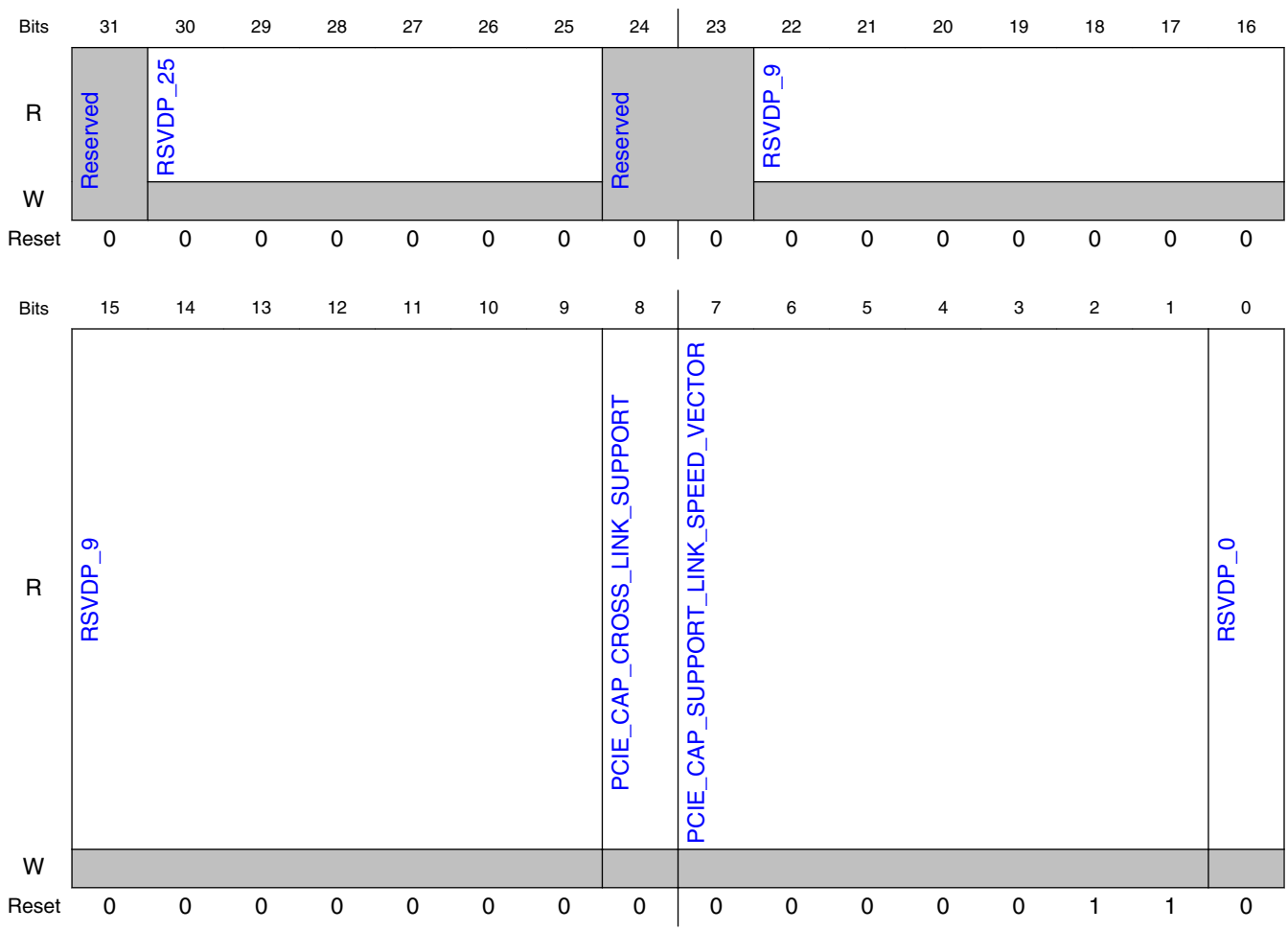
11.3.5.1.35.1 Offset

Register	Offset
LINK_CAPABILITIES2_REG	9Ch

11.3.5.1.35.2 Function

Link Capabilities 2 Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.35.3 Diagram



11.3.5.1.35.4 Fields

Field	Function
31 —	Reserved.
30-25 RSVDP_25	Reserved for future use.
24-23 —	Reserved.
22-9 RSVDP_9	Reserved for future use.
8 PCIE_CAP_CROSS_LINK_SUPPORT	Cross Link Supported. For a description of this standard PCIe register field, see the PCI Express Specification.
7-1 PCIE_CAP_SUPPORT_LINK_SPEED_VECTOR	Supported Link Speeds Vector. For a description of this standard PCIe register field, see the PCI Express Specification. This field has a default of (PCIE_CAP_MAX_LINK_SPEED == 0100) ? 0001111 : (PCIE_CAP_MAX_LINK_SPEED == 0011) ? 0000111 : (PCIE_CAP_MAX_LINK_SPEED == 0010) ? 0000011 : 0000001 where PCIE_CAP_MAX_LINK_SPEED is a field in the LINK_CAPABILITIES_REG register.
0 RSVDP_0	Reserved for future use.

11.3.5.1.36 Link Control 2 and Status 2 Register. (LINK_CONTROL2_LINK_STATUS2_REG)

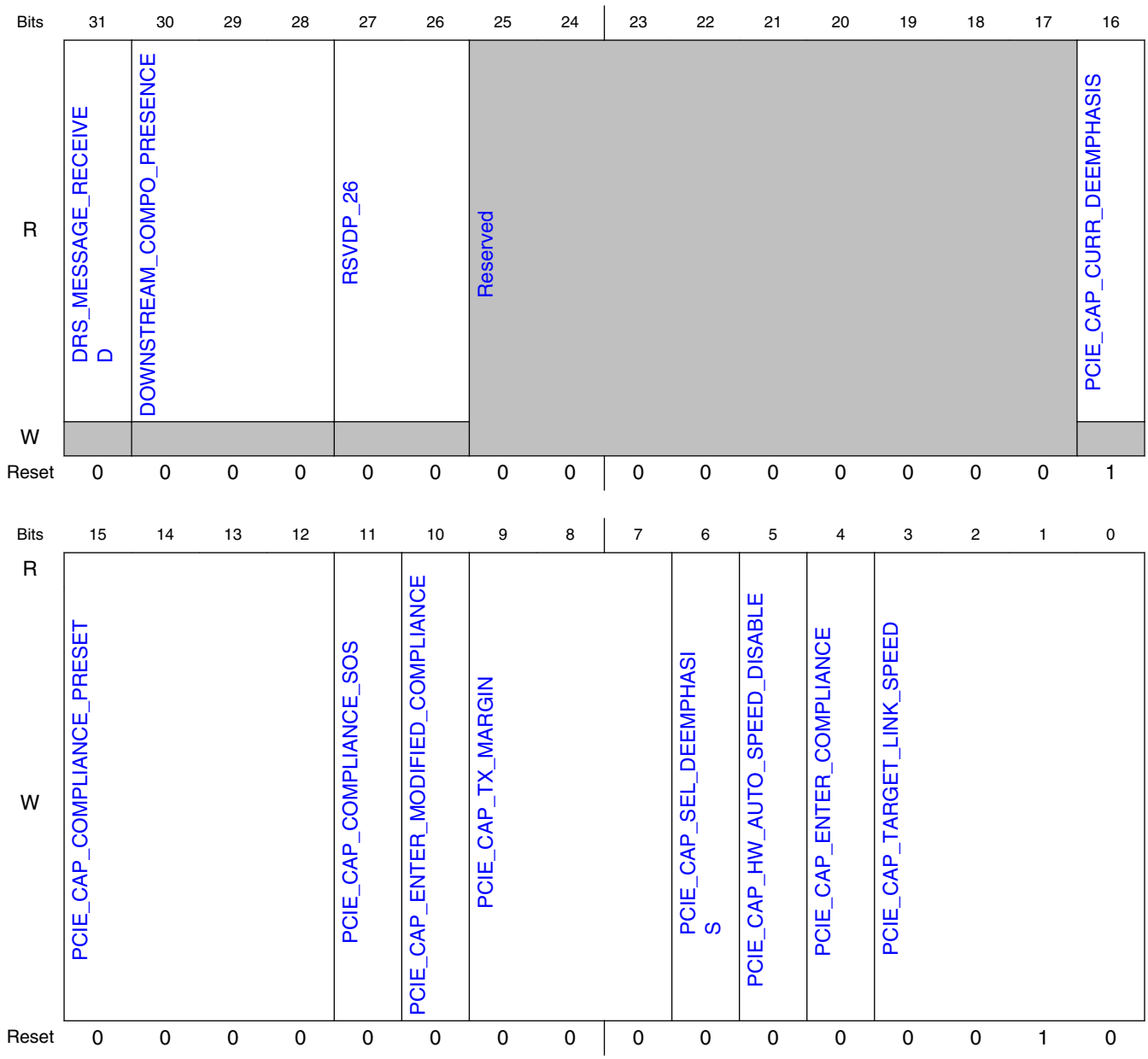
11.3.5.1.36.1 Offset

Register	Offset
LINK_CONTROL2_LINK_STATUS2_REG	A0h

11.3.5.1.36.2 Function

Link Control 2 and Status 2 Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.36.3 Diagram



11.3.5.1.36.4 Fields

Field	Function
31 DRS_MESSAGE_RECEIVED	DRS Message Received. For a description of this standard PCIe register field, see the PCI Express Base Specification 4.0. Note: The access attributes of this field are as follows: - Dbi: RW1C
30-28	Downstream Component Presence. For a description of this standard PCIe register field, see the PCI Express Base Specification 4.0.

Table continues on the next page...

Field	Function
DOWNSTREAM_COMPO_PRESENCE	
27-26 RSVDP_26	Reserved for future use.
25-17 —	Reserved.
16 PCIE_CAP_CURR_DEEMPHASIS	Current De-emphasis Level. For a description of this standard PCIe register field, see the PCI Express Specification. In M-PCIe mode this register is always 0x0. In C-PCIe mode, its contents are derived by sampling the PIPE
15-12 PCIE_CAP_COMPLIANCE_PRESET	Sets Compliance Preset/De-emphasis for 5 GT/s and 8 GT/s. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
11 PCIE_CAP_COMPLIANCE_SOS	Sets Compliance Skip Ordered Sets transmission. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
10 PCIE_CAP_ENTER_MODIFIED_COMPLIANCE	Enter Modified Compliance. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
9-7 PCIE_CAP_TX_MARGIN	Controls Transmit Margin for Debug or Compliance. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
6 PCIE_CAP_SEL_DEEMPHASIS	Controls Selectable De-emphasis for 5 GT/s. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
5 PCIE_CAP_HW_AUTO_SPEED_DISABLE	Hardware Autonomous Speed Disable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
4 PCIE_CAP_ENTER_COMPLIANCE	Enter Compliance Mode. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
3-0 PCIE_CAP_TARGET_LINK_SPEED	Target Link Speed. For a description of this standard PCIe register field, see the PCI Express Specification. In M-PCIe mode, the contents of this field are derived from other registers. Note: This register field is sticky.

11.3.5.1.37 Advanced Error Reporting Extended Capability Header. (AER_EXT_CAP_HDR_OFF)

11.3.5.1.37.1 Offset

Register	Offset
AER_EXT_CAP_HDR_OFF	100h

11.3.5.1.37.2 Function

Advanced Error Reporting Extended Capability Header. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.37.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NEXT_OFFSET												CAP_VERSION			
Reset	0	0	0	1	0	1	0	0	1	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CAP_ID															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

11.3.5.1.37.4 Fields

Field	Function
31-20 NEXT_OFFSET	Next Capability Offset. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
19-16 CAP_VERSION	Capability Version. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
15-0 CAP_ID	AER Extended Capability ID. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

11.3.5.1.38 Uncorrectable Error Status Register. (UNCORR_ERR_STATUS_OFF)

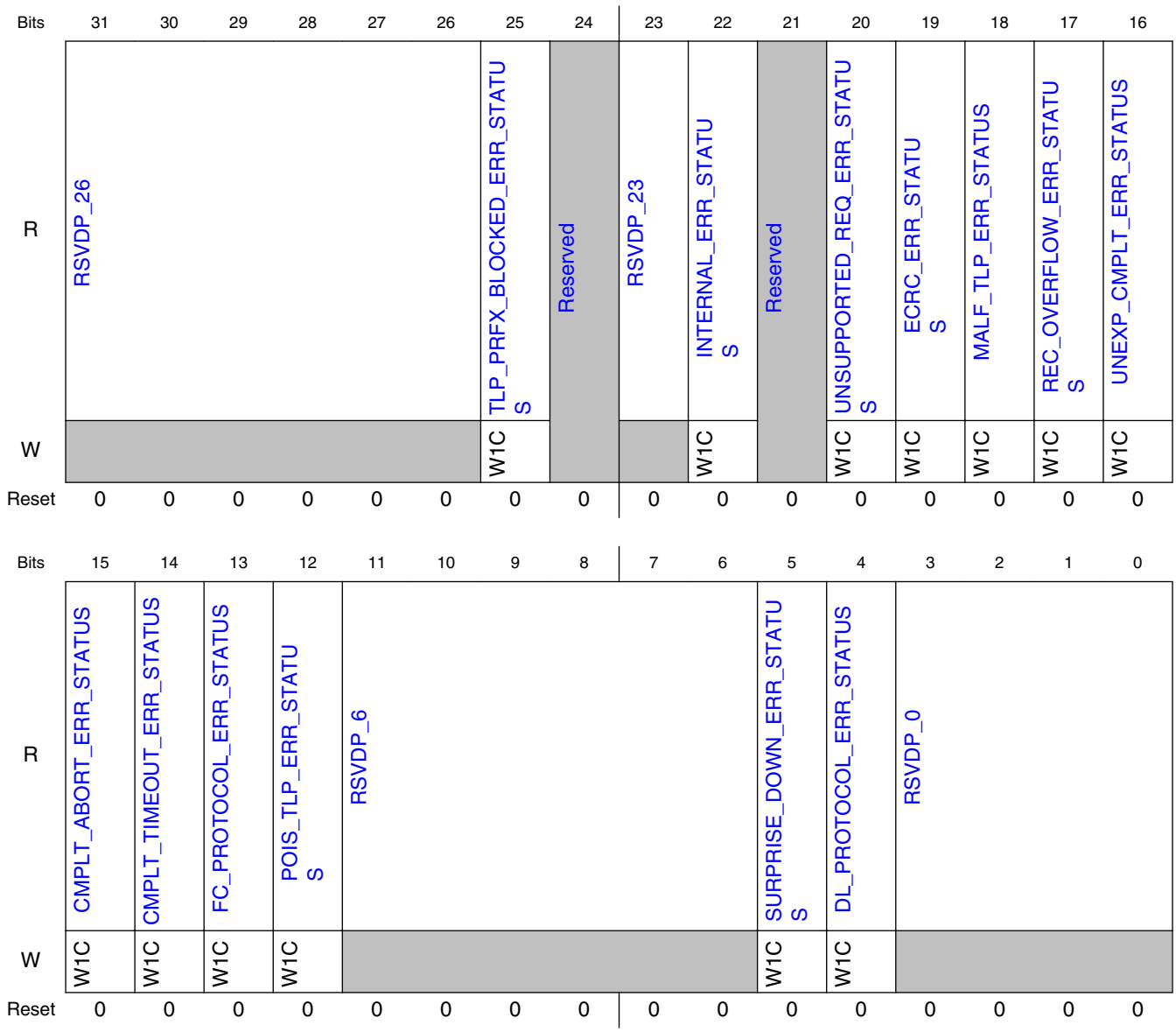
11.3.5.1.38.1 Offset

Register	Offset
UNCORR_ERR_STATUS_OFF	104h

11.3.5.1.38.2 Function

Uncorrectable Error Status Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.38.3 Diagram



11.3.5.1.38.4 Fields

Field	Function
31-26 RSVDP_26	Reserved for future use.
25 TLP_PRFX_BLOCKED_ERR_STATUS	TLP Prefix Blocked Error Status. For a description of this standard PCIe register field, see the PCI Express Specification. Note: Not supported.
24	Reserved.

Table continues on the next page...

Field	Function
—	
23 RSVDP_23	Reserved for future use.
22 INTERNAL_ERR_STATUS	Uncorrectable Internal Error Status. For a description of this standard PCIe register field, see the PCI Express Specification. The controller sets this bit when your application asserts app_err_bus[9]. It does not set this bit when it detects internal uncorrectable internal errors such as parity and ECC failures. You should use the outputs from these errors to drive the app_err_bus[9] input. For more details, see the "Data Integrity (Wire, Datapath, and RAM Protection)" section in the Databook.
21 —	Reserved.
20 UNSUPPORTED_REQ_ERR_STATUS	Unsupported Request Error Status. For a description of this standard PCIe register field, see the PCI Express Specification.
19 ECRC_ERR_STATUS	ECRC Error Status. For a description of this standard PCIe register field, see the PCI Express Specification.
18 MALF_TLP_ERR_STATUS	Malformed TLP Status. For a description of this standard PCIe register field, see the PCI Express Specification.
17 REC_OVERFLOW_ERR_STATUS	Receiver Overflow Status. For a description of this standard PCIe register field, see the PCI Express Specification.
16 UNEXP_CMPLT_ERR_STATUS	Unexpected Completion Status. For a description of this standard PCIe register field, see the PCI Express Specification.
15 CMPLT_ABORT_ERR_STATUS	Completer Abort Status. For a description of this standard PCIe register field, see the PCI Express Specification.
14 CMPLT_TIMEOUT_ERR_STATUS	Completion Timeout Status. For a description of this standard PCIe register field, see the PCI Express Specification.
13 FC_PROTOCOL_ERR_STATUS	Flow Control Protocol Error Status. For a description of this standard PCIe register field, see the PCI Express Specification.
12 POIS_TLP_ERR_STATUS	Poisoned TLP Status. For a description of this standard PCIe register field, see the PCI Express Specification.
11-6 RSVDP_6	Reserved for future use.
5	Surprise Down Error Status (Optional). For a description of this standard PCIe register field, see the PCI Express Specification.

Table continues on the next page...

Field	Function
SURPRISE_DOWN_ERR_STATUS	
4 DL_PROTOCOL_ERR_STATUS	Data Link Protocol Error Status. For a description of this standard PCIe register field, see the PCI Express Specification.
3-0 RSVDP_0	Reserved for future use.

11.3.5.1.39 Uncorrectable Error Mask Register. (UNCORR_ERR_MASK_OFF)

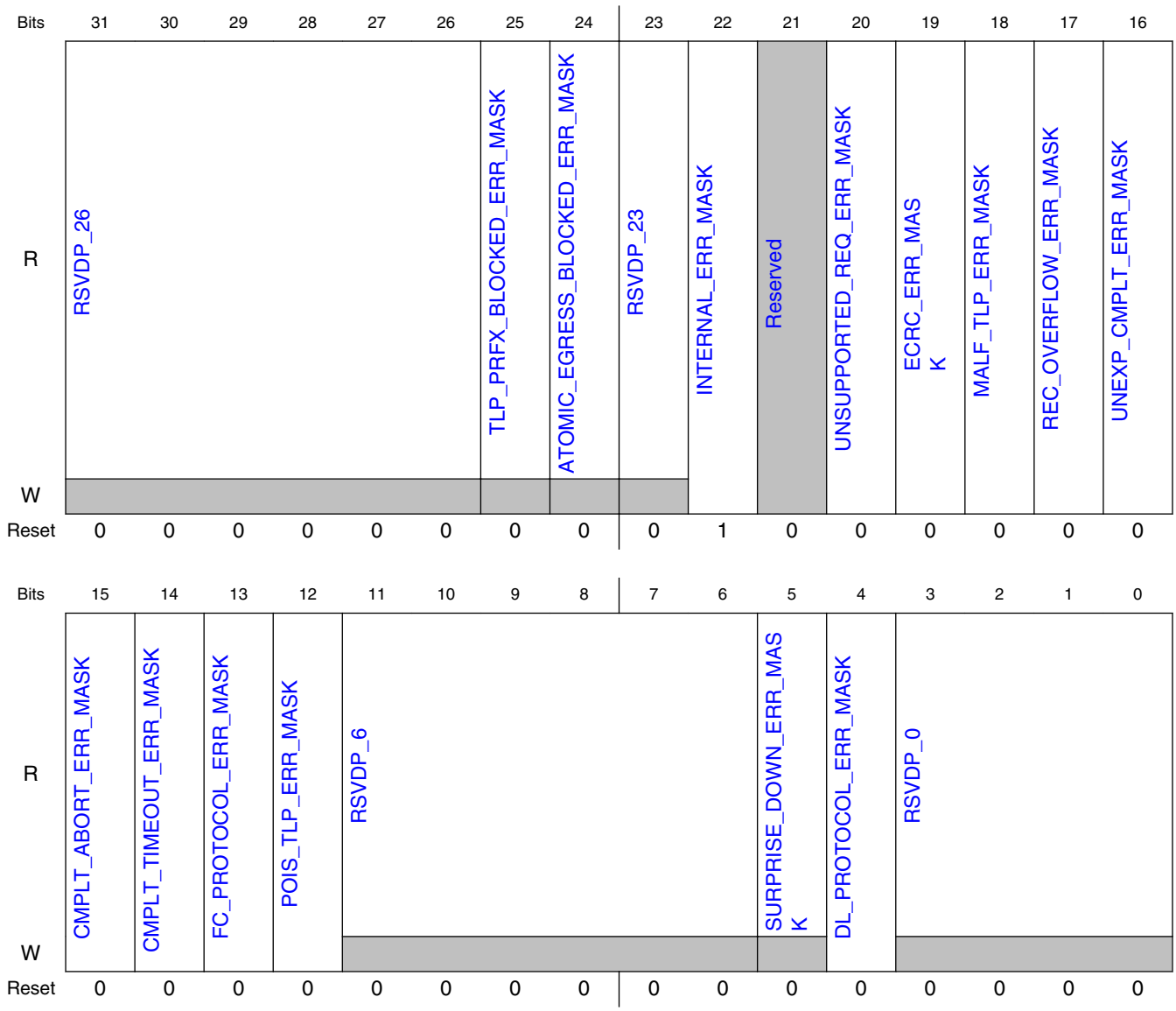
11.3.5.1.39.1 Offset

Register	Offset
UNCORR_ERR_MASK_OFF	108h

11.3.5.1.39.2 Function

Uncorrectable Error Mask Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.39.3 Diagram



11.3.5.1.39.4 Fields

Field	Function
31-26 RSVDP_26	Reserved for future use.
25 TLP_PRFX_BLOCKED_ERR_MASK	TLP Prefix Blocked Error Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: Not supported. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.

Table continues on the next page...

Field	Function
24 ATOMIC_EGRESS_BLOCKED_ERR_MASK	AtomicOp Egress Block Mask (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
23 RSVDP_23	Reserved for future use.
22 INTERNAL_ERR_MASK	Uncorrectable Internal Error Mask (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
21 —	Reserved.
20 UNSUPPORTED_REQ_ERR_MASK	Unsupported Request Error Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
19 ECRC_ERR_MASK	ECRC Error Mask (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
18 MALF_TLP_ERR_MASK	Malformed TLP Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
17 REC_OVERFLOW_ERR_MASK	Receiver Overflow Mask (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
16 UNEXP_CMPLT_ERR_MASK	Unexpected Completion Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
15 CMPLT_ABORT_ERR_MASK	Completer Abort Error Mask (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
14 CMPLT_TIMEOUT_ERR_MASK	Completion Timeout Error Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
13 FC_PROTOCOL_ERR_MASK	Flow Control Protocol Error Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
12 POIS_TLP_ERR_MASK	Poisoned TLP Error Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
11-6 RSVDP_6	Reserved for future use.
5	Surprise Down Error Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi:

Table continues on the next page...

Field	Function
SURPRISE_DOWN_ERR_MASK	LINK_CAPABILITIES_REG.PCIE_CAP_SURPRISE_DOWN_ERR_REP_CAP ? RW : RO Note: This register field is sticky.
4 DL_PROTOCOL_ERR_MASK	Data Link Protocol Error Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
3-0 RSVDP_0	Reserved for future use.

11.3.5.1.40 Uncorrectable Error Severity Register. (UNCORR_ERR_SEV_OFF)

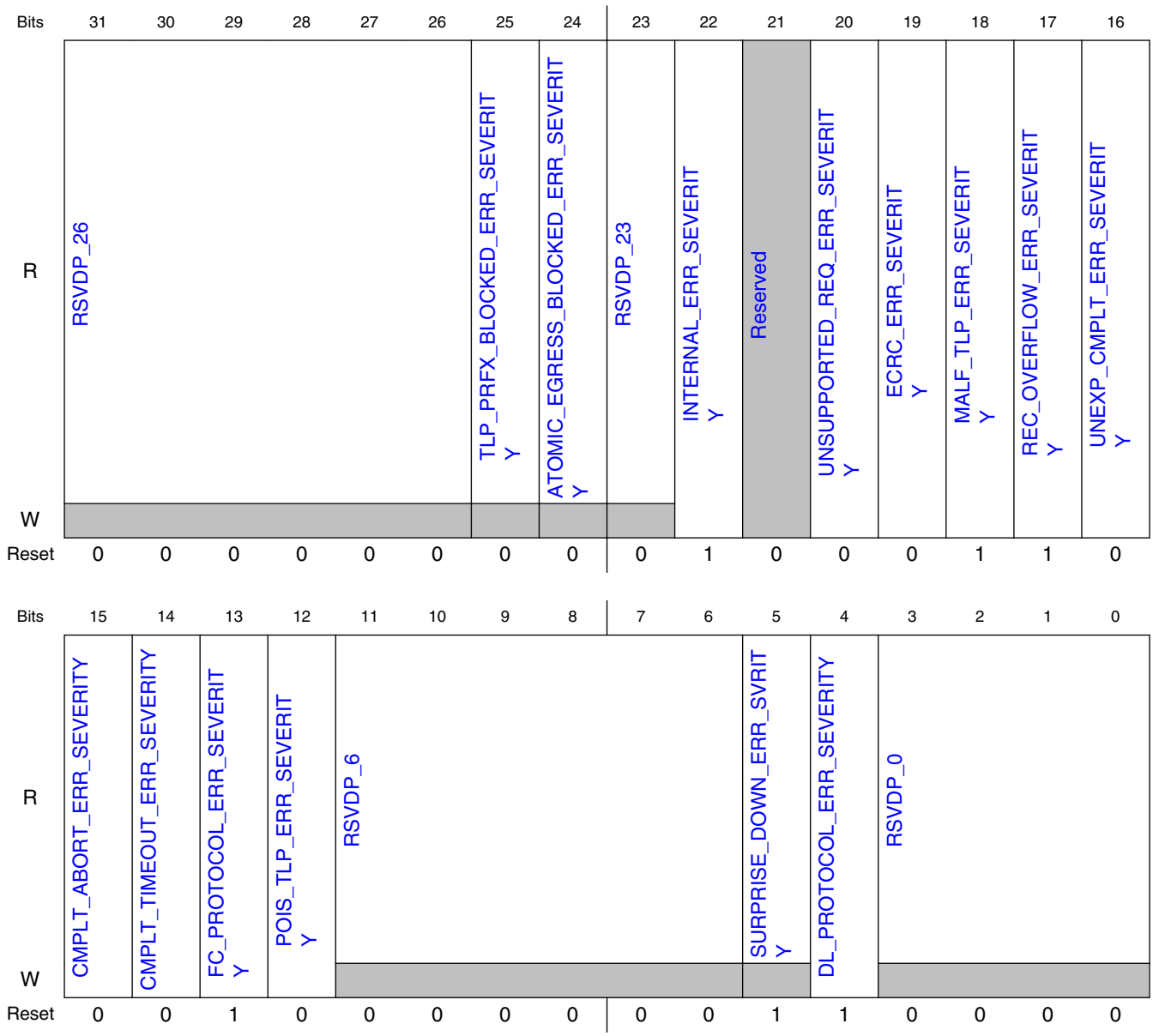
11.3.5.1.40.1 Offset

Register	Offset
UNCORR_ERR_SEV_OFF	10Ch

11.3.5.1.40.2 Function

Uncorrectable Error Severity Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.40.3 Diagram



11.3.5.1.40.4 Fields

Field	Function
31-26 RSVDP_26	Reserved for future use.
25 TLP_PRFX_BLOCKED_ERR_SEVERITY	TLP Prefix Blocked Error Severity (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: Not supported. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.

Table continues on the next page...

Field	Function
24 ATOMIC_EGRESS_BLOCKED_ERR_SEVERITY	AtomicOp Egress Blocked Severity (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
23 RSVDP_23	Reserved for future use.
22 INTERNAL_ERR_SEVERITY	Uncorrectable Internal Error Severity (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
21 —	Reserved.
20 UNSUPPORTED_REQ_ERR_SEVERITY	Unsupported Request Error Severity. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
19 ECRC_ERR_SEVERITY	ECRC Error Severity (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
18 MALF_TLP_ERR_SEVERITY	Malformed TLP Severity. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
17 REC_OVERFLOW_ERR_SEVERITY	Receiver Overflow Error Severity (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
16 UNEXP_CMPLT_ERR_SEVERITY	Unexpected Completion Error Severity. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
15 CMPLT_ABORT_ERR_SEVERITY	Completer Abort Error Severity (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
14 CMPLT_TIMEOUT_ERR_SEVERITY	Completion Timeout Error Severity. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
13 FC_PROTOCOL_ERR_SEVERITY	Flow Control Protocol Error Severity (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
12 POIS_TLP_ERR_SEVERITY	Poisoned TLP Severity. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

Table continues on the next page...

Field	Function
11-6 RSVDP_6	Reserved for future use.
5 SURPRISE_DOWN_ERR_SVRITY	Surprise Down Error Severity (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: LINK_CAPABILITIES_REG.PCIE_CAP_SURPRISE_DOWN_ERR_REP_CAP ? RW : RO Note: This register field is sticky.
4 DL_PROTOCOL_ERR_SEVERITY	Data Link Protocol Error Severity. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
3-0 RSVDP_0	Reserved for future use.

11.3.5.1.41 Correctable Error Status Register. (CORR_ERR_STATUS_OFF)

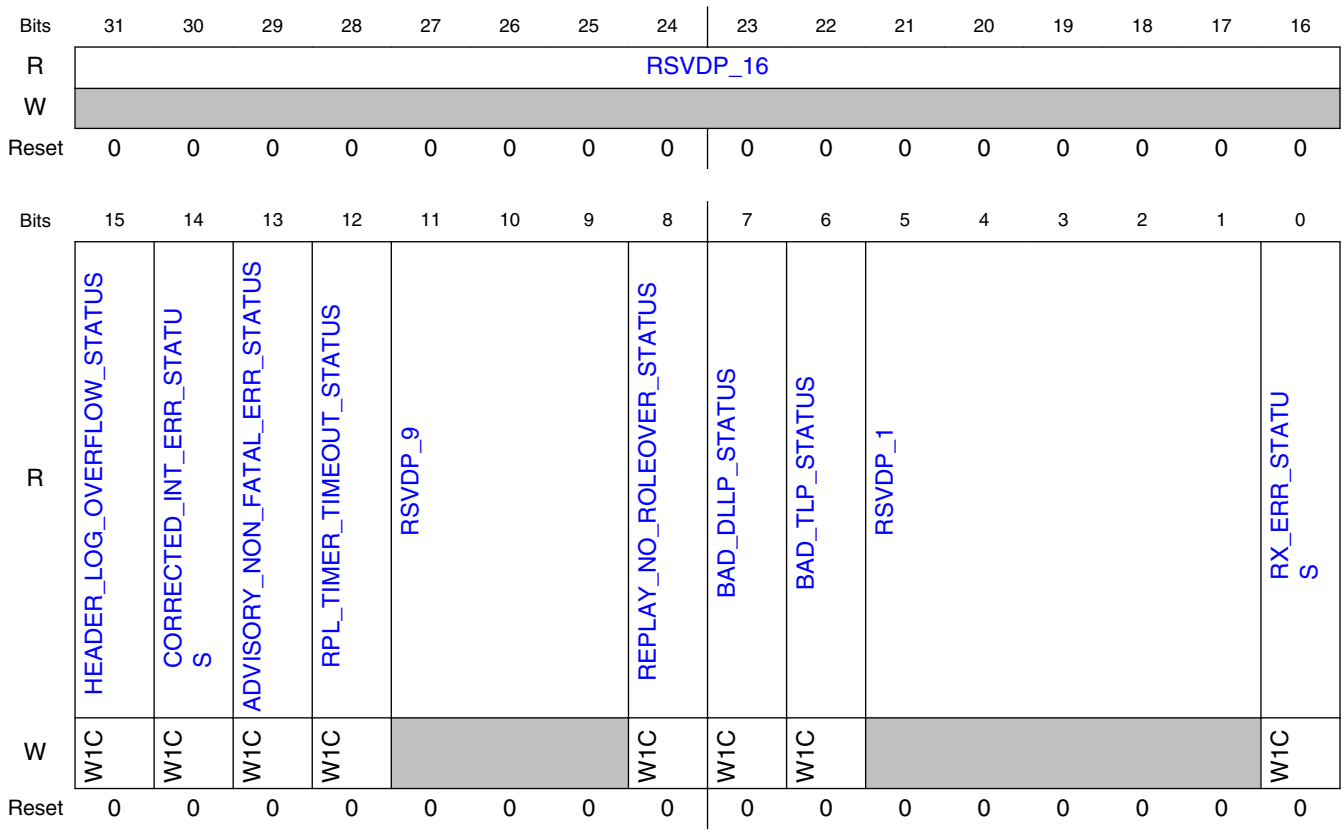
11.3.5.1.41.1 Offset

Register	Offset
CORR_ERR_STATUS_OFF	110h

11.3.5.1.41.2 Function

Correctable Error Status Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.41.3 Diagram



11.3.5.1.41.4 Fields

Field	Function
31-16 RSVDP_16	Reserved for future use.
15 HEADER_LOG_OVERFLOW_STATUS	Header Log Overflow Error Status (Optional). For a description of this standard PCIe register field, see the PCI Express Specification.
14 CORRECTED_INT_ERR_STATUS	Corrected Internal Error Status (Optional). For a description of this standard PCIe register field, see the PCI Express Specification.
13 ADVISORY_NON_FATAL_ERR_STATUS	Advisory Non-Fatal Error Status. For a description of this standard PCIe register field, see the PCI Express Specification.
12	Replay Timer Timeout Status. For a description of this standard PCIe register field, see the PCI Express Specification.

Table continues on the next page...

Field	Function
RPL_TIMER_TIMEOUT_STATUS	
11-9 RSVDP_9	Reserved for future use.
8 REPLAY_NO_ROLEOVER_STATUS	REPLAY_NUM Rollover Status. For a description of this standard PCIe register field, see the PCI Express Specification.
7 BAD_DLLP_STATUS	Bad DLLP Status. For a description of this standard PCIe register field, see the PCI Express Specification.
6 BAD_TLP_STATUS	Bad TLP Status. For a description of this standard PCIe register field, see the PCI Express Specification.
5-1 RSVDP_1	Reserved for future use.
0 RX_ERR_STATUS	Receiver Error Status (Optional). For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.42 Correctable Error Mask Register. (CORR_ERR_MASK_OFF)

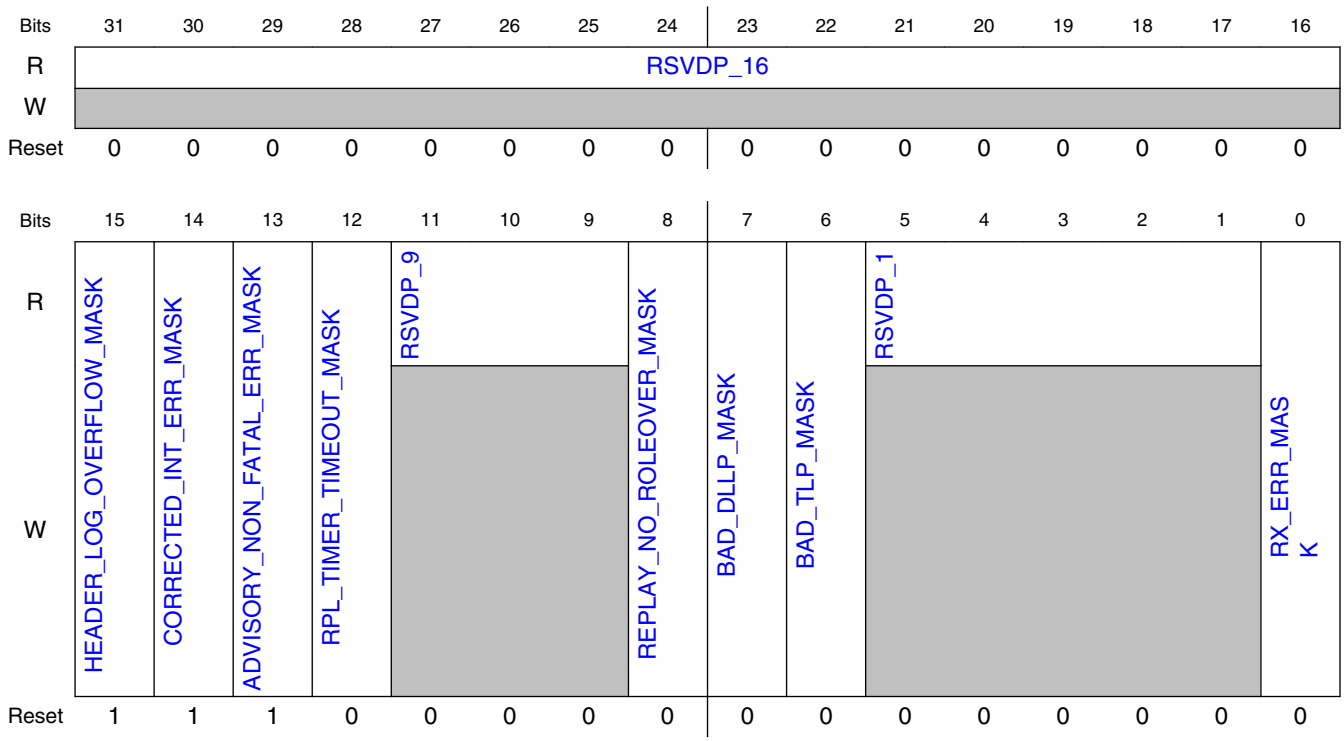
11.3.5.1.42.1 Offset

Register	Offset
CORR_ERR_MASK_OFF	114h

11.3.5.1.42.2 Function

Correctable Error Mask Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.42.3 Diagram



11.3.5.1.42.4 Fields

Field	Function
31-16 RSVDP_16	Reserved for future use.
15 HEADER_LOG_OVERFLOW_MASK	Header Log Overflow Error Mask (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
14 CORRECTED_INT_ERR_MASK	Corrected Internal Error Mask (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
13 ADVISORY_NON_FATAL_ERR_MASK	Advisory Non-Fatal Error Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
12 RPL_TIMER_TIMEOUT_MASK	Replay Timer Timeout Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
11-9 RSVDP_9	Reserved for future use.

Table continues on the next page...

Field	Function
8 REPLAY_NO_ROLEOVER_MASK	REPLAY_NUM Rollover Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
7 BAD_DLLP_MASK	Bad DLLP Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
6 BAD_TLP_MASK	Bad TLP Mask. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
5-1 RSVDP_1	Reserved for future use.
0 RX_ERR_MASK	Receiver Error Mask (Optional). For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

11.3.5.1.43 Advanced Error Capabilities and Control Register. (ADV_ERR_CAP_CTRL_OFF)

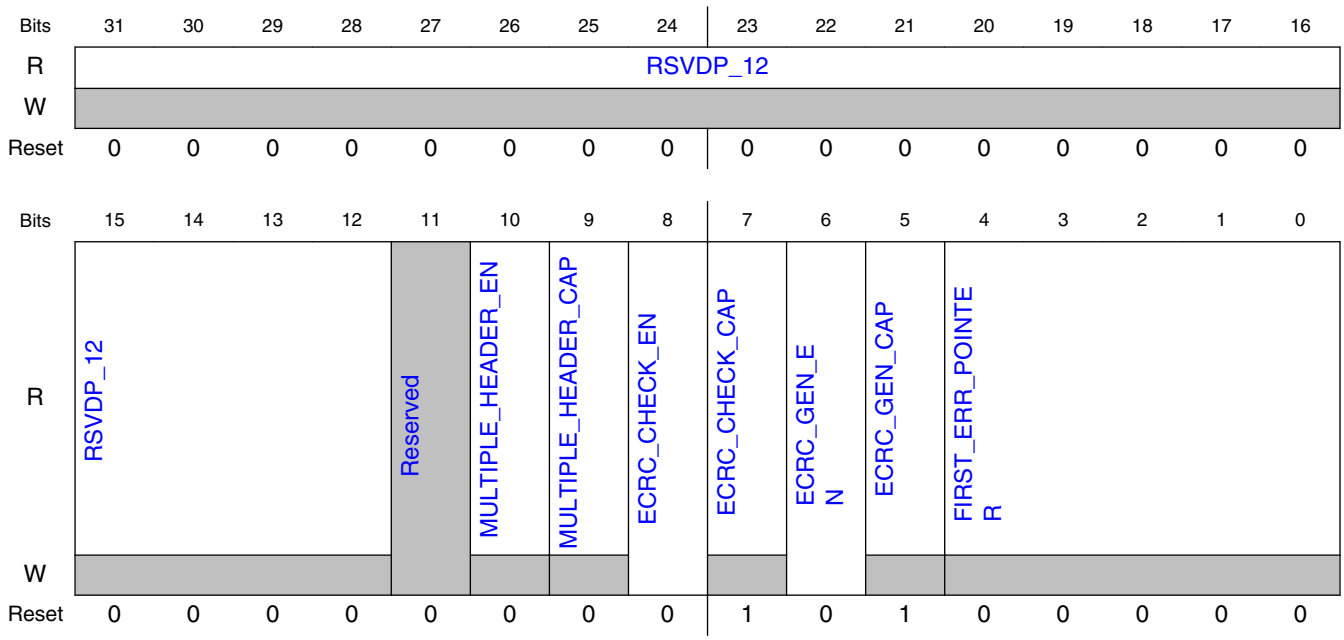
11.3.5.1.43.1 Offset

Register	Offset
ADV_ERR_CAP_CTRL_OFF	118h

11.3.5.1.43.2 Function

Advanced Error Capabilities and Control Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.43.3 Diagram



11.3.5.1.43.4 Fields

Field	Function
31-12 RSVDP_12	Reserved for future use.
11 —	Reserved.
10 MULTIPLE_HEADER_EN	Multiple Header Recording Enable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
9 MULTIPLE_HEADER_CAP	Multiple Header Recording Capable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
8 ECRC_CHECK_EN	ECRC Check Enable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
7 ECRC_CHECK_CAP	ECRC Check Capable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
6 ECRC_GEN_EN	ECRC Generation Enable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
5	ECRC Generation Capable. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

Table continues on the next page...

Field	Function
ECRC_GEN_CAP	
4-0 FIRST_ERR_POINTER	First Error Pointer. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

11.3.5.1.44 Header Log Register 0. (HDR_LOG_0_OFF)

11.3.5.1.44.1 Offset

Register	Offset
HDR_LOG_0_OFF	11Ch

11.3.5.1.44.2 Function

Header Log Register 0. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.44.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FIRST_DWORD_FOURTH_BYTE								FIRST_DWORD_THIRD_BYTE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FIRST_DWORD_SECOND_BYTE								FIRST_DWORD_FIRST_BYTE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.44.4 Fields

Field	Function
31-24 FIRST_DWORD_FOURTH_BYTE	Byte 3 of Header log register of First 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

Table continues on the next page...

Field	Function
23-16 FIRST_DWORD_THIRD_BYTE	Byte 2 of Header log register of First 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
15-8 FIRST_DWORD_SECOND_BYTE	Byte 1 of Header log register of First 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
7-0 FIRST_DWORD_FIRST_BYTE	Byte 0 of Header log register of First 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

11.3.5.1.45 Header Log Register 1. (HDR_LOG_1_OFF)

11.3.5.1.45.1 Offset

Register	Offset
HDR_LOG_1_OFF	120h

11.3.5.1.45.2 Function

Header Log Register 1. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.45.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SECOND_DWORD_FOURTH_BYTE								SECOND_DWORD_THIRD_BYTE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SECOND_DWORD_SECOND_BYTE								SECOND_DWORD_FIRST_BYTE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.45.4 Fields

Field	Function
31-24 SECOND_DWORD_FOURTH_BYTE	Byte 3 of Header log register of Second 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
23-16 SECOND_DWORD_THIRD_BYTE	Byte 2 of Header log register of Second 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
15-8 SECOND_DWORD_SECOND_BYTE	Byte 1 of Header log register of Second 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
7-0 SECOND_DWORD_FIRST_BYTE	Byte 0 of Header log register of Second 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

11.3.5.1.46 Header Log Register 2. (HDR_LOG_2_OFF)

11.3.5.1.46.1 Offset

Register	Offset
HDR_LOG_2_OFF	124h

11.3.5.1.46.2 Function

Header Log Register 2. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.46.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	THIRD_DWORD_FOURTH_BYTE								THIRD_DWORD_THIRD_BYTE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	THIRD_DWORD_SECOND_BYTE								THIRD_DWORD_FIRST_BYTE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.46.4 Fields

Field	Function
31-24 THIRD_DWORD_FOURTH_BYTE	Byte 3 of Header log register of Third 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
23-16 THIRD_DWORD_THIRD_BYTE	Byte 2 of Header log register of Third 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
15-8 THIRD_DWORD_SECOND_BYTE	Byte 1 of Header log register of Third 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
7-0 THIRD_DWORD_FIRST_BYTE	Byte 0 of Header log register of Third 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

11.3.5.1.47 Header Log Register 3. (HDR_LOG_3_OFF)

11.3.5.1.47.1 Offset

Register	Offset
HDR_LOG_3_OFF	128h

11.3.5.1.47.2 Function

Header Log Register 3. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.47.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FOURTH_DWORD_FOURTH_BYTE								FOURTH_DWORD_THIRD_BYTE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FOURTH_DWORD_SECOND_BYTE								FOURTH_DWORD_FIRST_BYTE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.47.4 Fields

Field	Function
31-24 FOURTH_DWORD_FOURTH_BYTE	Byte 3 of Header log register of Fourth 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
23-16 FOURTH_DWORD_THIRD_BYTE	Byte 2 of Header log register of Fourth 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
15-8 FOURTH_DWORD_SECOND_BYTE	Byte 1 of Header log register of Fourth 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
7-0 FOURTH_DWORD_FIRST_BYTE	Byte 0 of Header log register of Fourth 32 bit Data Word. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

11.3.5.1.48 Root Error Command Register. (ROOT_ERR_CMD_OFF)

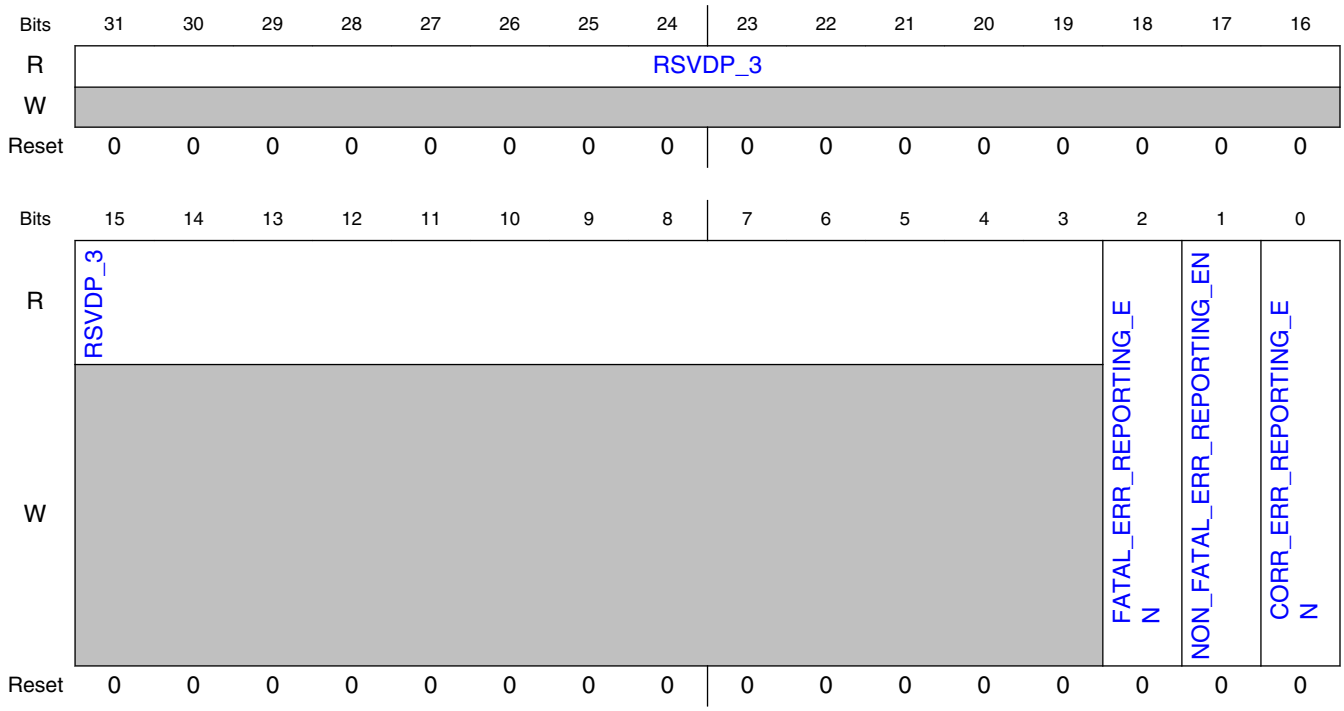
11.3.5.1.48.1 Offset

Register	Offset
ROOT_ERR_CMD_OFF	12Ch

11.3.5.1.48.2 Function

Root Error Command Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.48.3 Diagram



11.3.5.1.48.4 Fields

Field	Function
31-3 RSVDP_3	Reserved for future use.
2 FATAL_ERR_REPORTING_EN	Fatal Error Reporting Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
1 NON_FATAL_ERR_REPORTING_EN	Non-Fatal Error Reporting Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
0 CORR_ERR_REPORTING_EN	Correctable Error Reporting Enable. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.49 Root Error Status Register. (ROOT_ERR_STATUS_OFF)

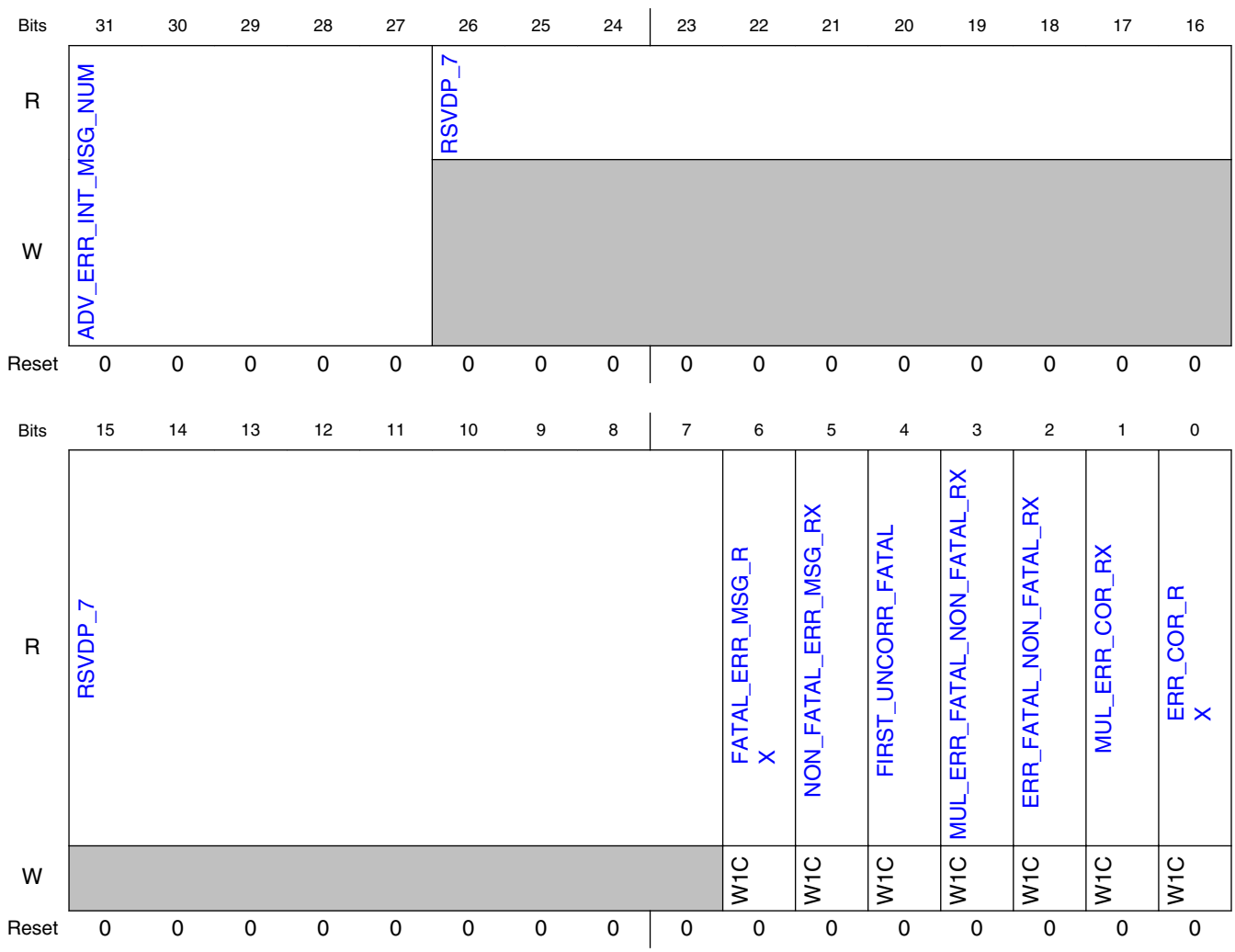
11.3.5.1.49.1 Offset

Register	Offset
ROOT_ERR_STATUS_OFF	130h

11.3.5.1.49.2 Function

Root Error Status Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.49.3 Diagram



11.3.5.1.49.4 Fields

Field	Function
31-27 ADV_ERR_INT_MSG_NUM	Advanced Error Interrupt Message Number. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
26-7 RSVDP_7	Reserved for future use.
6 FATAL_ERR_MSG_RX	One or more Fatal Error Messages Received. For a description of this standard PCIe register field, see the PCI Express Specification.
5 NON_FATAL_ERR_MSG_RX	One or more Non-Fatal Error Messages Received. For a description of this standard PCIe register field, see the PCI Express Specification.
4 FIRST_UNCOR_FATAL	First Uncorrectable Error is Fatal. For a description of this standard PCIe register field, see the PCI Express Specification.
3 MUL_ERR_FATAL_NON_FATAL_RX	Multiple Fatal or Non-Fatal Errors Received. For a description of this standard PCIe register field, see the PCI Express Specification.
2 ERR_FATAL_NON_FATAL_RX	Fatal or Non-Fatal Error Received. For a description of this standard PCIe register field, see the PCI Express Specification.
1 MUL_ERR_COR_RX	Multiple Correctable Errors Received. For a description of this standard PCIe register field, see the PCI Express Specification.
0 ERR_COR_RX	Correctable Error Received. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.50 Error Source Identification Register. (ERR_SRC_ID_OFF)

11.3.5.1.50.1 Offset

Register	Offset
ERR_SRC_ID_OFF	134h

11.3.5.1.50.2 Function

Error Source Identification Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.50.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ERR_FATAL_NON_FATAL_SOURCE_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERR_COR_SOURCE_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.50.4 Fields

Field	Function
31-16 ERR_FATAL_NON_FATAL_SOURCE_ID	Source of Fatal/Non-Fatal Error. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
15-0 ERR_COR_SOURCE_ID	Source of Correctable Error. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

11.3.5.1.51 TLP Prefix Log Register 1. (TLP_PREFIX_LOG_1_OFF)

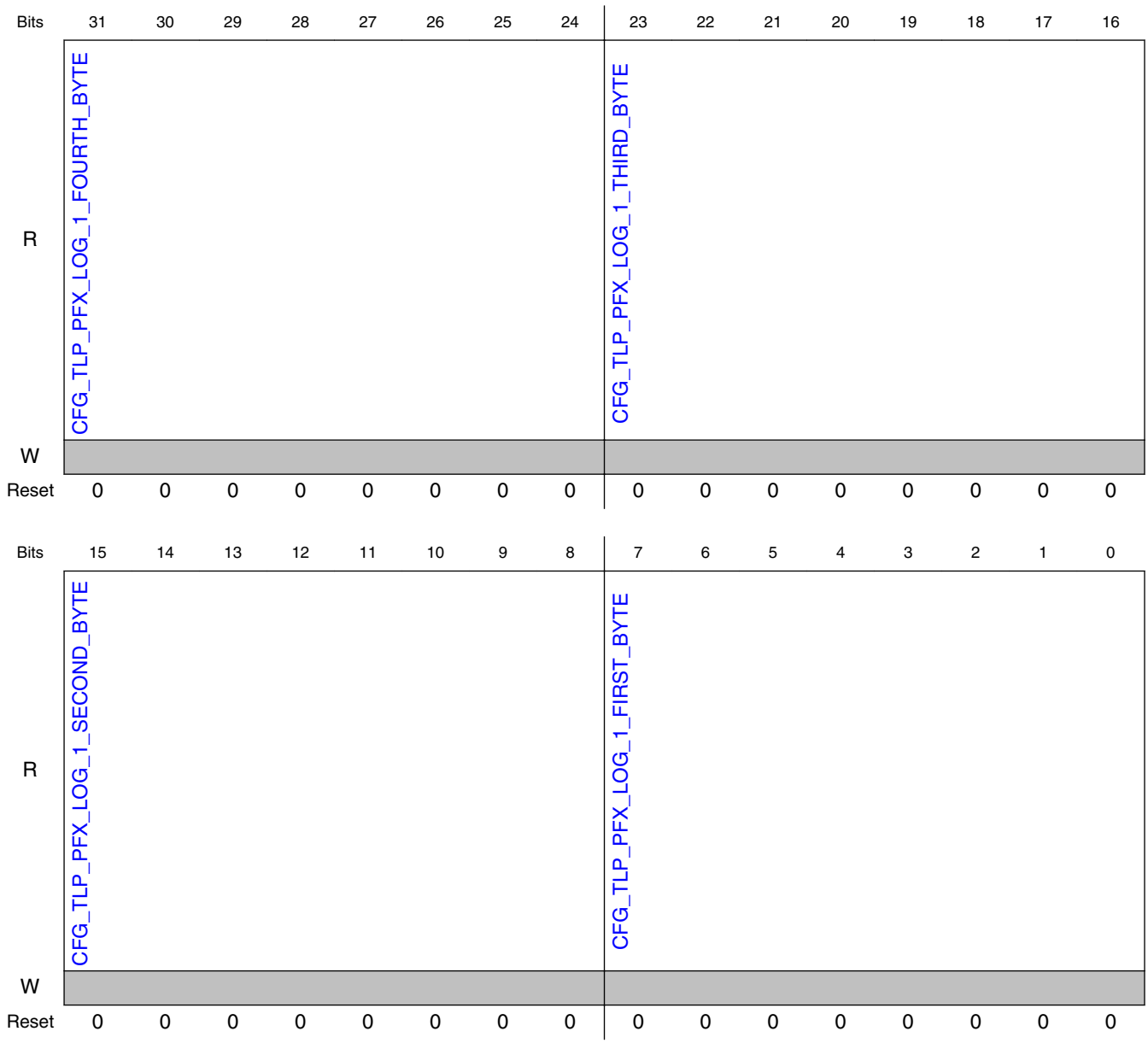
11.3.5.1.51.1 Offset

Register	Offset
TLP_PREFIX_LOG_1_OFF	138h

11.3.5.1.51.2 Function

TLP Prefix Log Register 1. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.51.3 Diagram



11.3.5.1.51.4 Fields

Field	Function
31-24 CFG_TLP_PFX_LOG_1_FOURTH_BYTE	Byte 3 of Error TLP Prefix Log 1. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
23-16	Byte 2 of Error TLP Prefix Log 1. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

Table continues on the next page...

Field	Function
CFG_TLP_PFX_LOG_1_THIRD_BYTE	
15-8 CFG_TLP_PFX_LOG_1_SECOND_BYTE	Byte 1 of Error TLP Prefix Log 1. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
7-0 CFG_TLP_PFX_LOG_1_FIRST_BYTE	Byte 0 of Error TLP Prefix Log 1. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

11.3.5.1.52 TLP Prefix Log Register 2. (TLP_PREFIX_LOG_2_OFF)

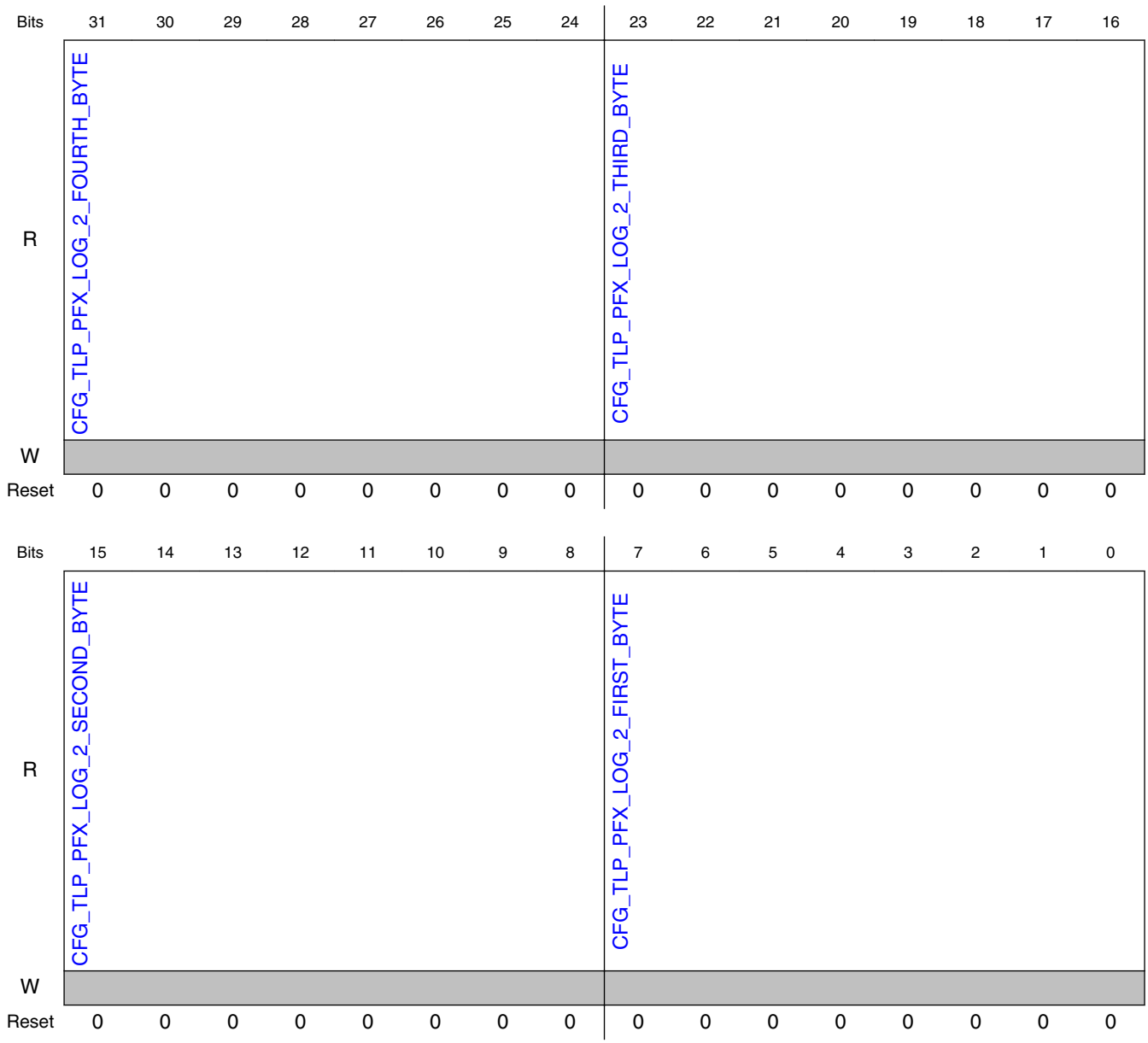
11.3.5.1.52.1 Offset

Register	Offset
TLP_PREFIX_LOG_2_OFF	13Ch

11.3.5.1.52.2 Function

TLP Prefix Log Register 2. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.52.3 Diagram



11.3.5.1.52.4 Fields

Field	Function
31-24 CFG_TLP_PFX_LOG_2_FOURTH_BYTE	Byte 3 Error TLP Prefix Log 2. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
23-16	Byte 2 Error TLP Prefix Log 2. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

Table continues on the next page...

Field	Function
CFG_TLP_PFX_LOG_2_THIRD_BYTE	
15-8 CFG_TLP_PFX_LOG_2_SECOND_BYTE	Byte 1 Error TLP Prefix Log 2. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
7-0 CFG_TLP_PFX_LOG_2_FIRST_BYTE	Byte 0 Error TLP Prefix Log 2. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

11.3.5.1.53 TLP Prefix Log Register 3. (TLP_PREFIX_LOG_3_OFF)

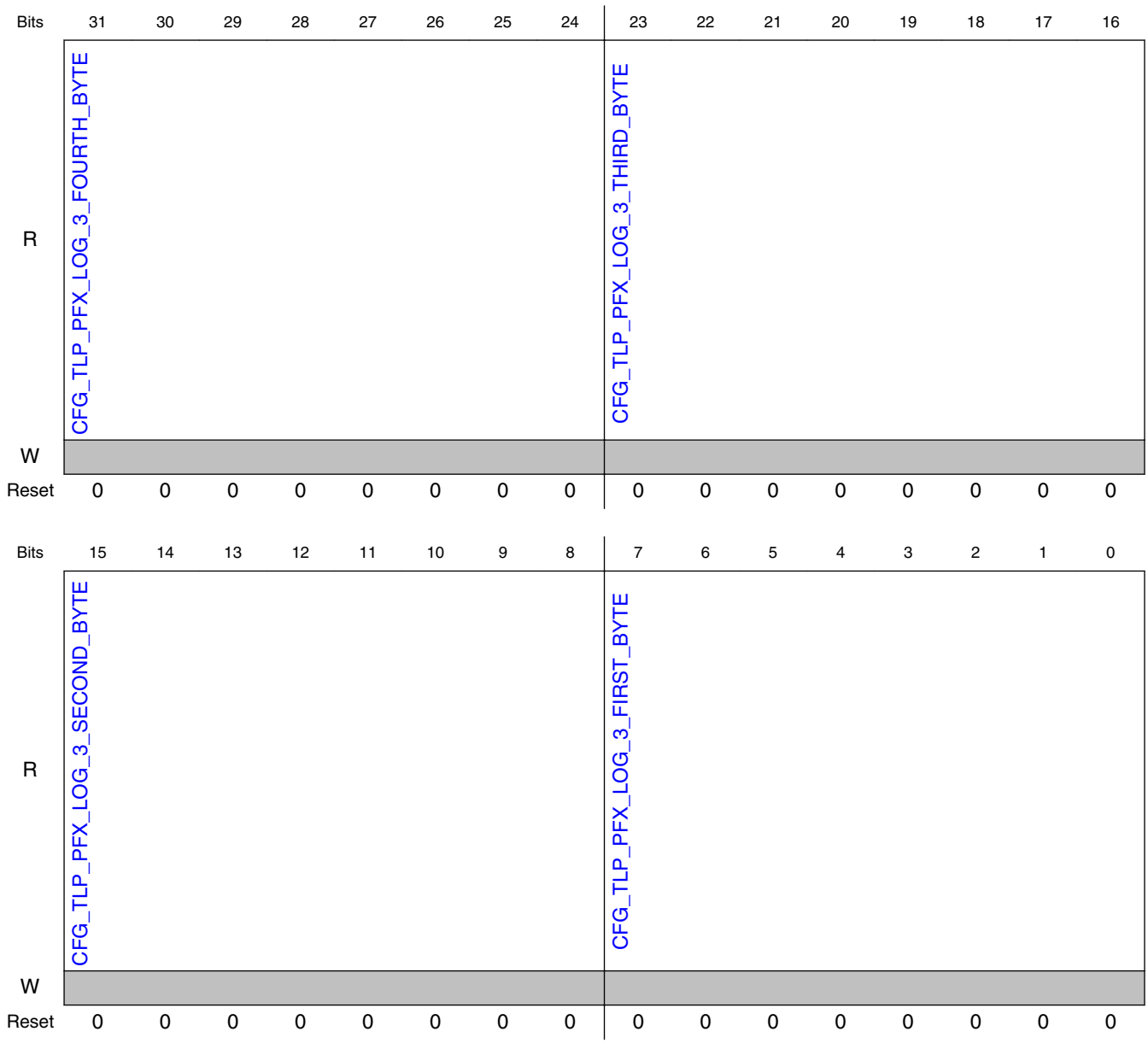
11.3.5.1.53.1 Offset

Register	Offset
TLP_PREFIX_LOG_3_OFF	140h

11.3.5.1.53.2 Function

TLP Prefix Log Register 3. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.53.3 Diagram



11.3.5.1.53.4 Fields

Field	Function
31-24 CFG_TLP_PFX_LOG_3_FOURTH_BYTE	Byte 3 Error TLP Prefix Log 3. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
23-16	Byte 2 Error TLP Prefix Log 3. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

Table continues on the next page...

Field	Function
CFG_TLP_PFX_LOG_3_THIRD_BYTE	
15-8 CFG_TLP_PFX_LOG_3_SECOND_BYTE	Byte 1 Error TLP Prefix Log 3. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
7-0 CFG_TLP_PFX_LOG_3_FIRST_BYTE	Byte 0 Error TLP Prefix Log 3. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

11.3.5.1.54 TLP Prefix Log Register 4. (TLP_PREFIX_LOG_4_OFF)

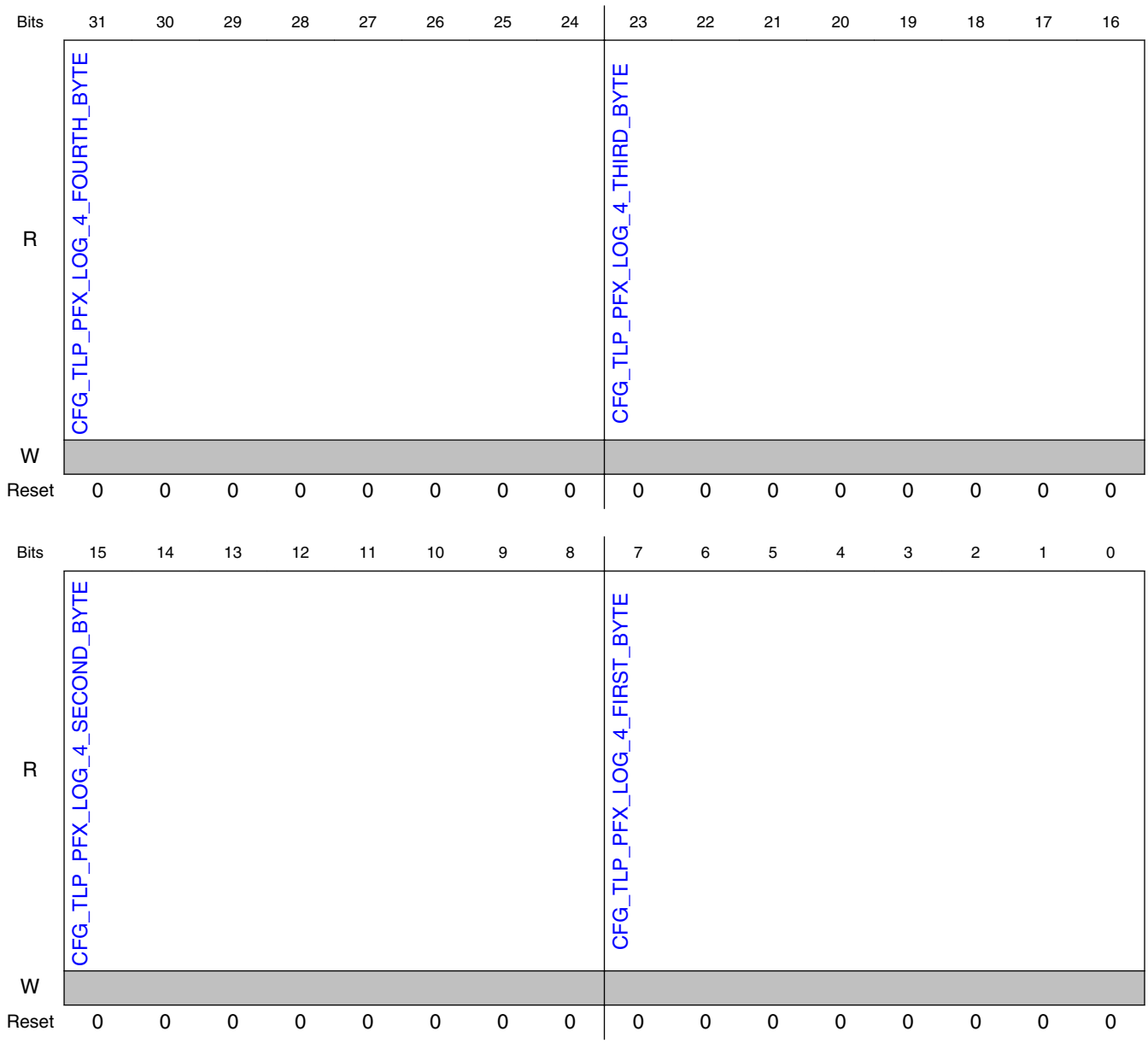
11.3.5.1.54.1 Offset

Register	Offset
TLP_PREFIX_LOG_4_OFF	144h

11.3.5.1.54.2 Function

TLP Prefix Log Register 4. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.54.3 Diagram



11.3.5.1.54.4 Fields

Field	Function
31-24 CFG_TLP_PFX_LOG_4_FOURTH_BYTE	Byte 3 Error TLP Prefix Log 4. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
23-16	Byte 2 Error TLP Prefix Log 4. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

Table continues on the next page...

Field	Function
CFG_TLP_PFX_LOG_4_THIRD_BYTE	
15-8 CFG_TLP_PFX_LOG_4_SECOND_BYTE	Byte 1 Error TLP Prefix Log 4. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.
7-0 CFG_TLP_PFX_LOG_4_FIRST_BYTE	Byte 0 Error TLP Prefix Log 4. For a description of this standard PCIe register field, see the PCI Express Specification. Note: This register field is sticky.

11.3.5.1.55 L1 Substates Extended Capability Header. (L1SUB_CAP_HEADER_REG)

11.3.5.1.55.1 Offset

Register	Offset
L1SUB_CAP_HEADER_REG	148h

11.3.5.1.55.2 Function

L1 Substates Extended Capability Header. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.55.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NEXT_OFFSET												CAP_VERSION			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EXTENDED_CAP_ID															
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0

11.3.5.1.55.4 Fields

Field	Function
31-20 NEXT_OFFSET	Next Capability Offset. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
19-16 CAP_VERSION	Capability Version. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.
15-0 EXTENDED_CAP_ID	L1SUB Extended Capability ID. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: if (DBI_RO_WR_EN == 1) then R/W(sticky) else R(sticky) Note: This register field is sticky.

11.3.5.1.56 L1 Substates Capability Register. (L1SUB_CAPABILITY_REG)

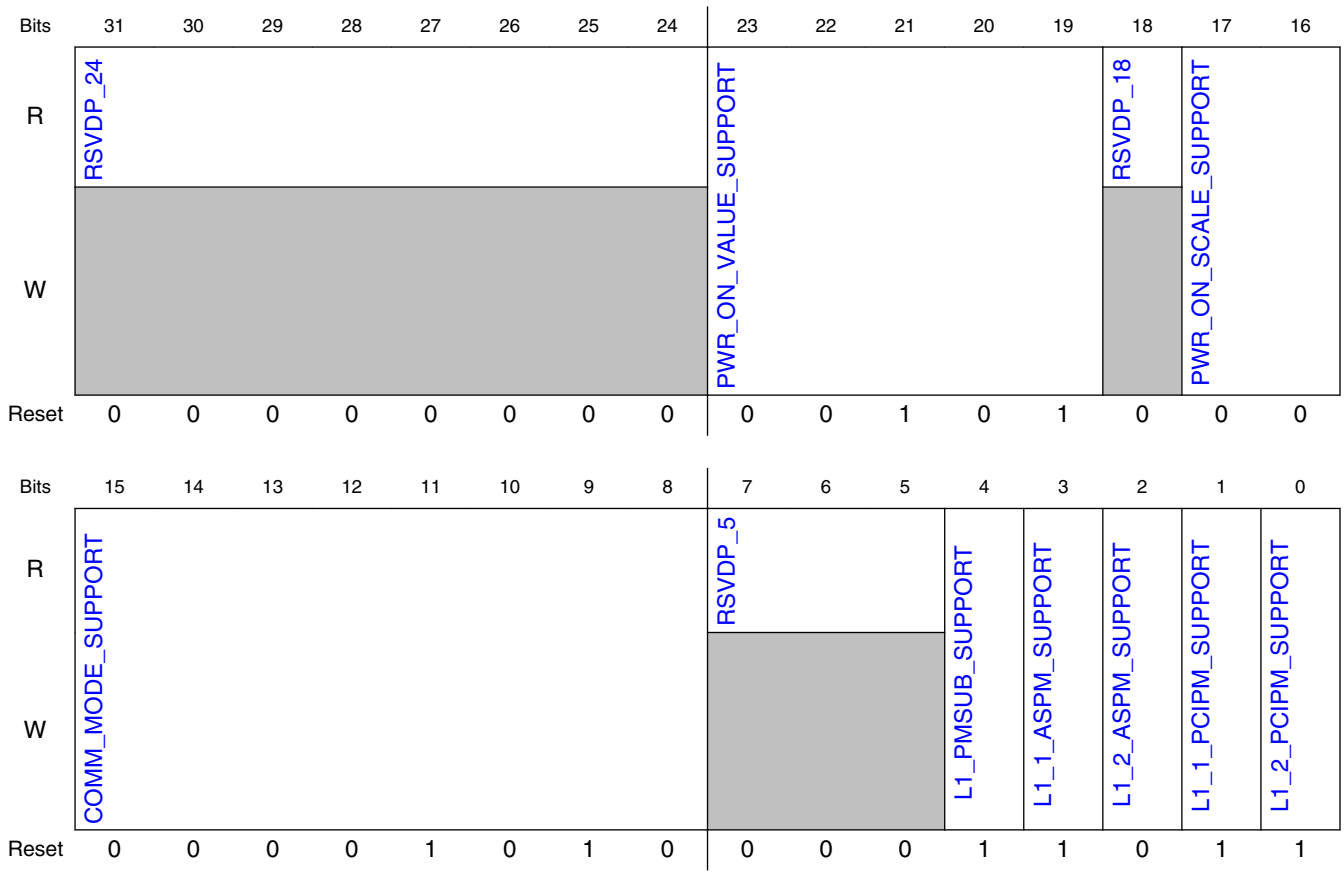
11.3.5.1.56.1 Offset

Register	Offset
L1SUB_CAPABILITY_REG	14Ch

11.3.5.1.56.2 Function

L1 Substates Capability Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.56.3 Diagram



11.3.5.1.56.4 Fields

Field	Function
31-24 RSVDP_24	Reserved for future use.
23-19 PWR_ON_VALUE_SUPPORT	Port T Power On Value. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W
18 RSVDP_18	Reserved for future use.
17-16 PWR_ON_SCALE_SUPPORT	Port T Power On Scale. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W
15-8 COMM_MODE_SUPPORT	Port Common Mode Restore Time. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W
7-5	Reserved for future use.

Table continues on the next page...

Field	Function
RSVDP_5	
4 L1_PMSUB_SUPPORT	L1 PM Substates ECN Supported. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W
3 L1_1_ASPM_SUPPORT	ASPM L11 Supported. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W
2 L1_2_ASPM_SUPPORT	ASPM L12 Supported. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W
1 L1_1_PCIPM_SUPPORT	PCI-PM L11 Supported. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W
0 L1_2_PCIPM_SUPPORT	PCI-PM L12 Supported. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.57 L1 Substates Control 1 Register. (L1SUB_CONTROL1_REG)

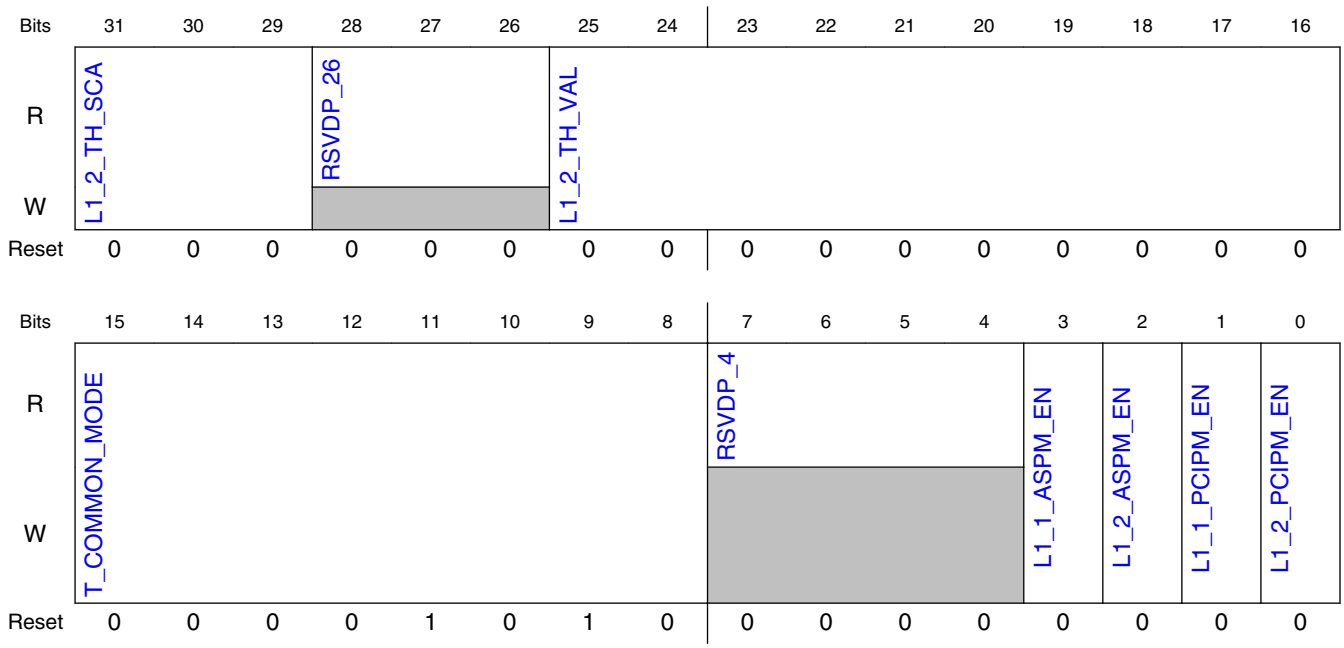
11.3.5.1.57.1 Offset

Register	Offset
L1SUB_CONTROL1_REG	150h

11.3.5.1.57.2 Function

L1 Substates Control 1 Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.57.3 Diagram



11.3.5.1.57.4 Fields

Field	Function
31-29 L1_2_TH_SCA	LTR L12 Threshold Scale. For a description of this standard PCIe register field, see the PCI Express Specification.
28-26 RSVDP_26	Reserved for future use.
25-16 L1_2_TH_VAL	LTR L12 Threshold Value. For a description of this standard PCIe register field, see the PCI Express Specification.
15-8 T_COMMON_MODE	Common Mode Restore Time. For a description of this standard PCIe register field, see the PCI Express Specification. Note: The access attributes of this field are as follows: - Dbi: R/W
7-4 RSVDP_4	Reserved for future use.
3 L1_1_ASPM_EN	ASPM L11 Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
2 L1_2_ASPM_EN	ASPM L12 Enable. For a description of this standard PCIe register field, see the PCI Express Specification.
1 L1_1_PCIPM_EN	PCI-PM L11 Enable. For a description of this standard PCIe register field, see the PCI Express Specification.

Table continues on the next page...

Field	Function
0 L1_2_PCIPM_EN	PCI-PM L12 Enable. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.58 L1 Substates Control 2 Register. (L1SUB_CONTROL2_REG)

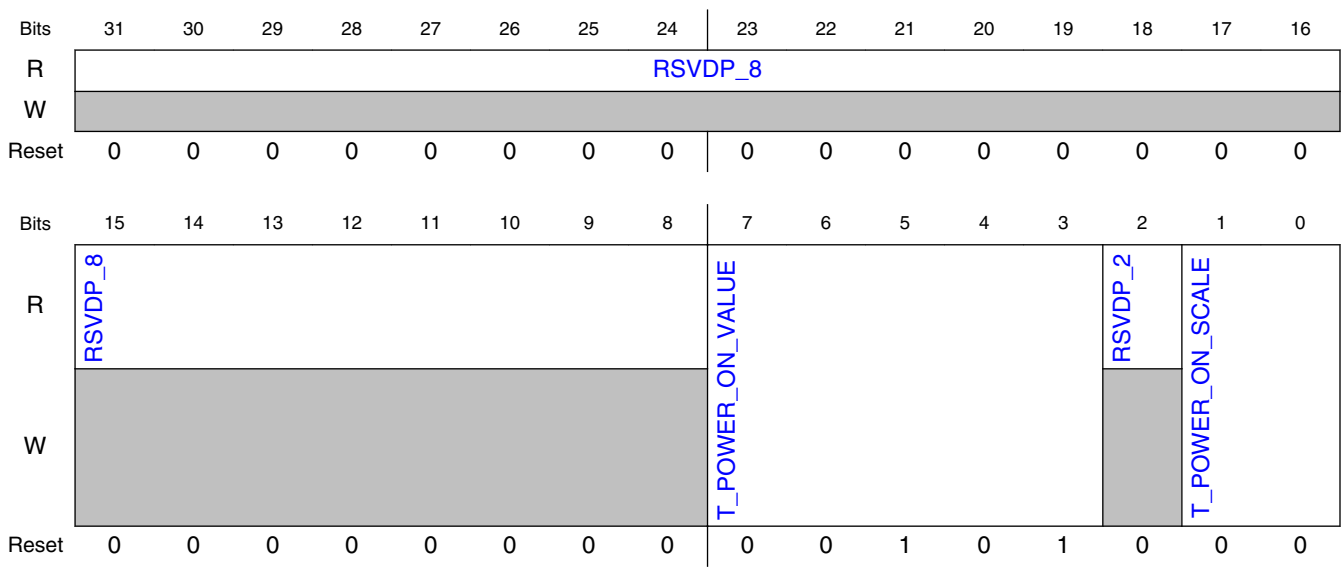
11.3.5.1.58.1 Offset

Register	Offset
L1SUB_CONTROL2_REG	154h

11.3.5.1.58.2 Function

L1 Substates Control 2 Register. For a description of this standard PCIe register, see the PCI Express Specification.

11.3.5.1.58.3 Diagram



11.3.5.1.58.4 Fields

Field	Function
31-8	Reserved for future use.

Table continues on the next page...

Field	Function
RSVDP_8	
7-3 T_POWER_ON_VALUE	T Power On Value. For a description of this standard PCIe register field, see the PCI Express Specification.
2 RSVDP_2	Reserved for future use.
1-0 T_POWER_ON_SCALE	T Power On Scale. For a description of this standard PCIe register field, see the PCI Express Specification.

11.3.5.1.59 Ack Latency Timer and Replay Timer Register. (ACK_LATENCY_TIMER_OFF)

11.3.5.1.59.1 Offset

Register	Offset
ACK_LATENCY_TIMER_OFF	700h

11.3.5.1.59.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	REPLAY_TIME_LIMIT															
Reset	0	0	0	1	1	0	0	0	0	1	0	0	0	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ROUND_TRIP_LATENCY_TIME_LIMIT															
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	1	1	1

11.3.5.1.59.3 Fields

Field	Function
31-16 REPLAY_TIME_LIMIT	Replay Timer Limit. The replay timer expires when it reaches this limit. The controller initiates a replay upon reception of a NAK or when the replay timer expires. For more details, see "Transmit Replay". You can modify the effective timer limit with the TIMER_MOD_REPLAY_TIMER field of the TIMER_CTRL_MAX_FUNC_NUM_OFF register. After reset, the controller updates the default according to the Negotiated Link Width, Max_Payload_Size, and speed. The value is determined from Tables 3-4,

Table continues on the next page...

Field	Function
	3-5, and 3-6 of the PCIe 3.0 specification. If there is a change in the payload size or link speed, the controller will override any value that you have written to this register field, and reset the field back to the specification-defined value. It will not change the value in the TIMER_MOD_REPLAY_TIMER field of the TIMER_CTRL_MAX_FUNC_NUM_OFF register.
15-0 ROUND_TRIP_LATENCY_TIME_LIMIT	Ack Latency Timer Limit. The Ack latency timer expires when it reaches this limit. For more details, see "Ack Scheduling". You can modify the effective timer limit with the TIMER_MOD_ACK_NAK field of the TIMER_CTRL_MAX_FUNC_NUM_OFF register. After reset, the controller updates the default according to the Negotiated Link Width, Max_Payload_Size, and speed. The value is determined from Tables 3-7, 3-8, and 3-9 of the PCIe 3.0 specification. The limit must reflect the round trip latency from requester to completer. If there is a change in the payload size or link width, the controller will override any value that you have written to this register field, and reset the field back to the specification-defined value. It will not change the value in the TIMER_MOD_ACK_NAK field of the TIMER_CTRL_MAX_FUNC_NUM_OFF register.

11.3.5.1.60 Vendor Specific DLLP Register. (VENDOR_SPEC_DLLP_OFF)

11.3.5.1.60.1 Offset

Register	Offset
VENDOR_SPEC_DLLP_OFF	704h

11.3.5.1.60.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	VENDOR_SPEC_DLLP															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VENDOR_SPEC_DLLP															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

11.3.5.1.60.3 Fields

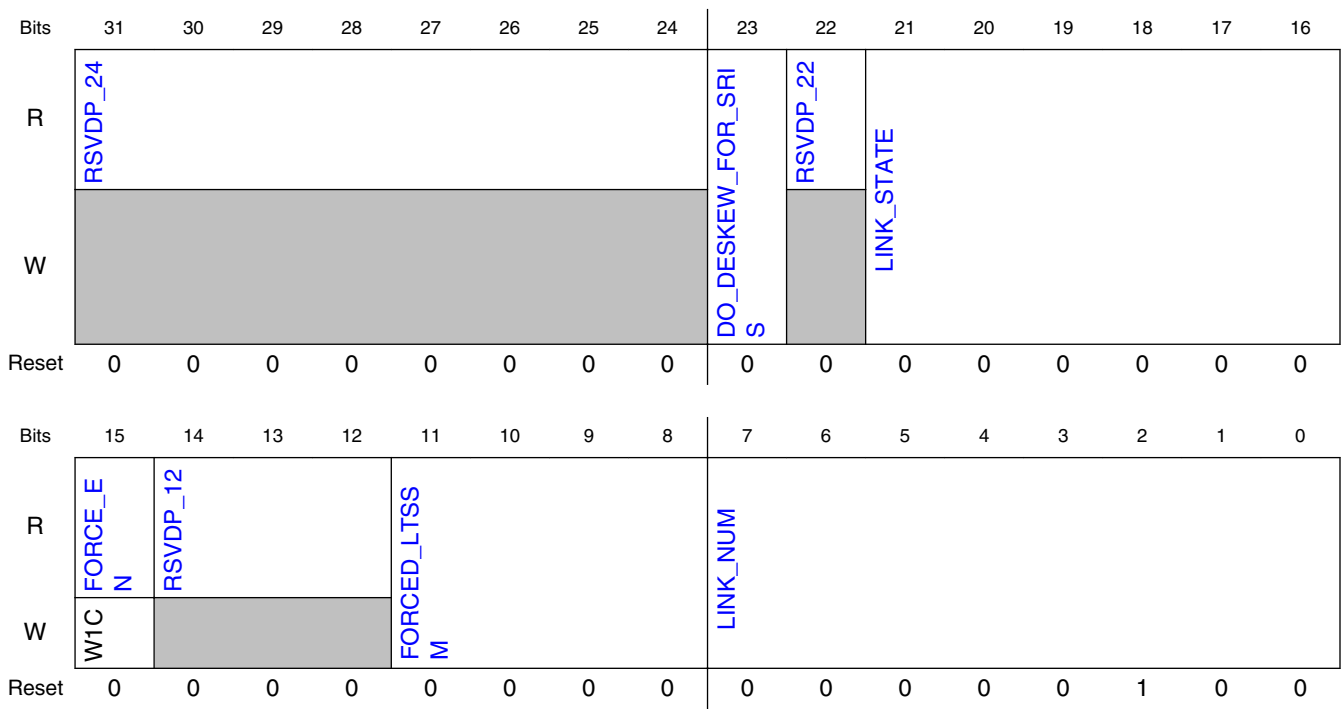
Field	Function
31-0 VENDOR_SPEC_DLLP	Vendor Specific DLLP Register. Used to send a specific PCI Express DLLP. Your application writes the 8-bit DLLP Type and 24-bits of Payload data into this register, then sets the field VENDOR_SPECIFIC_DLLP_REQ of PORT_LINK_CTRL_OFF to send the DLLP. - [7:0] = Type - [31:8] = Payload (24 bits) The dllp type is in bits [7:0] while the remainder is the vendor defined payload. Note: This register field is sticky.

11.3.5.1.61 Port Force Link Register. (PORT_FORCE_OFF)

11.3.5.1.61.1 Offset

Register	Offset
PORT_FORCE_OFF	708h

11.3.5.1.61.2 Diagram



11.3.5.1.61.3 Fields

Field	Function
31-24 RSVDP_24	Reserved for future use.
23 DO_DESKEW_FOR_SRIS	Use the transitions from TS2 to Logical Idle Symbol, SKP OS to Logical Idle Symbol, and FTS Sequence to SKP OS to do deskew for SRIS instead of using received SKP OS if DO_DESKEW_FOR_SRIS is set to 1. Note: This register field is sticky.
22 RSVDP_22	Reserved for future use.

Table continues on the next page...

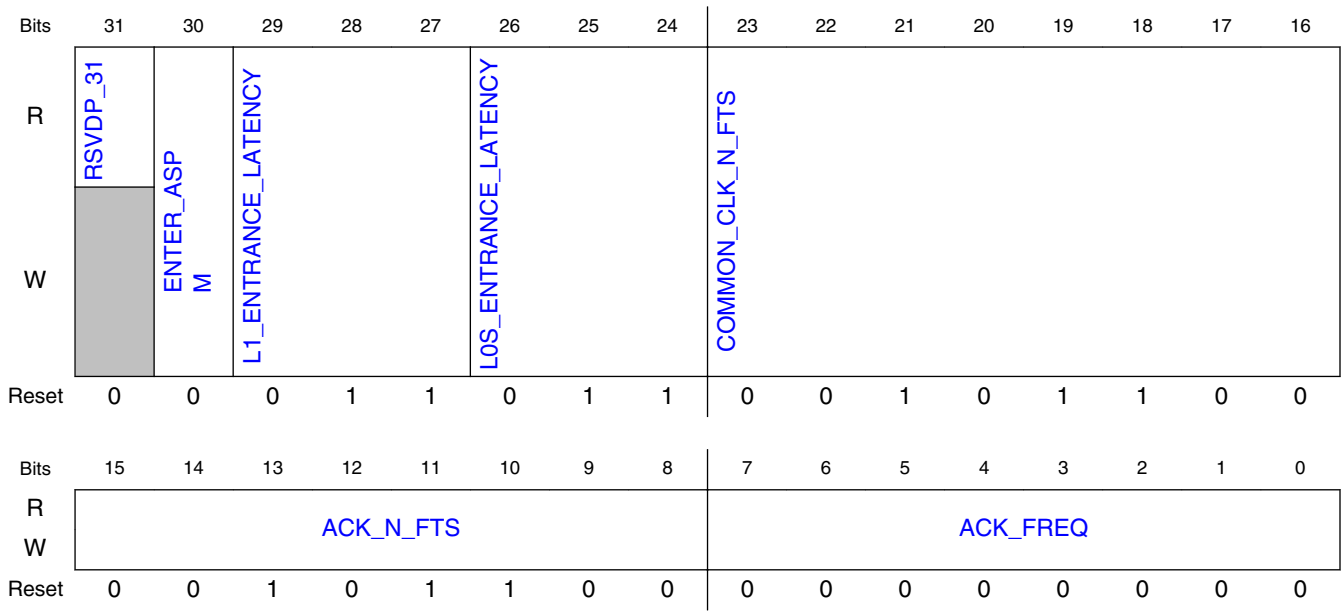
Field	Function
21-16 LINK_STATE	Forced LTSSM State. The LTSSM state that the controller is forced to when you set the FORCE_EN bit (Force Link). LTSSM state encoding is defined by the lts_state variable in workspace/src/Layer1/smlh_ltssm.v. Note: This register field is sticky.
15 FORCE_EN	Force Link. The controller supports a testing and debug capability to allow your software to force the LTSSM state machine into a specific state, and to force the controller to transmit a specific Link Command. Asserting this bit triggers the following actions: - Forces the LTSSM to the state specified by the Forced LTSSM State field. - Forces the controller to transmit the command specified by the Forced Link Command field. This is a self-clearing register field. Reading from this register field always returns a "0".
14-12 RSVDP_12	Reserved for future use.
11-8 FORCED_LTSSM	Forced Link Command. The link command that the controller is forced to transmit when you set FORCE_EN bit (Force Link). Link command encoding is defined by the ltssm_cmd variable in workspace/src/Layer1/smlh_ltssm.v. Note: This register field is sticky.
7-0 LINK_NUM	Link Number. Not used for endpoint. Not used for M-PCIe. Note: This register field is sticky.

11.3.5.1.62 Ack Frequency and L0-L1 ASPM Control Register. (ACK_F_ASPM_CTRL_OFF)

11.3.5.1.62.1 Offset

Register	Offset
ACK_F_ASPM_CTRL_OFF	70Ch

11.3.5.1.62.2 Diagram



11.3.5.1.62.3 Fields

Field	Function
31 RSVDP_31	Reserved for future use.
30 ENTER_ASPM	ASPM L1 Entry Control. - 1: Core enters ASPM L1 after a period in which it has been idle. - 0: Core enters ASPM L1 only after idle period during which both receive and transmit are in L0s. Note: This register field is sticky.
29-27 L1_ENTRANCE_LATENCY	L1 Entrance Latency. Value range is: - 000: 1 us - 001: 2 us - 010: 4 us - 011: 8 us - 100: 16 us - 101: 32 us - 110 or 111: 64 us Note: Programming this timer with a value greater that 32us has no effect unless extended sync is used, or all of the credits are infinite. Note: This register field is sticky.
26-24 L0S_ENTRANCE_LATENCY	L0s Entrance Latency. Values correspond to: - 000: 1 us - 001: 2 us - 010: 3 us - 011: 4 us - 100: 5 us - 101: 6 us - 110 or 111: 7 us This field is applicable to STALL while in L0 for M-PCIe. Note: This register field is sticky.
23-16 COMMON_CLK_N_FTS	Common Clock N_FTS. This is the N_FTS when common clock is used. The number of Fast Training Sequence ordered sets to be transmitted when transitioning from L0s to L0. The maximum number of FTS ordered-sets that a component can request is 255. The controller does not support a value of zero; a value of zero can cause the LTSSM to go into the recovery state when exiting from L0s. This field is reserved (fixed to '0') for M-PCIe. Note: The access attributes of this field are as follows: - Dbi: R
15-8 ACK_N_FTS	N_FTS. The number of Fast Training Sequence ordered sets to be transmitted when transitioning from L0s to L0. The maximum number of FTS ordered-sets that a component can request is 255. The controller does not support a value of zero; a value of zero can cause the LTSSM to go into the recovery state when exiting from L0s. This field is reserved (fixed to '0') for M-PCIe. Note: This register field is sticky.
7-0 ACK_FREQ	Ack Frequency. The controller accumulates the number of pending ACKs specified here (up to 255) before sending an ACK DLLP. - 0: Indicates that this Ack frequency control feature is turned off. The controller schedules a low-priority ACK DLLP for every TLP that it receives. - 1-255: Indicates that the

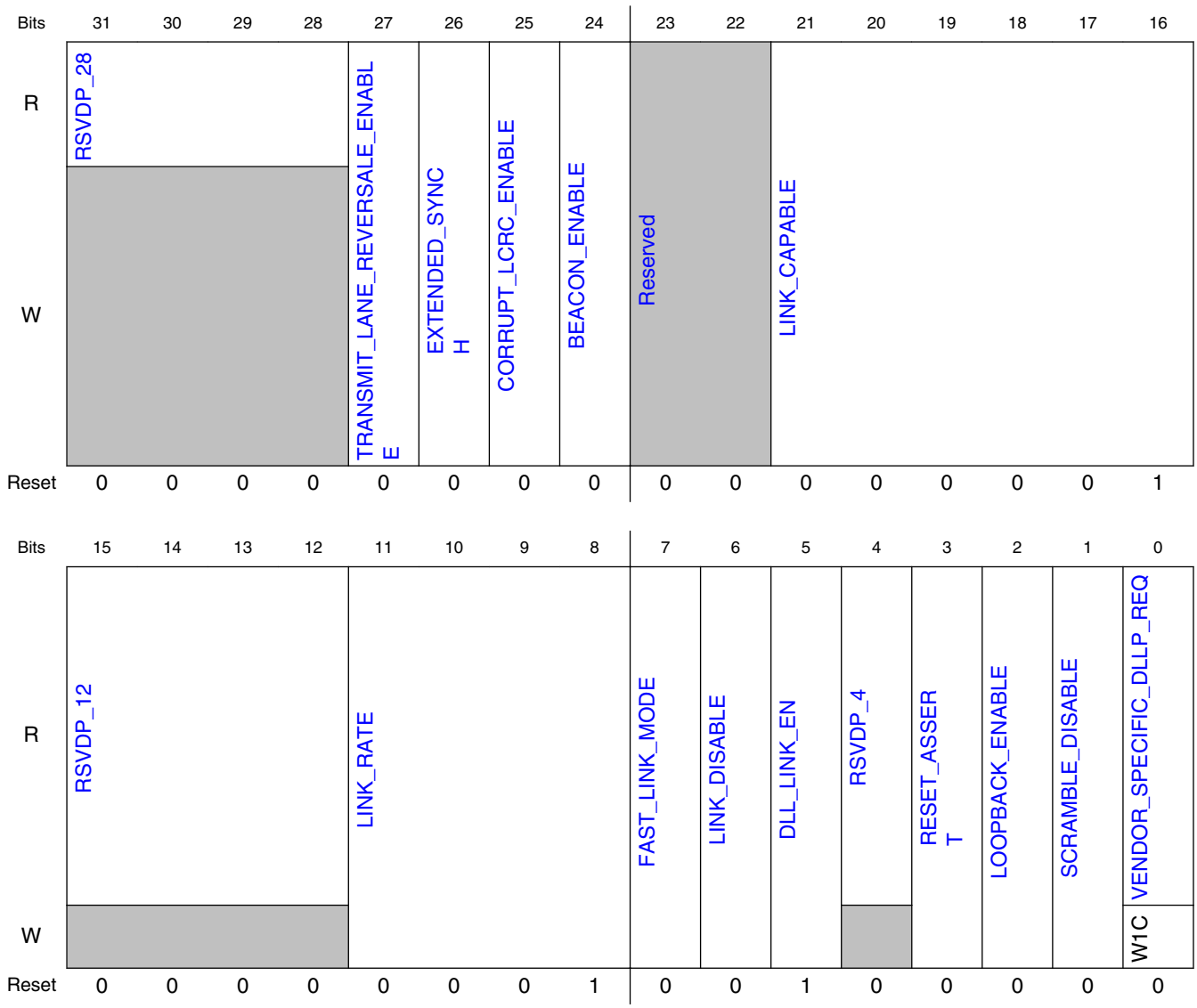
Field	Function
	controller will schedule a high-priority ACK after receiving this number of TLPs. It might schedule the ACK before receiving this number of TLPs, but never later. For a typical system, you do not have to modify the default setting. For more details, see "ACK/NAK Scheduling". Note: This register field is sticky.

11.3.5.1.63 Port Link Control Register. (PORT_LINK_CTRL_OFF)

11.3.5.1.63.1 Offset

Register	Offset
PORT_LINK_CTRL_OFF	710h

11.3.5.1.63.2 Diagram



11.3.5.1.63.3 Fields

Field	Function
31-28 RSVDP_28	Reserved for future use.
27 TRANSMIT_LANE_REVERSALE_ENABLE	TRANSMIT_LANE_REVERSALE_ENABLE is an internally reserved field. Do not use. Note: This register field is sticky.
26	EXTENDED_SYNCH is an internally reserved field. Do not use. Note: This register field is sticky.

Table continues on the next page...

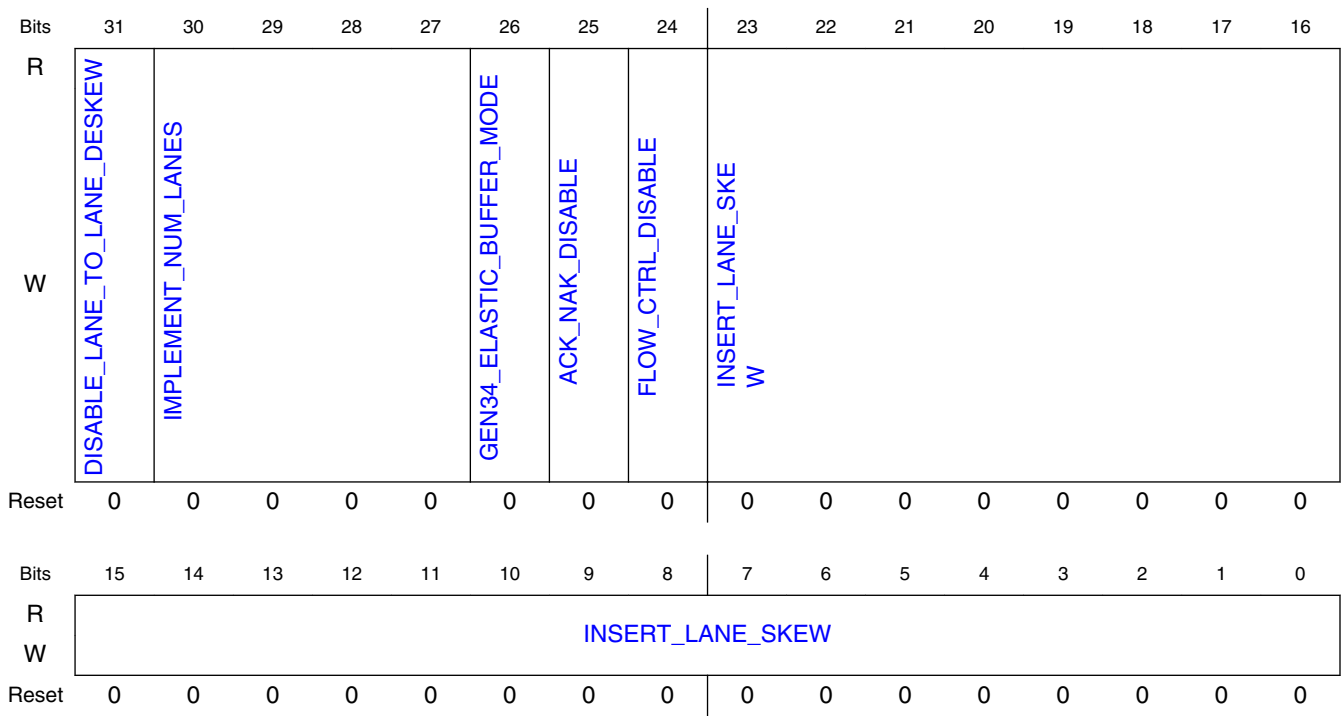
Field	Function
EXTENDED_SYNCH	
25 CORRUPT_LCRC_ENABLE	CORRUPT_LCRC_ENABLE is an internally reserved field. Do not use. Note: This register field is sticky.
24 BEACON_ENABLE	BEACON_ENABLE is an internally reserved field. Do not use. Note: This register field is sticky.
23-22 —	Reserved.
21-16 LINK_CAPABLE	Link Mode Enable. Sets the number of lanes in the link that you want to connect to the link partner. When you have unused lanes in your system, then you must change the value in this register to reflect the number of lanes. You must also change the value in the "Predetermined Number of Lanes" field of the "Link Width and Speed Change Control Register". For more information, see "How to Tie Off Unused Lanes". For information on upsizing and downsizing the link width, see "Link Establishment". - 000001: x1 - 000011: x2 - 000111: x4 - 001111: x8 - 011111: x16 - 111111: x32 (not supported) This field is reserved (fixed to '0') for M-PCIe. Note: This register field is sticky.
15-12 RSVDP_12	Reserved for future use.
11-8 LINK_RATE	LINK_RATE is an internally reserved field. Do not use. Note: This register field is sticky.
7 FAST_LINK_MODE	Fast Link Mode. Sets all internal LTSSM millisecond timers to Fast Mode for speeding up simulation. Forces the LTSSM training (link initialization) to use shorter time-outs and to link up faster. The default scaling factor can be changed through the FAST_LINK_SCALING_FACTOR field in the TIMER_CTRL_MAX_FUNC_NUM_OFF register. If this bit is set to '1', tRRAPInitiatorResponse is set to 1.88 ms(60 ms/32). Note: This register field is sticky.
6 LINK_DISABLE	LINK_DISABLE is an internally reserved field. Do not use. Note: This register field is sticky.
5 DLL_LINK_EN	DLL Link Enable. Enables link initialization. When DLL Link Enable =0, the controller does not transmit InitFC DLLPs and does not establish a link. Note: This register field is sticky.
4 RSVDP_4	Reserved for future use.
3 RESET_ASSERT	Reset Assert. Triggers a recovery and forces the LTSSM to the hot reset state (downstream port only). Note: This register field is sticky.
2 LOOPBACK_ENABLE	Loopback Enable. Turns on loopback. For more details, see "Loopback". For M-PCIe, to force the master to enter Digital Loopback mode, you must set this field to "1" during Configuration.start state(initial discovery/configuration). M-PCIe doesn't support loopback mode from L0 state - only from Configuration.start. Note: This register field is sticky.
1 SCRAMBLE_DISABLE	Scramble Disable. Turns off data scrambling. Note: This register field is sticky.
0 VENDOR_SPECIFIC_DLLP_REQ	Vendor Specific DLLP Request. When software writes a '1' to this bit, the controller transmits the DLLP contained in the VENDOR_SPEC_DLLP field of VENDOR_SPEC_DLLP_OFF. Reading from this self-clearing register field always returns a '0'.

11.3.5.1.64 Lane Skew Register. (LANE_SKEW_OFF)

11.3.5.1.64.1 Offset

Register	Offset
LANE_SKEW_OFF	714h

11.3.5.1.64.2 Diagram



11.3.5.1.64.3 Fields

Field	Function
31 DISABLE_LANE_TO_LANE_DESKEW	Disable Lane-to-Lane Deskew. Causes the controller to disable the internal Lane-to-Lane deskew logic. Note: This register field is sticky.
30-27 IMPLEMENT_NUM_LANES	Implementation-specific Number of Lanes. Set the implementation-specific number of lanes. Allowed values are: - 4'b0000: 1 lane - 4'b0001: 2 lanes - 4'b0011: 4 lanes - 4'b0111: 8 lanes - 4'b1111: 16 lanes The number of lanes to be used when in Loopback Master. The number of lanes programmed must be equal to or less than the valid number of lanes set in LINK_CAPABLE field. You must configure this field before initiating Loopback by writing in the LOOPBACK_ENABLE field. The controller will transition from Loopback.Entry to Loopback.Active after receiving two consecutive TS1 Ordered Sets with the Loopback

Table continues on the next page...

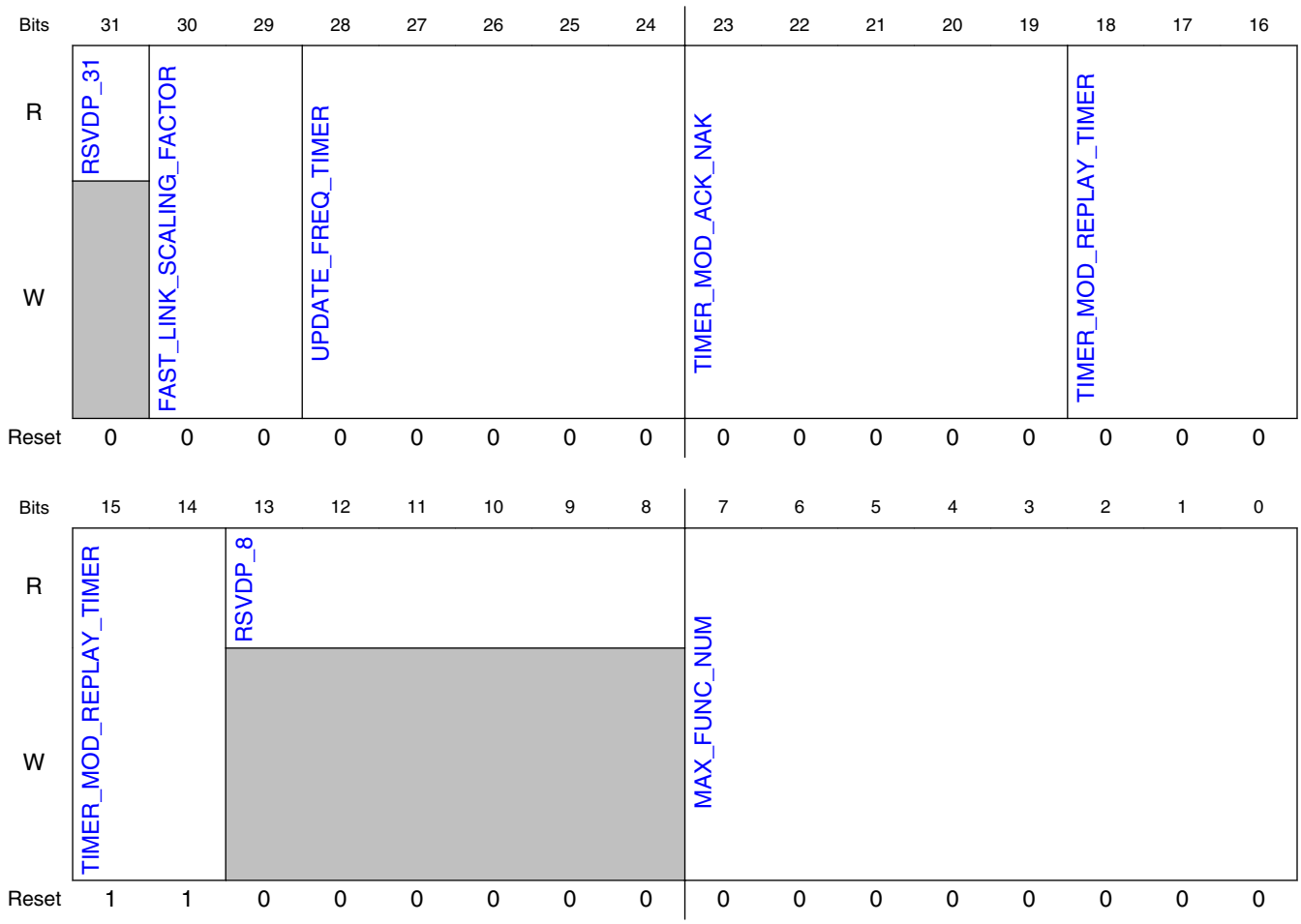
Field	Function
	bit asserted on the implementation specific number of lanes configured in this field. Note: This register field is sticky.
26 GEN34_ELASTIC_BUFFER_MODE	Selects Elasticity Buffer operating mode in Gen3 or Gen4 rate: 0: Nominal Half Full Buffer mode 1: Nominal Empty Buffer Mode This register bit only affects Gen3 or Gen4 operating rate. For Gen1 or Gen2 operating rate the Elasticity Buffer operating mode is always the Nominal Half Full Buffer mode. Note: This register field is sticky.
25 ACK_NAK_DISABLE	Ack/Nak Disable. Prevents the controller from sending ACK and NAK DLLPs. Note: This register field is sticky.
24 FLOW_CTRL_DISABLE	Flow Control Disable. Prevents the controller from sending FC DLLPs. Note: This register field is sticky.
23-0 INSERT_LANE_SKEW	INSERT_LANE_SKEW is an internally reserved field. Do not use. Note: This register field is sticky.

11.3.5.1.65 Timer Control and Max Function Number Register. (TIMER_CTRL_MAX_FUNC_NUM_OFF)

11.3.5.1.65.1 Offset

Register	Offset
TIMER_CTRL_MAX_FUNC_NUM_OFF	718h

11.3.5.1.65.2 Diagram



11.3.5.1.65.3 Fields

Field	Function
31 RSVDP_31	Reserved for future use.
30-29 FAST_LINK_SCALING_FACTOR	Fast Link Timer Scaling Factor. Sets the scaling factor of LTSSM timer when FAST_LINK_MODE field in PORT_LINK_CTRL_OFF is set to '1'. - 0: Scaling Factor is 1024 (1ms is 1us) - 1: Scaling Factor is 256 (1ms is 4us) - 2: Scaling Factor is 64 (1ms is 16us) - 3: Scaling Factor is 16 (1ms is 64us). Not used for M-PCIe. Note: This register field is sticky.
28-24 UPDATE_FREQ_TIMER	UPDATE_FREQ_TIMER is an internally reserved field. Do not use. Note: This register field is sticky.
23-19 TIMER_MOD_ACK_NAK	Ack Latency Timer Modifier. Increases the timer value for the Ack latency timer in increments of 64 clock cycles. A value of "0" represents no modification to the timer value. For more details, see the ROUND_TRIP_LATENCY_TIME_LIMIT field of the ACK_LATENCY_TIMER_OFF register. Note: This register field is sticky.

Table continues on the next page...

Field	Function
18-14 TIMER_MOD_REPLAY_TIMER	Replay Timer Limit Modifier. Increases the time-out value for the replay timer in increments of 64 clock cycles at Gen1 or Gen2 speed, and in increments of 256 clock cycles at Gen3 speed. A value of "0" represents no modification to the timer limit. For more details, see the REPLAY_TIME_LIMIT field of the ACK_LATENCY_TIMER_OFF register. At Gen3 speed, the controller automatically changes the value of this field to DEFAULT_GEN3_REPLAY_ADJ. For M-PCIe, this field increases the time-out value for the replay timer in increments of 64 clock cycles at HS-Gear1, HS-Gear2, or HS-Gear3 speed. Note: This register field is sticky.
13-8 RSVDP_8	Reserved for future use.
7-0 MAX_FUNC_NUM	Maximum function number that can be used in a request. Configuration requests targeted at function numbers above this value are returned with UR (unsupported request). Note: This register field is sticky.

11.3.5.1.66 Symbol Timer Register and Filter Mask 1 Register. (SYMBOL_TIMER_FILTER_1_OFF)

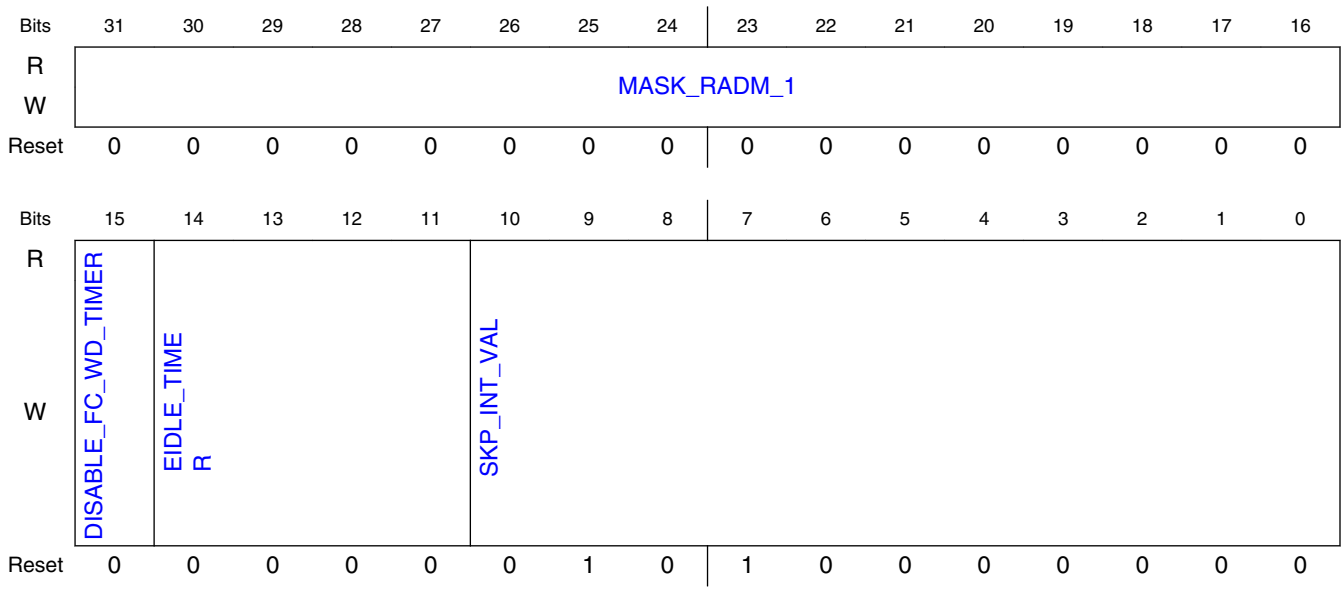
11.3.5.1.66.1 Offset

Register	Offset
SYMBOL_TIMER_FILTER_1_OFF	71Ch

11.3.5.1.66.2 Function

Symbol Timer Register and Filter Mask 1 Register. The Filter Mask 1 Register modifies the RADM filtering and error handling rules. For more details, see the "Receive Filtering" section. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule.

11.3.5.1.66.3 Diagram



11.3.5.1.66.4 Fields

Field	Function
31-16 MASK_RADM_1	Filter Mask 1. The Filter Mask 1 Register modifies the RADM filtering and error handling rules. For more details, see the "Receive Filtering" section. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule. Note: This register field is sticky.
15 DISABLE_FC_WD_TIMER	Disable FC Watchdog Timer. Note: This register field is sticky.
14-11 EIDLE_TIMER	EIDLE_TIMER is an internally reserved field. Do not use. Note: This register field is sticky.
10-0 SKP_INT_VAL	SKP Interval Value. The number of symbol times to wait between transmitting SKP ordered sets. Note that the controller actually waits the number of symbol times in this register plus 1 between transmitting SKP ordered sets. Your application must program this register accordingly. For example, if 1536 were programmed into this register (in a 250 MHz controller), then the controller actually transmits SKP ordered sets once every 1537 symbol times. The value programmed to this register is actually clock ticks and not symbol times. In a 125 MHz controller, programming the value programmed to this register should be scaled down by a factor of 2 (because one clock tick = two symbol times in this case). Note: This value is not used at Gen3 speed; the skip interval is hardcoded to 370 blocks. Note: This register field is sticky.

11.3.5.1.67 Filter Mask 2 Register. (FILTER_MASK_2_OFF)

11.3.5.1.67.1 Offset

Register	Offset
FILTER_MASK_2_OFF	720h

11.3.5.1.67.2 Function

Filter Mask 2 Register. This register modifies the RADM filtering and error handling rules. For more details, see the "Receive Filtering" section. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule.

11.3.5.1.67.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MASK_RADM_2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MASK_RADM_2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.67.4 Fields

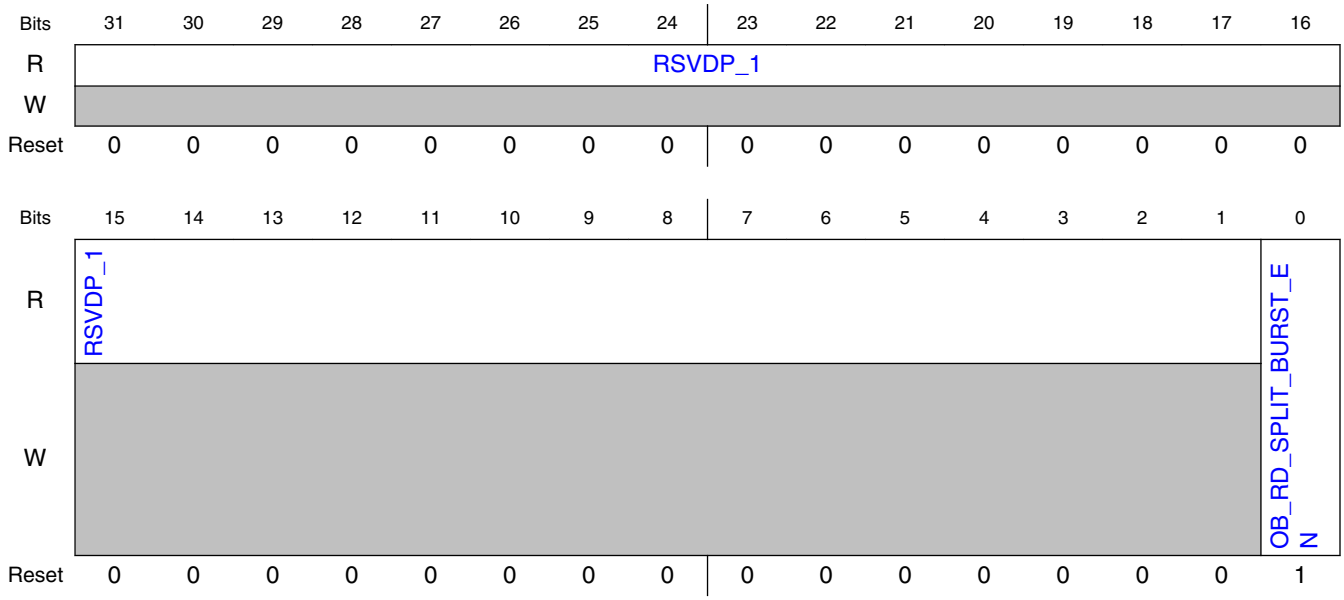
Field	Function
31-0 MASK_RADM_2	Filter Mask 2. This field modifies the RADM filtering and error handling rules. For more details, see the "Receive Filtering" section. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule. Note: This register field is sticky.

11.3.5.1.68 AMBA Multiple Outbound Decomposed NP SubRequests Control Register. (AMBA_MUL_OB_DECOMP_NP_SUB_REQ_CTRL_OFF)

11.3.5.1.68.1 Offset

Register	Offset
AMBA_MUL_OB_DECOMP_NP_SUB_REQ_CTRL_OFF	724h

11.3.5.1.68.2 Diagram



11.3.5.1.68.3 Fields

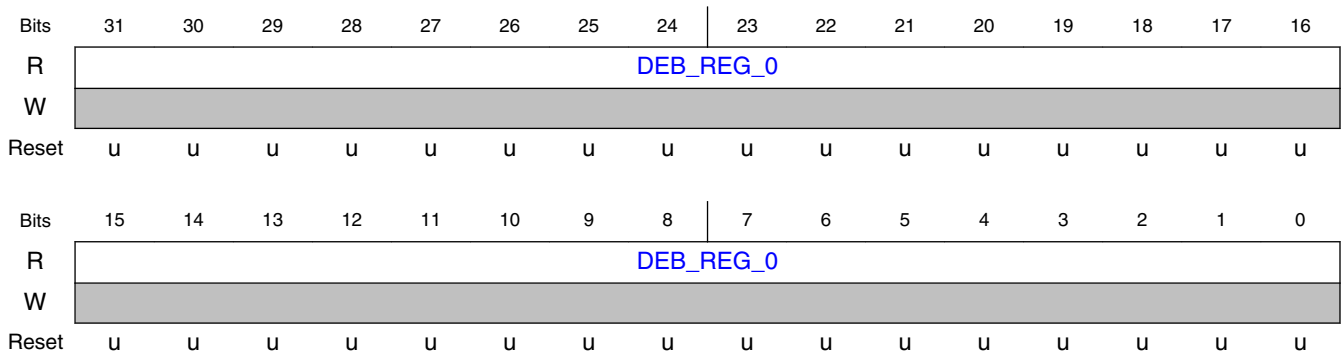
Field	Function
31-1 RSVDP_1	Reserved for future use.
0 OB_RD_SPLIT_BURST_EN	Enable AMBA Multiple Outbound Decomposed NP SubRequests. This bit when set to "0" disables the possibility of having multiple outstanding non-posted requests that were derived from decomposition of an outbound AMBA request. For more details, see "AXI Bridge Ordering" in the AXI chapter of the Databook. You should not clear this register unless your application master is requesting an amount of read data greater than Max_Read_Request_Size, and the remote device (or switch) is reordering completions that have different tags. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.

11.3.5.1.69 Debug Register 0 (PL_DEBUG0_OFF)

11.3.5.1.69.1 Offset

Register	Offset
PL_DEBUG0_OFF	728h

11.3.5.1.69.2 Diagram



11.3.5.1.69.3 Fields

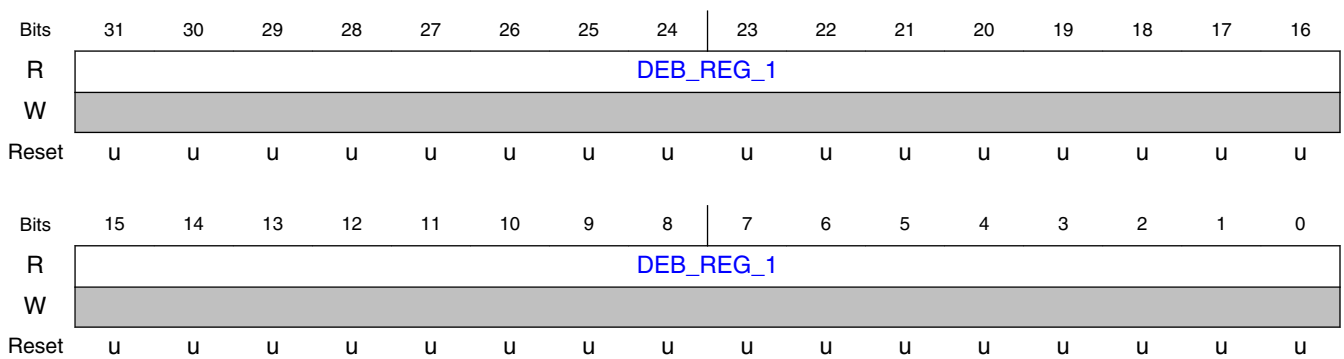
Field	Function
31-0 DEB_REG_0	The value on cxpl_debug_info[31:0].

11.3.5.1.70 Debug Register 1 (PL_DEBUG1_OFF)

11.3.5.1.70.1 Offset

Register	Offset
PL_DEBUG1_OFF	72Ch

11.3.5.1.70.2 Diagram



11.3.5.1.70.3 Fields

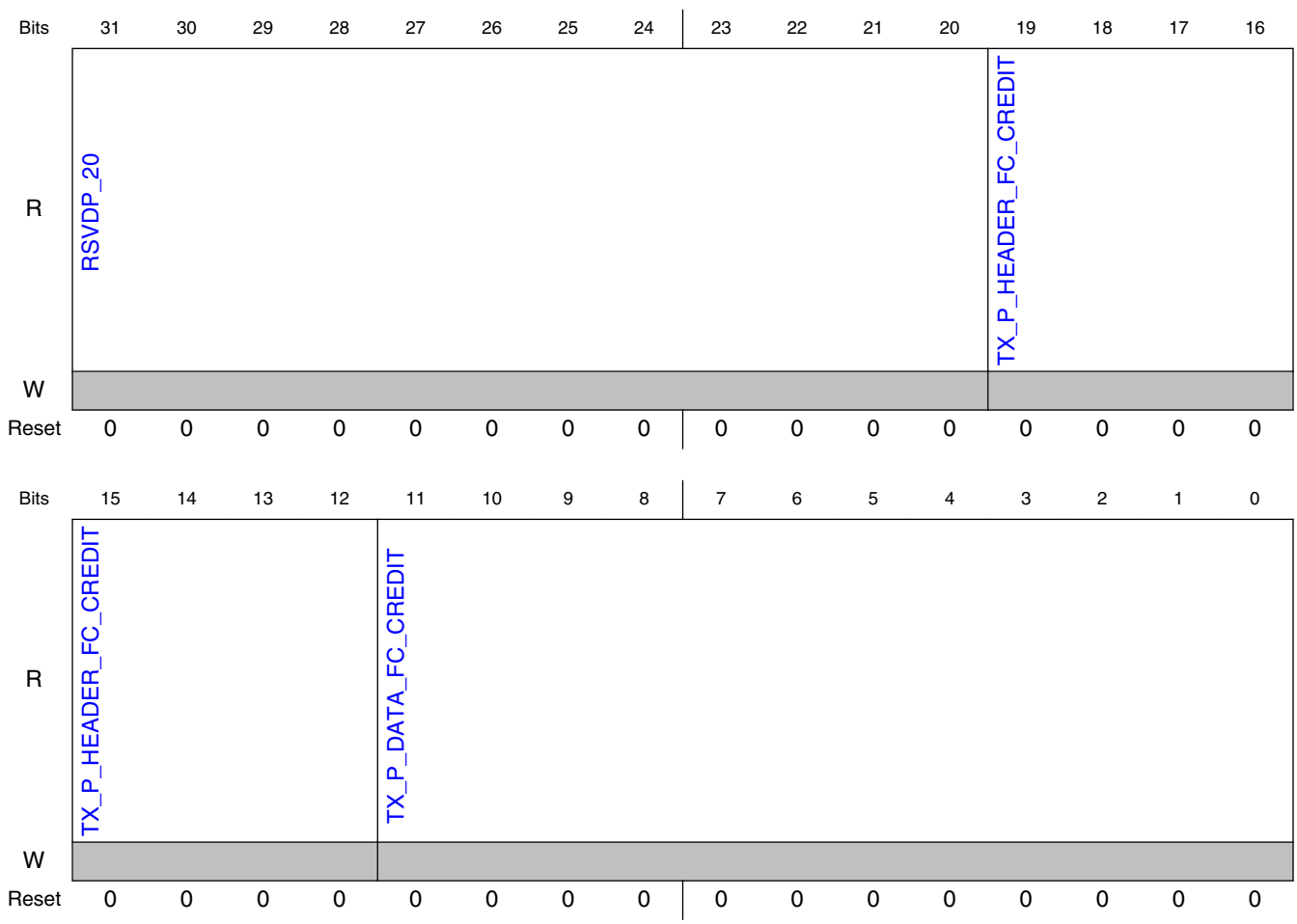
Field	Function
31-0 DEB_REG_1	The value on cxpl_debug_info[63:32].

11.3.5.1.71 Transmit Posted FC Credit Status (TX_P_FC_CREDIT_STATUS_OFF)

11.3.5.1.71.1 Offset

Register	Offset
TX_P_FC_CREDIT_STATUS_OFF	730h

11.3.5.1.71.2 Diagram



11.3.5.1.71.3 Fields

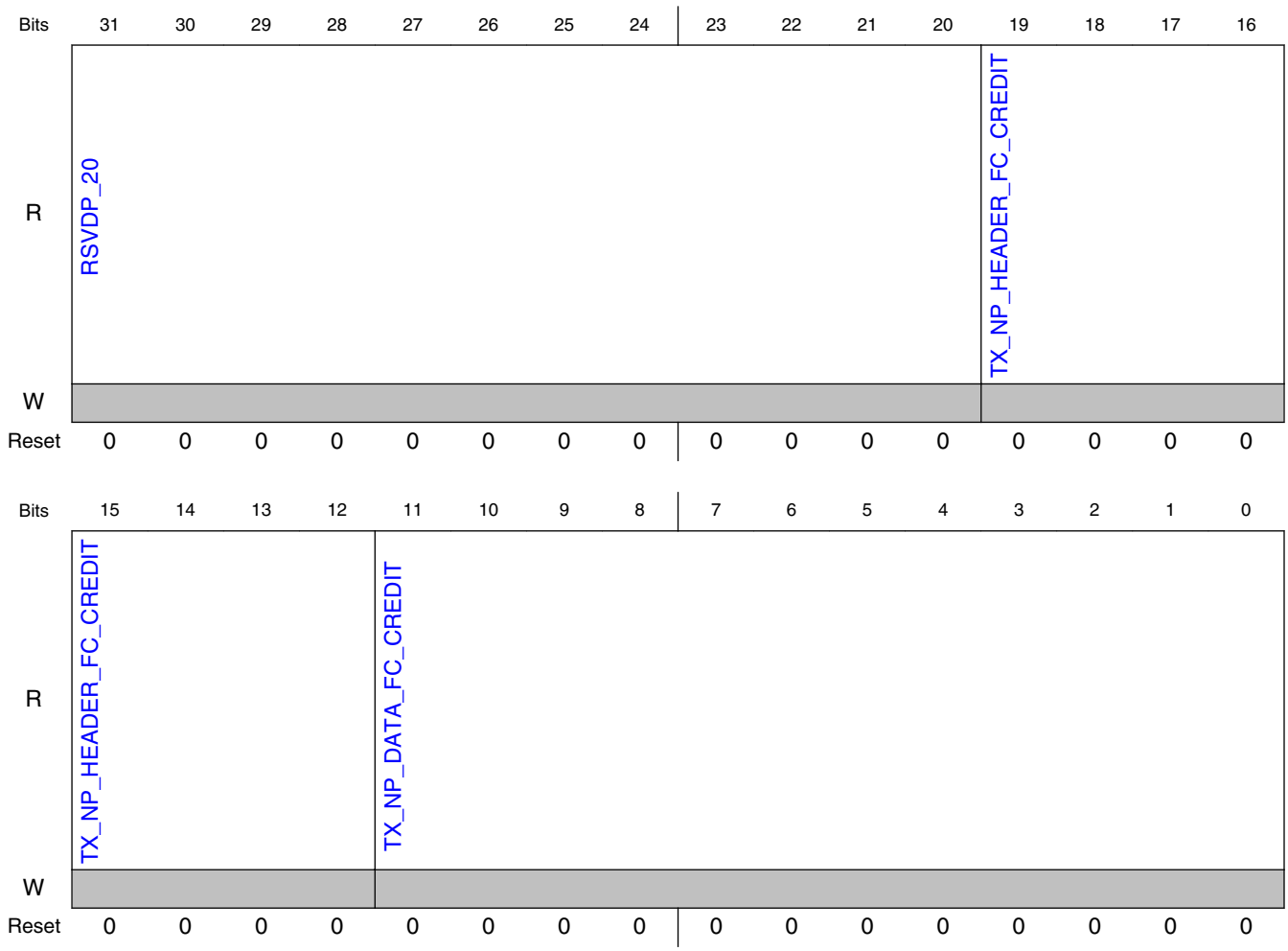
Field	Function
31-20 RSVDP_20	Reserved for future use.
19-12 TX_P_HEADER_FC_CREDIT	Transmit Posted Header FC Credits. The posted Header credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP. Default value depends on the number of advertised credits for header and data [12'b0, xtlh_xadm_ph_cdts, xtlh_xadm_pd_cdts]; When the number of advertised completion credits (both header and data) are infinite, then the default would be [12'b0, 8'hFF, 12'hFFF].
11-0 TX_P_DATA_FC_CREDIT	Transmit Posted Data FC Credits. The posted Data credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP. Default value depends on the number of advertised credits for header and data [12'b0, xtlh_xadm_ph_cdts, xtlh_xadm_pd_cdts]; When the number of advertised completion credits (both header and data) are infinite, then the default would be [12'b0, 8'hFF, 12'hFFF].

11.3.5.1.72 Transmit Non-Posted FC Credit Status (TX_NP_FC_CREDIT_STATUS_OFF)

11.3.5.1.72.1 Offset

Register	Offset
TX_NP_FC_CREDIT_STATUS_OFF	734h

11.3.5.1.72.2 Diagram



11.3.5.1.72.3 Fields

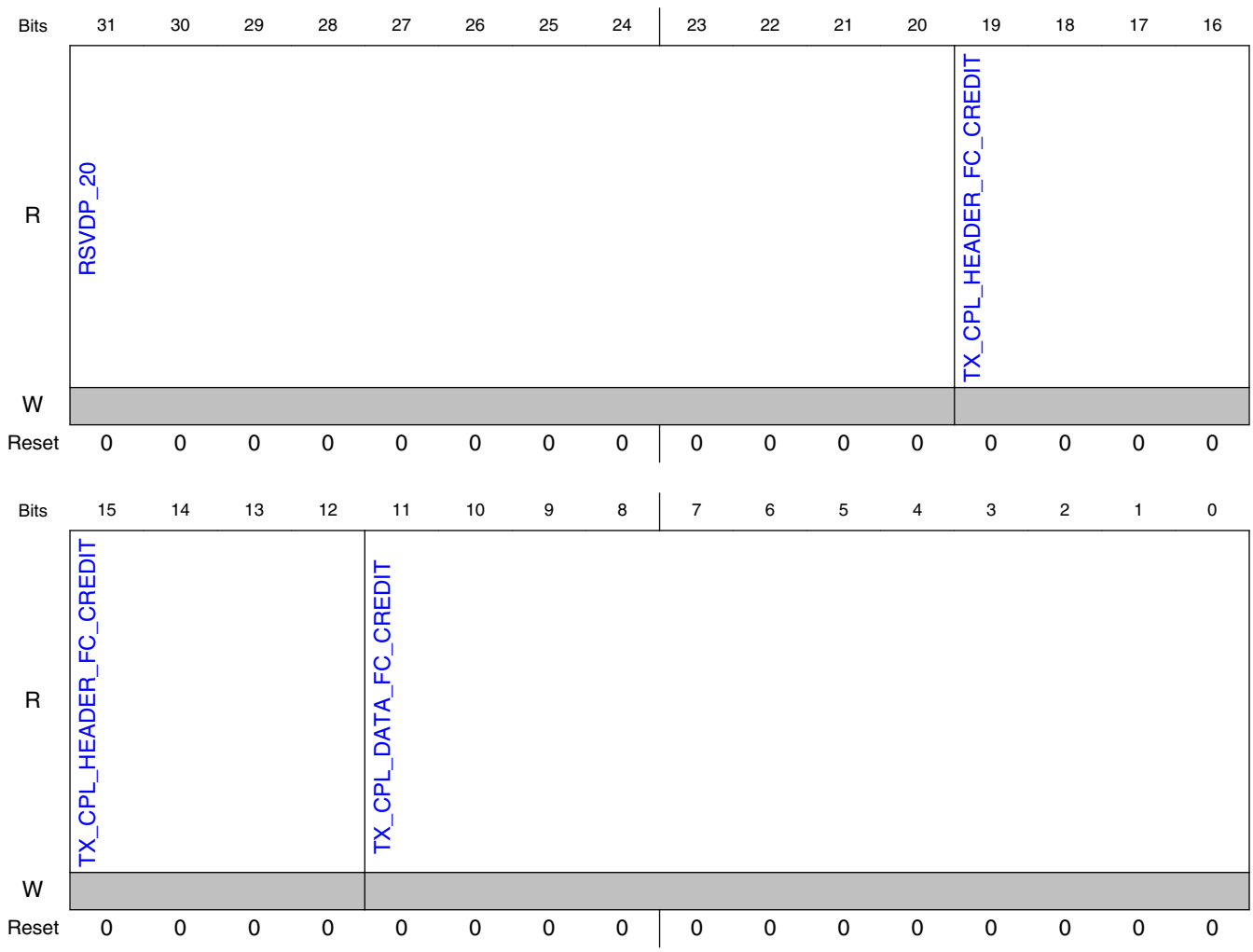
Field	Function
31-20 RSVDP_20	Reserved for future use.
19-12 TX_NP_HEADER_FC_CREDIT	Transmit Non-Posted Header FC Credits. The non-posted Header credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP. Default value depends on the number of advertised credits for header and data [12'b0, xtlh_xadm_nph_cdts, xtlh_xadm_npd_cdts]; When the number of advertised completion credits (both header and data) are infinite, then the default would be [12'b0, 8'hFF, 12'hFFF].
11-0 TX_NP_DATA_FC_CREDIT	Transmit Non-Posted Data FC Credits. The non-posted Data credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP. Default value depends on the number of advertised credits for header and data [12'b0, xtlh_xadm_nph_cdts, xtlh_xadm_npd_cdts]; When the number of advertised completion credits (both header and data) are infinite, then the default would be [12'b0, 8'hFF, 12'hFFF].

11.3.5.1.73 Transmit Completion FC Credit Status (TX_CPL_FC_CREDIT_STATUS_OFF)

11.3.5.1.73.1 Offset

Register	Offset
TX_CPL_FC_CREDIT_STATUS_OFF	738h

11.3.5.1.73.2 Diagram



11.3.5.1.73.3 Fields

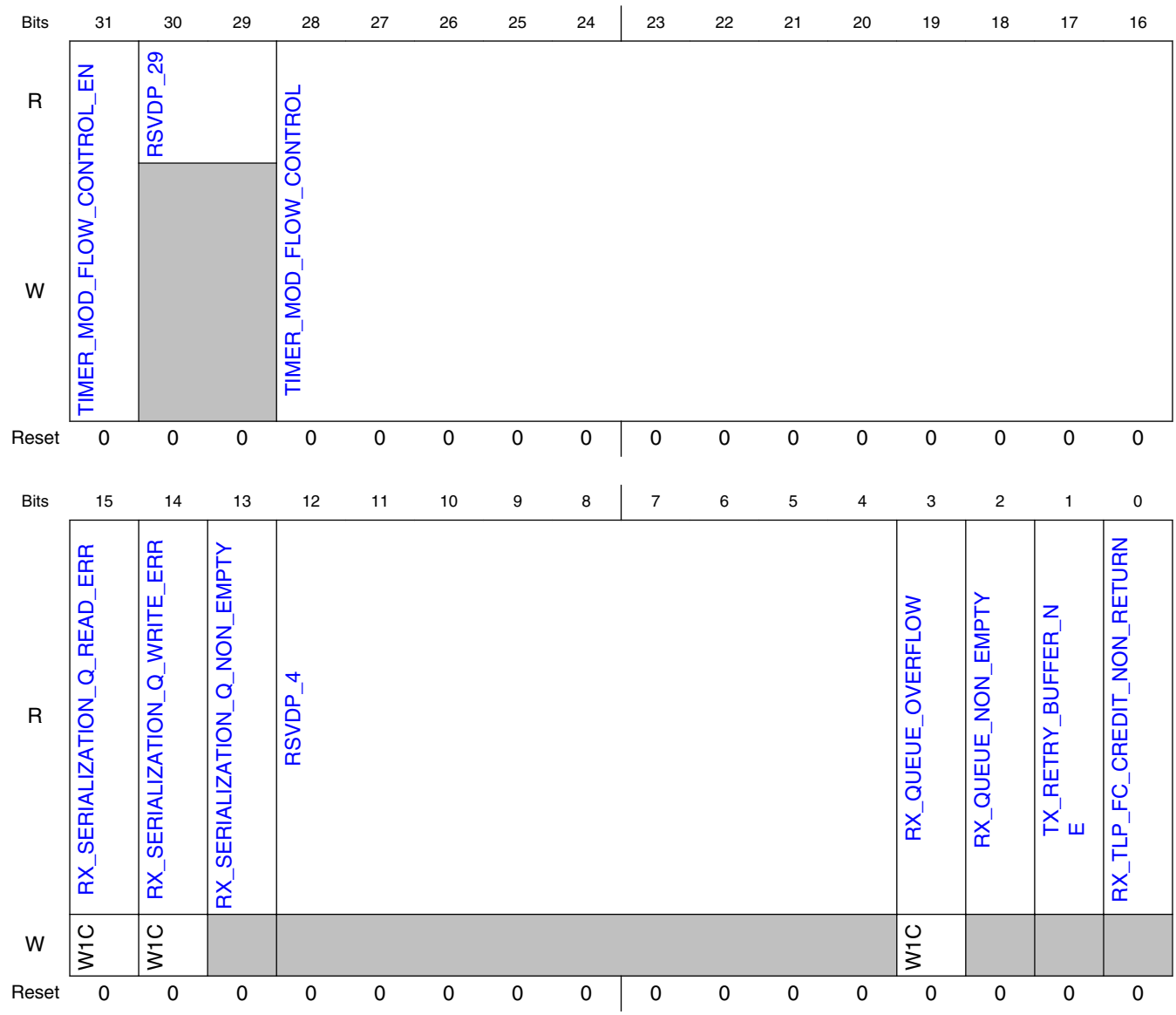
Field	Function
31-20 RSVDP_20	Reserved for future use.
19-12 TX_CPL_HEADER_FC_CREDIT	Transmit Completion Header FC Credits. The Completion Header credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP. Default value depends on the number of advertised credits for header and data [12'b0, xtlh_xadm_cplh_cdts, xtlh_xadm_cpld_cdts]; When the number of advertised completion credits (both header and data) are infinite, then the default would be [12'b0, 8'hFF, 12'hFFF].
11-0 TX_CPL_DATA_FC_CREDIT	Transmit Completion Data FC Credits. The Completion Data credits advertised by the receiver at the other end of the link, updated with each UpdateFC DLLP. Default value depends on the number of advertised credits for header and data [12'b0, xtlh_xadm_cplh_cdts, xtlh_xadm_cpld_cdts]; When the number of advertised completion credits (both header and data) are infinite, then the default would be [12'b0, 8'hFF, 12'hFFF].

11.3.5.1.74 Queue Status (QUEUE_STATUS_OFF)

11.3.5.1.74.1 Offset

Register	Offset
QUEUE_STATUS_OFF	73Ch

11.3.5.1.74.2 Diagram



11.3.5.1.74.3 Fields

Field	Function
31 TIMER_MOD_FLOW_CONTROL_EN	FC Latency Timer Override Enable. When this bit is set, the value from the "FC Latency Timer Override Value" field in this register will override the FC latency timer value that the controller calculates according to the PCIe specification. Note: This register field is sticky.
30-29 RSVDP_29	Reserved for future use.

Table continues on the next page...

Field	Function
28-16 TIMER_MOD_FLOW_CONTROL	FC Latency Timer Override Value. When you set the "FC Latency Timer Override Enable" in this register, the value in this field will override the FC latency timer value that the controller calculates according to the PCIe specification. For more details, see "Flow Control". Note: This register field is sticky.
15 RX_SERIALIZATION_Q_READ_ERR	Receive Serialization Read Error. Indicates the serialization queue has attempted to read an incorrectly formatted TLP.
14 RX_SERIALIZATION_Q_WRITE_ERR	Receive Serialization Queue Write Error. Indicates insufficient buffer space available to write to the serialization queue.
13 RX_SERIALIZATION_Q_NON_EMPTY	Receive Serialization Queue Not Empty. Indicates there is data in the serialization queue.
12-4 RSVDP_4	Reserved for future use.
3 RX_QUEUE_OVERFLOW	Receive Credit Queue Overflow. Indicates insufficient buffer space available to write to the P/NP/CPL credit queue.
2 RX_QUEUE_NON_EMPTY	Receive Credit Queue Not Empty. Indicates there is data in one or more of the receive buffers.
1 TX_RETRY_BUFFER_NE	Transmit Retry Buffer Not Empty. Indicates that there is data in the transmit retry buffer.
0 RX_TLP_FC_CREDIT_NON_RETURN	Received TLP FC Credits Not Returned. Indicates that the controller has received a TLP but has not yet sent an UpdateFC DLLP indicating that the credits for that TLP have been restored by the receiver at the other end of the link.

11.3.5.1.75 VC Transmit Arbitration Register 1 (VC_TX_ARBI_1_OFF)

11.3.5.1.75.1 Offset

Register	Offset
VC_TX_ARBI_1_OFF	740h

11.3.5.1.75.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WRR_WEIGHT_VC_3								WRR_WEIGHT_VC_2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WRR_WEIGHT_VC_1								WRR_WEIGHT_VC_0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

11.3.5.1.75.3 Fields

Field	Function
31-24 WRR_WEIGHT_VC_3	WRR Weight for VC3. Note: The access attributes of this field are as follows: - Dbi: R
23-16 WRR_WEIGHT_VC_2	WRR Weight for VC2. Note: The access attributes of this field are as follows: - Dbi: R
15-8 WRR_WEIGHT_VC_1	WRR Weight for VC1. Note: The access attributes of this field are as follows: - Dbi: R
7-0 WRR_WEIGHT_VC_0	WRR Weight for VC0. Note: The access attributes of this field are as follows: - Dbi: R

11.3.5.1.76 VC Transmit Arbitration Register 2 (VC_TX_ARBI_2_OFF)

11.3.5.1.76.1 Offset

Register	Offset
VC_TX_ARBI_2_OFF	744h

11.3.5.1.76.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WRR_WEIGHT_VC_7								WRR_WEIGHT_VC_6							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WRR_WEIGHT_VC_5								WRR_WEIGHT_VC_4							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.76.3 Fields

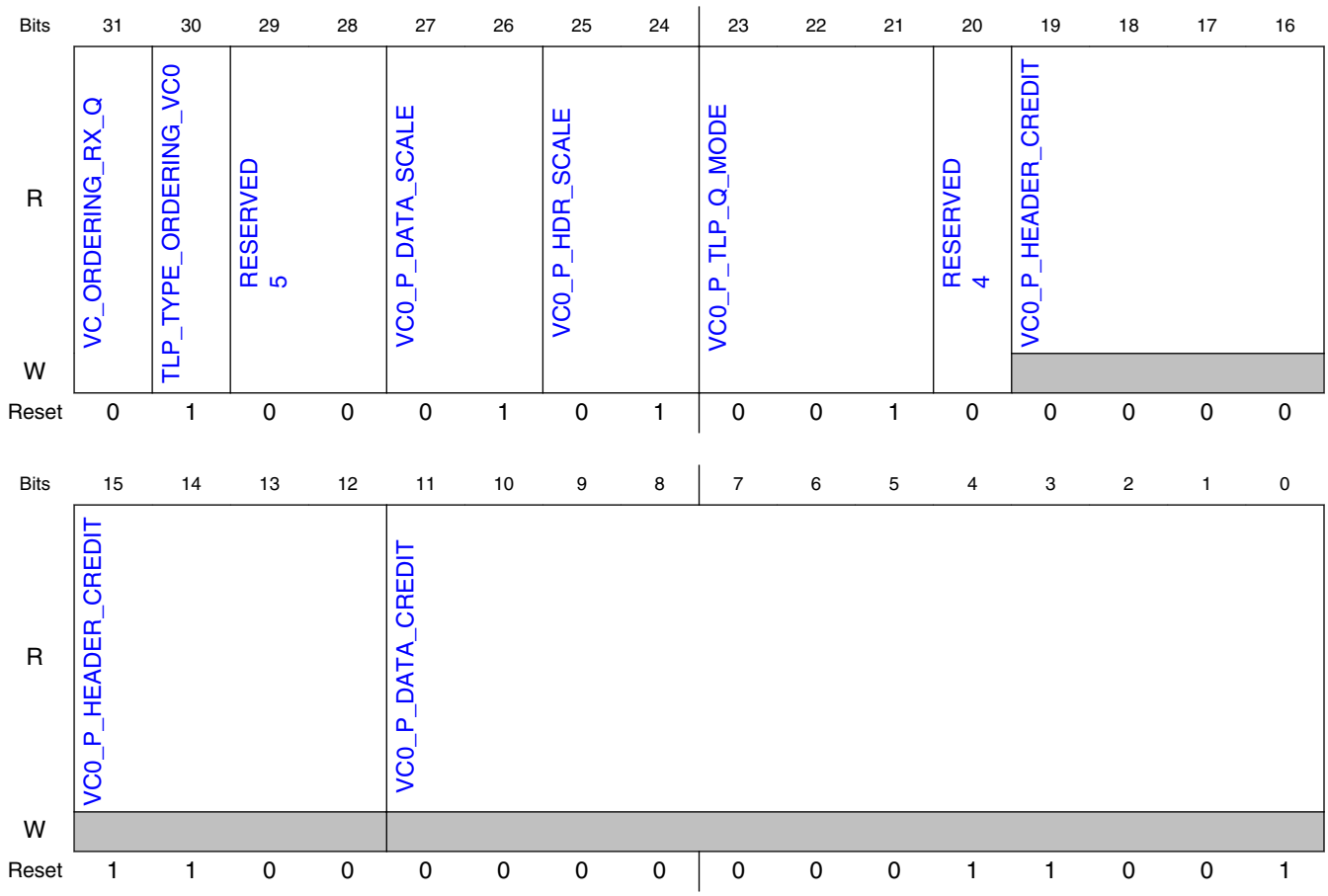
Field	Function
31-24 WRR_WEIGHT_VC_7	WRR Weight for VC7. Note: The access attributes of this field are as follows: - Dbi: R
23-16 WRR_WEIGHT_VC_6	WRR Weight for VC6. Note: The access attributes of this field are as follows: - Dbi: R
15-8 WRR_WEIGHT_VC_5	WRR Weight for VC5. Note: The access attributes of this field are as follows: - Dbi: R
7-0 WRR_WEIGHT_VC_4	WRR Weight for VC4. Note: The access attributes of this field are as follows: - Dbi: R

11.3.5.1.77 Segmented-Buffer VC0 Posted Receive Queue Control. (VC0_P_RX_Q_CTRL_OFF)

11.3.5.1.77.1 Offset

Register	Offset
VC0_P_RX_Q_CTRL_OFF	748h

11.3.5.1.77.2 Diagram



11.3.5.1.77.3 Fields

Field	Function
31 VC_ORDERING_RX_Q	VC Ordering for Receive Queues. Determines the VC ordering rule for the receive queues, used only in the segmented-buffer configuration: - 1: Strict ordering, higher numbered VCs have higher priority - 0: Round robin Note: This register field is sticky.
30 TLP_TYPE_ORDERING_VC0	TLP Type Ordering for VC0. Determines the TLP type ordering rule for VC0 receive queues, used only in the segmented-buffer configuration: - 1: PCIe ordering rules (recommended) - 0: Strict ordering: posted, completion, then non-posted Note: This register field is sticky.
29-28 RESERVED5	Reserved. Note: This register field is sticky.
27-26 VC0_P_DATA_SCALE	VC0 Scale Posted Data Credites. Note: This register field is sticky.
25-24 VC0_P_HDR_SCALE	VC0 Scale Posted Header Credites. Note: This register field is sticky.

Table continues on the next page...

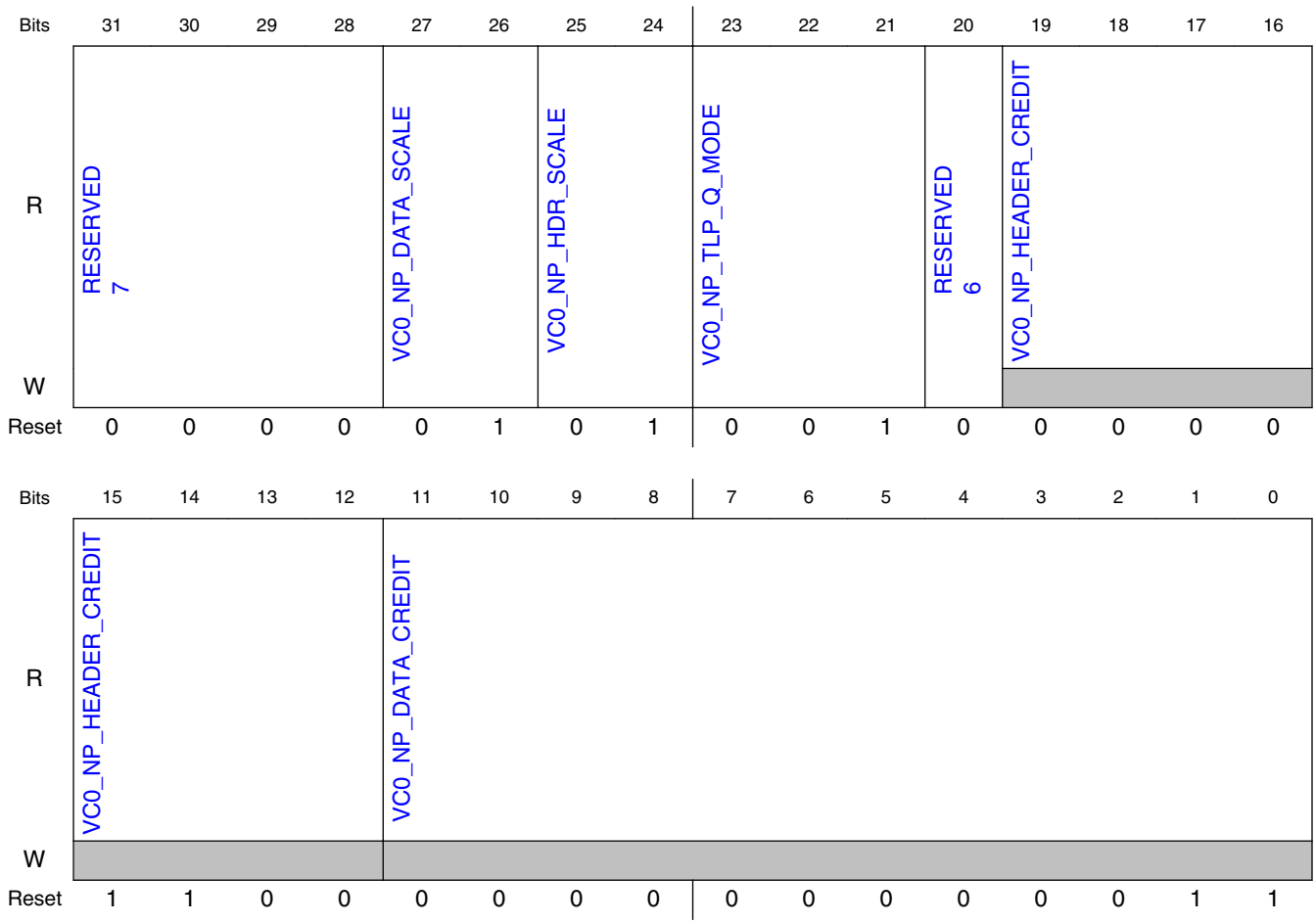
Field	Function
23-21 VC0_P_TLP_Q_MODE	Reserved. Note: This register field is sticky.
20 RESERVED4	Reserved. Note: This register field is sticky.
19-12 VC0_P_HEADER_CREDIT	VC0 Posted Header Credits. The number of initial posted header credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Dbi: R (sticky) Note: This register field is sticky.
11-0 VC0_P_DATA_CREDIT	VC0 Posted Data Credits. The number of initial posted data credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Dbi: R (sticky) Note: This register field is sticky.

11.3.5.1.78 Segmented-Buffer VC0 Non-Posted Receive Queue Control. (VC0_NP_RX_Q_CTRL_OFF)

11.3.5.1.78.1 Offset

Register	Offset
VC0_NP_RX_Q_CTRL_OFF	74Ch

11.3.5.1.78.2 Diagram



11.3.5.1.78.3 Fields

Field	Function
31-28 RESERVED7	Reserved. Note: This register field is sticky.
27-26 VC0_NP_DATA_SCALE	VC0 Scale Non-Posted Data Credites. Note: This register field is sticky.
25-24 VC0_NP_HDR_SCALE	VC0 Scale Non-Posted Header Credites. Note: This register field is sticky.
23-21 VC0_NP_TLP_Q_MODE	Reserved. Note: This register field is sticky.
20 RESERVED6	Reserved. Note: This register field is sticky.

Table continues on the next page...

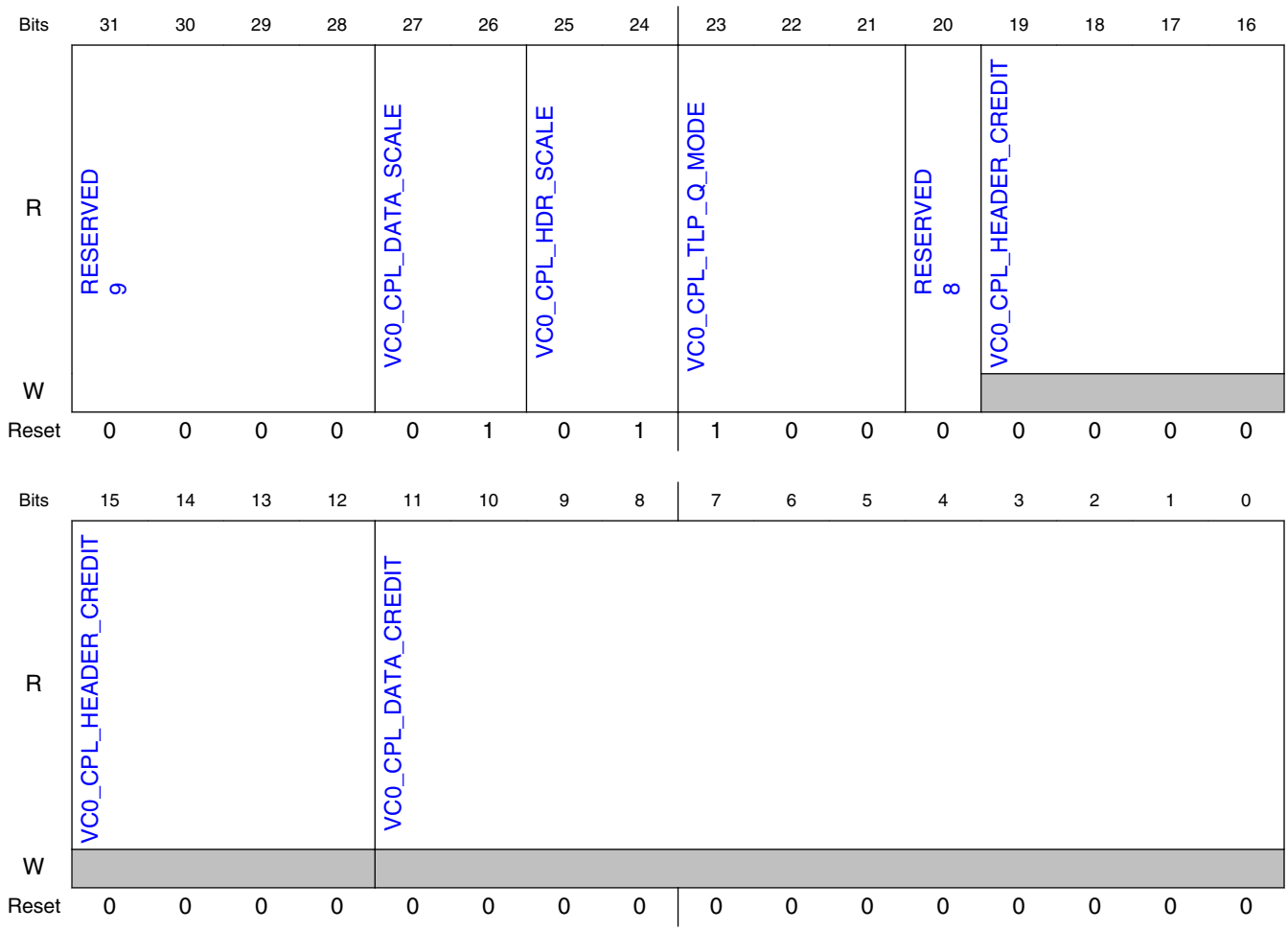
Field	Function
19-12 VC0_NP_HEADER_CREDIT	VC0 Non-Posted Header Credits. The number of initial non-posted header credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Dbi: R (sticky) Note: This register field is sticky.
11-0 VC0_NP_DATA_CREDIT	VC0 Non-Posted Data Credits. The number of initial non-posted data credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Dbi: R (sticky) Note: This register field is sticky.

11.3.5.1.79 Segmented-Buffer VC0 Completion Receive Queue Control. (VC0_CPL_RX_Q_CTRL_OFF)

11.3.5.1.79.1 Offset

Register	Offset
VC0_CPL_RX_Q_CTRL_OFF	750h

11.3.5.1.79.2 Diagram



11.3.5.1.79.3 Fields

Field	Function
31-28 RESERVED9	Reserved. Note: This register field is sticky.
27-26 VC0_CPL_DATA_SCALE	VC0 Scale CPL Data Credites. Note: This register field is sticky.
25-24 VC0_CPL_HDR_SCALE	VC0 Scale CPL Header Credites. Note: This register field is sticky.
23-21 VC0_CPL_TLP_Q_MODE	Reserved. Note: This register field is sticky.
20	Reserved. Note: This register field is sticky.

Table continues on the next page...

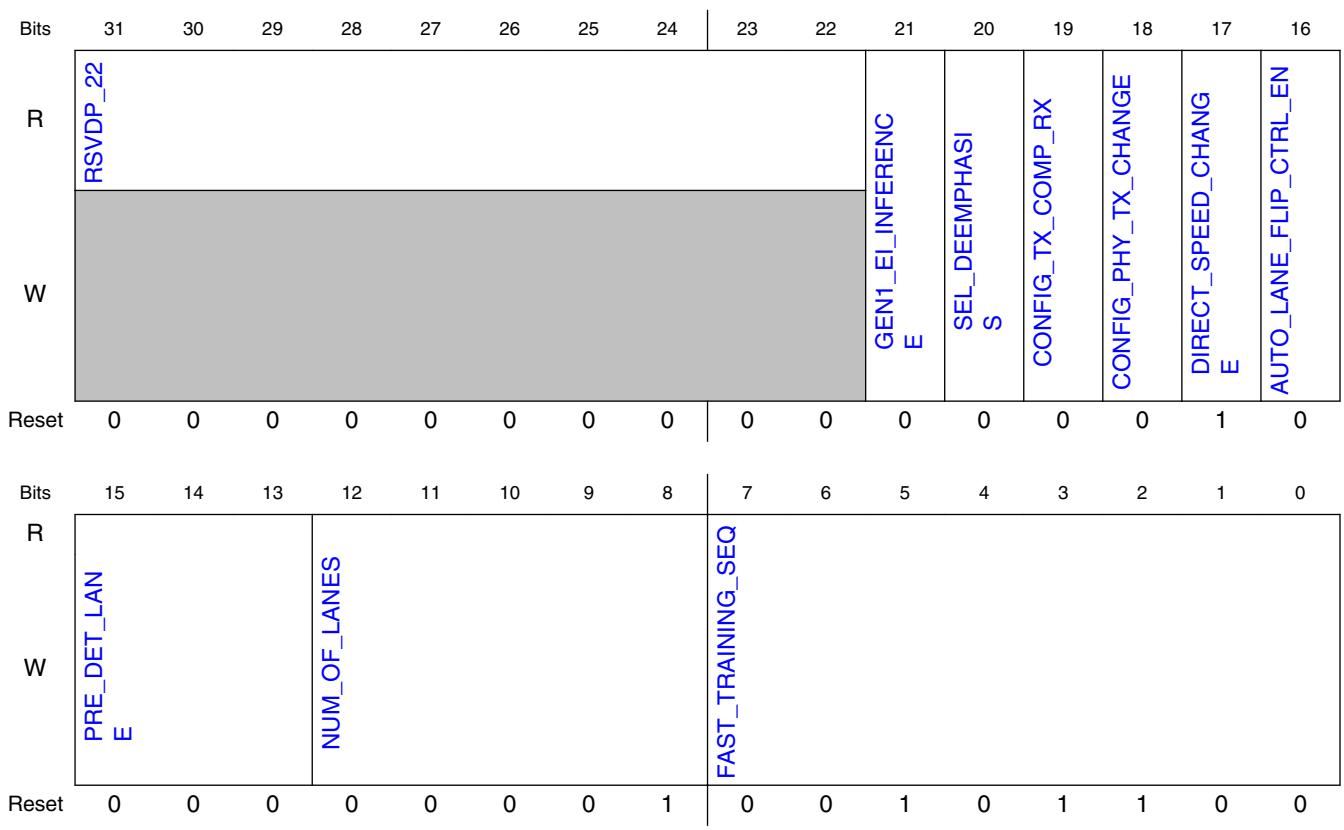
Field	Function
RESERVED8	
19-12 VC0_CPL_HEADER_CREDIT	VC0 Completion Header Credits. The number of initial Completion header credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Dbi: R (sticky) Note: This register field is sticky.
11-0 VC0_CPL_DATA_CREDIT	VC0 Completion Data Credits. The number of initial Completion data credits for VC0, used only in the segmented-buffer configuration. Note: The access attributes of this field are as follows: - Dbi: R (sticky) Note: This register field is sticky.

11.3.5.1.80 Link Width and Speed Change Control Register. (GEN2_CTRL_OFF)

11.3.5.1.80.1 Offset

Register	Offset
GEN2_CTRL_OFF	80Ch

11.3.5.1.80.2 Diagram



11.3.5.1.80.3 Fields

Field	Function
31-22 RSVDP_22	Reserved for future use.
21 GEN1_EI_INFERENCE	Electrical Idle Inference Mode at Gen1 Rate. Programmable mode to determine inferred electrical idle (EI) in Recovery.Speed or Loopback.Active (as slave) state at Gen1 speed by looking for a "1" value on RxElecIdle instead of looking for a "0" on RxValid. If the PHY fails to deassert the RxValid signal in Recovery.Speed or Loopback.Active (because of corrupted EIOS for example), then EI cannot be inferred successfully in the controller by just detecting the condition RxValid=0. - 0: Use RxElecIdle signal to infer Electrical Idle - 1: Use RxValid signal to infer Electrical Idle Note: This register field is sticky.
20 SEL_DEEMPHASIS	Used to set the de-emphasis level for upstream ports. This bit selects the level of de-emphasis the link operates at. - 0: -6 dB - 1: -3.5 dB This field is reserved (fixed to '0') for M-PCIe. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
19 CONFIG_TX_COMP_RX	Config Tx Compliance Receive Bit. When set to 1, signals LTSSM to transmit TS ordered sets with the compliance receive bit assert (equal to "1"). This field is reserved (fixed to '0') for M-PCIe. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
18 CONFIG_PHY_TX_CHANGE	Config PHY Tx Swing. Controls the PHY transmitter voltage swing level. The controller drives the mac_phy_txswing output from this register bit field. - 0: Full Swing - 1: Low Swing This field is reserved (fixed to '0') for M-PCIe. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
17 DIRECT_SPEED_CHANGE	Directed Speed Change. Writing "1" to this field instructs the LTSSM to initiate a speed change to Gen2 or Gen3 after the link is initialized at Gen1 speed. When the speed change occurs, the controller will clear the contents of this field; and a read to this field by your software will return a "0". To manually initiate the speed change: - Write to LINK_CONTROL2_LINK_STATUS2_REG[PCIE_CAP_TARGET_LINK_SPEED] in the local device, deassert this field, and then assert this field. This field is reserved (fixed to '0') for M-PCIe. Note: The access attributes of this field are as follows: - Dbi: R/W
16 AUTO_LANE_FLIP_CTRL_EN	Enable Auto flipping of the lanes. This field is reserved (fixed to '0') for M-PCIe. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
15-13 PRE_DET_LANE	Predetermined Lane for Auto Flip. This field defines which physical lane is connected to logical Lane0 by the flip operation performed in Detect. Allowed values are: - 3'b000: Connect logical Lane0 to physical lane 0 - 3'b001: Connect logical Lane0 to physical lane 1 - 3'b010: Connect logical Lane0 to physical lane 3 - 3'b011: Connect logical Lane0 to physical lane 7 - 3'b100: Connect logical Lane0 to physical lane 15 This field is used to restrict the receiver detect procedure to a particular lane when the default detect and polling procedure performed on all lanes cannot be successful. A notable example of when it is useful to program this field to a value different from the default, is when a lane is asymmetrically broken, that is, it is detected in Detect LTSSM state but it cannot exit Electrical Idle in Polling LTSSM state. Note: This field is reserved (fixed to '0') for M-PCIe. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
12-8 NUM_OF_LANES	Predetermined Number of Lanes. Defines the number of lanes which are connected and not bad. Used to limit the effective link width to ignore 'broken" or "unused" lanes that detect a receiver. Indicates the number of lanes to check for exit from Electrical Idle in Polling.Active and L2.Idle. It is possible that the LTSSM might detect a receiver on a bad or broken lane during the Detect Substate. However, it is also possible that such a lane might also fail to exit Electrical Idle and therefore prevent a valid link from being configured. This value is referred to as the "Predetermined Number of Lanes" in section 4.2.6.2.1 of the PCI Express Base 3.0 Specification, revision 1.0. Encoding is as follows: - 0x01: 1 lane - 0x02: 2 lanes - 0x03: 3 lanes - .. When you have unused lanes in your system, then you must change the value in this register to reflect the number of lanes. You must also change the value in the "Link Mode Enable" field of PORT_LINK_CTRL_OFF. The value in this register is normally the same as the encoded value in

Table continues on the next page...

Field	Function
	PORT_LINK_CTRL_OFF. If you find that one of your used lanes is bad then you must reduce the value in this register. For more information, see "How to Tie Off Unused Lanes." For information on upsizing and downsizing the link width, see "Link Establishment." This field is reserved (fixed to '0') for M-PCIe. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.
7-0 FAST_TRAINING_SEQ	Sets the Number of Fast Training Sequences (N_FTS) that the controller advertises as its N_FTS during Gen2 or Gen3 link training. This value is used to inform the link partner about the PHY's ability to recover synchronization after a low power state. The number should be provided by the PHY vendor. Do not set N_FTS to zero; doing so can cause the LTSSM to go into the recovery state when exiting from L0s. This field is reserved (fixed to '0') for M-PCIe. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.

11.3.5.1.81 PHY Status Register. (PHY_STATUS_OFF)

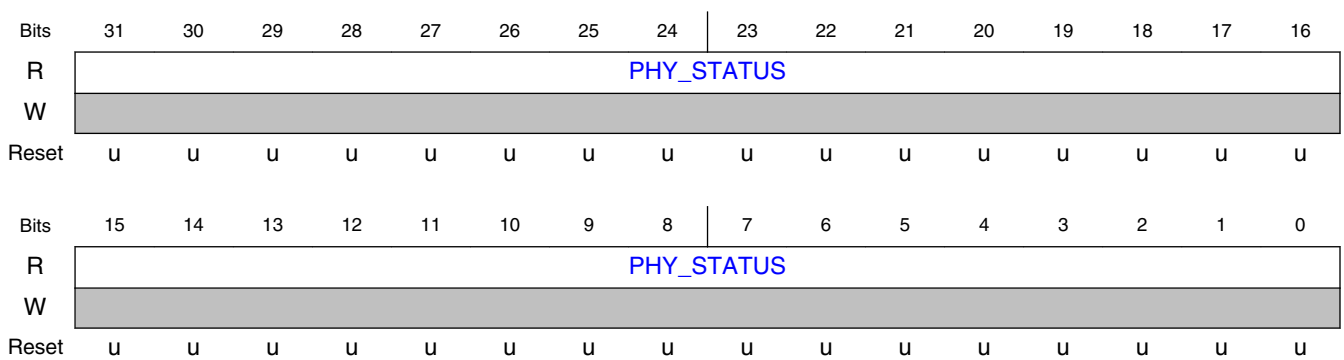
11.3.5.1.81.1 Offset

Register	Offset
PHY_STATUS_OFF	810h

11.3.5.1.81.2 Function

PHY Status Register. Memory mapped register from phy_cfg_status GPIO input pins.

11.3.5.1.81.3 Diagram



11.3.5.1.81.4 Fields

Field	Function
31-0 PHY_STATUS	PHY Status. Data received directly from the phy_cfg_status bus. These is a GPIO register reflecting the values on the static phy_cfg_status input signals. The usage is left completely to the user and does not in any way influence controller functionality. You can use it for any static sideband status signalling

Field	Function
	requirements that you have for your PHY. This field is reserved (fixed to '0') for M-PCIe. Note: This register field is sticky.

11.3.5.1.82 PHY Control Register. (PHY_CONTROL_OFF)

11.3.5.1.82.1 Offset

Register	Offset
PHY_CONTROL_OFF	814h

11.3.5.1.82.2 Function

PHY Control Register. Memory mapped register to cfg_phy_control GPIO output pins.

11.3.5.1.82.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PHY_CONTROL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PHY_CONTROL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.82.4 Fields

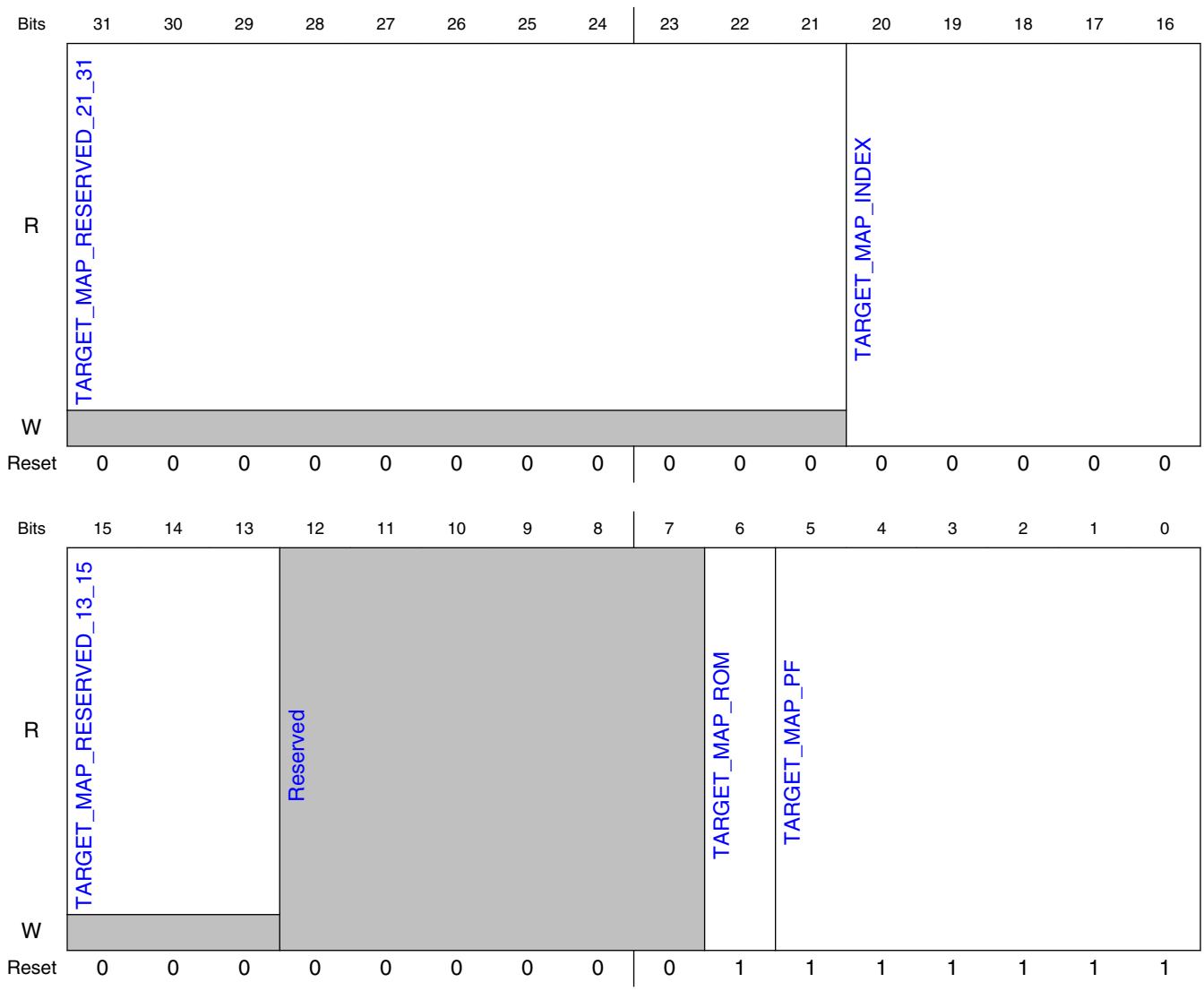
Field	Function
31-0 PHY_CONTROL	PHY Control. Data sent directly to the cfg_phy_control bus. These is a GPIO register driving the values on the static cfg_phy_control output signals. The usage is left completely to the user and does not in any way influence controller functionality. You can use it for any static sideband control signalling requirements that you have for your PHY. This field is reserved (fixed to '0') for M-PCIe. Note: This register field is sticky.

11.3.5.1.83 Programmable Target Map Control Register. (TRGT_MAP_CTRL_OFF)

11.3.5.1.83.1 Offset

Register	Offset
TRGT_MAP_CTRL_OFF	81Ch

11.3.5.1.83.2 Diagram



11.3.5.1.83.3 Fields

Field	Function
31-21 TARGET_MAP_RESERVED_21_31	Reserved. Note: The access attributes of this field are as follows: - Dbi: R (sticky)
20-16 TARGET_MAP_INDEX	The number of the PF Function on which the Target Values are set. This register does not respect the Byte Enable setting. any write will affect all register bits.
15-13 TARGET_MAP_RESERVED_13_15	Reserved. Note: The access attributes of this field are as follows: - Dbi: R (sticky)
12-7 —	Reserved.
6 TARGET_MAP_ROM	Target Value for the ROM page of the PF Function selected by the index number. This register does not respect the Byte Enable setting. any write will affect all register bits.
5-0 TARGET_MAP_PF	Target Values for each BAR on the PF Function selected by the index number. This register does not respect the Byte Enable setting. any write will affect all register bits.

11.3.5.1.84 Integrated MSI Reception Module (iMRM) Address Register. (MSI_CTRL_ADDR_OFF)

11.3.5.1.84.1 Offset

Register	Offset
MSI_CTRL_ADDR_OFF	820h

11.3.5.1.84.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.84.3 Fields

Field	Function
31-0 MSI_CTRL_ADDR	Integrated MSI Reception Module Address. System specified address for MSI memory write transaction termination. Within the AXI Bridge, every received Memory Write request is examined to see if it targets the MSI Address that has been specified in this register; and also to see if it satisfies the definition of an MSI interrupt request. When these conditions are satisfied the Memory Write request is marked as an MSI request. Note: This register field is sticky.

11.3.5.1.85 Integrated MSI Reception Module Upper Address Register. (MSI_CTRL_UPPER_ADDR_OFF)

11.3.5.1.85.1 Offset

Register	Offset
MSI_CTRL_UPPER_ADDR_OFF	824h

11.3.5.1.85.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_UPPER_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_UPPER_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.85.3 Fields

Field	Function
31-0 MSI_CTRL_UPPER_ADDR	Integrated MSI Reception Module Upper Address. System specified upper address for MSI memory write transaction termination. Allows functions to support a 64-bit MSI address. Note: This register field is sticky.

11.3.5.1.86 Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_0_EN_OFF)

11.3.5.1.86.1 Offset

Register	Offset
MSI_CTRL_INT_0_EN_OFF	828h

11.3.5.1.86.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_0_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_0_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.86.3 Fields

Field	Function
31-0 MSI_CTRL_INT_0_EN	MSI Interrupt#i Enable. Specifies which interrupts are enabled. When an MSI is received from a disabled interrupt, no status bit gets set in MSI controller interrupt status register. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.87 Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_0_MASK_OFF)

11.3.5.1.87.1 Offset

Register	Offset
MSI_CTRL_INT_0_MASK_OFF	82Ch

11.3.5.1.87.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_0_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_0_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.87.3 Fields

Field	Function
31-0 MSI_CTRL_INT_0_MASK	MSI Interrupt#i Mask. Allows enabled interrupts to be masked. When an MSI is received for a masked interrupt, the corresponding status bit gets set in the interrupt status register but the msi_ctrl_int output is not set HIGH. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.88 Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_0_STATUS_OFF)

11.3.5.1.88.1 Offset

Register	Offset
MSI_CTRL_INT_0_STATUS_OFF	830h

11.3.5.1.88.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MSI_CTRL_INT_0_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MSI_CTRL_INT_0_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.88.3 Fields

Field	Function
31-0 MSI_CTRL_INT_0_STATUS	MSI Interrupt#i Status. When an MSI is detected for EP#i, one bit in this register is set. The decoding of the data payload of the MSI Memory Write request determines which bit gets set. A status is bit is cleared by writing a 1 to the bit. Each bit corresponds to a single MSI Interrupt Vector.

11.3.5.1.89 Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_1_EN_OFF)

11.3.5.1.89.1 Offset

Register	Offset
MSI_CTRL_INT_1_EN_OFF	834h

11.3.5.1.89.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_1_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_1_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.89.3 Fields

Field	Function
31-0 MSI_CTRL_INT_1_EN	MSI Interrupt#i Enable. Specifies which interrupts are enabled. When an MSI is received from a disabled interrupt, no status bit gets set in MSI controller interrupt status register. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.90 Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_1_MASK_OFF)

11.3.5.1.90.1 Offset

Register	Offset
MSI_CTRL_INT_1_MASK_OFF	838h

11.3.5.1.90.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_1_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_1_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.90.3 Fields

Field	Function
31-0 MSI_CTRL_INT_1_MASK	MSI Interrupt#i Mask. Allows enabled interrupts to be masked. When an MSI is received for a masked interrupt, the corresponding status bit gets set in the interrupt status register but the msi_ctrl_int output is not set HIGH. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.91 Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_1_STATUS_OFF)

11.3.5.1.91.1 Offset

Register	Offset
MSI_CTRL_INT_1_STATUS_OFF	83Ch

11.3.5.1.91.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MSI_CTRL_INT_1_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MSI_CTRL_INT_1_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.91.3 Fields

Field	Function
31-0 MSI_CTRL_INT_1_STATUS	MSI Interrupt#i Status. When an MSI is detected for EP#i, one bit in this register is set. The decoding of the data payload of the MSI Memory Write request determines which bit gets set. A status is bit is cleared by writing a 1 to the bit. Each bit corresponds to a single MSI Interrupt Vector.

11.3.5.1.92 Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_2_EN_OFF)

11.3.5.1.92.1 Offset

Register	Offset
MSI_CTRL_INT_2_EN_OFF	840h

11.3.5.1.92.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_2_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_2_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.92.3 Fields

Field	Function
31-0 MSI_CTRL_INT_2_EN	MSI Interrupt#i Enable. Specifies which interrupts are enabled. When an MSI is received from a disabled interrupt, no status bit gets set in MSI controller interrupt status register. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.93 Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_2_MASK_OFF)

11.3.5.1.93.1 Offset

Register	Offset
MSI_CTRL_INT_2_MASK_OFF	844h

11.3.5.1.93.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_2_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_2_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.93.3 Fields

Field	Function
31-0 MSI_CTRL_INT_2_MASK	MSI Interrupt#i Mask. Allows enabled interrupts to be masked. When an MSI is received for a masked interrupt, the corresponding status bit gets set in the interrupt status register but the msi_ctrl_int output is not set HIGH. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.94 Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_2_STATUS_OFF)

11.3.5.1.94.1 Offset

Register	Offset
MSI_CTRL_INT_2_STATUS_OFF	848h

11.3.5.1.94.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MSI_CTRL_INT_2_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MSI_CTRL_INT_2_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.94.3 Fields

Field	Function
31-0 MSI_CTRL_INT_2_STATUS	MSI Interrupt#i Status. When an MSI is detected for EP#i, one bit in this register is set. The decoding of the data payload of the MSI Memory Write request determines which bit gets set. A status is bit is cleared by writing a 1 to the bit. Each bit corresponds to a single MSI Interrupt Vector.

11.3.5.1.95 Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_3_EN_OFF)

11.3.5.1.95.1 Offset

Register	Offset
MSI_CTRL_INT_3_EN_OFF	84Ch

11.3.5.1.95.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_3_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_3_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.95.3 Fields

Field	Function
31-0 MSI_CTRL_INT_3_EN	MSI Interrupt#i Enable. Specifies which interrupts are enabled. When an MSI is received from a disabled interrupt, no status bit gets set in MSI controller interrupt status register. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.96 Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_3_MASK_OFF)

11.3.5.1.96.1 Offset

Register	Offset
MSI_CTRL_INT_3_MASK_OFF	850h

11.3.5.1.96.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_3_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_3_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.96.3 Fields

Field	Function
31-0 MSI_CTRL_INT_3_MASK	MSI Interrupt#i Mask. Allows enabled interrupts to be masked. When an MSI is received for a masked interrupt, the corresponding status bit gets set in the interrupt status register but the msi_ctrl_int output is not set HIGH. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.97 Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_3_STATUS_OFF)

11.3.5.1.97.1 Offset

Register	Offset
MSI_CTRL_INT_3_STATUS_OFF	854h

11.3.5.1.97.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MSI_CTRL_INT_3_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MSI_CTRL_INT_3_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.97.3 Fields

Field	Function
31-0 MSI_CTRL_INT_3_STATUS	MSI Interrupt#i Status. When an MSI is detected for EP#i, one bit in this register is set. The decoding of the data payload of the MSI Memory Write request determines which bit gets set. A status is bit is cleared by writing a 1 to the bit. Each bit corresponds to a single MSI Interrupt Vector.

11.3.5.1.98 Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_4_EN_OFF)

11.3.5.1.98.1 Offset

Register	Offset
MSI_CTRL_INT_4_EN_OFF	858h

11.3.5.1.98.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_4_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_4_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.98.3 Fields

Field	Function
31-0 MSI_CTRL_INT_4_EN	MSI Interrupt#i Enable. Specifies which interrupts are enabled. When an MSI is received from a disabled interrupt, no status bit gets set in MSI controller interrupt status register. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.99 Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_4_MASK_OFF)

11.3.5.1.99.1 Offset

Register	Offset
MSI_CTRL_INT_4_MASK_OFF	85Ch

11.3.5.1.99.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_4_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_4_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.99.3 Fields

Field	Function
31-0 MSI_CTRL_INT_4_MASK	MSI Interrupt#i Mask. Allows enabled interrupts to be masked. When an MSI is received for a masked interrupt, the corresponding status bit gets set in the interrupt status register but the msi_ctrl_int output is not set HIGH. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.100 Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_4_STATUS_OFF)

11.3.5.1.100.1 Offset

Register	Offset
MSI_CTRL_INT_4_STATUS_OFF	860h

11.3.5.1.100.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MSI_CTRL_INT_4_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MSI_CTRL_INT_4_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.100.3 Fields

Field	Function
31-0 MSI_CTRL_INT_4_STATUS	MSI Interrupt#i Status. When an MSI is detected for EP#i, one bit in this register is set. The decoding of the data payload of the MSI Memory Write request determines which bit gets set. A status is bit is cleared by writing a 1 to the bit. Each bit corresponds to a single MSI Interrupt Vector.

11.3.5.1.101 Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_5_EN_OFF)

11.3.5.1.101.1 Offset

Register	Offset
MSI_CTRL_INT_5_EN_OFF	864h

11.3.5.1.101.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_5_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_5_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.101.3 Fields

Field	Function
31-0 MSI_CTRL_INT_5_EN	MSI Interrupt#i Enable. Specifies which interrupts are enabled. When an MSI is received from a disabled interrupt, no status bit gets set in MSI controller interrupt status register. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.102 Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_5_MASK_OFF)

11.3.5.1.102.1 Offset

Register	Offset
MSI_CTRL_INT_5_MASK_OFF	868h

11.3.5.1.102.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_5_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_5_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.102.3 Fields

Field	Function
31-0 MSI_CTRL_INT_5_MASK	MSI Interrupt#i Mask. Allows enabled interrupts to be masked. When an MSI is received for a masked interrupt, the corresponding status bit gets set in the interrupt status register but the msi_ctrl_int output is not set HIGH. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.103 Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_5_STATUS_OFF)

11.3.5.1.103.1 Offset

Register	Offset
MSI_CTRL_INT_5_STATUS_OFF	86Ch

11.3.5.1.103.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MSI_CTRL_INT_5_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MSI_CTRL_INT_5_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.103.3 Fields

Field	Function
31-0 MSI_CTRL_INT_5_STATUS	MSI Interrupt#i Status. When an MSI is detected for EP#i, one bit in this register is set. The decoding of the data payload of the MSI Memory Write request determines which bit gets set. A status is bit is cleared by writing a 1 to the bit. Each bit corresponds to a single MSI Interrupt Vector.

11.3.5.1.104 Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_6_EN_OFF)

11.3.5.1.104.1 Offset

Register	Offset
MSI_CTRL_INT_6_EN_OFF	870h

11.3.5.1.104.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_6_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_6_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.104.3 Fields

Field	Function
31-0 MSI_CTRL_INT_6_EN	MSI Interrupt#i Enable. Specifies which interrupts are enabled. When an MSI is received from a disabled interrupt, no status bit gets set in MSI controller interrupt status register. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.105 Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_6_MASK_OFF)

11.3.5.1.105.1 Offset

Register	Offset
MSI_CTRL_INT_6_MASK_OFF	874h

11.3.5.1.105.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_6_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_6_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.105.3 Fields

Field	Function
31-0 MSI_CTRL_INT_6_MASK	MSI Interrupt#i Mask. Allows enabled interrupts to be masked. When an MSI is received for a masked interrupt, the corresponding status bit gets set in the interrupt status register but the msi_ctrl_int output is not set HIGH. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.106 Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_6_STATUS_OFF)

11.3.5.1.106.1 Offset

Register	Offset
MSI_CTRL_INT_6_STATUS_OFF	878h

11.3.5.1.106.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MSI_CTRL_INT_6_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MSI_CTRL_INT_6_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.106.3 Fields

Field	Function
31-0 MSI_CTRL_INT_6_STATUS	MSI Interrupt#i Status. When an MSI is detected for EP#i, one bit in this register is set. The decoding of the data payload of the MSI Memory Write request determines which bit gets set. A status is bit is cleared by writing a 1 to the bit. Each bit corresponds to a single MSI Interrupt Vector.

11.3.5.1.107 Integrated MSI Reception Module Interrupt#i Enable Register. (MSI_CTRL_INT_7_EN_OFF)

11.3.5.1.107.1 Offset

Register	Offset
MSI_CTRL_INT_7_EN_OFF	87Ch

11.3.5.1.107.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_7_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_7_EN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.107.3 Fields

Field	Function
31-0 MSI_CTRL_INT_7_EN	MSI Interrupt#i Enable. Specifies which interrupts are enabled. When an MSI is received from a disabled interrupt, no status bit gets set in MSI controller interrupt status register. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.108 Integrated MSI Reception Module Interrupt#i Mask Register. (MSI_CTRL_INT_7_MASK_OFF)

11.3.5.1.108.1 Offset

Register	Offset
MSI_CTRL_INT_7_MASK_OFF	880h

11.3.5.1.108.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_CTRL_INT_7_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_CTRL_INT_7_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.108.3 Fields

Field	Function
31-0 MSI_CTRL_INT_7_MASK	MSI Interrupt#i Mask. Allows enabled interrupts to be masked. When an MSI is received for a masked interrupt, the corresponding status bit gets set in the interrupt status register but the msi_ctrl_int output is not set HIGH. Each bit corresponds to a single MSI Interrupt Vector. Note: This register field is sticky.

11.3.5.1.109 Integrated MSI Reception Module Interrupt#i Status Register. (MSI_CTRL_INT_7_STATUS_OFF)

11.3.5.1.109.1 Offset

Register	Offset
MSI_CTRL_INT_7_STATUS_OFF	884h

11.3.5.1.109.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MSI_CTRL_INT_7_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MSI_CTRL_INT_7_STATUS															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.109.3 Fields

Field	Function
31-0 MSI_CTRL_INT_7_STATUS	MSI Interrupt#i Status. When an MSI is detected for EP#i, one bit in this register is set. The decoding of the data payload of the MSI Memory Write request determines which bit gets set. A status is bit is cleared by writing a 1 to the bit. Each bit corresponds to a single MSI Interrupt Vector.

11.3.5.1.110 Integrated MSI Reception Module General Purpose IO Register. (MSI_GPIO_IO_OFF)

11.3.5.1.110.1 Offset

Register	Offset
MSI_GPIO_IO_OFF	888h

11.3.5.1.110.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MSI_GPIO_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MSI_GPIO_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.110.3 Fields

Field	Function
31-0 MSI_GPIO_REG	MSI GPIO Register. The contents of this register drives the top-level GPIO msi_ctrl_io[31:0] Note: This register field is sticky.

11.3.5.1.111 RADM clock gating enable control register. (CLOCK_GATING_CTRL_OFF)

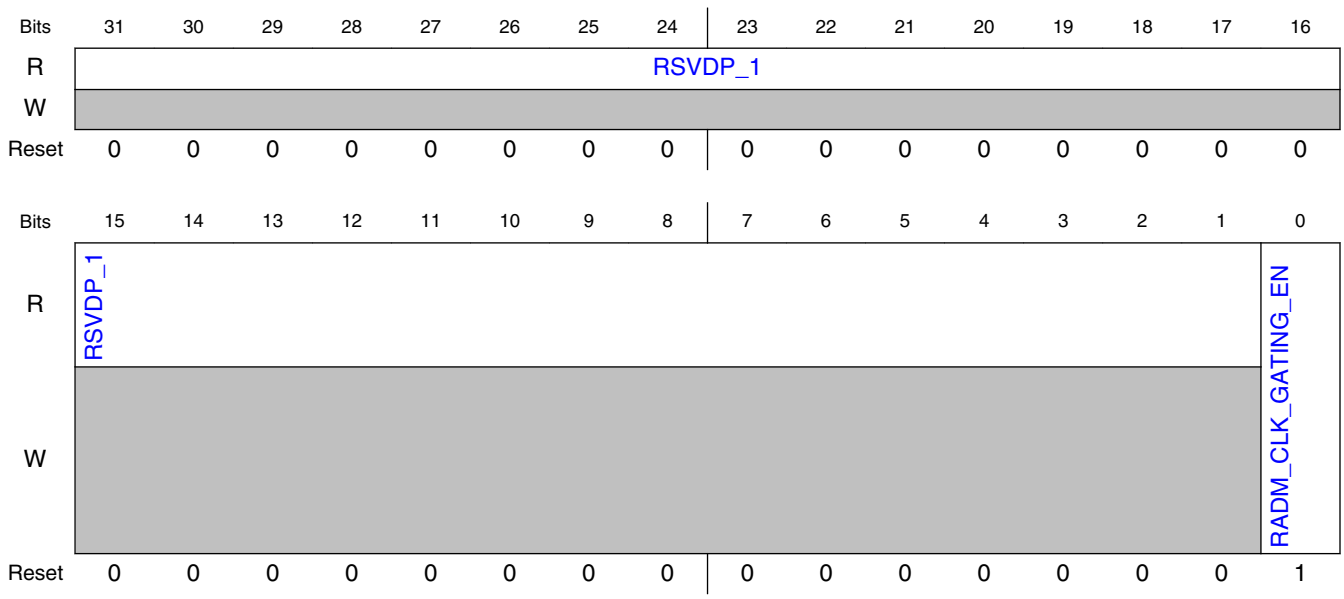
11.3.5.1.111.1 Offset

Register	Offset
CLOCK_GATING_CTRL_OFF	88Ch

11.3.5.1.111.2 Function

RADM clock gating enable control register. Using this register you can disable the RADM clock gating feature. The DWC_pcie_clk_rst.v modules uses the en_radm_clk_g output to gate core_clk and create the radm_clk_g clock input. The controller de-asserts the en_radm_clk_g output when there is no Rx traffic, Rx queues and pre/post-queue pipelines are empty, RADM completion LUT is empty, and there are no FLR actions pending. You must set the RADM_CLK_GATING_EN field to enable this functionality; otherwise the en_radm_clk_g output will always be set to '1'.

11.3.5.1.111.3 Diagram



11.3.5.1.111.4 Fields

Field	Function
31-1 RSVDP_1	Reserved for future use.
0 RADM_CLK_GATING_EN	Enable Radm clock gating feature. - 0: Disable - 1: Enable(default)

11.3.5.1.112 Order Rule Control Register. (ORDER_RULE_CTRL_OFF)

11.3.5.1.112.1 Offset

Register	Offset
ORDER_RULE_CTRL_OFF	8B4h

11.3.5.1.112.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVDP_16															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CPL_PASS_P								NP_PASS_P							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.112.3 Fields

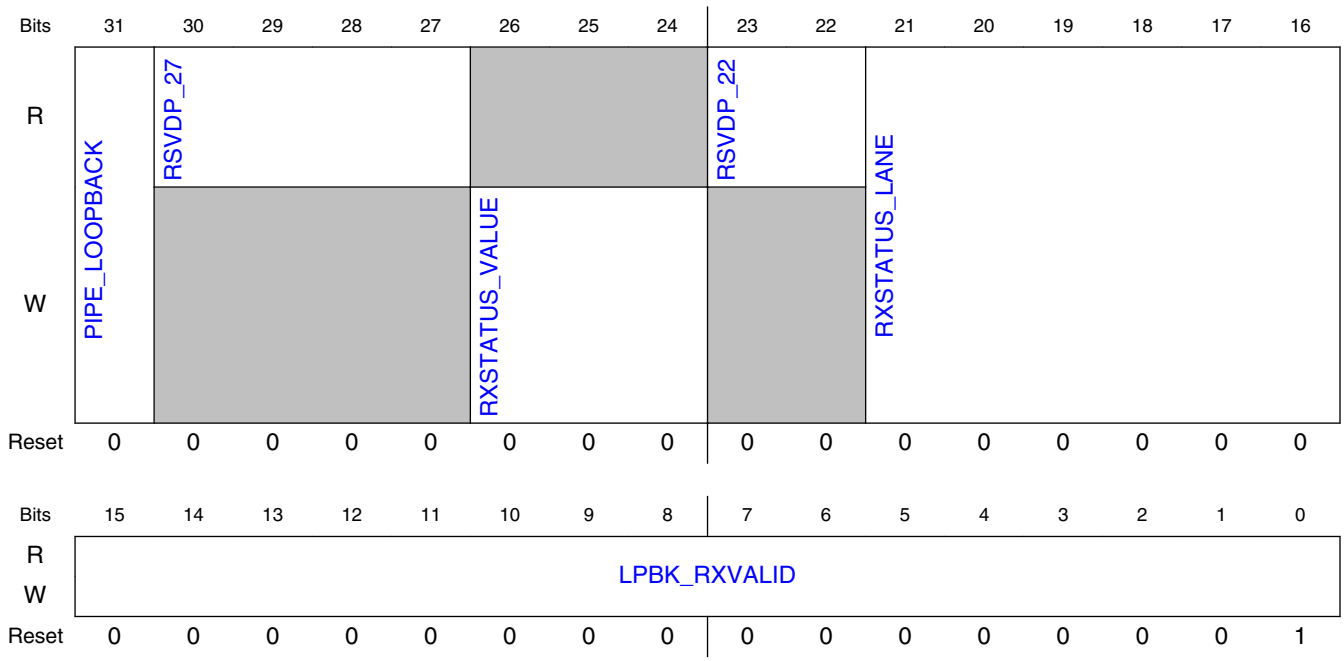
Field	Function
31-16 RSVDP_16	Reserved for future use.
15-8 CPL_PASS_P	Completion Passing Posted Ordering Rule Control. Determines if CPL can pass halted P queue. - 0: CPL can not pass P (recommended) - 1: CPL can pass P
7-0 NP_PASS_P	Non-Posted Passing Posted Ordering Rule Control. Determines if NP can pass halted P queue. - 0 : NP can not pass P (recommended). - 1 : NP can pass P

11.3.5.1.113 PIPE Loopback Control Register. (PIPE_LOOPBACK_CONTROL_OFF)

11.3.5.1.113.1 Offset

Register	Offset
PIPE_LOOPBACK_CONTROL_OFF	8B8h

11.3.5.1.113.2 Diagram



11.3.5.1.113.3 Fields

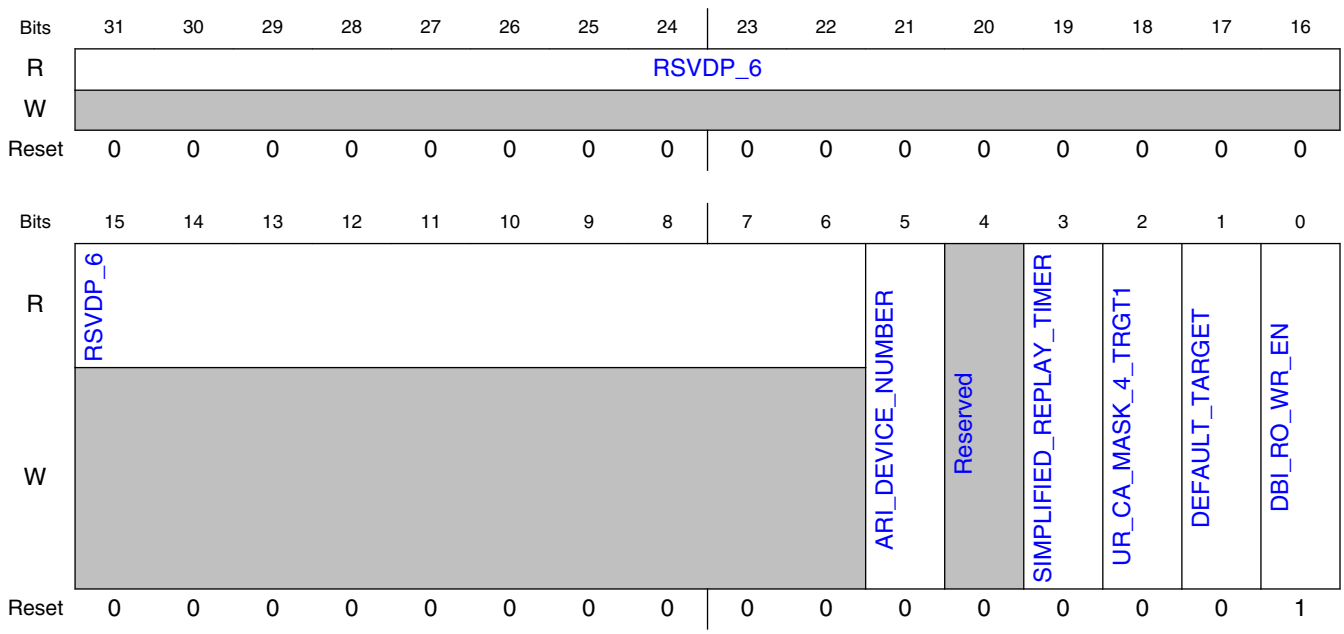
Field	Function
31 PIPE_LOOPBACK	PIPE Loopback Enable. Indicates RMMI Loopback if M-PCIe. Note: This register field is sticky.
30-27 RSVDP_27	Reserved for future use.
26-24 RXSTATUS_VALUE	RXSTATUS_VALUE is an internally reserved field. Do not use.
23-22 RSVDP_22	Reserved for future use.
21-16 RXSTATUS_LANE	RXSTATUS_LANE is an internally reserved field. Do not use. Note: This register field is sticky.
15-0 LPBK_RXVALID	LPBK_RXVALID is an internally reserved field. Do not use. Note: This register field is sticky.

11.3.5.1.114 DBI Read-Only Write Enable Register. (MISC_CONTROL_1_OFF)

11.3.5.1.114.1 Offset

Register	Offset
MISC_CONTROL_1_OFF	8BCh

11.3.5.1.114.2 Diagram



11.3.5.1.114.3 Fields

Field	Function
31-6 RSVDP_6	Reserved for future use.
5 ARI_DEVICE_NUMBER	When ARI is enabled, this field enables use of the device ID. Note: This register field is sticky.
4 —	Reserved.
3	Enables Simplified Replay Timer (Gen4). For more details, see "Transmit Replay" in the Controller Operations chapter of the Databook. Simplified Replay Timer Values are: - A value from 24,000 to 31,000

Table continues on the next page...

Field	Function
SIMPLIFIED_REPLAY_TIMER	Symbol Times when Extended Synch is 0b. - A value from 80,000 to 100,000 Symbol Times when Extended Synch is 1b. Must not be changed while link is in use. Note: This register field is sticky.
2 UR_CA_MASK_4_TRGT1	This field only applies to request TLPs (with UR filtering status) that you have chosen to forward to the application (when you set DEFAULT_TARGET in this register). - When you set this field to '1', the core suppresses error logging, Error Message generation, and CPL generation (for non-posted requests). Note: This register field is sticky.
1 DEFAULT_TARGET	Default target a received IO or MEM request with UR/CA/CRS is sent to by the controller. - 0: The controller drops all incoming I/O or MEM requests (after corresponding error reporting). A completion with UR status will be generated for non-posted requests. - 1: The controller forwards all incoming I/O or MEM requests with UR/CA/CRS status to your application. Note: This register field is sticky.
0 DBI_RO_WR_EN	Write to RO Registers Using DBI. When you set this field to "1", then some RO and HwInit bits are writable from the local application through the DBI. Note: This register field is sticky.

11.3.5.1.115 UpConfigure Multi-lane Control Register. (MULTI_LANE_CONTROL_OFF)

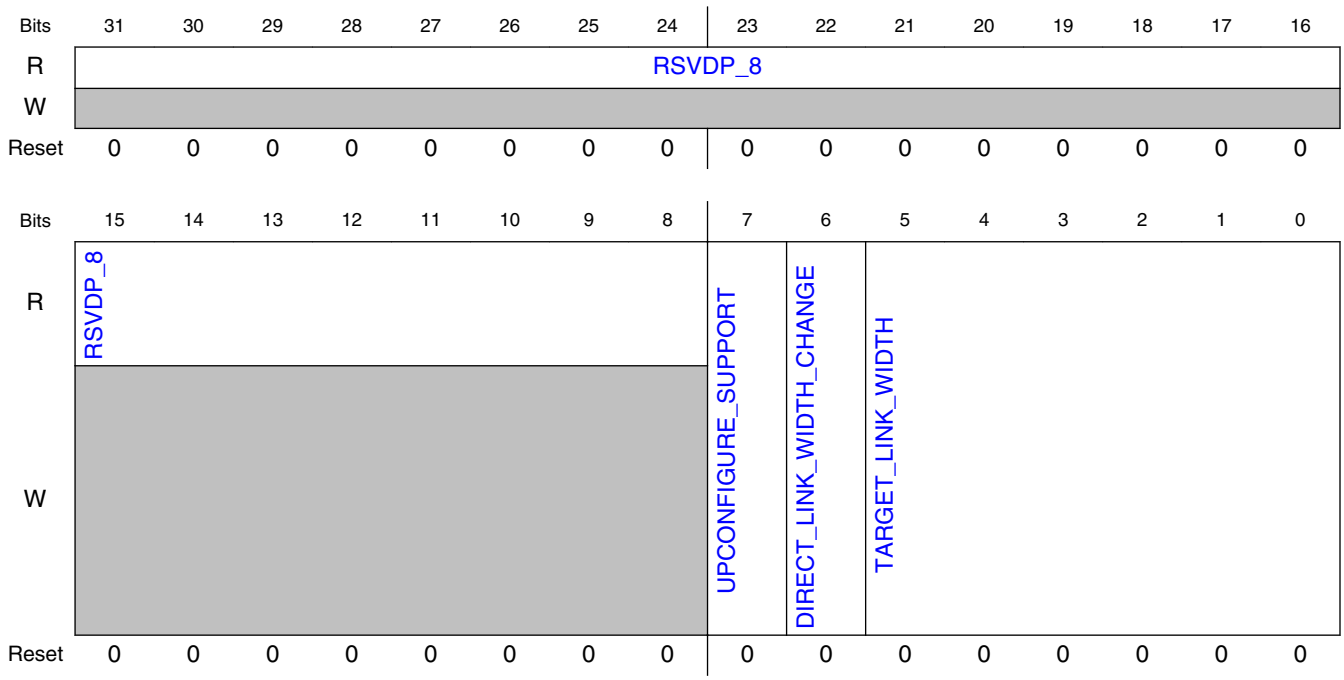
11.3.5.1.115.1 Offset

Register	Offset
MULTI_LANE_CONTROL_OFF	8C0h

11.3.5.1.115.2 Function

UpConfigure Multi-lane Control Register. Used when upsizing or downsizing the link width through Configuration state without bringing the link down. For more details, see the "Link Establishment" section in the "Controller Operations" chapter of the Databook.

11.3.5.1.115.3 Diagram



11.3.5.1.115.4 Fields

Field	Function
31-8 RSVDP_8	Reserved for future use.
7 UPCONFIGURE_SUPPORT	Upconfigure Support. The controller sends this value as the Link Upconfigure Capability in TS2 Ordered Sets in Configuration.Complete state. This field is reserved (fixed to '0') for M-PCIe. Note: This register field is sticky.
6 DIRECT_LINK_WIDTH_CHANGE	Directed Link Width Change. The controller always moves to Configuration state through Recovery state when this bit is set to '1'. - If the upconfigure_capable variable is '1' and the PCIE_CAP_HW_AUTO_WIDTH_DISABLE bit in LINK_CONTROL_LINK_STATUS_REG is '0', the controller starts upconfigure or autonomous width downsizing (to the TARGET_LINK_WIDTH value) in the Configuration state. - If TARGET_LINK_WIDTH value is 0x0, the controller does not start upconfigure or autonomous width downsizing in the Configuration state. The controller self-clears this field when the controller accepts this request. This field is reserved (fixed to '0') for M-PCIe.
5-0 TARGET_LINK_WIDTH	Target Link Width. Values correspond to: - 6'b000000: Core does not start upconfigure or autonomous width downsizing in the Configuration state. - 6'b000001: x1 - 6'b000010: x2 - 6'b000100: x4 - 6'b001000: x8 - 6'b010000: x16 - 6'b100000: x32 This field is reserved (fixed to '0') for M-PCIe.

11.3.5.1.116 PHY Interoperability Control Register. (PHY_INTEROP_CTRL_OFF)

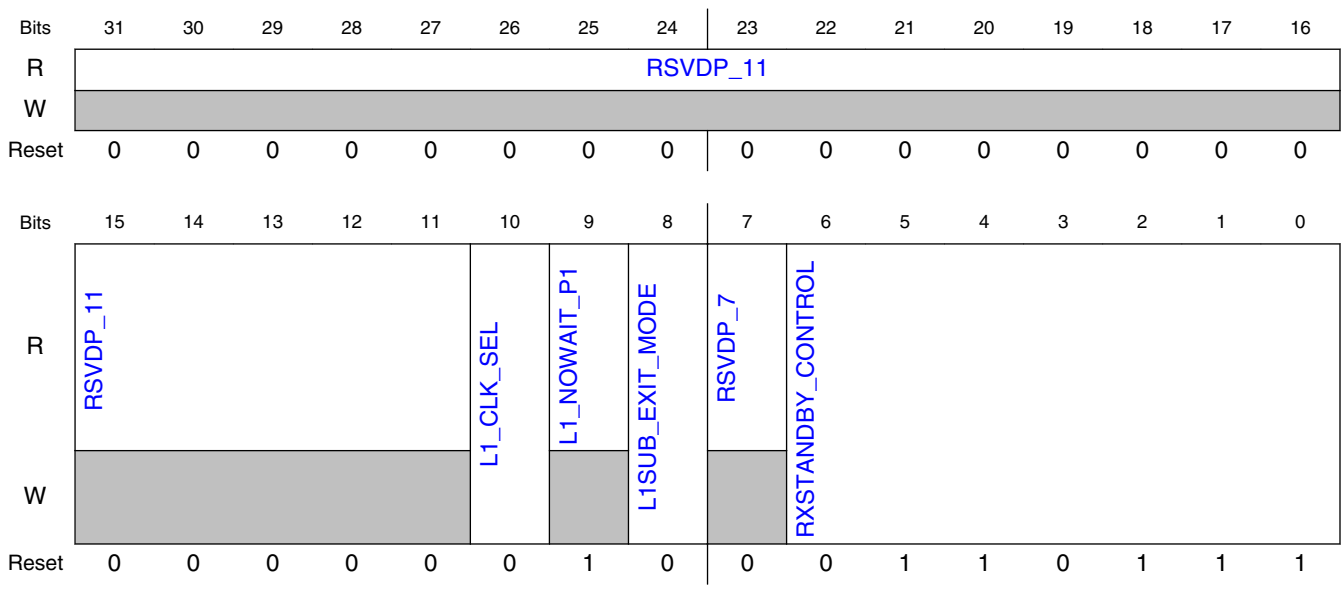
11.3.5.1.116.1 Offset

Register	Offset
PHY_INTEROP_CTRL_OFF	8C4h

11.3.5.1.116.2 Function

PHY Interoperability Control Register. This register is reserved for internal use. You should not write to this register and change the default unless specifically instructed by Synopsys support.

11.3.5.1.116.3 Diagram



11.3.5.1.116.4 Fields

Field	Function
31-11 RSVDP_11	Reserved for future use.
10 L1_CLK_SEL	L1 Clock control bit. - 1: Controller does not request aux_clk switch and core_clk gating in L1. - 0: Controller requests aux_clk switch and core_clk gating in L1. Note: This register field is sticky.
9 L1_NOWAIT_P1	L1 entry control bit. - 1: Core does not wait for PHY to acknowledge transition to P1 before entering L1. - 0: Core waits for the PHY to acknowledge transition to P1 before entering L1. Note: The access attributes of this field are as follows: - Dbi: R/W (sticky) Note: This register field is sticky.

Table continues on the next page...

Field	Function
8 L1SUB_EXIT_MODE	L1 Exit Control Using phy_mac_pclkack_n. - 1: Core exits L1 without waiting for the PHY to assert phy_mac_pclkack_n. - 0: Core waits for the PHY to assert phy_mac_pclkack_n before exiting L1. Note: This register field is sticky.
7 RSVDP_7	Reserved for future use.
6-0 RXSTANDBY_CONTROL	Rxstandby Control. Bits 0..5 determine if the controller asserts the RxStandby signal (mac_phy_rxstandby) in the indicated condition. Bit 6 enables the controller to perform the RxStandby/ RxStandbyStatus handshake. - [0]: Rx EIOS and subsequent T TX-IDLE-MIN - [1]: Rate Change - [2]: Inactive lane for upconfigure/downconfigure - [3]: PowerDown=P1orP2 - [4]: RxL0s.Idle - [5]: EI Infer in L0 - [6]: Execute RxStandby/RxStandbyStatus Handshake This field is reserved (fixed to '0') for M-PCIe. Note: This register field is sticky.

11.3.5.1.117 TRGT_CPL_LUT Delete Entry Control register. (TRGT_CPL_LUT_DELETE_ENTRY_OFF)

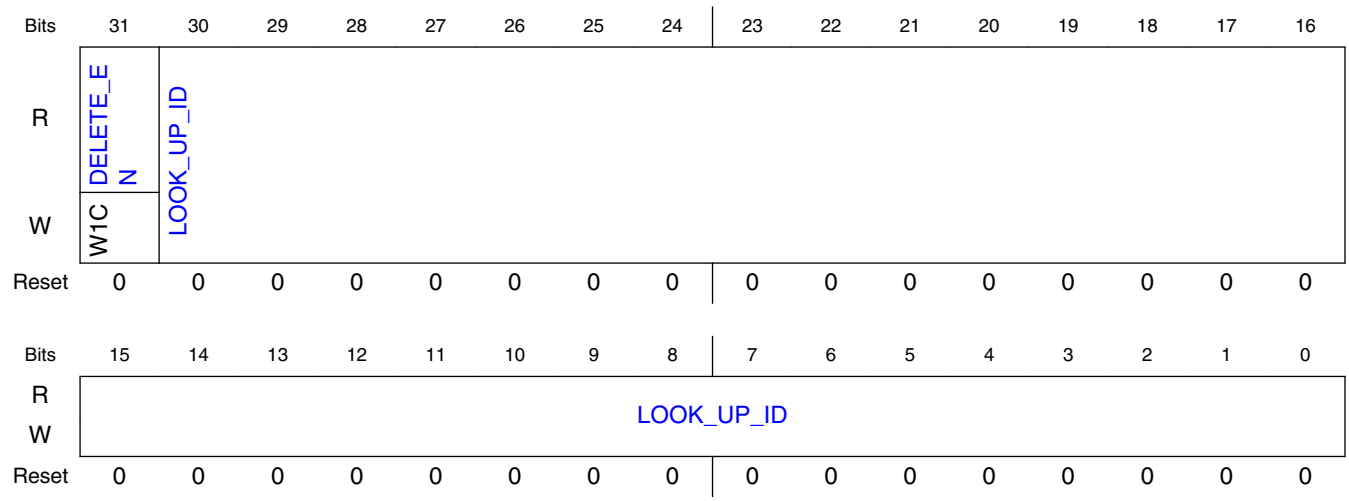
11.3.5.1.117.1 Offset

Register	Offset
TRGT_CPL_LUT_DELETE_ENTRY_OFF	8C8h

11.3.5.1.117.2 Function

TRGT_CPL_LUT Delete Entry Control register. Using this register you can delete one entry in the target completion LUT. You should only use this register when you know that your application will never send the completion because of an FLR or any other reason. Note:: The target completion LUT (and associated target completion timeout event) is watching for completions (from your application on XALI0/1/2 or AXI master read channel) corresponding to previously received non-posted requests from the PCIe wire.

11.3.5.1.117.3 Diagram



11.3.5.1.117.4 Fields

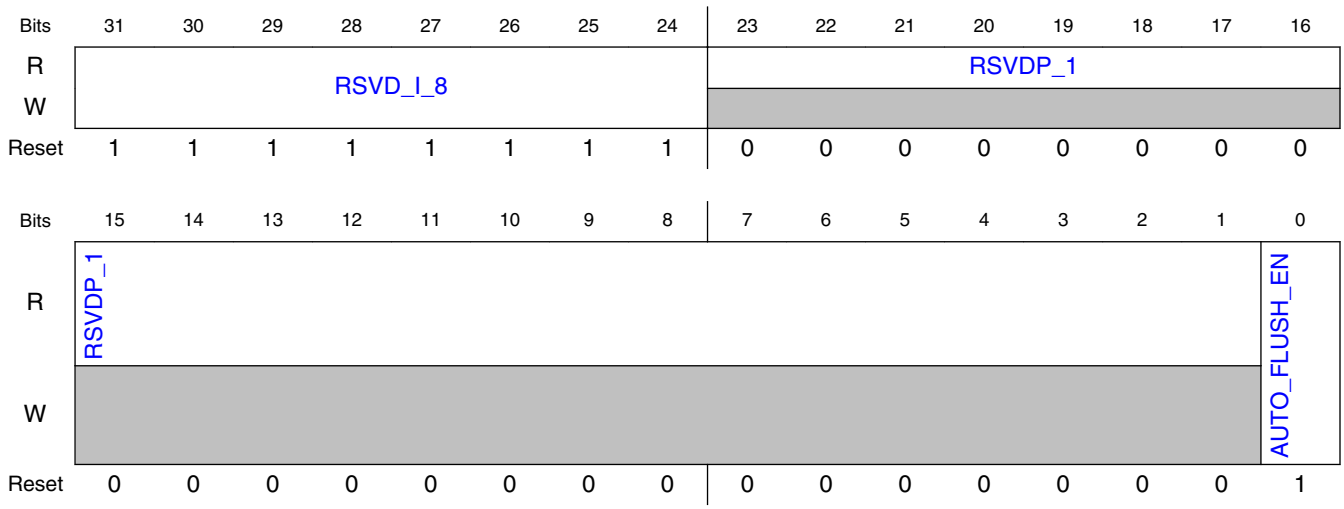
Field	Function
31 DELETE_EN	This is a one-shot bit. A '1' write to this bit triggers the deletion of the target completion LUT entry that is specified in the LOOK_UP_ID field. This is a self-clearing register field. Reading from this register field always returns a '0'.
30-0 LOOK_UP_ID	This number selects one entry to delete of the TRGT_CPL_LUT.

11.3.5.1.118 Link Reset Request Flush Control Register. (LINK_FLUSH_CONTROL_OFF)

11.3.5.1.118.1 Offset

Register	Offset
LINK_FLUSH_CONTROL_OFF	8CCh

11.3.5.1.118.2 Diagram



11.3.5.1.118.3 Fields

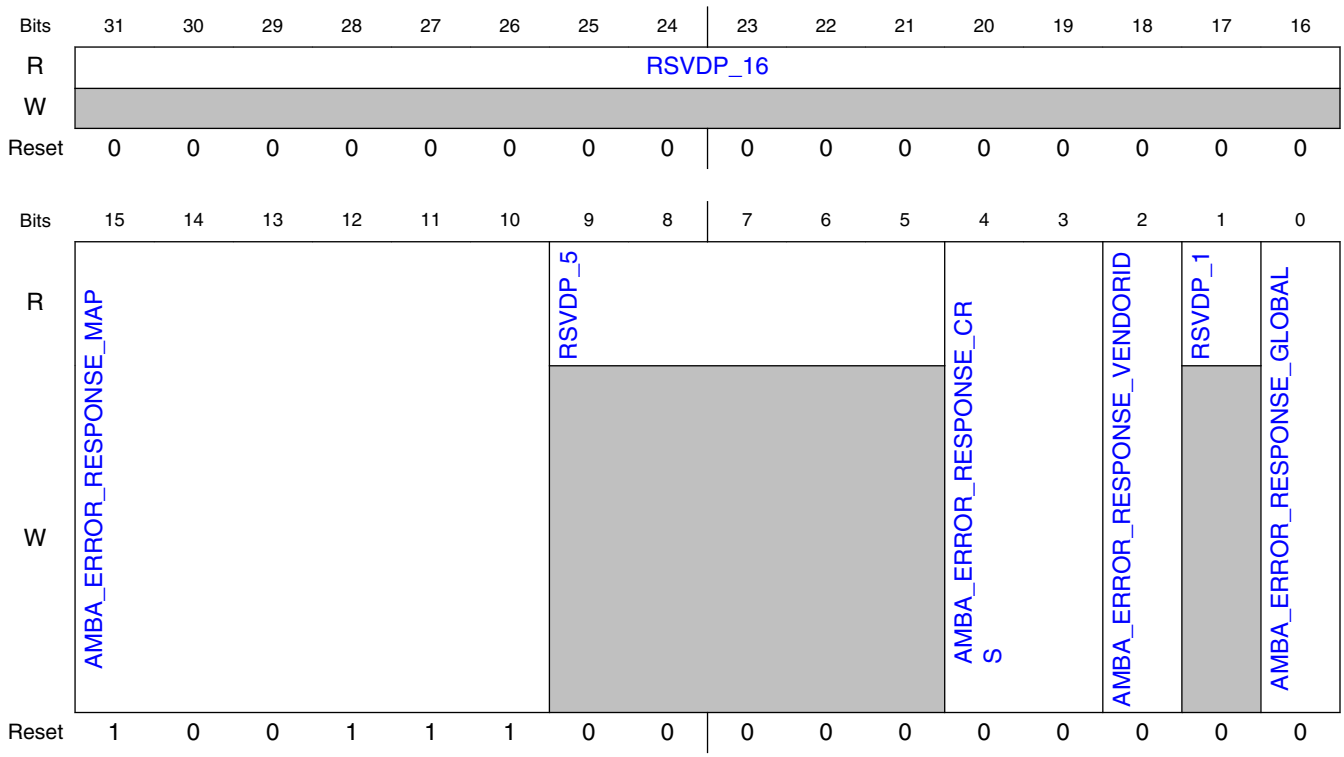
Field	Function
31-24 RSVD_I_8	This is an internally reserved field. Do not use. Note: This register field is sticky.
23-1 RSVDP_1	Reserved for future use.
0 AUTO_FLUSH_EN	Enables automatic flushing of pending requests before sending the reset request to the application logic to reset the PCIe controller and the AXI Bridge. The flushing process is initiated if any of the following events occur: - Hot reset request. A downstream port (DSP) can "hot reset" an upstream port (USP) by sending two consecutive TS1 ordered sets with the hot reset bit asserted. - Warm (Soft) reset request. Generated when exiting from D3 to D0 and cfg_pm_no_soft_rst=0. - Link down reset request. A high to low transition on smlh_req_rst_not indicates the link has gone down and the controller is requesting a reset. If you disable automatic flushing, your application is responsible for resetting the PCIe controller and the AXI Bridge. For more details see "Warm and Hot Resets" section in the Architecture chapter of the Databook. Note: This register field is sticky.

11.3.5.1.119 AXI Bridge Slave Error Response Register. (AMBA_ERROR_RESPONSE_DEFAULT_OFF)

11.3.5.1.119.1 Offset

Register	Offset
AMBA_ERROR_RESPONSE_DEFAULT_OFF	8D0h

11.3.5.1.119.2 Diagram



11.3.5.1.119.3 Fields

Field	Function
31-16 RSVDP_16	Reserved for future use.
15-10 AMBA_ERROR_RESPONSE_MAP	AXI Slave Response Error Map. Allows you to selectively map the errors received from the PCIe completion (for non-posted requests) to the AXI slave responses, slv_rresp or slv_bresp. The recommended setting is SLVERR. CRS is always mapped to OKAY. - [0] -- 0: UR (unsupported request) -> DECERR -- 1: UR (unsupported request) -> SLVERR - [1] -- 0: CRS (configuration retry status) -> DECERR -- 1: CRS (configuration retry status) -> SLVERR - [2] -- 0: CA (completer abort) -> DECERR -- 1: CA (completer abort) -> SLVERR - [3]: Reserved - [4]: Reserved - [5]: -- 0: Completion Timeout -> DECERR -- 1: Completion Timeout -> SLVERR The AXI bridge internally drops (processes internally but not passed to your application) a completion that has been marked by the Rx filter as UC or MLF, and does not pass its status directly down to the slave interface. It waits for a timeout and then signals "Completion Timeout" to the slave interface. The controller sets the AXI slave read databus to 0xFFFF for all error responses. Note: This register field is sticky.
9-5 RSVDP_5	Reserved for future use.
4-3 AMBA_ERROR_RESPONSE_CRS	CRS Slave Error Response Mapping. Determines the AXI slave response for CRS completions. For more details see "Error Handling" in the AXI chapter of the Databook. AHB: - always returns OKAY AXI: - 00: OKAY - 01: OKAY with all FFFF_FFFF data for all CRS completions - 10: OKAY with FFFF_0001 data for CRS completions to vendor ID read requests, OKAY with FFFF_FFFF data for all other CRS completions - 11: SLVERR/DECERR (the AXI_ERROR_RESPONSE_MAP field determines the PCIe-to-AXI Slave error response mapping) Note: This register field is sticky.

Table continues on the next page...

Field	Function
2 AMBA_ERROR_RESPONSE_VENDORID	Vendor ID Non-existent Slave Error Response Mapping. Determines the AXI slave response for errors on reads to non-existent Vendor ID register. For more details see "Error Handling" in the AXI chapter of the Databook. AHB: - 0: OKAY (with FFFF data). The controller ignores the setting in the bit when bit 0 of this register is '0'. - 1: ERROR AXI: - 0: OKAY (with FFFF data). - 1: SLVERR/DECERR (the AXI_ERROR_RESPONSE_MAP field determines the PCIe-to-AXI Slave error response mapping) Note: This register field is sticky.
1 RSVDP_1	Reserved for future use.
0 AMBA_ERROR_RESPONSE_GLOBAL	Global Slave Error Response Mapping. Determines the AXI slave response for all error scenarios on non-posted requests. For more details see "Error Handling" in the AXI chapter of the Databook. AHB: - 0: OKAY (with FFFF data for non-posted requests) and ignore the setting in bit [2] of this register. - 1: ERROR for normal link (data) accesses and look at bit [2] for other scenarios. AXI: - 0: OKAY (with FFFF data for non-posted requests) - 1: SLVERR/DECERR (the AXI_ERROR_RESPONSE_MAP field determines the PCIe-to-AXI Slave error response mapping) The error response mapping is not applicable to Non-existent Vendor ID register reads. Note: This register field is sticky.

11.3.5.1.120 Link Down AXI Bridge Slave Timeout Register. (AMBA_LINK_TIMEOUT_OFF)

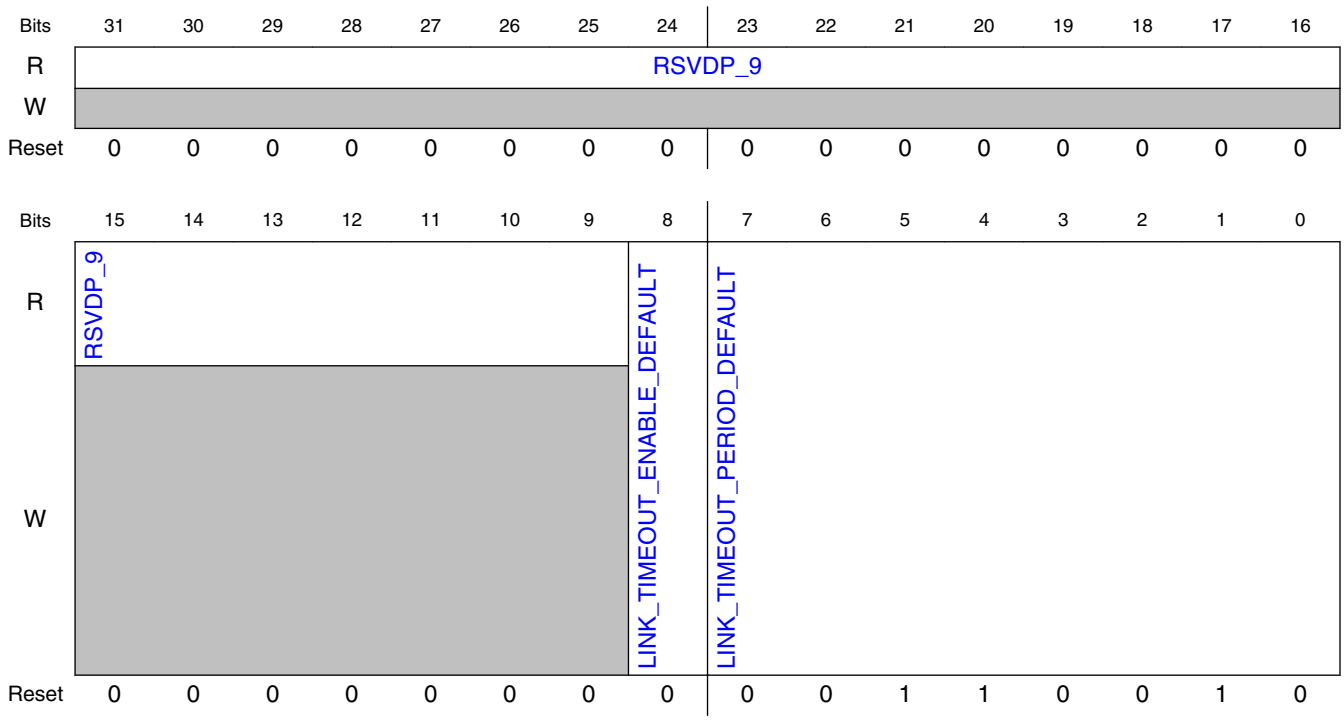
11.3.5.1.120.1 Offset

Register	Offset
AMBA_LINK_TIMEOUT_OFF	8D4h

11.3.5.1.120.2 Function

Link Down AXI Bridge Slave Timeout Register. If your application AXI master issues outbound requests to the AXI bridge slave interface before the PCIe link is operational, the controller starts a "flush" timer. The timeout value of the timer is set by this register. The timer will timeout and then flush the bridge TX request queues after this amount of time. The timer counts when there are pending outbound AXI slave interface (or DMA) requests and the PCIe TX link is not transmitting any of these requests. For more details, see the "AXI Bridge Initialization, Clocking and Reset" section in the AXI chapter of the Databook.

11.3.5.1.120.3 Diagram



11.3.5.1.120.4 Fields

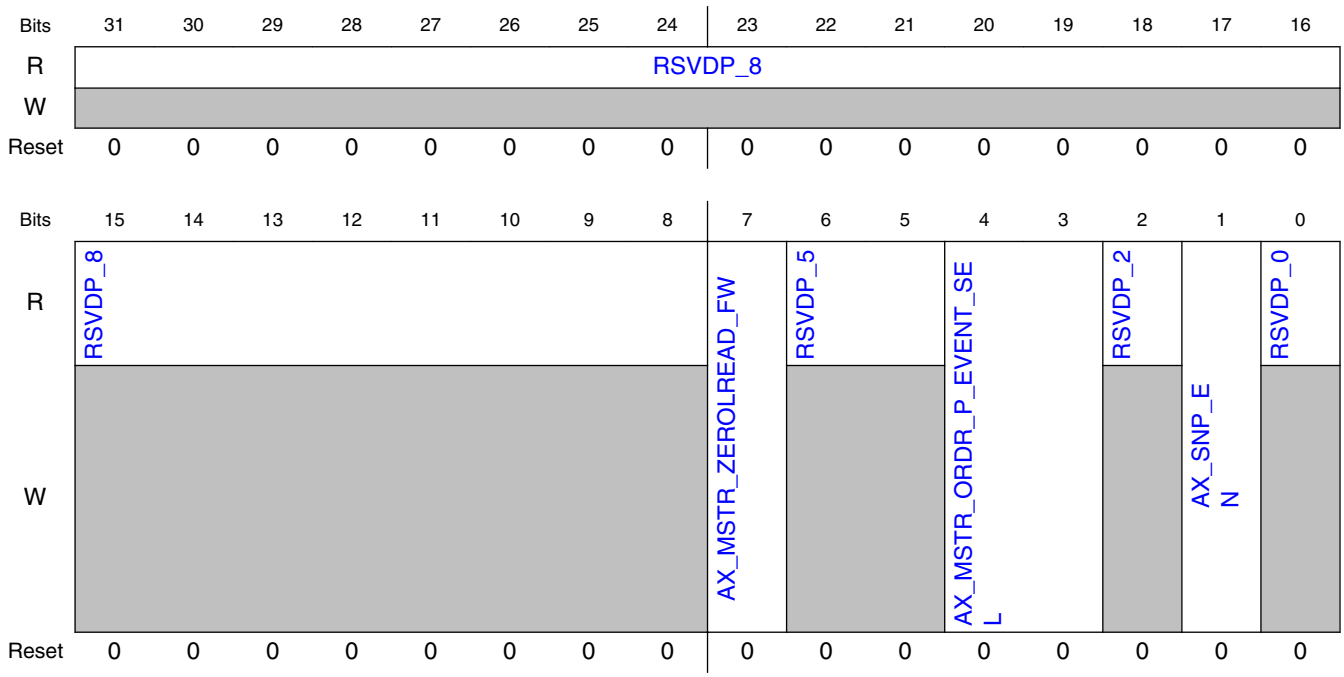
Field	Function
31-9 RSVDP_9	Reserved for future use.
8 LINK_TIMEOUT_ENABLE_DEFAULT	Disable Flush. You can disable the flush feature by setting this field to "1". Note: This register field is sticky.
7-0 LINK_TIMEOUT_PERIOD_DEFAULT	Timeout Value (ms). The timer will timeout and then flush the bridge TX request queues after this amount of time. The timer counts when there are pending outbound AXI slave interface requests and the PCIe TX link is not transmitting any of these requests. The timer is clocked by core_clk. For an M-PCIe configuration: - Time unit of this field is 4 ms. - Margin of error for RateA clock is < 1%. - Margin of error for RateB clock is between 16% and 17%. Note: This register field is sticky.

11.3.5.1.121 AMBA Ordering Control. (AMBA_ORDERING_CTRL_OFF)

11.3.5.1.121.1 Offset

Register	Offset
AMBA_ORDERING_CTRL_OFF	8D8h

11.3.5.1.121.2 Diagram



11.3.5.1.121.3 Fields

Field	Function
31-8 RSVDP_8	Reserved for future use.
7 AX_MSTR_ZEROLREAD_FW	AXI Master Zero Length Read Forward to the application. The DW PCIe controller AXI bridge is able to terminate in order with the Posted transactions the zero length read, implementing the PCIe express flush semantics of the Posted transactions. - 0x0: The zero length Read is terminated at the DW PCIe AXI bridge master - 0x1: The zero length Read is forward to the application.
6-5 RSVDP_5	Reserved for future use.
4-3 AX_MSTR_ORDR_P_EVENT_SEL	AXI Master Posted Ordering Event Selector. This field selects how the master interface determines when a P write is completed when enforcing the PCIe ordering rule, "NP must not pass P" at the AXI Master Interface. The AXI protocol does not support ordering between channels. Therefore, NP reads can pass P on your AXI bus fabric. This can result in an ordering violation when the read overtakes a P that is going to the same address. Therefore, the bridge master does not issue any NP requests until all outstanding P writes reach their destination. It does this by waiting for the all of the write responses on

Table continues on the next page...

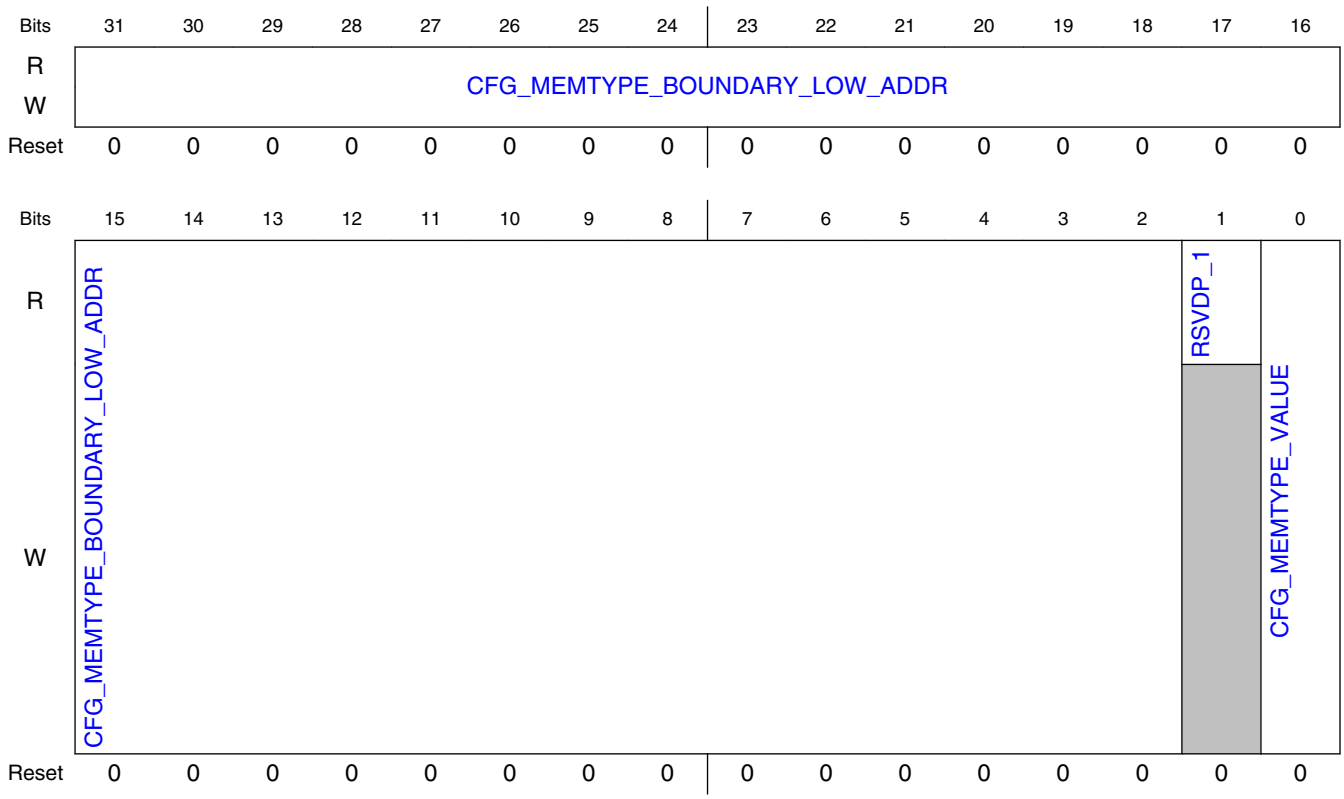
Field	Function
	the B channel. This can affect the performance of the master read channel. For scenarios where the interconnect serializes the AXI master "AW", "W" and "AR" channels,you can increase the performance by reducing the need to wait until the complete Posted transaction has effectively reached the application slave. - 00: B'last event: wait for the all of the write responses on the B channel thereby ensuring that the complete Posted transaction has effectively reached the application slave (default). - 01: AW'last event: wait until the complete Posted transaction has left the AXI address channel at the bridge master. - 10: W'last event: wait until the complete Posted transaction has left the AXI data channel at the bridge master. - 11: Reserved Note 2: This setting will not affect: - MSI interrupt catcher and P data ordering. This is always driven by the B'last event. - DMA read engine TLP ordering. This is always driven by the B'last event. - NP write transactions which are always serialized with P write transactions.
2 RSVDP_2	Reserved for future use.
1 AX_SNP_EN	AXI Serialize Non-Posted Requests Enable. This field enables the AXI Bridge to serialize same ID Non-Posted Read/Write Requests on the wire. Serialization implies one outstanding same ID NP Read or Write on the wire and used to avoid AXI RAR and WAW hazards at the remote link partner. For more details, see the "Optional Serialization of AXI Slave Non-posted Requests" section in the AXI chapter of the Databook.
0 RSVDP_0	Reserved for future use.

11.3.5.1.122 ACE Cache Coherency Control Register 1 (COHERENCY_CONTROL_1_OFF)

11.3.5.1.122.1 Offset

Register	Offset
COHERENCY_CONTROL_1_OFF	8E0h

11.3.5.1.122.2 Diagram



11.3.5.1.122.3 Fields

Field	Function
31-2 CFG_MEMTYPE_BOUNDARY_LOW_ADDR	Boundary Lower Address For Memory Type. Bits [31:0] of dword-aligned address of the boundary for Memory type. The two lower address LSBs are "00". Addresses up to but not including this value are in the lower address space region; addresses equal or greater than this value are in the upper address space region. Note: This register field is sticky.
1 RSVDP_1	Reserved for future use.
0 CFG_MEMTYPE_VALUE	Sets the memory type for the lower and upper parts of the address space: - 0: lower = Peripheral; upper = Memory - 1: lower = Memory type; upper = Peripheral Note: This register field is sticky.

11.3.5.1.123 ACE Cache Coherency Control Register 2 (COHERENCY_CONTROL_2_OFF)

11.3.5.1.123.1 Offset

Register	Offset
COHERENCY_CONTROL_2_OFF	8E4h

11.3.5.1.123.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CFG_MEMTYPE_BOUNDARY_HIGH_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CFG_MEMTYPE_BOUNDARY_HIGH_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.123.3 Fields

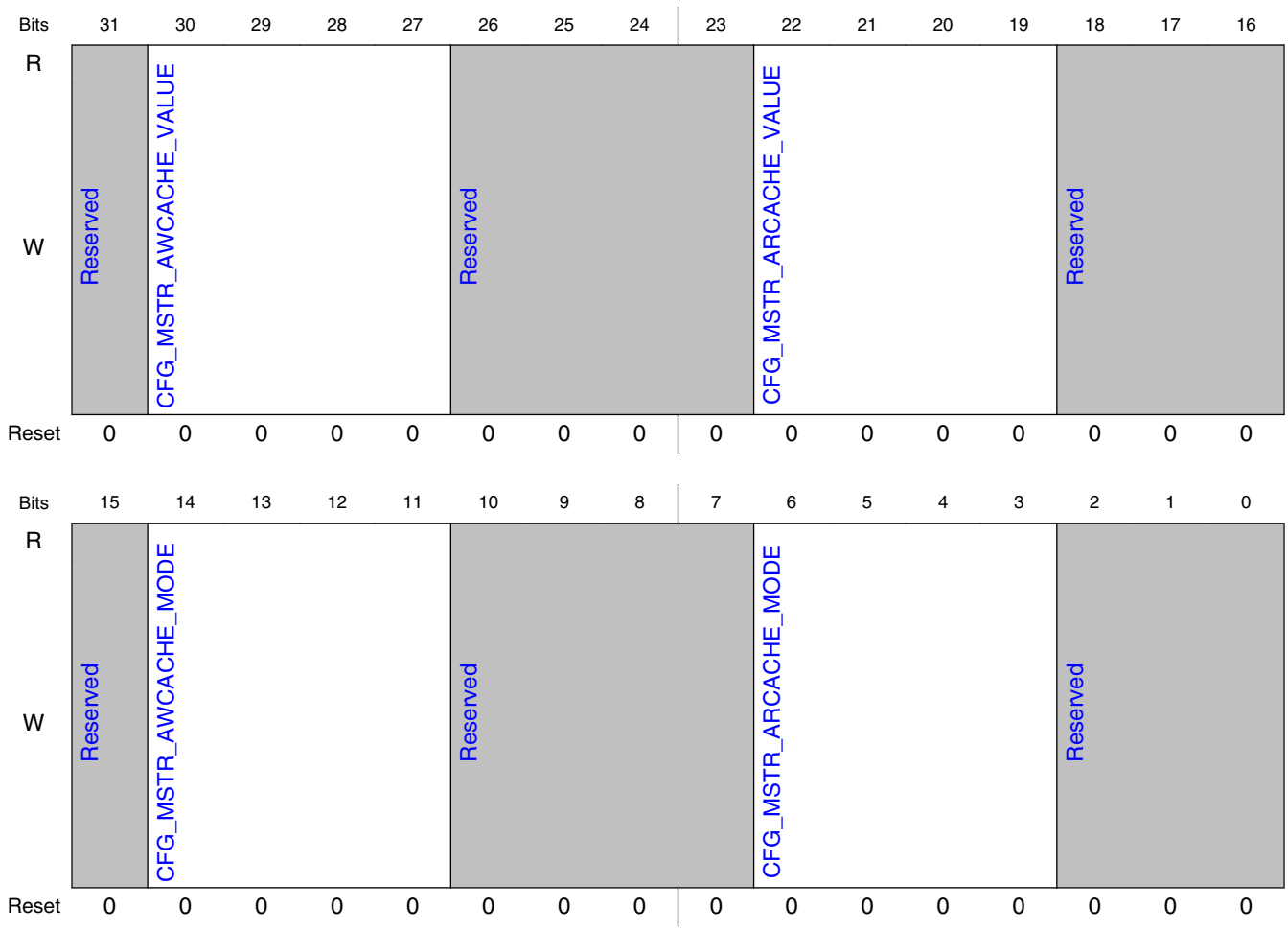
Field	Function
31-0 CFG_MEMTYPE_BOUNDARY_HIGH_ADDR	Boundary Upper Address For Memory Type. Bits [63:32] of the 64-bit dword-aligned address of the boundary for Memory type. Note: This register field is sticky.

11.3.5.1.124 ACE Cache Coherency Control Register 3 (COHERENCY_CONTROL_3_OFF)

11.3.5.1.124.1 Offset

Register	Offset
COHERENCY_CONTROL_3_OFF	8E8h

11.3.5.1.124.2 Diagram



11.3.5.1.124.3 Fields

Field	Function
31 —	Reserved.
30-27 CFG_MSTR_AWCACHE_VALUE	Master Write CACHE Signal Value. Value of the individual bits in mstr_awcache when CFG_MSTR_AWCACHE_MODE is '1'. Note: not applicable to message requests; for message requests the value of mstr_awcache is always "0000" Note: This register field is sticky.
26-23 —	Reserved.
22-19 CFG_MSTR_ARCACHE_VALUE	Master Read CACHE Signal Value. Value of the individual bits in mstr_arcache when CFG_MSTR_ARCACHE_MODE is '1'. Note: This register field is sticky.
18-15	Reserved.

Table continues on the next page...

Field	Function
—	
14-11 CFG_MSTR_AWCACHE_MODE	Master Write CACHE Signal Behavior. Defines how the individual bits in mstr_awcache are controlled: - 0: set automatically by the AXI master - 1: set by the value of the corresponding bit of the CFG_MSTR_AWCACHE_VALUE field Note: for message requests the value of mstr_awcache is always "0000" regardless of the value of this bit Note: This register field is sticky.
10-7 —	Reserved.
6-3 CFG_MSTR_ARCACHE_MODE	Master Read CACHE Signal Behavior. Defines how the individual bits in mstr_arcache are controlled: - 0: set automatically by the AXI master - 1: set by the value of the corresponding bit of the CFG_MSTR_ARCACHE_VALUE field Note: This register field is sticky.
2-0 —	Reserved.

11.3.5.1.125 Lower 20 bits of the programmable AXI address where Messages coming from wire are mapped to. (AXI_MSTR_MSG_ADDR_LOW_OFF)

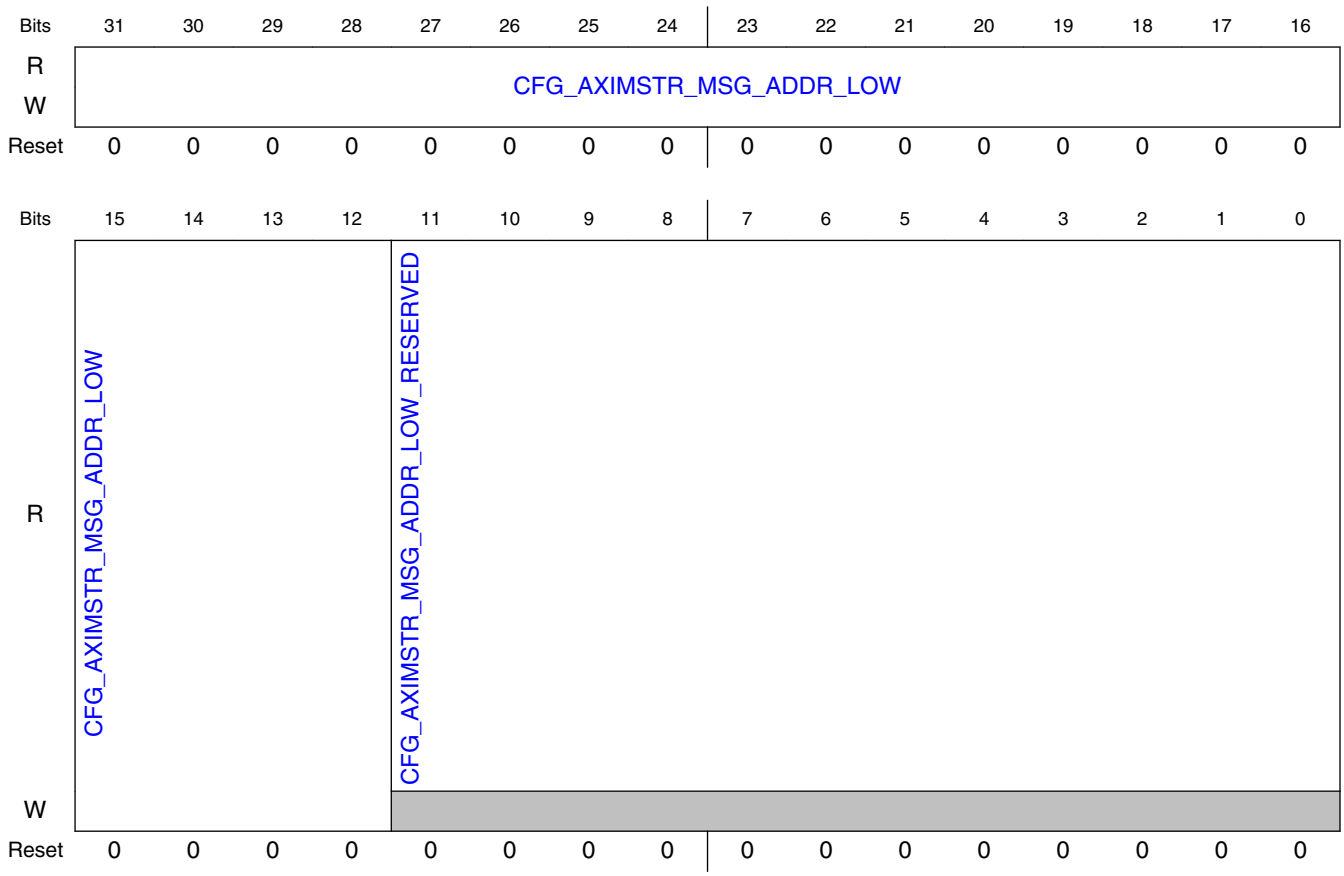
11.3.5.1.125.1 Offset

Register	Offset
AXI_MSTR_MSG_ADDR_LOW_OFF	8F0h

11.3.5.1.125.2 Function

Lower 20 bits of the programmable AXI address where Messages coming from wire are mapped to. Bits [11:0] of the register are tied to zero for the address to be 4k-aligned. In previous releases, the third and fourth DWORDs of a message (Msg/MsgD) TLP header were delivered though the AXI master address bus (mstr_awaddr). These DWORDS are now supplied through the mstr_awmisc_info_hdr_34dw[63:0] output; and the value on mstr_awaddr is driven to the value you have programmed into the AXI_MSTR_MSG_ADDR_LOW_OFF and AXI_MSTR_MSG_ADDR_HIGH_OFF registers.

11.3.5.1.125.3 Diagram



11.3.5.1.125.4 Fields

Field	Function
31-12 CFG_AXIMSTR_MSG_ADDR_LOW	Lower 20 bits of the programmable AXI address for Messages. Note: This register field is sticky.
11-0 CFG_AXIMSTR_MSG_ADDR_LOW_RESERVED	Reserved for future use. Note: This register field is sticky.

11.3.5.1.126 Upper 32 bits of the programmable AXI address where Messages coming from wire are mapped to. (AXI_MSTR_MSG_ADDR_HIGH_OFF)

11.3.5.1.126.1 Offset

Register	Offset
AXI_MSTR_MSG_ADDR_HIGH_OFF	8F4h

11.3.5.1.126.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CFG_AXIMSTR_MSG_ADDR_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CFG_AXIMSTR_MSG_ADDR_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.126.3 Fields

Field	Function
31-0 CFG_AXIMSTR_MSG_ADDR_HIGH	Upper 32 bits of the programmable AXI address for Messages. Note: This register field is sticky.

11.3.5.1.127 PCIe Controller IIP Release Version Number. (PCIE_VERSION_NUMBER_OFF)

11.3.5.1.127.1 Offset

Register	Offset
PCIE_VERSION_NUMBER_OFF	8F8h

11.3.5.1.127.2 Function

PCIe Controller IIP Release Version Number. The version number is given in hex format. You should convert each pair of hex characters to ASCII to interpret. Using 4.70a (GA)

as an example: - VERSION_NUMBER = 0x3437302a which translates to 470* - VERSION_TYPE = 0x67612a2a which translates to ga** Using 4.70a-ea01 as an example: - VERSION_NUMBER = 0x3437302a which translates to 470* - VERSION_TYPE = 0x65613031 which translates to ea01 GA is a general release

11.3.5.1.127.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VERSION_NUMBER															
W																
Reset	0	0	1	1	0	1	0	1	0	0	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VERSION_NUMBER															
W																
Reset	0	0	1	1	0	0	0	0	0	0	1	0	1	0	1	0

11.3.5.1.127.4 Fields

Field	Function
31-0 VERSION_NUMBER	Version Number.

11.3.5.1.128 PCIe Controller IIP Release Version Type. (PCIE_VERSION_TYPE_OFF)

11.3.5.1.128.1 Offset

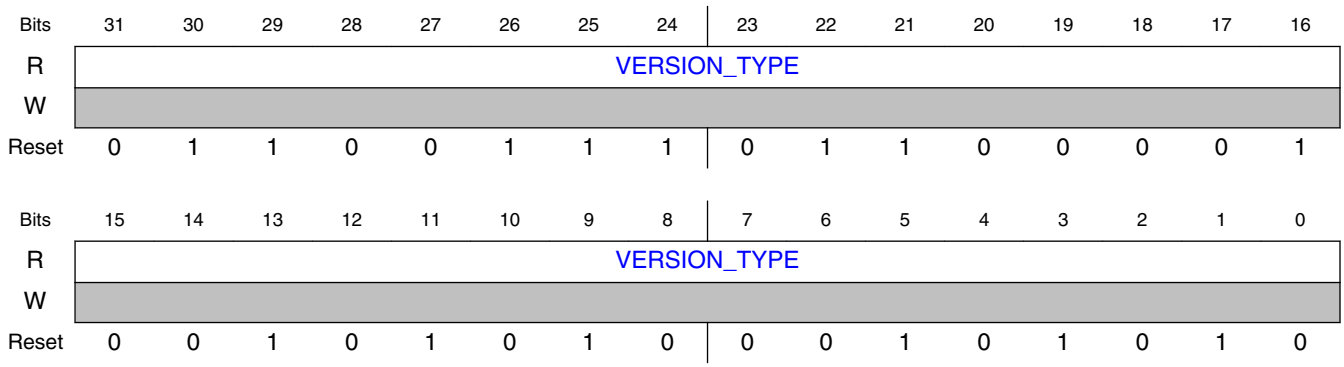
Register	Offset
PCIE_VERSION_TYPE_OFF	8FCh

11.3.5.1.128.2 Function

PCIe Controller IIP Release Version Type. The type is given in hex format. You should convert each pair of hex characters to ASCII to interpret. Using 4.70a (GA) as an example: - VERSION_NUMBER = 0x3437302a which translates to 470* - VERSION_TYPE = 0x67612a2a which translates to ga** Using 4.70a-ea01 as an

example: - VERSION_NUMBER = 0x3437302a which translates to 470* - VERSION_TYPE = 0x65613031 which translates to ea01 GA is a general release.

11.3.5.1.128.3 Diagram



11.3.5.1.128.4 Fields

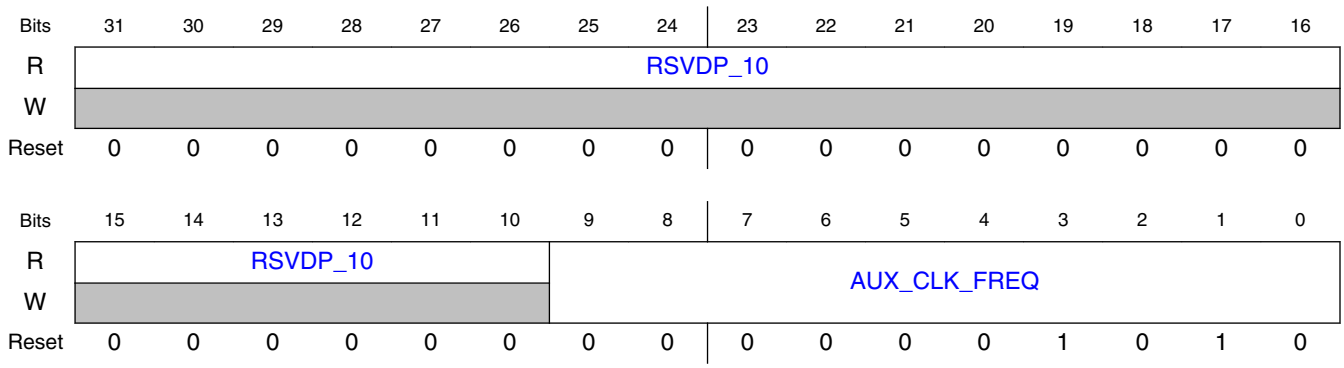
Field	Function
31-0 VERSION_TYPE	Version Type.

11.3.5.1.129 Auxiliary Clock Frequency Control Register. (AUX_CLK_FREQ_OFF)

11.3.5.1.129.1 Offset

Register	Offset
AUX_CLK_FREQ_OFF	B40h

11.3.5.1.129.2 Diagram



11.3.5.1.129.3 Fields

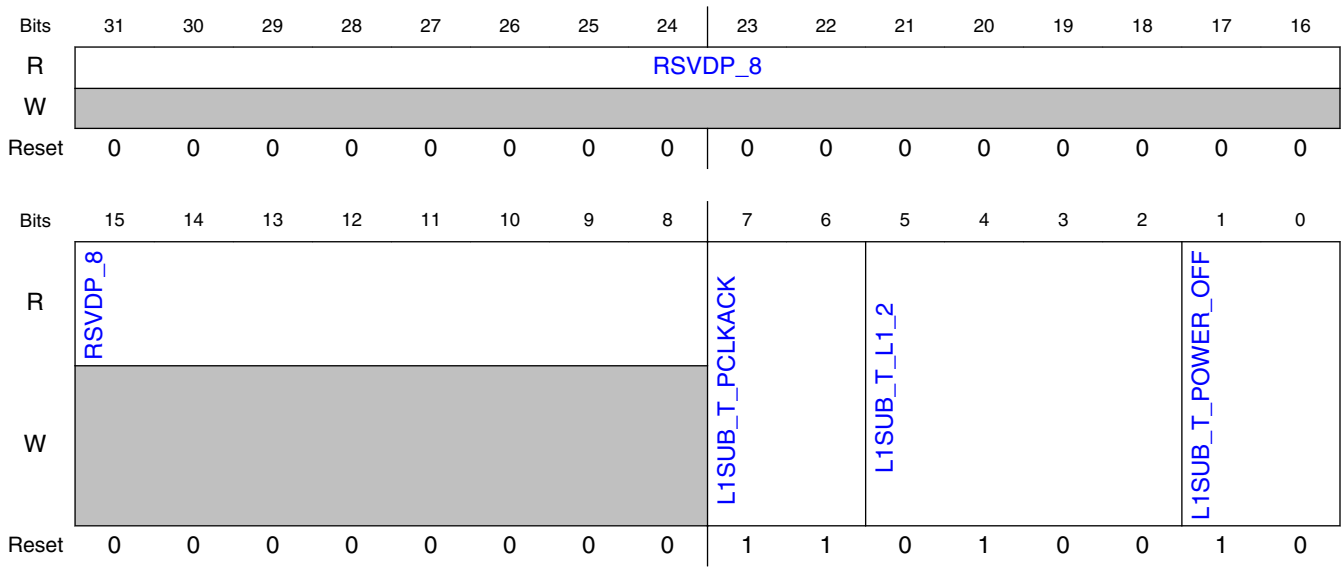
Field	Function
31-10 RSVDP_10	Reserved for future use.
9-0 AUX_CLK_FREQ	The aux_clk frequency in MHz. This value is used to provide a 1 us reference for counting time during low-power states with aux_clk when the PHY has removed the pipe_clk. Frequencies lower than 1 MHz are possible but with a loss of accuracy in the time counted. If the actual frequency (f) of aux_clk does not exactly match the programmed frequency (f_prog), then there is an error in the time counted by the controller that can be expressed in percentage as: err% = (f_prog/f-1)*100. For example if f=2.5 MHz and f_prog=3 MHz, then err% =(3/2.5-1)*100 =20%, meaning that the time counted by the controller on aux_clk will be 20% greater than the time in us programmed in the corresponding time register (for example T_POWER_ON). Note: This register field is sticky.

11.3.5.1.130 L1 Substates Timing Register. (L1_SUBSTATES_OFF)

11.3.5.1.130.1 Offset

Register	Offset
L1_SUBSTATES_OFF	B44h

11.3.5.1.130.2 Diagram



11.3.5.1.130.3 Fields

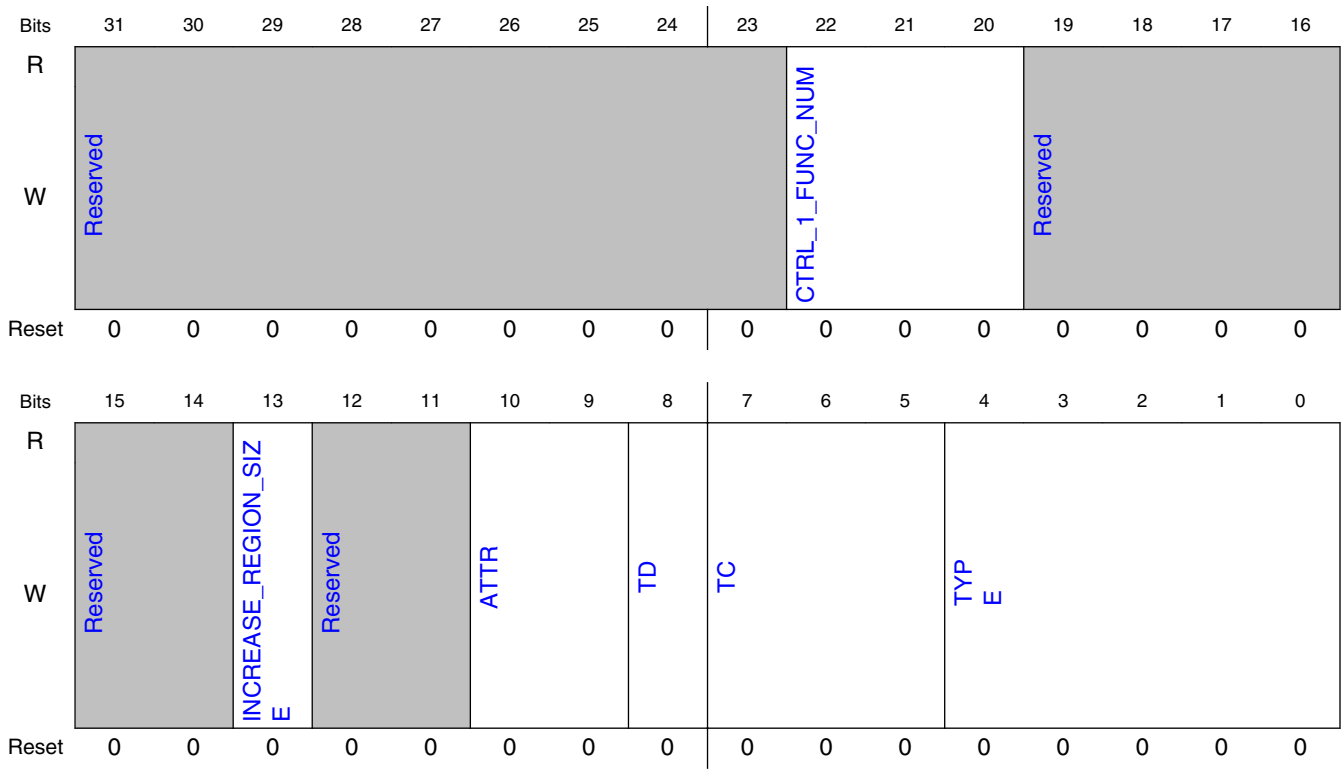
Field	Function
31-8 RSVDP_8	Reserved for future use.
7-6 L1SUB_T_PCLKACK	Max delay (in 1us units) between a MAC request to remove the clock on mac_phy_pclkreq_n and a PHY response on phy_mac_pclkack_n. If the PHY does not respond within this time the request is aborted. Range is 0..3 Note: This register field is sticky.
5-2 L1SUB_T_L1_2	Duration (in 1us units) of L1.2. Range is 0.15. Note: The timeout value can vary by 50%. Note: This register field is sticky.
1-0 L1SUB_T_POWER_OFF	Duration (in 1us units) of L1.2.Entry. Range is 0.3. Note: The timeout value can vary by 50%. Note: This register field is sticky.

11.3.5.1.131 iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_OUTBOUND_0)

11.3.5.1.131.1 Offset

Register	Offset
IATU_REGION_CTRL_1_OFF_OUTBOUND_0	8000_0000h

11.3.5.1.131.2 Diagram



11.3.5.1.131.3 Fields

Field	Function
31-23 —	Reserved.
22-20 CTRL_1_FUNC_NUM	Function Number. - When the address of an outbound TLP is matched to this region and the FUNC_BYPASS field in the "iATU Control 2 Register" is '0', then the function number used in generating the function part of the requester ID (RID) field of the TLP is taken from this 5-bit register. When you are using the AXI Bridge, then this field is swapped before AXI decomposition occurs so that the correct "Max_Read_Request_Size" and "Max_Payload_Size" values are used. Note: This register field is sticky.
19-14 —	Reserved.
13 INCREASE_REGION_SIZE	Increase the maximum ATU Region size. When set, the size is determined by the Maximum Size of iATU Region. When clear, the maximum ATU Region size is 4 GB (default). Note: This register field is sticky.
12-11 —	Reserved.
10-9 ATTR	When the address of an outbound TLP is matched to this region, then the ATTR field of the TLP is changed to the value in this register. Note: This register field is sticky.

Table continues on the next page...

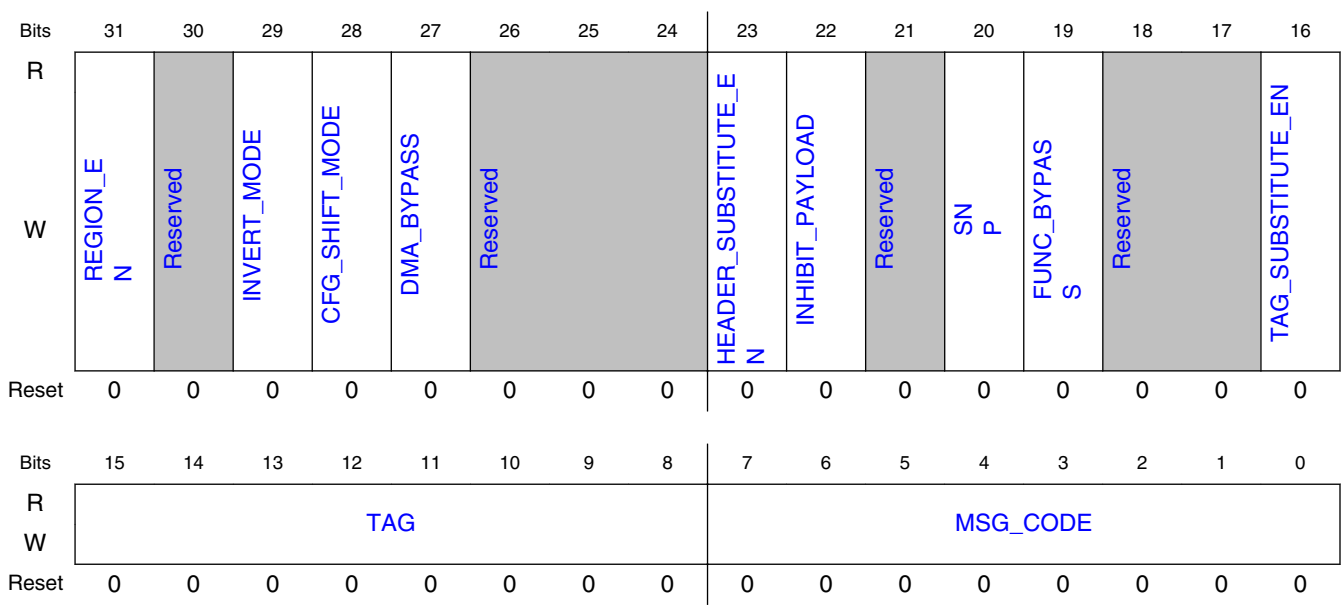
Field	Function
8 TD	When the address of an outbound TLP is matched to this region, then the TD field of the TLP is changed to the value in this register. Note: This register field is sticky.
7-5 TC	When the address of an outbound TLP is matched to this region, then the TC field of the TLP is changed to the value in this register. Note: This register field is sticky.
4-0 TYPE	When the address of an outbound TLP is matched to this region, then the TYPE field of the TLP is changed to the value in this register. Note: This register field is sticky.

11.3.5.1.132 iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_OUTBOUND_0)

11.3.5.1.132.1 Offset

Register	Offset
IATU_REGION_CTRL_2_OFF_OUTBOUND_0	8000_0004h

11.3.5.1.132.2 Diagram



11.3.5.1.132.3 Fields

Field	Function
31 REGION_EN	Region Enable. This bit must be set to '1' for address translation to take place. Note: This register field is sticky.
30 —	Reserved.
29 INVERT_MODE	Invert Mode. When set the address matching region is inverted. Therefore, an address match occurs when the untranslated address is in the region outside the defined range (Base Address to Limit Address). Note: This register field is sticky.
28 CFG_SHIFT_MODE	CFG Shift Mode. The iATU uses bits [27:12] of the untranslated address (on the XALI0/1/2 interface or AXI slave interface address) to form the BDF number of the outgoing CFG TLP. This supports the Enhanced Configuration Address Mapping (ECAM) mechanism (Section 7.2.2 of the PCI Express Base 3.1 Specification, revision 1.0) by allowing all outgoing I/O and MEM TLPs (that have been translated to CFG) to be mapped from memory space into any 256 MB region of the PCIe configuration space. Note: This register field is sticky.
27 DMA_BYPASS	DMA Bypass Mode. Allows request TLPs which are initiated by the DMA controller to pass through the iATU untranslated. Note: This field is reserved for the SW product. You must set it to '0'. Note: This register field is sticky.
26-24 —	Reserved.
23 HEADER_SUBSTITUTE_EN	Header Substitute Enable. When enabled and region address is matched, the iATU fully substitutes bytes 8-11 (for 3 DWORD header) or bytes 12-15 (for 4 DWORD header) of the outbound TLP header with the contents of the LWR_TARGET_RW field in IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i. - 1: LWR_TARGET_RW in the iATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register is used to fill bytes 8-to-11 (for 3 DWORD header) or bytes 12-to-15 (for 4 DWORD header) of the translated TLP header. - 0: LWR_TARGET_RW in the iATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register forms the new address of the translated region. Note: This register field is sticky.
22 INHIBIT_PAYLOAD	Inhibit TLP Payload Data for TLP's in Matched Region; assign iATU region to be TLP without data. When enabled and region address is matched, the iATU marks all TLPs as having no payload data by forcing the TLP header Fmt[1] bit =0, regardless of the application inputs such as slv_wstrb. - 1: Fmt[1] =0 so that only TLP type without data is sent. For example, a Msg instead of MsgD will be sent. - 0: Fmt[1] =0/1 so that TLPs with or without data can be sent. Note: This register field is sticky.
21 —	Reserved.
20 SNP	Serialize Non-Posted Requests. In this mode, when the AXI Bridge is populated, same AXI ID Non-Posted Read/Write Requests are transmitted on the wire if there are no other same ID Non-Posted Requests outstanding. Note: This register field is sticky.
19 FUNC_BYPASS	Function Number Translation Bypass. In this mode, the function number of the translated TLP is taken from your application transmit interface and not from the CTRL_1_FUNC_NUM field of the "iATU Control 1 Register" or the VF_NUMBER field of the "iATU Control 3 Register." Note: This register field is sticky.
18-17 —	Reserved.
16 TAG_SUBSTITUTE_EN	TAG Substitute Enable. When enabled and region address is matched, the iATU substitutes the TAG field of the outbound TLP header with the contents of the TAG field in this register. The expected usage scenario is translation from AXI MWr to Vendor Defined Msg/MsgD. Note: This register field is sticky.
15-8 TAG	TAG. The substituted TAG field (byte 6) in the outgoing TLP header when TAG_SUBSTITUTE_EN is set. Note: This register field is sticky.

Table continues on the next page...

Field	Function
7-0 MSG_CODE	MSG TLPs (Message Code). When the address of an outbound TLP is matched to this region, and the translated TLP TYPE field is Msg or MsgD; then the message field of the TLP is changed to the value in this register. Memory TLPs: (ST;Steering Tag). When the ST field of an outbound TLP is matched to this region, and the translated TLP TYPE field targets memory space; then the ST field of the TLP is changed to the value in this register. Note: This register field is sticky.

11.3.5.1.133 iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_OUTBOUND_0)

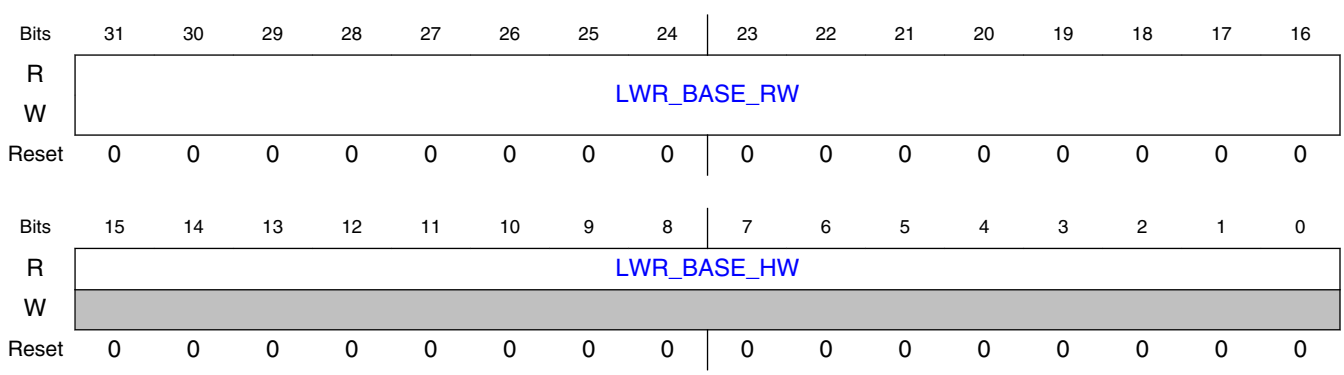
11.3.5.1.133.1 Offset

Register	Offset
IATU_LWR_BASE_ADDR_OFF_OUTBOUND_0	8000_0008h

11.3.5.1.133.2 Function

iATU Lower Base Address Register. The Minimum Size of iATU Region (Value Range: 4 kB, 8 kB, 16 kB, 32 kB, 64 kB defaults to 64 kB) specifies the minimum size of an address translation region. For example, if set to 64 kB; the lower 16 bits of the Base, Limit and Target registers are zero and all address regions are aligned on 64 kB boundaries. More precisely, the lower log2(Minimum Size of iATU Region) bits are zero.

11.3.5.1.133.3 Diagram



11.3.5.1.133.4 Fields

Field	Function
31-16 LWR_BASE_RW	Forms bits [31:n] of the start address of the address region to be translated. n is log2(Minimum Size of iATU Region) Note: This register field is sticky.
15-0 LWR_BASE_HW	Forms bits [n-1:0] of the start address of the address region to be translated. The start address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. n is log2(Minimum Size of iATU Region)

11.3.5.1.134 iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_0)

11.3.5.1.134.1 Offset

Register	Offset
IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_0	8000_000Ch

11.3.5.1.134.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.134.3 Fields

Field	Function
31-0 UPPER_BASE_RW	Forms bits [63:32] of the start (and end) address of the address region to be translated. In systems with a 32-bit address space, this register is not used and therefore writing to this register has no effect. Note: This register field is sticky.

11.3.5.1.135 iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_OUTBOUND_0)

11.3.5.1.135.1 Offset

Register	Offset
IATU_LIMIT_ADDR_OFF_OUTBOUND_0	8000_0010h

11.3.5.1.135.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LIMIT_ADDR_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LIMIT_ADDR_HW															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

11.3.5.1.135.3 Fields

Field	Function
31-16 LIMIT_ADDR_RW	Forms upper bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. Note: This register field is sticky.
15-0 LIMIT_ADDR_HW	Forms lower bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller.

11.3.5.1.136 iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_0)

11.3.5.1.136.1 Offset

Register	Offset
IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_0	8000_0014h

11.3.5.1.136.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LWR_TARGET_RW_OUTBOUND															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LWR_TARGET_RW_OUTBOUND															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.136.3 Fields

Field	Function
31-0 LWR_TARGET_RW_OUTBOUND	When HEADER_SUBSTITUTE_EN in IATU_REGION_CTRL_2_OFF_OUTBOUND_ is '0' (normal operation): - LWR_TARGET_RW[31:n] forms MSB's of the Lower Target part of the new address of the translated region; - LWR_TARGET_RW[n-1:0] are not used. (The start address must be aligned to the Minimum Size of iATU Region kB boundary, so the lower bits of the start address of the new address of the translated region (bits n-1:0) are always '0'). - n is log2(Minimum Size of iATU Region). When HEADER_SUBSTITUTE_EN in IATU_REGION_CTRL_2_OFF_OUTBOUND_i is '1': - LWR_TARGET_RW[31:0] forms bytes 8-11 (for 3 dword header) or bytes 12-15 (for 4 dword header) of the outbound TLP header. Usage scenarios include the transmission of Vendor Defined Messages where the controller determines the content of bytes 12 to 15 of the TLP header. Note: This register field is sticky.

11.3.5.1.137 iATU Upper Target Address Register. (IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_0)

11.3.5.1.137.1 Offset

Register	Offset
IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_0	8000_0018h

11.3.5.1.137.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UPPER_TARGET_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	UPPER_TARGET_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.137.3 Fields

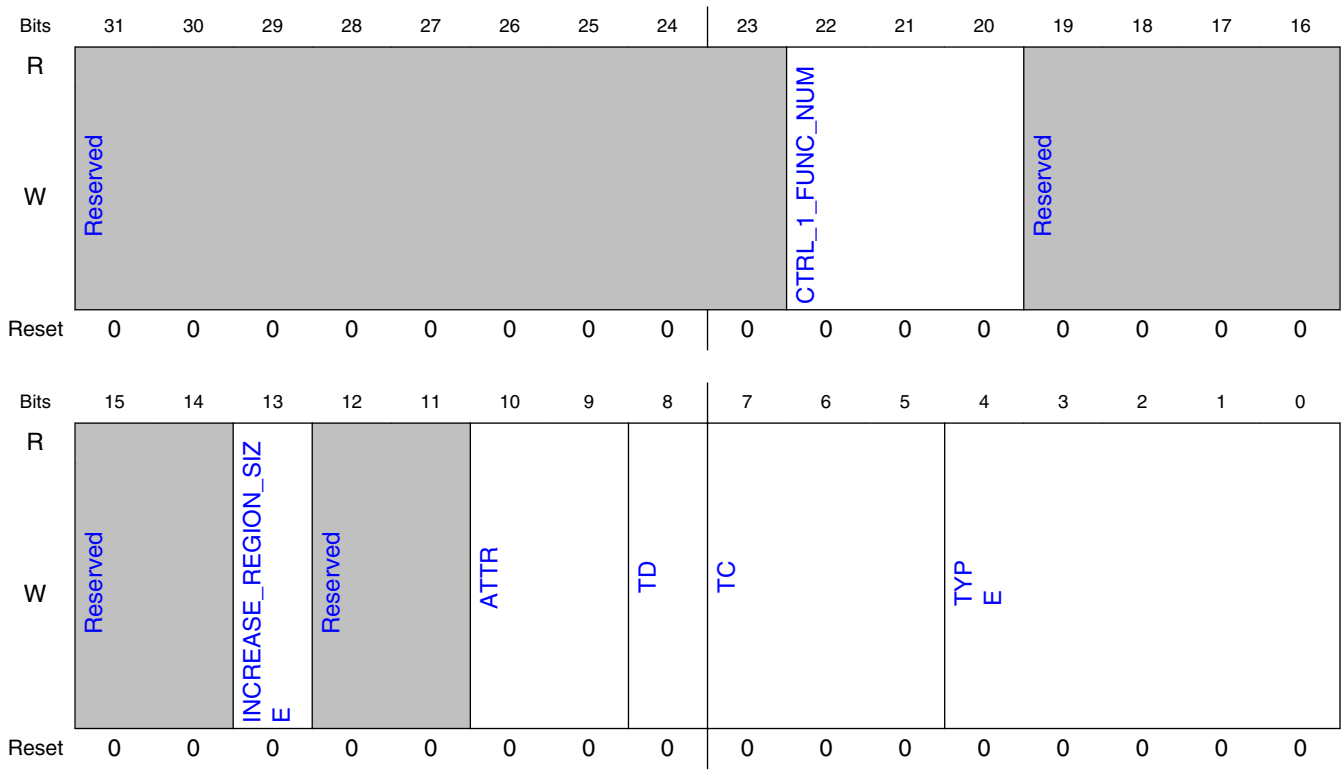
Field	Function
31-0 UPPER_TARGET_RW	Forms bits [63:32] of the start address (Upper Target part) of the new address of the translated region. Note: This register field is sticky.

11.3.5.1.138 iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_INBOUND_0)

11.3.5.1.138.1 Offset

Register	Offset
IATU_REGION_CTRL_1_OFF_INBOUND_0	8000_0100h

11.3.5.1.138.2 Diagram



11.3.5.1.138.3 Fields

Field	Function
31-23 —	Reserved.
22-20 CTRL_1_FUNC_NUM	Function Number. - MEM-I/O: When the Address and BAR matching logic in the controller indicate that a MEM-I/O transaction matches a BAR in the function corresponding to this value, then address translation proceeds. This check is only performed if the "Function Number Match Enable" bit of the "iATU Control 2 Register" is set. - CFG0/CFG1: When the destination function number as specified in the routing ID of the TLP header matches the function, then address translation proceeds. This check is only performed if the "Function Number Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
19-14 —	Reserved.
13 INCREASE_REGION_SIZE	Increase the maximum ATU Region size. When set, the size is determined by the Maximum Size of iATU Region. When clear, the maximum ATU Region size is 4 GB (default). Note: This register field is sticky.
12-11 —	Reserved.
10-9 ATTR	When the ATTR field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "ATTR Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.

Table continues on the next page...

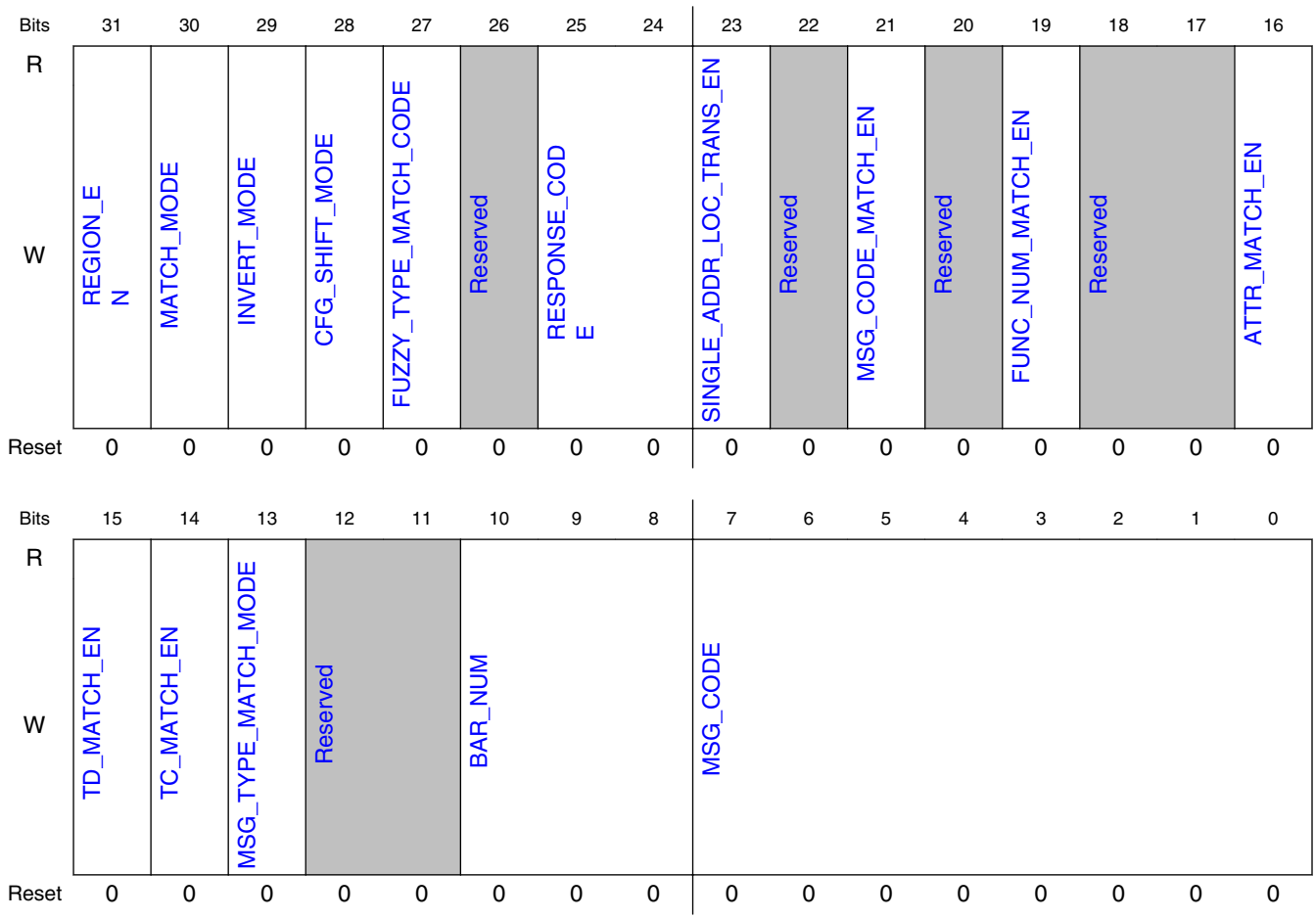
Field	Function
8 TD	When the TD field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "TD Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
7-5 TC	When the TC field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "TC Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
4-0 TYPE	When the TYPE field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). Note: This register field is sticky.

11.3.5.1.139 iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_INBOUND_0)

11.3.5.1.139.1 Offset

Register	Offset
IATU_REGION_CTRL_2_OFF_INBOUND_0	8000_0104h

11.3.5.1.139.2 Diagram



11.3.5.1.139.3 Fields

Field	Function
31 REGION_EN	Region Enable. This bit must be set to '1' for address translation to take place. Note: This register field is sticky.
30 MATCH_MODE	Match Mode. Determines Inbound matching mode for TLPs. The mode depends on the type of TLP that is received as follows: For MEM-I/O TLPs, this field is interpreted as follows: - 0: Address Match Mode. The iATU operates using addresses as in the outbound direction. The Region Base and Limit Registers must be setup. - 1:BAR Match Mode. BAR matching is used. The "BAR Number" field is relevant. Not used for RC. For CFG0 TLPs, this field is interpreted as follows: - 0: Routing ID Match Mode. The iATU interprets the Routing ID (Bytes 8 to 11 of TLP header) as an address. This corresponds to the upper 16 bits of the address in MEM-I/O transactions. The Routing ID of the TLP must be within the base and limit of the iATU region for matching to proceed. - 1: Accept Mode. The iATU accepts all CFG0 transactions as address matches. The routing ID in the CFG0 TLP is ignored. This is useful as all received CFG0 TLPs should be processed regardless of the Bus number. For MSG/MSGD TLPs, this field is interpreted as follows: - 0: Address Match Mode. The iATU treats the third dword and fourth dword of the inbound MSG/MSGD TLP as an address and it is matched against the Region Base and Limit Registers. - 1: Vendor ID Match Mode. This mode is relevant for ID-routed Vendor Defined Messages. The iATU ignores the Routing ID (Bus, Device, Function) in bits [31:16] of the third dword of the TLP header, but matches

Table continues on the next page...

Field	Function
	against the Vendor ID in bits [15:0] of the third dword of the TLP header. Bits [15:0] of the Region Upper Base register should be programmed with the required Vendor ID. The lower Base and Limit Register should be programmed to translate TLPs based on vendor specific information in the fourth dword of the TLP header. - If SINGLE_ADDRESS_LOCATION_TRANSLATE_EN = 1 AND MSG_TYPE_MATCH_MODE = 1, then Match Mode is ignored. Note: This register field is sticky.
29 INVERT_MODE	Invert Mode. When set the address matching region is inverted. Therefore, an address match occurs when the untranslated address is in the region outside the defined range (Base Address to Limit Address). Note: This register field is sticky.
28 CFG_SHIFT_MODE	CFG Shift Mode. This is useful for CFG transactions where the PCIe configuration mechanism maps bits [27:12] of the address to the bus/device and function number. This allows a CFG configuration space to be located in any 256MB window of your application memory space using a 28-bit effective address. Shifts bits [31:16] of the untranslated address to form bits [27:12] of the translated address. Note: This register field is sticky.
27 FUZZY_TYPE_MATCH_CODE	Fuzzy Type Match Mode. When enabled, the iATU relaxes the matching of the TLP TYPE field against the expected TYPE field so that - CfgRd0 and CfgRd1 TLPs are seen as identical. Similarly with CfgWr0 and CfgWr1. - MWr, MRd and MRdLk TLPs are seen as identical - The Routing field of Msg/MsgD TLPs is ignored - FetchAdd, Swap and CAS are seen as identical. For example, CFG0 in the TYPE field in the "iATU Control 1 Register" matches against an inbound CfgRd0, CfgRd1, CfgWr0 or CfgWr1 TLP. Note: This register field is sticky.
26 —	Reserved.
25-24 RESPONSE_CODE	Response Code. Defines the type of response to give for accesses matching this region. This overrides the normal RADM filter response. Note that this feature is not available for any region where Single Address Location Translate is enabled. - 00 - Normal RADM filter response is used. - 01 - Unsupported request (UR) - 10 - Completer abort (CA) - 11 - Not used / undefined / reserved. Note: This register field is sticky.
23 SINGLE_ADDR_LOC_TRANS_EN	Single Address Location Translate Enable. When enabled, Rx TLPs can be translated to a single address location as determined by the target address register of the iATU region. The main usage scenario is translation of Messages (such as Vendor Defined or ATS Messages) to MWr TLPs when the AXI bridge is enabled. Note: This register field is sticky.
22 —	Reserved.
21 MSG_CODE_MATCH_EN	Message Code Match Enable (Msg TLPS). Ensures that a successful message Code TLP field comparison match (see Message Code field of the "iATU Control 2 Register") occurs (in MSG transactions) for address translation to proceed. ST Match Enable (Mem TLPs). Ensures that a successful ST TLP field comparison match (see ST field of the "iATU Control 2 Register") occurs (in MEM transactions) for address translation to proceed. Note: This register field is sticky.
20 —	Reserved.
19 FUNC_NUM_MATCH_EN	Function Number Match Enable. Ensures that a successful Function Number TLP field comparison match (see Function Number field of the "iATU Control 1 Register") occurs (in MEM-I/O and CFG0/CFG1 transactions) for address translation to proceed. Note: This register field is sticky.
18-17 —	Reserved.
16 ATTR_MATCH_EN	ATTR Match Enable. Ensures that a successful ATTR TLP field comparison match (see ATTR field of the "iATU Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky.

Table continues on the next page...

Field	Function
15 TD_MATCH_EN	TD Match Enable. Ensures that a successful TD TLP field comparison match (see TD field of the "iATU Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky.
14 TC_MATCH_EN	TC Match Enable. Ensures that a successful TC TLP field comparison match (see TC field of the "iATU Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky.
13 MSG_TYPE_MATCH_MODE	Message Type Match Mode. When enabled, and if single address location translate enable is set, then inbound TLPs of type MSG/MSGd which match the type field of the iatu_region_ctrl_1_OFF_inbound register (=>TYPE[4:3]=2'b10) will be translated. Message type match mode overrides any value of MATCH_MODE field in this register. Usage scenarios for this are translation of VDM or ATS messages when AXI bridge is configured on client interface. Note: This register field is sticky.
12-11 —	Reserved.
10-8 BAR_NUM	BAR Number. When the BAR number of an inbound MEM or IO TLP " that is matched by the normal internal BAR address matching mechanism " is the same as this field, address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "Match Mode" bit of the "iATU Control 2 Register" is set. - 000b - BAR0 - 001b - BAR1 - 010b - BAR2 - 011b - BAR3 - 100b - BAR4 - 101b - BAR5 - 110b - ROM - 111b - reserved - IO translation would require either 00100b or 00101b in the inbound TLP TYPE; the BAR Number set in the range 000b - 101b and that BAR configured as an IO BAR. Note: This register field is sticky.
7-0 MSG_CODE	MSG TLPs: (Message Code). When the TYPE field of an inbound Msg/MsgD TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "Message Code Match Enable" bit of the "iATU Control 2 Register" is set. Memory TLPs: (ST;Steering Tag). When the ST field of an inbound TLP is matched to this value, then address translation proceeds. This check is only performed if the "ST Match Enable" bit of the "iATU Control2 Register" is set. The setting is independent of the setting of the TH field. Note: This register field is sticky.

11.3.5.1.140 iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_INBOUND_0)

11.3.5.1.140.1 Offset

Register	Offset
IATU_LWR_BASE_ADDR_OFF_INBOUND_0	8000_0108h

11.3.5.1.140.2 Function

iATU Lower Base Address Register. The Minimum Size of iATU Region (Value Range: 4 kB, 8 kB, 16 kB, 32 kB, 64 kB defaults to 64 kB) specifies the minimum size of an address translation region. For example, if set to 64 kB; the lower 16 bits of the Base, Limit and Target registers are zero and all address regions are aligned on 64 kB boundaries. More precisely, the lower log2(Minimum Size of iATU Region) bits are zero.

11.3.5.1.140.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LWR_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LWR_BASE_HW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.140.4 Fields

Field	Function
31-16 LWR_BASE_RW	Forms bits [31:n] of the start address of the address region to be translated. n is log2(Minimum Size of iATU Region) Note: This register field is sticky.
15-0 LWR_BASE_HW	Forms bits [n-1:0] of the start address of the address region to be translated. The start address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. n is log2(Minimum Size of iATU Region)

11.3.5.1.141 iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_INBOUND_0)

11.3.5.1.141.1 Offset

Register	Offset
IATU_UPPER_BASE_ADDR_OFF_INBOUND_0	8000_010Ch

11.3.5.1.141.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.141.3 Fields

Field	Function
31-0 UPPER_BASE_RW	Forms bits [63:32] of the start (and end) address of the address region to be translated. Note: This register field is sticky.

11.3.5.1.142 iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_INBOUND_0)

11.3.5.1.142.1 Offset

Register	Offset
IATU_LIMIT_ADDR_OFF_INBOUND_0	8000_0110h

11.3.5.1.142.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LIMIT_ADDR_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LIMIT_ADDR_HW															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

11.3.5.1.142.3 Fields

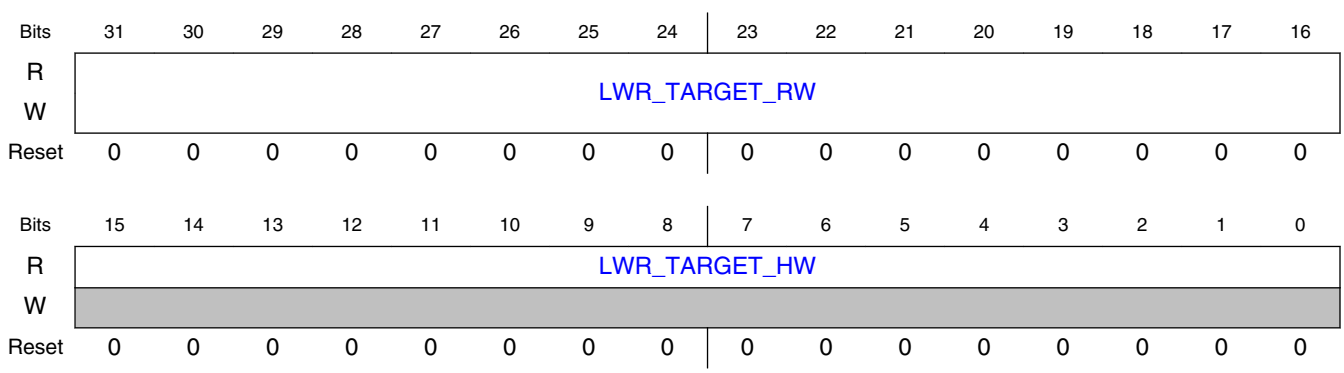
Field	Function
31-16 LIMIT_ADDR_RW	Forms upper bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. Note: This register field is sticky.
15-0 LIMIT_ADDR_HW	Forms lower bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller.

11.3.5.1.143 iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_INBOUND_0)

11.3.5.1.143.1 Offset

Register	Offset
IATU_LWR_TARGET_ADDR_OFF_INBOUND_0	8000_0114h

11.3.5.1.143.2 Diagram



11.3.5.1.143.3 Fields

Field	Function
31-16 LWR_TARGET_RW	Forms MSB's of the Lower Target part of the new address of the translated region. These bits are always '0'. - Field size depends on log2(Minimum Size of iATU Region) in address match mode. - Field size depends on log2(BAR_MASK+1) in BAR match mode. Note: This register field is sticky.

Table continues on the next page...

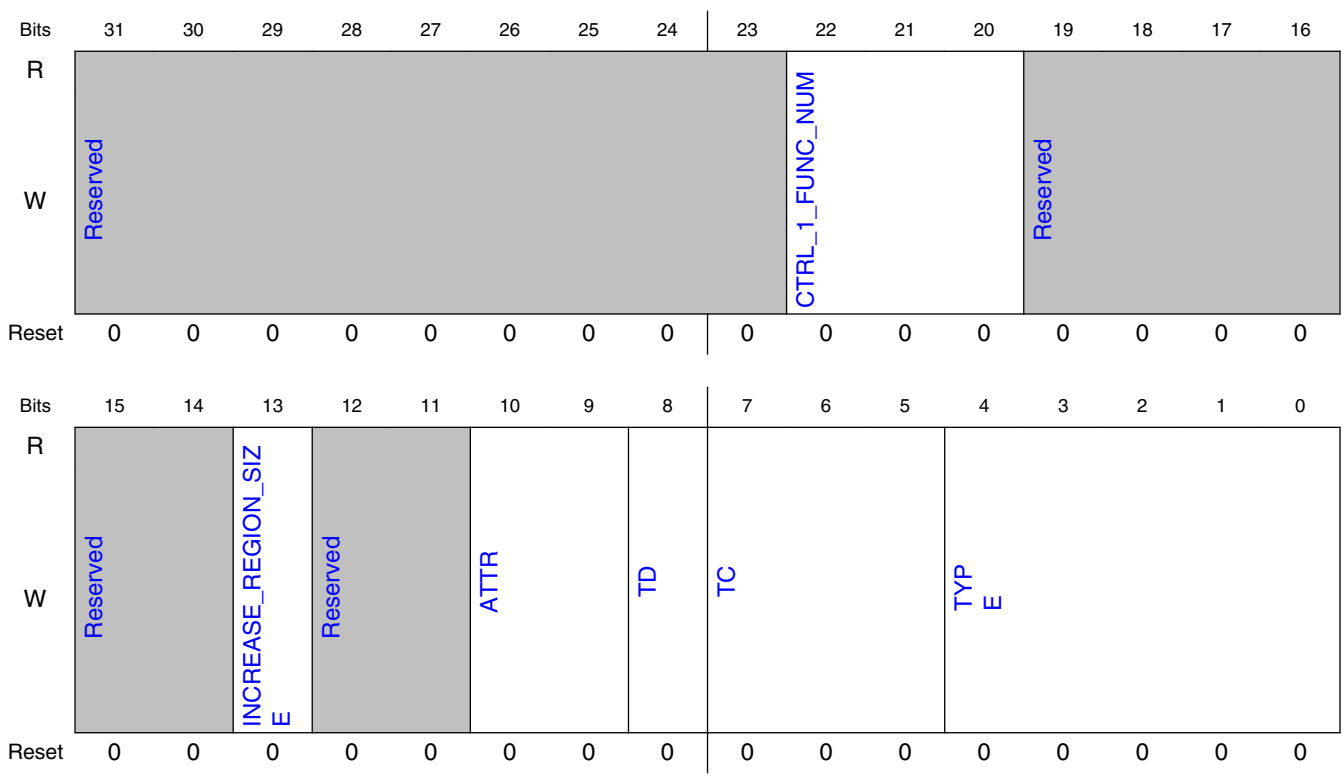
Field	Function
15-0 LWR_TARGET_HW	Forms the LSB's of the Lower Target part of the new address of the translated region. The start address must be aligned to the Minimum Size of iATU Region kB boundary (in address match mode); and to the Bar size boundary (in BAR match mode) so that these bits are always '0'. If the BAR is smaller than the iATU region size, then the iATU target address must align to the iATU region size; otherwise it must align to the BAR size. A write to this location is ignored by the PCIe controller. - Field size depends on log2(Minimum Size of iATU Region) in address match mode. - Field size depends on log2(BAR_MASK +1) in BAR match mode.

11.3.5.1.144 iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_OUTBOUND_1)

11.3.5.1.144.1 Offset

Register	Offset
IATU_REGION_CTRL_1_OFF_OUTBOUND_1	8000_0200h

11.3.5.1.144.2 Diagram



11.3.5.1.144.3 Fields

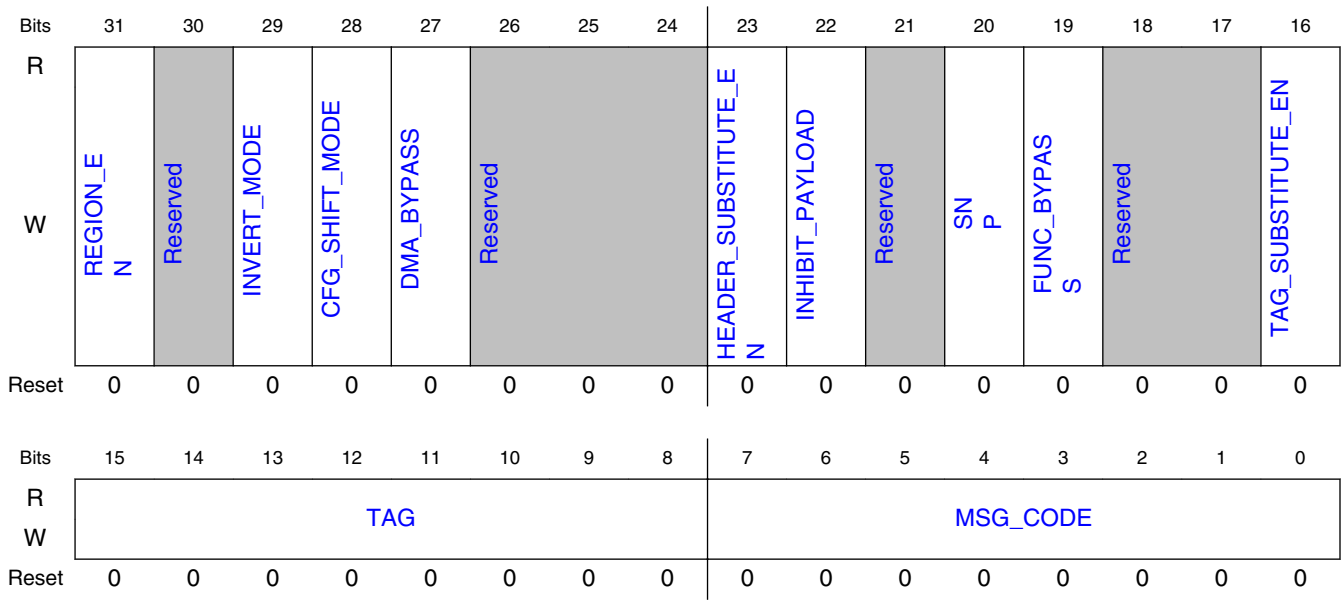
Field	Function
31-23 —	Reserved.
22-20 CTRL_1_FUNC_NUM	Function Number. - When the address of an outbound TLP is matched to this region and the FUNC_BYPASS field in the "iATU Control 2 Register" is '0', then the function number used in generating the function part of the requester ID (RID) field of the TLP is taken from this 5-bit register. When you are using the AXI Bridge, then this field is swapped before AXI decomposition occurs so that the correct "Max_Read_Request_Size" and "Max_Payload_Size" values are used. Note: This register field is sticky.
19-14 —	Reserved.
13 INCREASE_REGION_SIZE	Increase the maximum ATU Region size. When set, the size is determined by the Maximum Size of iATU Region. When clear, the maximum ATU Region size is 4 GB (default). Note: This register field is sticky.
12-11 —	Reserved.
10-9 ATTR	When the address of an outbound TLP is matched to this region, then the ATTR field of the TLP is changed to the value in this register. Note: This register field is sticky.
8 TD	When the address of an outbound TLP is matched to this region, then the TD field of the TLP is changed to the value in this register. Note: This register field is sticky.
7-5 TC	When the address of an outbound TLP is matched to this region, then the TC field of the TLP is changed to the value in this register. Note: This register field is sticky.
4-0 TYPE	When the address of an outbound TLP is matched to this region, then the TYPE field of the TLP is changed to the value in this register. Note: This register field is sticky.

11.3.5.1.145 iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_OUTBOUND_1)

11.3.5.1.145.1 Offset

Register	Offset
IATU_REGION_CTRL_2_OFF_OUTBOUND_1	8000_0204h

11.3.5.1.145.2 Diagram



11.3.5.1.145.3 Fields

Field	Function
31 REGION_EN	Region Enable. This bit must be set to '1' for address translation to take place. Note: This register field is sticky.
30 —	Reserved.
29 INVERT_MODE	Invert Mode. When set the address matching region is inverted. Therefore, an address match occurs when the untranslated address is in the region outside the defined range (Base Address to Limit Address). Note: This register field is sticky.
28 CFG_SHIFT_MODE	CFG Shift Mode. The iATU uses bits [27:12] of the untranslated address (on the XALI0/1/2 interface or AXI slave interface address) to form the BDF number of the outgoing CFG TLP. This supports the Enhanced Configuration Address Mapping (ECAM) mechanism (Section 7.2.2 of the PCI Express Base 3.1 Specification, revision 1.0) by allowing all outgoing I/O and MEM TLPs (that have been translated to CFG) to be mapped from memory space into any 256 MB region of the PCIe configuration space. Note: This register field is sticky.
27 DMA_BYPASS	DMA Bypass Mode. Allows request TLPs which are initiated by the DMA controller to pass through the iATU untranslated. Note: This field is reserved for the SW product. You must set it to '0'. Note: This register field is sticky.
26-24 —	Reserved.
23 HEADER_SUBSTITUTE_EN	Header Substitute Enable. When enabled and region address is matched, the iATU fully substitutes bytes 8-11 (for 3 DWORD header) or bytes 12-15 (for 4 DWORD header) of the outbound TLP header with the contents of the LWR_TARGET_RW field in IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i. - 1: LWR_TARGET_RW in the iATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register is used to fill bytes 8-to-11 (for 3 DWORD header) or bytes 12-to-15 (for 4 DWORD header) of the translated TLP header. - 0: LWR_TARGET_RW in the iATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register forms the new address of the translated region. Note: This register field is sticky.

Table continues on the next page...

Field	Function
22 INHIBIT_PAYLOAD	Inhibit TLP Payload Data for TLP's in Matched Region; assign iATU region to be TLP without data. When enabled and region address is matched, the iATU marks all TLPs as having no payload data by forcing the TLP header Fmt[1] bit =0, regardless of the application inputs such as slv_wstrb. - 1: Fmt[1] =0 so that only TLP type without data is sent. For example, a Msg instead of MsgD will be sent. - 0: Fmt[1] =0/1 so that TLPs with or without data can be sent. Note: This register field is sticky.
21 —	Reserved.
20 SNP	Serialize Non-Posted Requests. In this mode, when the AXI Bridge is populated, same AXI ID Non-Posted Read/Write Requests are transmitted on the wire if there are no other same ID Non-Posted Requests outstanding. Note: This register field is sticky.
19 FUNC_BYPASS	Function Number Translation Bypass. In this mode, the function number of the translated TLP is taken from your application transmit interface and not from the CTRL_1_FUNC_NUM field of the "iATU Control 1 Register" or the VF_NUMBER field of the "iATU Control 3 Register." Note: This register field is sticky.
18-17 —	Reserved.
16 TAG_SUBSTITUTE_EN	TAG Substitute Enable. When enabled and region address is matched, the iATU substitutes the TAG field of the outbound TLP header with the contents of the TAG field in this register. The expected usage scenario is translation from AXI MWr to Vendor Defined Msg/MsgD. Note: This register field is sticky.
15-8 TAG	TAG. The substituted TAG field (byte 6) in the outgoing TLP header when TAG_SUBSTITUTE_EN is set. Note: This register field is sticky.
7-0 MSG_CODE	MSG TLPs (Message Code). When the address of an outbound TLP is matched to this region, and the translated TLP TYPE field is Msg or MsgD; then the message field of the TLP is changed to the value in this register. Memory TLPs: (ST;Steering Tag). When the ST field of an outbound TLP is matched to this region, and the translated TLP TYPE field targets memory space; then the ST field of the TLP is changed to the value in this register. Note: This register field is sticky.

11.3.5.1.146 iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_OUTBOUND_1)

11.3.5.1.146.1 Offset

Register	Offset
IATU_LWR_BASE_ADDR_OFF_OUTBOUND_1	8000_0208h

11.3.5.1.146.2 Function

iATU Lower Base Address Register. The Minimum Size of iATU Region (Value Range: 4 kB, 8 kB, 16 kB, 32 kB, 64 kB defaults to 64 kB) specifies the minimum size of an address translation region. For example, if set to 64 kB; the lower 16 bits of the Base, Limit and Target registers are zero and all address regions are aligned on 64 kB boundaries. More precisely, the lower log2(Minimum Size of iATU Region) bits are zero.

11.3.5.1.146.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LWR_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LWR_BASE_HW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.146.4 Fields

Field	Function
31-16 LWR_BASE_RW	Forms bits [31:n] of the start address of the address region to be translated. n is log2(Minimum Size of iATU Region) Note: This register field is sticky.
15-0 LWR_BASE_HW	Forms bits [n-1:0] of the start address of the address region to be translated. The start address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. n is log2(Minimum Size of iATU Region)

11.3.5.1.147 iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_1)

11.3.5.1.147.1 Offset

Register	Offset
IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_1	8000_020Ch

11.3.5.1.147.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.147.3 Fields

Field	Function
31-0 UPPER_BASE_RW	Forms bits [63:32] of the start (and end) address of the address region to be translated. In systems with a 32-bit address space, this register is not used and therefore writing to this register has no effect. Note: This register field is sticky.

11.3.5.1.148 iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_OUTBOUND_1)

11.3.5.1.148.1 Offset

Register	Offset
IATU_LIMIT_ADDR_OFF_OUTBOUND_1	8000_0210h

11.3.5.1.148.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LIMIT_ADDR_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LIMIT_ADDR_HW															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

11.3.5.1.148.3 Fields

Field	Function
31-16 LIMIT_ADDR_RW	Forms upper bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. Note: This register field is sticky.
15-0 LIMIT_ADDR_HW	Forms lower bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller.

11.3.5.1.149 iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_1)

11.3.5.1.149.1 Offset

Register	Offset
IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_1	8000_0214h

11.3.5.1.149.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LWR_TARGET_RW_OUTBOUND															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LWR_TARGET_RW_OUTBOUND															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.149.3 Fields

Field	Function
31-0 LWR_TARGET_RW_OUTBOUND	When HEADER_SUBSTITUTE_EN in IATU_REGION_CTRL_2_OFF_OUTBOUND_ is '0' (normal operation): - LWR_TARGET_RW[31:n] forms MSB's of the Lower Target part of the new address of the translated region; - LWR_TARGET_RW[n-1:0] are not used. (The start address must be aligned to the Minimum Size of iATU Region kB boundary, so the lower bits of the start address of the new address of the translated region (bits n-1:0) are always '0'). - n is log2(Minimum Size of iATU Region). When

Field	Function
	HEADER_SUBSTITUTE_EN in IATU_REGION_CTRL_2_OFF_OUTBOUND_i is '1': - LWR_TARGET_RW[31:0] forms bytes 8-11 (for 3 dword header) or bytes 12-15 (for 4 dword header) of the outbound TLP header. Usage scenarios include the transmission of Vendor Defined Messages where the controller determines the content of bytes 12 to 15 of the TLP header. Note: This register field is sticky.

11.3.5.1.150 iATU Upper Target Address Register. (IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_1)

11.3.5.1.150.1 Offset

Register	Offset
IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_1	8000_0218h

11.3.5.1.150.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UPPER_TARGET_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	UPPER_TARGET_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.150.3 Fields

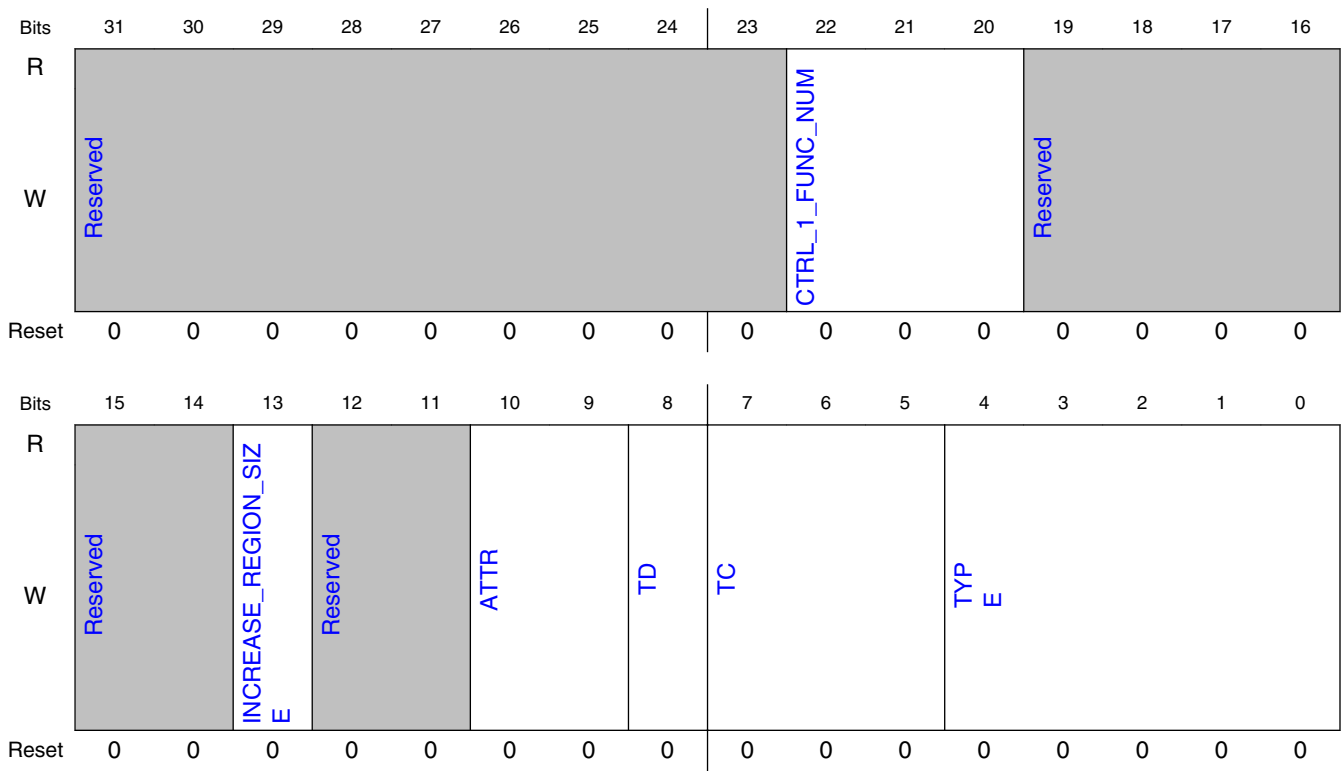
Field	Function
31-0 UPPER_TARGET_RW	Forms bits [63:32] of the start address (Upper Target part) of the new address of the translated region. Note: This register field is sticky.

11.3.5.1.151 iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_INBOUND_1)

11.3.5.1.151.1 Offset

Register	Offset
IATU_REGION_CTRL_1_OFF_INBOUND_1	8000_0300h

11.3.5.1.151.2 Diagram



11.3.5.1.151.3 Fields

Field	Function
31-23 —	Reserved.
22-20 CTRL_1_FUNC_NUM	Function Number. - MEM-I/O: When the Address and BAR matching logic in the controller indicate that a MEM-I/O transaction matches a BAR in the function corresponding to this value, then address translation proceeds. This check is only performed if the "Function Number Match Enable" bit of the "iATU Control 2 Register" is set. - CFG0/CFG1: When the destination function number as specified in the routing ID of the TLP header matches the function, then address translation proceeds. This check is only performed if the

Table continues on the next page...

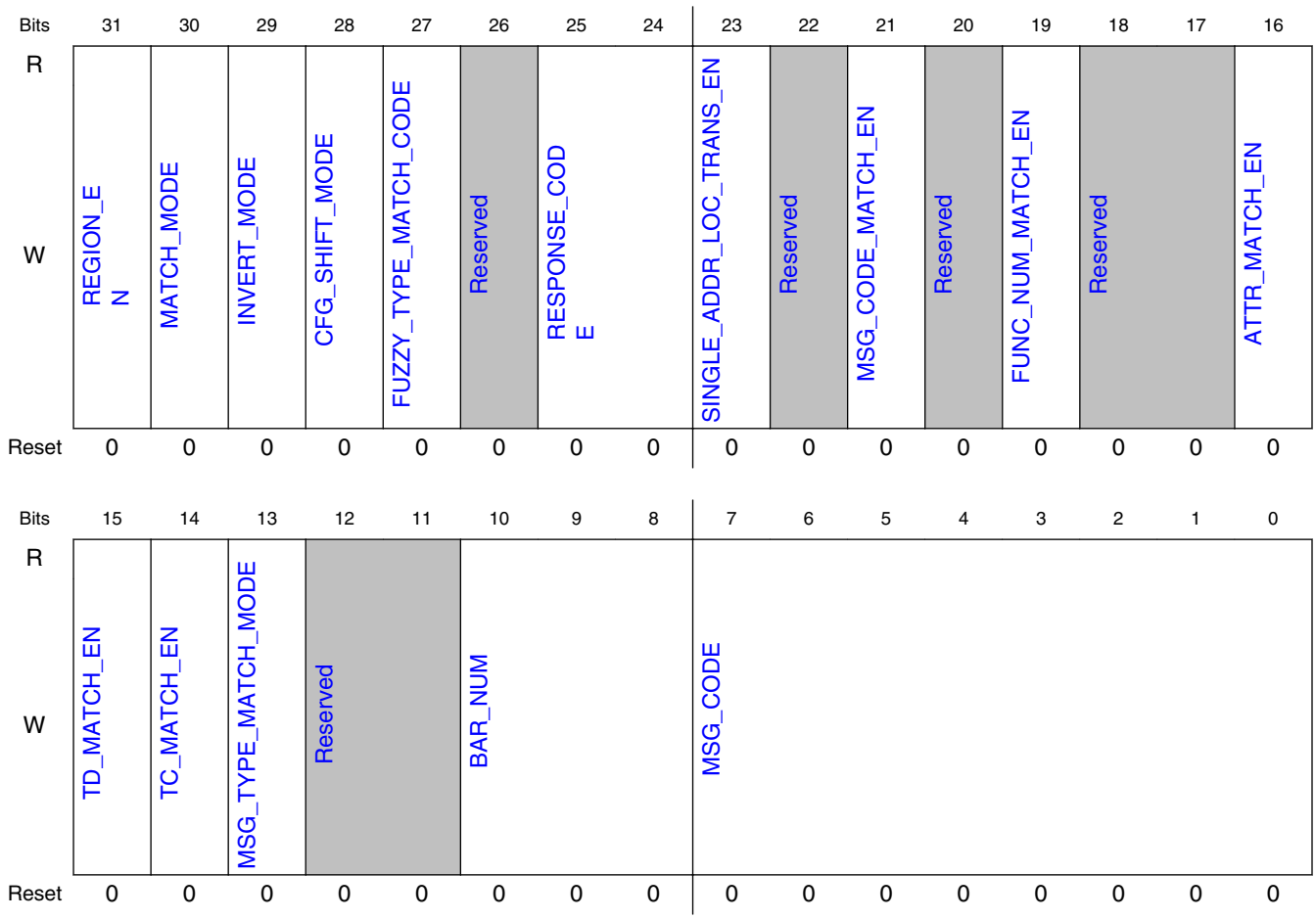
Field	Function
	"Function Number Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
19-14 —	Reserved.
13 INCREASE_REGION_SIZE	Increase the maximum ATU Region size. When set, the size is determined by the Maximum Size of iATU Region. When clear, the maximum ATU Region size is 4 GB (default). Note: This register field is sticky.
12-11 —	Reserved.
10-9 ATTR	When the ATTR field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "ATTR Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
8 TD	When the TD field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "TD Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
7-5 TC	When the TC field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "TC Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
4-0 TYPE	When the TYPE field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). Note: This register field is sticky.

11.3.5.1.152 iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_INBOUND_1)

11.3.5.1.152.1 Offset

Register	Offset
IATU_REGION_CTRL_2_OFF_INBOUND_1	8000_0304h

11.3.5.1.152.2 Diagram



11.3.5.1.152.3 Fields

Field	Function
31 REGION_EN	Region Enable. This bit must be set to '1' for address translation to take place. Note: This register field is sticky.
30 MATCH_MODE	Match Mode. Determines Inbound matching mode for TLPs. The mode depends on the type of TLP that is received as follows: For MEM-I/O TLPs, this field is interpreted as follows: - 0: Address Match Mode. The iATU operates using addresses as in the outbound direction. The Region Base and Limit Registers must be setup. - 1:BAR Match Mode. BAR matching is used. The "BAR Number" field is relevant. Not used for RC. For CFG0 TLPs, this field is interpreted as follows: - 0: Routing ID Match Mode. The iATU interprets the Routing ID (Bytes 8 to 11 of TLP header) as an address. This corresponds to the upper 16 bits of the address in MEM-I/O transactions. The Routing ID of the TLP must be within the base and limit of the iATU region for matching to proceed. - 1: Accept Mode. The iATU accepts all CFG0 transactions as address matches. The routing ID in the CFG0 TLP is ignored. This is useful as all received CFG0 TLPs should be processed regardless of the Bus number. For MSG/MSGD TLPs, this field is interpreted as follows: - 0: Address Match Mode. The iATU treats the third dword and fourth dword of the inbound MSG/MSGD TLP as an address and it is matched against the Region Base and Limit Registers. - 1: Vendor ID Match Mode. This mode is relevant for ID-routed Vendor Defined Messages. The iATU ignores the Routing ID (Bus, Device, Function) in bits [31:16] of the third dword of the TLP header, but matches

Table continues on the next page...

Field	Function
	against the Vendor ID in bits [15:0] of the third dword of the TLP header. Bits [15:0] of the Region Upper Base register should be programmed with the required Vendor ID. The lower Base and Limit Register should be programmed to translate TLPs based on vendor specific information in the fourth dword of the TLP header. - If SINGLE_ADDRESS_LOCATION_TRANSLATE_EN = 1 AND MSG_TYPE_MATCH_MODE = 1, then Match Mode is ignored. Note: This register field is sticky.
29 INVERT_MODE	Invert Mode. When set the address matching region is inverted. Therefore, an address match occurs when the untranslated address is in the region outside the defined range (Base Address to Limit Address). Note: This register field is sticky.
28 CFG_SHIFT_MODE	CFG Shift Mode. This is useful for CFG transactions where the PCIe configuration mechanism maps bits [27:12] of the address to the bus/device and function number. This allows a CFG configuration space to be located in any 256MB window of your application memory space using a 28-bit effective address. Shifts bits [31:16] of the untranslated address to form bits [27:12] of the translated address. Note: This register field is sticky.
27 FUZZY_TYPE_MATCH_CODE	Fuzzy Type Match Mode. When enabled, the iATU relaxes the matching of the TLP TYPE field against the expected TYPE field so that - CfgRd0 and CfgRd1 TLPs are seen as identical. Similarly with CfgWr0 and CfgWr1. - MWr, MRd and MRdLk TLPs are seen as identical - The Routing field of Msg/MsgD TLPs is ignored - FetchAdd, Swap and CAS are seen as identical. For example, CFG0 in the TYPE field in the "iATU Control 1 Register" matches against an inbound CfgRd0, CfgRd1, CfgWr0 or CfgWr1 TLP. Note: This register field is sticky.
26 —	Reserved.
25-24 RESPONSE_CODE	Response Code. Defines the type of response to give for accesses matching this region. This overrides the normal RADM filter response. Note that this feature is not available for any region where Single Address Location Translate is enabled. - 00 - Normal RADM filter response is used. - 01 - Unsupported request (UR) - 10 - Completer abort (CA) - 11 - Not used / undefined / reserved. Note: This register field is sticky.
23 SINGLE_ADDR_LOC_TRANS_EN	Single Address Location Translate Enable. When enabled, Rx TLPs can be translated to a single address location as determined by the target address register of the iATU region. The main usage scenario is translation of Messages (such as Vendor Defined or ATS Messages) to MWr TLPs when the AXI bridge is enabled. Note: This register field is sticky.
22 —	Reserved.
21 MSG_CODE_MATCH_EN	Message Code Match Enable (Msg TLPS). Ensures that a successful message Code TLP field comparison match (see Message Code field of the "iATU Control 2 Register") occurs (in MSG transactions) for address translation to proceed. ST Match Enable (Mem TLPs). Ensures that a successful ST TLP field comparison match (see ST field of the "iATU Control 2 Register") occurs (in MEM transactions) for address translation to proceed. Note: This register field is sticky.
20 —	Reserved.
19 FUNC_NUM_MATCH_EN	Function Number Match Enable. Ensures that a successful Function Number TLP field comparison match (see Function Number field of the "iATU Control 1 Register") occurs (in MEM-I/O and CFG0/CFG1 transactions) for address translation to proceed. Note: This register field is sticky.
18-17 —	Reserved.
16 ATTR_MATCH_EN	ATTR Match Enable. Ensures that a successful ATTR TLP field comparison match (see ATTR field of the "iATU Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky.

Table continues on the next page...

Field	Function
15 TD_MATCH_EN	TD Match Enable. Ensures that a successful TD TLP field comparison match (see TD field of the "iATU Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky.
14 TC_MATCH_EN	TC Match Enable. Ensures that a successful TC TLP field comparison match (see TC field of the "iATU Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky.
13 MSG_TYPE_MATCH_MODE	Message Type Match Mode. When enabled, and if single address location translate enable is set, then inbound TLPs of type MSG/MSGd which match the type field of the iatu_region_ctrl_1_OFF_inbound register (=>TYPE[4:3]=2'b10) will be translated. Message type match mode overrides any value of MATCH_MODE field in this register. Usage scenarios for this are translation of VDM or ATS messages when AXI bridge is configured on client interface. Note: This register field is sticky.
12-11 —	Reserved.
10-8 BAR_NUM	BAR Number. When the BAR number of an inbound MEM or IO TLP " that is matched by the normal internal BAR address matching mechanism " is the same as this field, address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "Match Mode" bit of the "iATU Control 2 Register" is set. - 000b - BAR0 - 001b - BAR1 - 010b - BAR2 - 011b - BAR3 - 100b - BAR4 - 101b - BAR5 - 110b - ROM - 111b - reserved - IO translation would require either 00100b or 00101b in the inbound TLP TYPE; the BAR Number set in the range 000b - 101b and that BAR configured as an IO BAR. Note: This register field is sticky.
7-0 MSG_CODE	MSG TLPs: (Message Code). When the TYPE field of an inbound Msg/MsgD TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "Message Code Match Enable" bit of the "iATU Control 2 Register" is set. Memory TLPs: (ST;Steering Tag). When the ST field of an inbound TLP is matched to this value, then address translation proceeds. This check is only performed if the "ST Match Enable" bit of the "iATU Control2 Register" is set. The setting is independent of the setting of the TH field. Note: This register field is sticky.

11.3.5.1.153 iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_INBOUND_1)

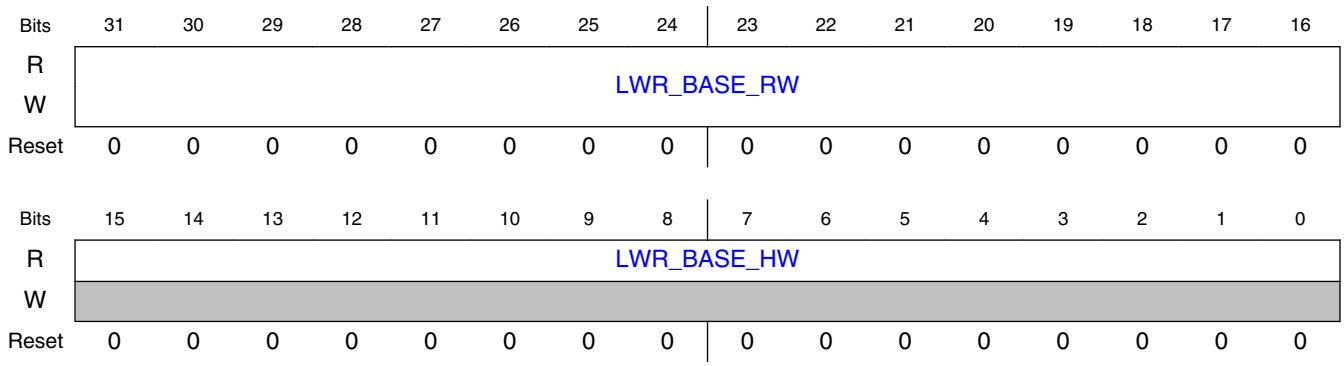
11.3.5.1.153.1 Offset

Register	Offset
IATU_LWR_BASE_ADDR_OFF_INBOUND_1	8000_0308h

11.3.5.1.153.2 Function

iATU Lower Base Address Register. The Minimum Size of iATU Region (Value Range: 4 kB, 8 kB, 16 kB, 32 kB, 64 kB defaults to 64 kB) specifies the minimum size of an address translation region. For example, if set to 64 kB; the lower 16 bits of the Base, Limit and Target registers are zero and all address regions are aligned on 64 kB boundaries. More precisely, the lower log2(Minimum Size of iATU Region) bits are zero.

11.3.5.1.153.3 Diagram



11.3.5.1.153.4 Fields

Field	Function
31-16 LWR_BASE_RW	Forms bits [31:n] of the start address of the address region to be translated. n is log2(Minimum Size of iATU Region) Note: This register field is sticky.
15-0 LWR_BASE_HW	Forms bits [n-1:0] of the start address of the address region to be translated. The start address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. n is log2(Minimum Size of iATU Region)

11.3.5.1.154 iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_INBOUND_1)

11.3.5.1.154.1 Offset

Register	Offset
IATU_UPPER_BASE_ADDR_OFF_INBOUND_1	8000_030Ch

11.3.5.1.154.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.154.3 Fields

Field	Function
31-0 UPPER_BASE_RW	Forms bits [63:32] of the start (and end) address of the address region to be translated. Note: This register field is sticky.

11.3.5.1.155 iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_INBOUND_1)

11.3.5.1.155.1 Offset

Register	Offset
IATU_LIMIT_ADDR_OFF_INBOUND_1	8000_0310h

11.3.5.1.155.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LIMIT_ADDR_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LIMIT_ADDR_HW															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

11.3.5.1.155.3 Fields

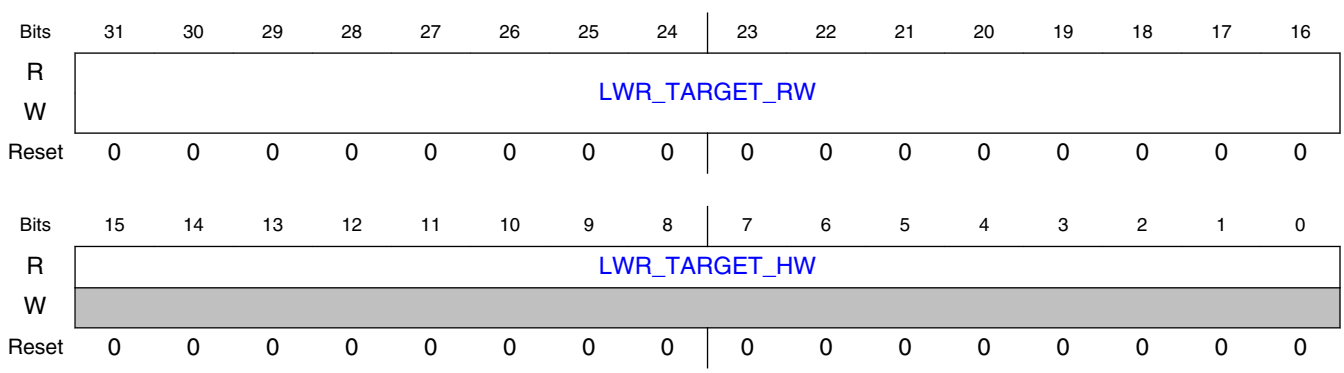
Field	Function
31-16 LIMIT_ADDR_RW	Forms upper bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. Note: This register field is sticky.
15-0 LIMIT_ADDR_HW	Forms lower bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller.

11.3.5.1.156 iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_INBOUND_1)

11.3.5.1.156.1 Offset

Register	Offset
IATU_LWR_TARGET_ADDR_OFF_INBOUND_1	8000_0314h

11.3.5.1.156.2 Diagram



11.3.5.1.156.3 Fields

Field	Function
31-16 LWR_TARGET_RW	Forms MSB's of the Lower Target part of the new address of the translated region. These bits are always '0'. - Field size depends on log2(Minimum Size of iATU Region) in address match mode. - Field size depends on log2(BAR_MASK+1) in BAR match mode. Note: This register field is sticky.

Table continues on the next page...

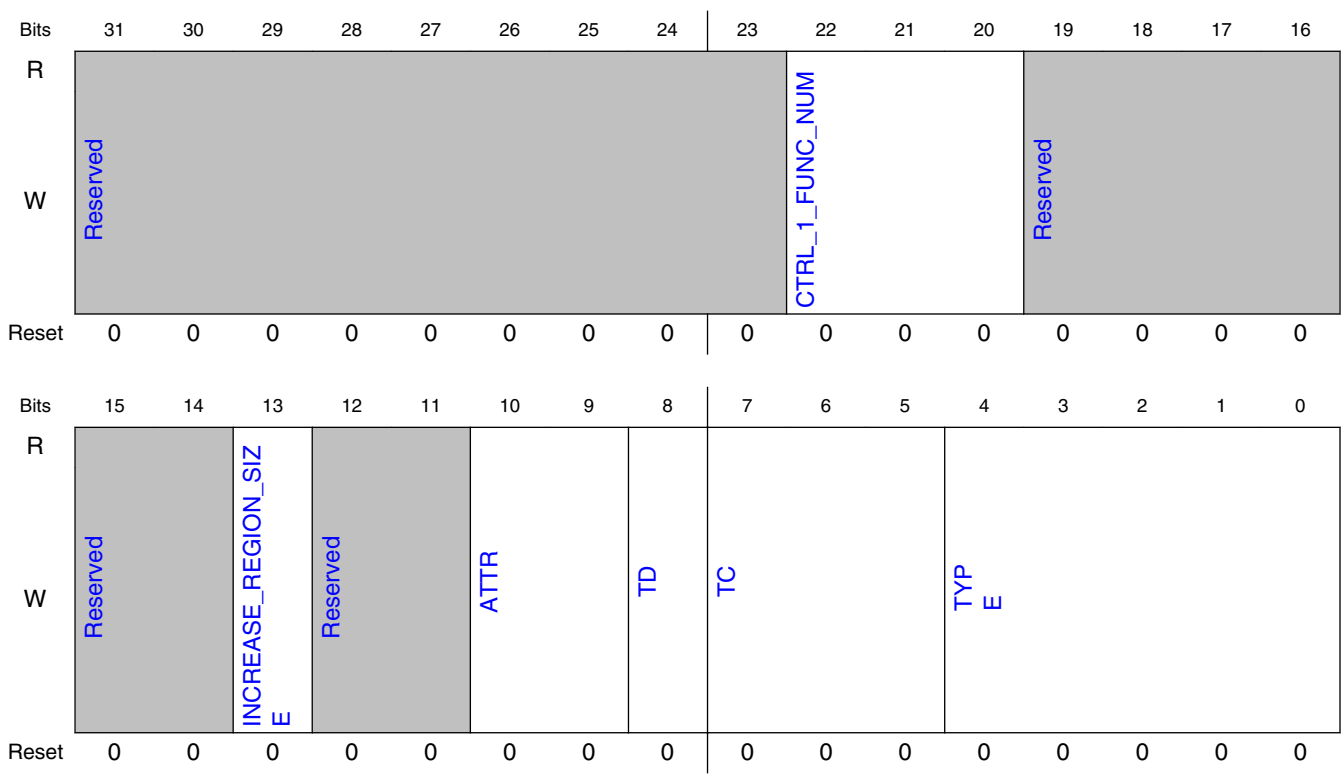
Field	Function
15-0 LWR_TARGET_HW	Forms the LSB's of the Lower Target part of the new address of the translated region. The start address must be aligned to the Minimum Size of iATU Region kB boundary (in address match mode); and to the Bar size boundary (in BAR match mode) so that these bits are always '0'. If the BAR is smaller than the iATU region size, then the iATU target address must align to the iATU region size; otherwise it must align to the BAR size. A write to this location is ignored by the PCIe controller. - Field size depends on log2(Minimum Size of iATU Region) in address match mode. - Field size depends on log2(BAR_MASK +1) in BAR match mode.

11.3.5.1.157 iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_OUTBOUND_2)

11.3.5.1.157.1 Offset

Register	Offset
IATU_REGION_CTRL_1_OFF_OUTBOUND_2	8000_0400h

11.3.5.1.157.2 Diagram



11.3.5.1.157.3 Fields

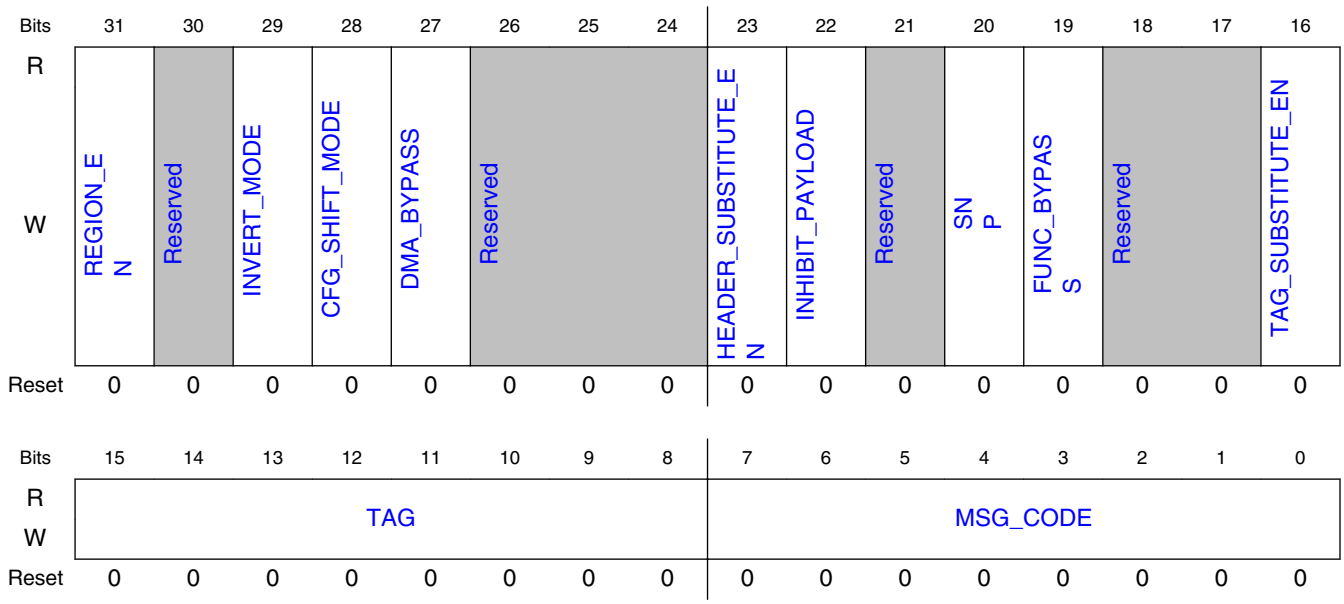
Field	Function
31-23 —	Reserved.
22-20 CTRL_1_FUNC_NUM	Function Number. - When the address of an outbound TLP is matched to this region and the FUNC_BYPASS field in the "iATU Control 2 Register" is '0', then the function number used in generating the function part of the requester ID (RID) field of the TLP is taken from this 5-bit register. When you are using the AXI Bridge, then this field is swapped before AXI decomposition occurs so that the correct "Max_Read_Request_Size" and "Max_Payload_Size" values are used. Note: This register field is sticky.
19-14 —	Reserved.
13 INCREASE_REGION_SIZE	Increase the maximum ATU Region size. When set, the size is determined by the Maximum Size of iATU Region. When clear, the maximum ATU Region size is 4 GB (default). Note: This register field is sticky.
12-11 —	Reserved.
10-9 ATTR	When the address of an outbound TLP is matched to this region, then the ATTR field of the TLP is changed to the value in this register. Note: This register field is sticky.
8 TD	When the address of an outbound TLP is matched to this region, then the TD field of the TLP is changed to the value in this register. Note: This register field is sticky.
7-5 TC	When the address of an outbound TLP is matched to this region, then the TC field of the TLP is changed to the value in this register. Note: This register field is sticky.
4-0 TYPE	When the address of an outbound TLP is matched to this region, then the TYPE field of the TLP is changed to the value in this register. Note: This register field is sticky.

11.3.5.1.158 iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_OUTBOUND_2)

11.3.5.1.158.1 Offset

Register	Offset
IATU_REGION_CTRL_2_OFF_OUTBOUND_2	8000_0404h

11.3.5.1.158.2 Diagram



11.3.5.1.158.3 Fields

Field	Function
31 REGION_EN	Region Enable. This bit must be set to '1' for address translation to take place. Note: This register field is sticky.
30 —	Reserved.
29 INVERT_MODE	Invert Mode. When set the address matching region is inverted. Therefore, an address match occurs when the untranslated address is in the region outside the defined range (Base Address to Limit Address). Note: This register field is sticky.
28 CFG_SHIFT_MODE	CFG Shift Mode. The iATU uses bits [27:12] of the untranslated address (on the XALI0/1/2 interface or AXI slave interface address) to form the BDF number of the outgoing CFG TLP. This supports the Enhanced Configuration Address Mapping (ECAM) mechanism (Section 7.2.2 of the PCI Express Base 3.1 Specification, revision 1.0) by allowing all outgoing I/O and MEM TLPs (that have been translated to CFG) to be mapped from memory space into any 256 MB region of the PCIe configuration space. Note: This register field is sticky.
27 DMA_BYPASS	DMA Bypass Mode. Allows request TLPs which are initiated by the DMA controller to pass through the iATU untranslated. Note: This field is reserved for the SW product. You must set it to '0'. Note: This register field is sticky.
26-24 —	Reserved.
23 HEADER_SUBSTITUTE_EN	Header Substitute Enable. When enabled and region address is matched, the iATU fully substitutes bytes 8-11 (for 3 DWORD header) or bytes 12-15 (for 4 DWORD header) of the outbound TLP header with the contents of the LWR_TARGET_RW field in IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i. - 1: LWR_TARGET_RW in the iATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register is used to fill bytes 8-to-11 (for 3 DWORD header) or bytes 12-to-15 (for 4 DWORD header) of the translated TLP header. - 0: LWR_TARGET_RW in the iATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register forms the new address of the translated region. Note: This register field is sticky.

Table continues on the next page...

Field	Function
22 INHIBIT_PAYLOAD	Inhibit TLP Payload Data for TLP's in Matched Region; assign iATU region to be TLP without data. When enabled and region address is matched, the iATU marks all TLPs as having no payload data by forcing the TLP header Fmt[1] bit =0, regardless of the application inputs such as slv_wstrb. - 1: Fmt[1] =0 so that only TLP type without data is sent. For example, a Msg instead of MsgD will be sent. - 0: Fmt[1] =0/1 so that TLPs with or without data can be sent. Note: This register field is sticky.
21 —	Reserved.
20 SNP	Serialize Non-Posted Requests. In this mode, when the AXI Bridge is populated, same AXI ID Non-Posted Read/Write Requests are transmitted on the wire if there are no other same ID Non-Posted Requests outstanding. Note: This register field is sticky.
19 FUNC_BYPASS	Function Number Translation Bypass. In this mode, the function number of the translated TLP is taken from your application transmit interface and not from the CTRL_1_FUNC_NUM field of the "iATU Control 1 Register" or the VF_NUMBER field of the "iATU Control 3 Register." Note: This register field is sticky.
18-17 —	Reserved.
16 TAG_SUBSTITUTE_EN	TAG Substitute Enable. When enabled and region address is matched, the iATU substitutes the TAG field of the outbound TLP header with the contents of the TAG field in this register. The expected usage scenario is translation from AXI MWr to Vendor Defined Msg/MsgD. Note: This register field is sticky.
15-8 TAG	TAG. The substituted TAG field (byte 6) in the outgoing TLP header when TAG_SUBSTITUTE_EN is set. Note: This register field is sticky.
7-0 MSG_CODE	MSG TLPs (Message Code). When the address of an outbound TLP is matched to this region, and the translated TLP TYPE field is Msg or MsgD; then the message field of the TLP is changed to the value in this register. Memory TLPs: (ST;Steering Tag). When the ST field of an outbound TLP is matched to this region, and the translated TLP TYPE field targets memory space; then the ST field of the TLP is changed to the value in this register. Note: This register field is sticky.

11.3.5.1.159 iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_OUTBOUND_2)

11.3.5.1.159.1 Offset

Register	Offset
IATU_LWR_BASE_ADDR_OFF_OUTBOUND_2	8000_0408h

11.3.5.1.159.2 Function

iATU Lower Base Address Register. The Minimum Size of iATU Region (Value Range: 4 kB, 8 kB, 16 kB, 32 kB, 64 kB defaults to 64 kB) specifies the minimum size of an address translation region. For example, if set to 64 kB; the lower 16 bits of the Base, Limit and Target registers are zero and all address regions are aligned on 64 kB boundaries. More precisely, the lower log2(Minimum Size of iATU Region) bits are zero.

11.3.5.1.159.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LWR_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LWR_BASE_HW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.159.4 Fields

Field	Function
31-16 LWR_BASE_RW	Forms bits [31:n] of the start address of the address region to be translated. n is log2(Minimum Size of iATU Region) Note: This register field is sticky.
15-0 LWR_BASE_HW	Forms bits [n-1:0] of the start address of the address region to be translated. The start address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. n is log2(Minimum Size of iATU Region)

11.3.5.1.160 iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_2)

11.3.5.1.160.1 Offset

Register	Offset
IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_2	8000_040Ch

11.3.5.1.160.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.160.3 Fields

Field	Function
31-0 UPPER_BASE_RW	Forms bits [63:32] of the start (and end) address of the address region to be translated. In systems with a 32-bit address space, this register is not used and therefore writing to this register has no effect. Note: This register field is sticky.

11.3.5.1.161 iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_OUTBOUND_2)

11.3.5.1.161.1 Offset

Register	Offset
IATU_LIMIT_ADDR_OFF_OUTBOUND_2	8000_0410h

11.3.5.1.161.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LIMIT_ADDR_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LIMIT_ADDR_HW															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

11.3.5.1.161.3 Fields

Field	Function
31-16 LIMIT_ADDR_RW	Forms upper bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. Note: This register field is sticky.
15-0 LIMIT_ADDR_HW	Forms lower bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller.

11.3.5.1.162 iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_2)

11.3.5.1.162.1 Offset

Register	Offset
IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_2	8000_0414h

11.3.5.1.162.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LWR_TARGET_RW_OUTBOUND															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LWR_TARGET_RW_OUTBOUND															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.162.3 Fields

Field	Function
31-0 LWR_TARGET_RW_OUTBOUND	When HEADER_SUBSTITUTE_EN in IATU_REGION_CTRL_2_OFF_OUTBOUND_ is '0' (normal operation): - LWR_TARGET_RW[31:n] forms MSB's of the Lower Target part of the new address of the translated region; - LWR_TARGET_RW[n-1:0] are not used. (The start address must be aligned to the Minimum Size of iATU Region kB boundary, so the lower bits of the start address of the new address of the translated region (bits n-1:0) are always '0'). - n is log2(Minimum Size of iATU Region). When

Field	Function
	HEADER_SUBSTITUTE_EN in IATU_REGION_CTRL_2_OFF_OUTBOUND_i is '1': - LWR_TARGET_RW[31:0] forms bytes 8-11 (for 3 dword header) or bytes 12-15 (for 4 dword header) of the outbound TLP header. Usage scenarios include the transmission of Vendor Defined Messages where the controller determines the content of bytes 12 to 15 of the TLP header. Note: This register field is sticky.

11.3.5.1.163 iATU Upper Target Address Register. (IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_2)

11.3.5.1.163.1 Offset

Register	Offset
IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_2	8000_0418h

11.3.5.1.163.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UPPER_TARGET_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	UPPER_TARGET_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.163.3 Fields

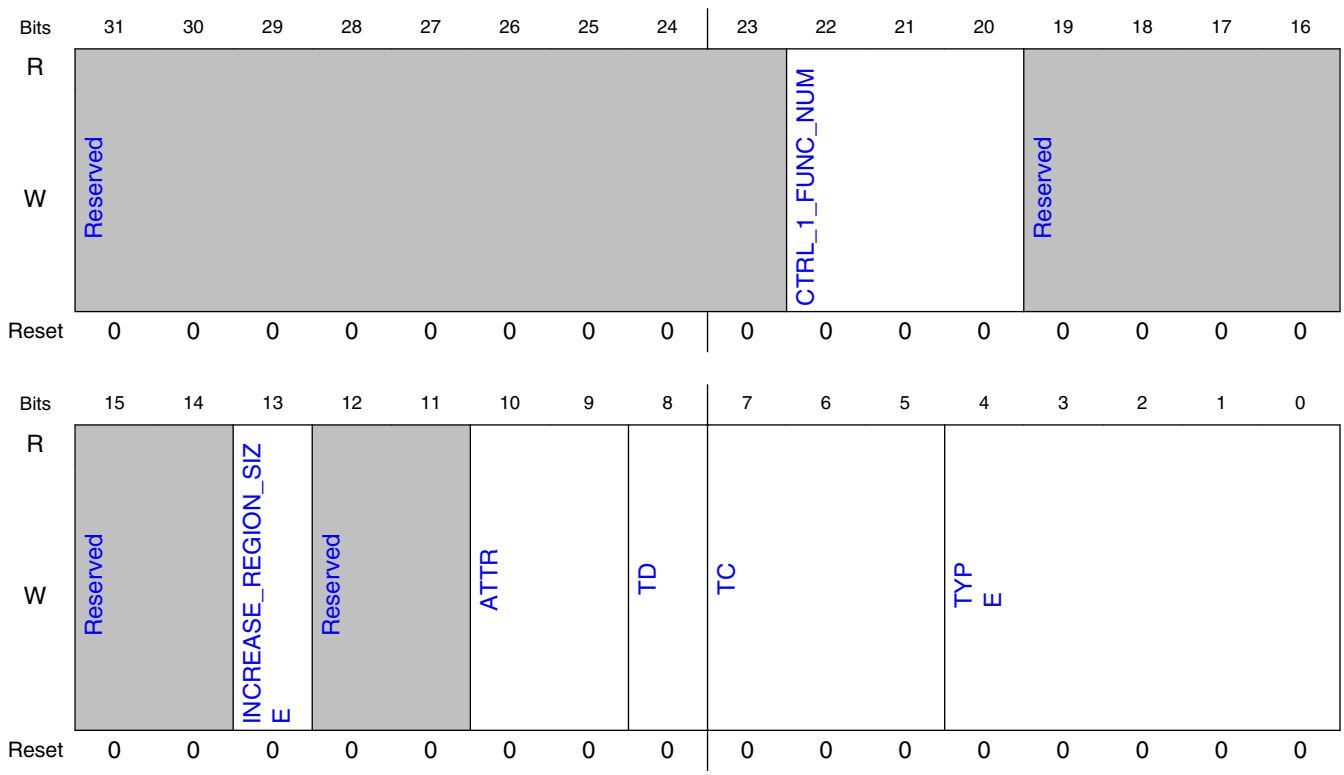
Field	Function
31-0 UPPER_TARGET_RW	Forms bits [63:32] of the start address (Upper Target part) of the new address of the translated region. Note: This register field is sticky.

11.3.5.1.164 iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_INBOUND_2)

11.3.5.1.164.1 Offset

Register	Offset
IATU_REGION_CTRL_1_OFF_INBOUND_2	8000_0500h

11.3.5.1.164.2 Diagram



11.3.5.1.164.3 Fields

Field	Function
31-23 —	Reserved.
22-20 CTRL_1_FUNC_NUM	Function Number. - MEM-I/O: When the Address and BAR matching logic in the controller indicate that a MEM-I/O transaction matches a BAR in the function corresponding to this value, then address translation proceeds. This check is only performed if the "Function Number Match Enable" bit of the "iATU Control 2 Register" is set. - CFG0/CFG1: When the destination function number as specified in the routing ID of the TLP header matches the function, then address translation proceeds. This check is only performed if the

Table continues on the next page...

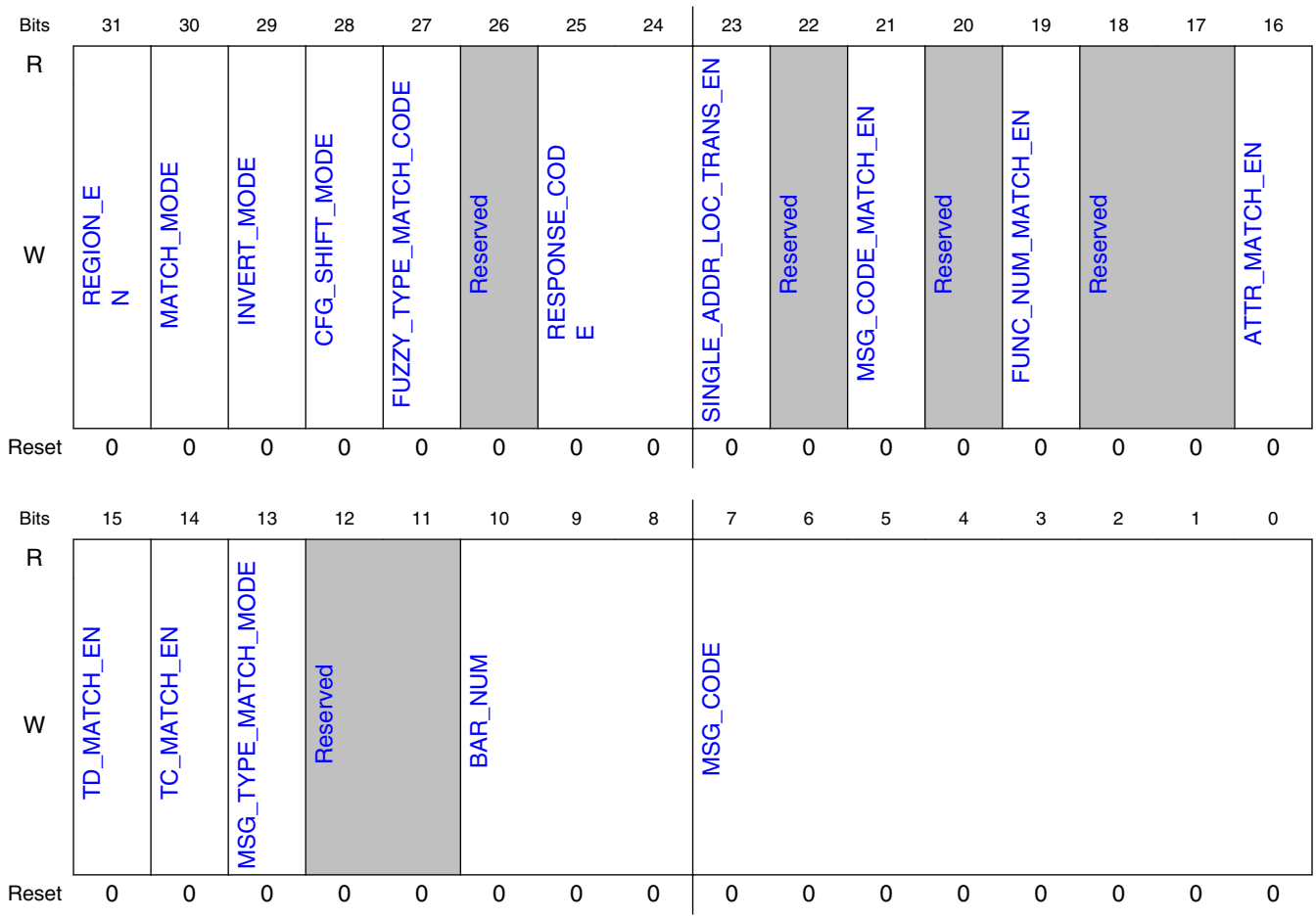
Field	Function
	"Function Number Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
19-14 —	Reserved.
13 INCREASE_REGION_SIZE	Increase the maximum ATU Region size. When set, the size is determined by the Maximum Size of iATU Region. When clear, the maximum ATU Region size is 4 GB (default). Note: This register field is sticky.
12-11 —	Reserved.
10-9 ATTR	When the ATTR field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "ATTR Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
8 TD	When the TD field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "TD Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
7-5 TC	When the TC field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "TC Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
4-0 TYPE	When the TYPE field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). Note: This register field is sticky.

11.3.5.1.165 iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_INBOUND_2)

11.3.5.1.165.1 Offset

Register	Offset
IATU_REGION_CTRL_2_OFF_INBOUND_2	8000_0504h

11.3.5.1.165.2 Diagram



11.3.5.1.165.3 Fields

Field	Function
31 REGION_EN	Region Enable. This bit must be set to '1' for address translation to take place. Note: This register field is sticky.
30 MATCH_MODE	Match Mode. Determines Inbound matching mode for TLPs. The mode depends on the type of TLP that is received as follows: For MEM-I/O TLPs, this field is interpreted as follows: - 0: Address Match Mode. The iATU operates using addresses as in the outbound direction. The Region Base and Limit Registers must be setup. - 1:BAR Match Mode. BAR matching is used. The "BAR Number" field is relevant. Not used for RC. For CFG0 TLPs, this field is interpreted as follows: - 0: Routing ID Match Mode. The iATU interprets the Routing ID (Bytes 8 to 11 of TLP header) as an address. This corresponds to the upper 16 bits of the address in MEM-I/O transactions. The Routing ID of the TLP must be within the base and limit of the iATU region for matching to proceed. - 1: Accept Mode. The iATU accepts all CFG0 transactions as address matches. The routing ID in the CFG0 TLP is ignored. This is useful as all received CFG0 TLPs should be processed regardless of the Bus number. For MSG/MSGD TLPs, this field is interpreted as follows: - 0: Address Match Mode. The iATU treats the third dword and fourth dword of the inbound MSG/MSGD TLP as an address and it is matched against the Region Base and Limit Registers. - 1: Vendor ID Match Mode. This mode is relevant for ID-routed Vendor Defined Messages. The iATU ignores the Routing ID (Bus, Device, Function) in bits [31:16] of the third dword of the TLP header, but matches

Table continues on the next page...

Field	Function
	against the Vendor ID in bits [15:0] of the third dword of the TLP header. Bits [15:0] of the Region Upper Base register should be programmed with the required Vendor ID. The lower Base and Limit Register should be programmed to translate TLPs based on vendor specific information in the fourth dword of the TLP header. - If SINGLE_ADDRESS_LOCATION_TRANSLATE_EN = 1 AND MSG_TYPE_MATCH_MODE = 1, then Match Mode is ignored. Note: This register field is sticky.
29 INVERT_MODE	Invert Mode. When set the address matching region is inverted. Therefore, an address match occurs when the untranslated address is in the region outside the defined range (Base Address to Limit Address). Note: This register field is sticky.
28 CFG_SHIFT_MODE	CFG Shift Mode. This is useful for CFG transactions where the PCIe configuration mechanism maps bits [27:12] of the address to the bus/device and function number. This allows a CFG configuration space to be located in any 256MB window of your application memory space using a 28-bit effective address. Shifts bits [31:16] of the untranslated address to form bits [27:12] of the translated address. Note: This register field is sticky.
27 FUZZY_TYPE_MATCH_CODE	Fuzzy Type Match Mode. When enabled, the iATU relaxes the matching of the TLP TYPE field against the expected TYPE field so that - CfgRd0 and CfgRd1 TLPs are seen as identical. Similarly with CfgWr0 and CfgWr1. - MWr, MRd and MRdLk TLPs are seen as identical - The Routing field of Msg/MsgD TLPs is ignored - FetchAdd, Swap and CAS are seen as identical. For example, CFG0 in the TYPE field in the "iATU Control 1 Register" matches against an inbound CfgRd0, CfgRd1, CfgWr0 or CfgWr1 TLP. Note: This register field is sticky.
26 —	Reserved.
25-24 RESPONSE_CODE	Response Code. Defines the type of response to give for accesses matching this region. This overrides the normal RADM filter response. Note that this feature is not available for any region where Single Address Location Translate is enabled. - 00 - Normal RADM filter response is used. - 01 - Unsupported request (UR) - 10 - Completer abort (CA) - 11 - Not used / undefined / reserved. Note: This register field is sticky.
23 SINGLE_ADDR_LOC_TRANS_EN	Single Address Location Translate Enable. When enabled, Rx TLPs can be translated to a single address location as determined by the target address register of the iATU region. The main usage scenario is translation of Messages (such as Vendor Defined or ATS Messages) to MWr TLPs when the AXI bridge is enabled. Note: This register field is sticky.
22 —	Reserved.
21 MSG_CODE_MATCH_EN	Message Code Match Enable (Msg TLPS). Ensures that a successful message Code TLP field comparison match (see Message Code field of the "iATU Control 2 Register") occurs (in MSG transactions) for address translation to proceed. ST Match Enable (Mem TLPs). Ensures that a successful ST TLP field comparison match (see ST field of the "iATU Control 2 Register") occurs (in MEM transactions) for address translation to proceed. Note: This register field is sticky.
20 —	Reserved.
19 FUNC_NUM_MATCH_EN	Function Number Match Enable. Ensures that a successful Function Number TLP field comparison match (see Function Number field of the "iATU Control 1 Register") occurs (in MEM-I/O and CFG0/CFG1 transactions) for address translation to proceed. Note: This register field is sticky.
18-17 —	Reserved.
16 ATTR_MATCH_EN	ATTR Match Enable. Ensures that a successful ATTR TLP field comparison match (see ATTR field of the "iATU Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky.

Table continues on the next page...

Field	Function
15 TD_MATCH_EN	TD Match Enable. Ensures that a successful TD TLP field comparison match (see TD field of the "iATU Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky.
14 TC_MATCH_EN	TC Match Enable. Ensures that a successful TC TLP field comparison match (see TC field of the "iATU Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky.
13 MSG_TYPE_MATCH_MODE	Message Type Match Mode. When enabled, and if single address location translate enable is set, then inbound TLPs of type MSG/MSGd which match the type field of the iatu_region_ctrl_1_OFF_inbound register (=>TYPE[4:3]=2'b10) will be translated. Message type match mode overrides any value of MATCH_MODE field in this register. Usage scenarios for this are translation of VDM or ATS messages when AXI bridge is configured on client interface. Note: This register field is sticky.
12-11 —	Reserved.
10-8 BAR_NUM	BAR Number. When the BAR number of an inbound MEM or IO TLP " that is matched by the normal internal BAR address matching mechanism " is the same as this field, address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "Match Mode" bit of the "iATU Control 2 Register" is set. - 000b - BAR0 - 001b - BAR1 - 010b - BAR2 - 011b - BAR3 - 100b - BAR4 - 101b - BAR5 - 110b - ROM - 111b - reserved - IO translation would require either 00100b or 00101b in the inbound TLP TYPE; the BAR Number set in the range 000b - 101b and that BAR configured as an IO BAR. Note: This register field is sticky.
7-0 MSG_CODE	MSG TLPs: (Message Code). When the TYPE field of an inbound Msg/MsgD TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "Message Code Match Enable" bit of the "iATU Control 2 Register" is set. Memory TLPs: (ST;Steering Tag). When the ST field of an inbound TLP is matched to this value, then address translation proceeds. This check is only performed if the "ST Match Enable" bit of the "iATU Control2 Register" is set. The setting is independent of the setting of the TH field. Note: This register field is sticky.

11.3.5.1.166 iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_INBOUND_2)

11.3.5.1.166.1 Offset

Register	Offset
IATU_LWR_BASE_ADDR_OFF_INBOUND_2	8000_0508h

11.3.5.1.166.2 Function

iATU Lower Base Address Register. The Minimum Size of iATU Region (Value Range: 4 kB, 8 kB, 16 kB, 32 kB, 64 kB defaults to 64 kB) specifies the minimum size of an address translation region. For example, if set to 64 kB; the lower 16 bits of the Base, Limit and Target registers are zero and all address regions are aligned on 64 kB boundaries. More precisely, the lower log2(Minimum Size of iATU Region) bits are zero.

11.3.5.1.166.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LWR_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LWR_BASE_HW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.166.4 Fields

Field	Function
31-16 LWR_BASE_RW	Forms bits [31:n] of the start address of the address region to be translated. n is log2(Minimum Size of iATU Region) Note: This register field is sticky.
15-0 LWR_BASE_HW	Forms bits [n-1:0] of the start address of the address region to be translated. The start address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. n is log2(Minimum Size of iATU Region)

11.3.5.1.167 iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_INBOUND_2)

11.3.5.1.167.1 Offset

Register	Offset
IATU_UPPER_BASE_ADDR_OFF_INBOUND_2	8000_050Ch

11.3.5.1.167.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.167.3 Fields

Field	Function
31-0 UPPER_BASE_RW	Forms bits [63:32] of the start (and end) address of the address region to be translated. Note: This register field is sticky.

11.3.5.1.168 iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_INBOUND_2)

11.3.5.1.168.1 Offset

Register	Offset
IATU_LIMIT_ADDR_OFF_INBOUND_2	8000_0510h

11.3.5.1.168.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LIMIT_ADDR_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LIMIT_ADDR_HW															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

11.3.5.1.168.3 Fields

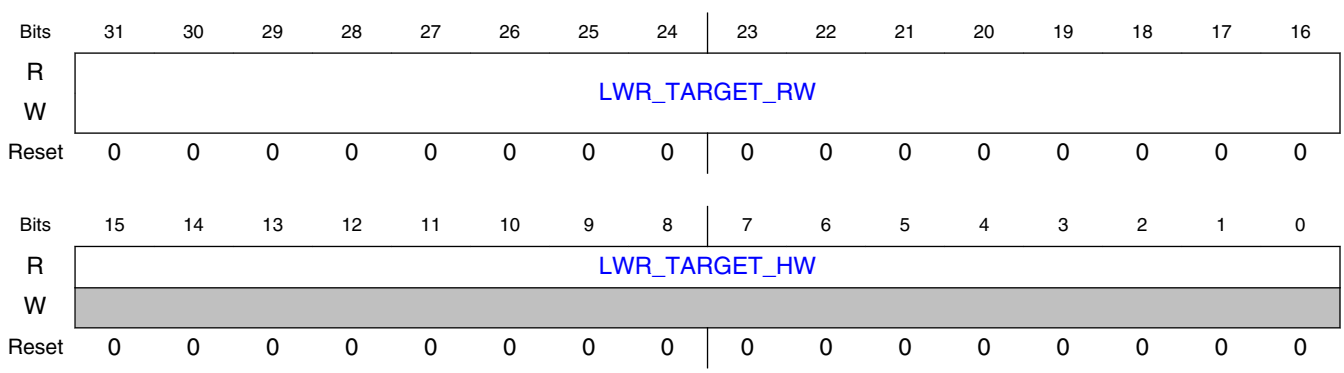
Field	Function
31-16 LIMIT_ADDR_RW	Forms upper bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. Note: This register field is sticky.
15-0 LIMIT_ADDR_HW	Forms lower bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller.

11.3.5.1.169 iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_INBOUND_2)

11.3.5.1.169.1 Offset

Register	Offset
IATU_LWR_TARGET_ADDR_OFF_INBOUND_2	8000_0514h

11.3.5.1.169.2 Diagram



11.3.5.1.169.3 Fields

Field	Function
31-16 LWR_TARGET_RW	Forms MSB's of the Lower Target part of the new address of the translated region. These bits are always '0'. - Field size depends on log2(Minimum Size of iATU Region) in address match mode. - Field size depends on log2(BAR_MASK+1) in BAR match mode. Note: This register field is sticky.

Table continues on the next page...

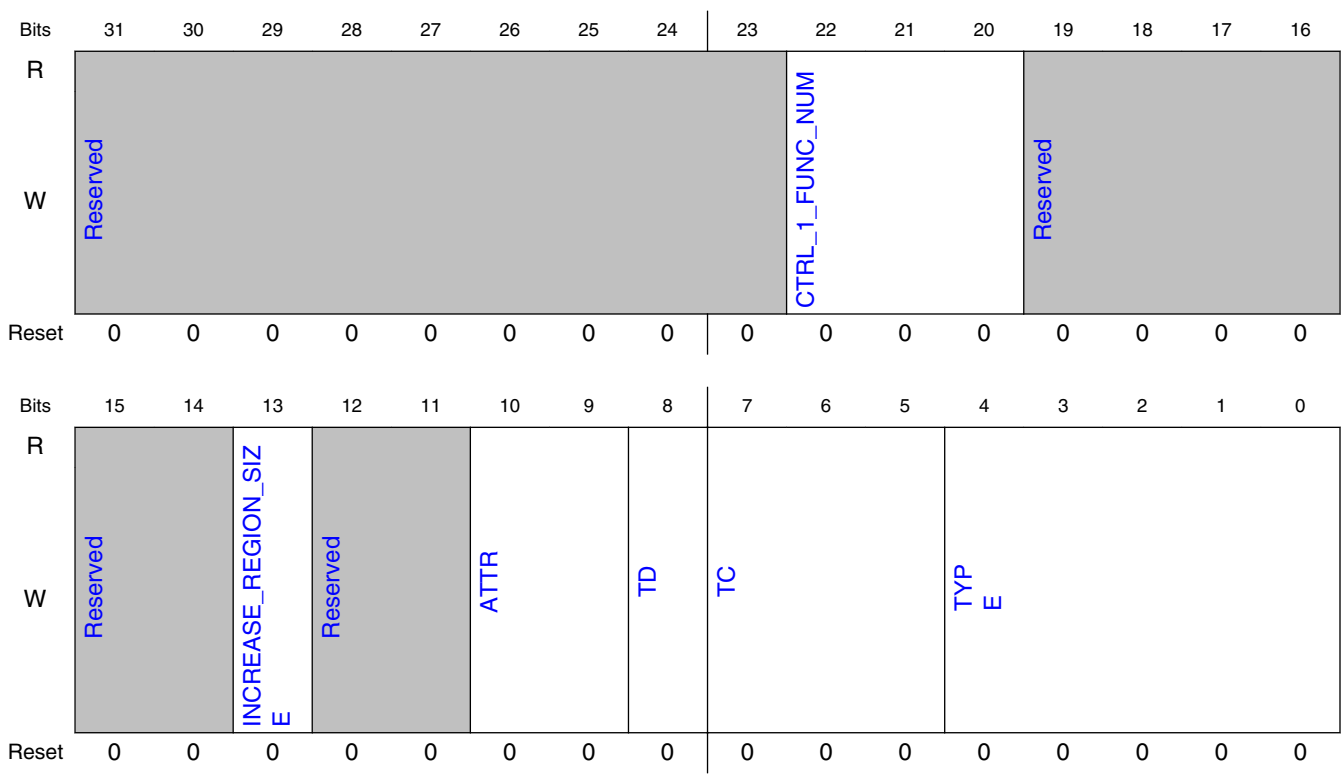
Field	Function
15-0 LWR_TARGET_HW	Forms the LSB's of the Lower Target part of the new address of the translated region. The start address must be aligned to the Minimum Size of iATU Region kB boundary (in address match mode); and to the Bar size boundary (in BAR match mode) so that these bits are always '0'. If the BAR is smaller than the iATU region size, then the iATU target address must align to the iATU region size; otherwise it must align to the BAR size. A write to this location is ignored by the PCIe controller. - Field size depends on log2(Minimum Size of iATU Region) in address match mode. - Field size depends on log2(BAR_MASK +1) in BAR match mode.

11.3.5.1.170 iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_OUTBOUND_3)

11.3.5.1.170.1 Offset

Register	Offset
IATU_REGION_CTRL_1_OFF_OUTBOUND_3	8000_0600h

11.3.5.1.170.2 Diagram



11.3.5.1.170.3 Fields

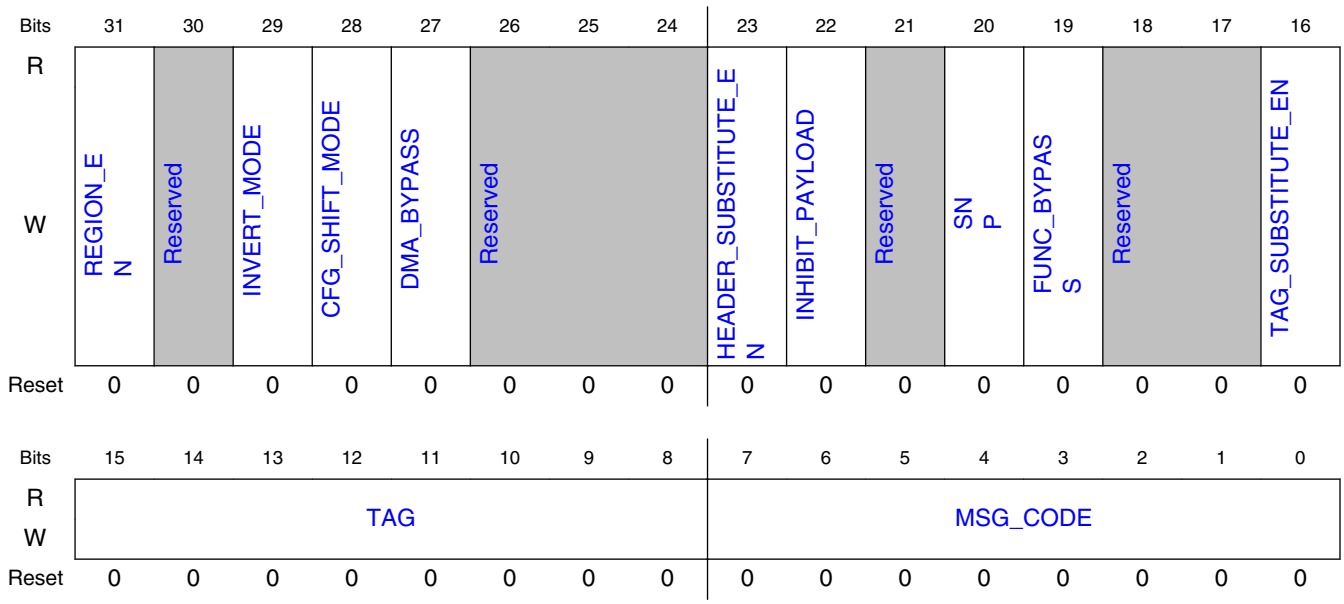
Field	Function
31-23 —	Reserved.
22-20 CTRL_1_FUNC_NUM	Function Number. - When the address of an outbound TLP is matched to this region and the FUNC_BYPASS field in the "iATU Control 2 Register" is '0', then the function number used in generating the function part of the requester ID (RID) field of the TLP is taken from this 5-bit register. When you are using the AXI Bridge, then this field is swapped before AXI decomposition occurs so that the correct "Max_Read_Request_Size" and "Max_Payload_Size" values are used. Note: This register field is sticky.
19-14 —	Reserved.
13 INCREASE_REGION_SIZE	Increase the maximum ATU Region size. When set, the size is determined by the Maximum Size of iATU Region. When clear, the maximum ATU Region size is 4 GB (default). Note: This register field is sticky.
12-11 —	Reserved.
10-9 ATTR	When the address of an outbound TLP is matched to this region, then the ATTR field of the TLP is changed to the value in this register. Note: This register field is sticky.
8 TD	When the address of an outbound TLP is matched to this region, then the TD field of the TLP is changed to the value in this register. Note: This register field is sticky.
7-5 TC	When the address of an outbound TLP is matched to this region, then the TC field of the TLP is changed to the value in this register. Note: This register field is sticky.
4-0 TYPE	When the address of an outbound TLP is matched to this region, then the TYPE field of the TLP is changed to the value in this register. Note: This register field is sticky.

11.3.5.1.171 iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_OUTBOUND_3)

11.3.5.1.171.1 Offset

Register	Offset
IATU_REGION_CTRL_2_OFF_OUTBOUND_3	8000_0604h

11.3.5.1.171.2 Diagram



11.3.5.1.171.3 Fields

Field	Function
31 REGION_EN	Region Enable. This bit must be set to '1' for address translation to take place. Note: This register field is sticky.
30 —	Reserved.
29 INVERT_MODE	Invert Mode. When set the address matching region is inverted. Therefore, an address match occurs when the untranslated address is in the region outside the defined range (Base Address to Limit Address). Note: This register field is sticky.
28 CFG_SHIFT_MODE	CFG Shift Mode. The iATU uses bits [27:12] of the untranslated address (on the XALI0/1/2 interface or AXI slave interface address) to form the BDF number of the outgoing CFG TLP. This supports the Enhanced Configuration Address Mapping (ECAM) mechanism (Section 7.2.2 of the PCI Express Base 3.1 Specification, revision 1.0) by allowing all outgoing I/O and MEM TLPs (that have been translated to CFG) to be mapped from memory space into any 256 MB region of the PCIe configuration space. Note: This register field is sticky.
27 DMA_BYPASS	DMA Bypass Mode. Allows request TLPs which are initiated by the DMA controller to pass through the iATU untranslated. Note: This field is reserved for the SW product. You must set it to '0'. Note: This register field is sticky.
26-24 —	Reserved.
23 HEADER_SUBSTITUTE_EN	Header Substitute Enable. When enabled and region address is matched, the iATU fully substitutes bytes 8-11 (for 3 DWORD header) or bytes 12-15 (for 4 DWORD header) of the outbound TLP header with the contents of the LWR_TARGET_RW field in IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i. - 1: LWR_TARGET_RW in the iATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register is used to fill bytes 8-to-11 (for 3 DWORD header) or bytes 12-to-15 (for 4 DWORD header) of the translated TLP header. - 0: LWR_TARGET_RW in the iATU_LWR_TARGET_ADDR_OFF_OUTBOUND_i register forms the new address of the translated region. Note: This register field is sticky.

Table continues on the next page...

Field	Function
22 INHIBIT_PAYLOAD	Inhibit TLP Payload Data for TLP's in Matched Region; assign iATU region to be TLP without data. When enabled and region address is matched, the iATU marks all TLPs as having no payload data by forcing the TLP header Fmt[1] bit =0, regardless of the application inputs such as slv_wstrb. - 1: Fmt[1] =0 so that only TLP type without data is sent. For example, a Msg instead of MsgD will be sent. - 0: Fmt[1] =0/1 so that TLPs with or without data can be sent. Note: This register field is sticky.
21 —	Reserved.
20 SNP	Serialize Non-Posted Requests. In this mode, when the AXI Bridge is populated, same AXI ID Non-Posted Read/Write Requests are transmitted on the wire if there are no other same ID Non-Posted Requests outstanding. Note: This register field is sticky.
19 FUNC_BYPASS	Function Number Translation Bypass. In this mode, the function number of the translated TLP is taken from your application transmit interface and not from the CTRL_1_FUNC_NUM field of the "iATU Control 1 Register" or the VF_NUMBER field of the "iATU Control 3 Register." Note: This register field is sticky.
18-17 —	Reserved.
16 TAG_SUBSTITUTE_EN	TAG Substitute Enable. When enabled and region address is matched, the iATU substitutes the TAG field of the outbound TLP header with the contents of the TAG field in this register. The expected usage scenario is translation from AXI MWr to Vendor Defined Msg/MsgD. Note: This register field is sticky.
15-8 TAG	TAG. The substituted TAG field (byte 6) in the outgoing TLP header when TAG_SUBSTITUTE_EN is set. Note: This register field is sticky.
7-0 MSG_CODE	MSG TLPs (Message Code). When the address of an outbound TLP is matched to this region, and the translated TLP TYPE field is Msg or MsgD; then the message field of the TLP is changed to the value in this register. Memory TLPs: (ST;Steering Tag). When the ST field of an outbound TLP is matched to this region, and the translated TLP TYPE field targets memory space; then the ST field of the TLP is changed to the value in this register. Note: This register field is sticky.

11.3.5.1.172 iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_OUTBOUND_3)

11.3.5.1.172.1 Offset

Register	Offset
IATU_LWR_BASE_ADDR_OFF_OUTBOUND_3	8000_0608h

11.3.5.1.172.2 Function

iATU Lower Base Address Register. The Minimum Size of iATU Region (Value Range: 4 kB, 8 kB, 16 kB, 32 kB, 64 kB defaults to 64 kB) specifies the minimum size of an address translation region. For example, if set to 64 kB; the lower 16 bits of the Base, Limit and Target registers are zero and all address regions are aligned on 64 kB boundaries. More precisely, the lower log2(Minimum Size of iATU Region) bits are zero.

11.3.5.1.172.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LWR_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LWR_BASE_HW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.172.4 Fields

Field	Function
31-16 LWR_BASE_RW	Forms bits [31:n] of the start address of the address region to be translated. n is log2(Minimum Size of iATU Region) Note: This register field is sticky.
15-0 LWR_BASE_HW	Forms bits [n-1:0] of the start address of the address region to be translated. The start address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. n is log2(Minimum Size of iATU Region)

11.3.5.1.173 iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_3)

11.3.5.1.173.1 Offset

Register	Offset
IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_3	8000_060Ch

11.3.5.1.173.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.173.3 Fields

Field	Function
31-0 UPPER_BASE_RW	Forms bits [63:32] of the start (and end) address of the address region to be translated. In systems with a 32-bit address space, this register is not used and therefore writing to this register has no effect. Note: This register field is sticky.

11.3.5.1.174 iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_OUTBOUND_3)

11.3.5.1.174.1 Offset

Register	Offset
IATU_LIMIT_ADDR_OFF_OUTBOUND_3	8000_0610h

11.3.5.1.174.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LIMIT_ADDR_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LIMIT_ADDR_HW															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

11.3.5.1.174.3 Fields

Field	Function
31-16 LIMIT_ADDR_RW	Forms upper bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. Note: This register field is sticky.
15-0 LIMIT_ADDR_HW	Forms lower bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller.

11.3.5.1.175 iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_3)

11.3.5.1.175.1 Offset

Register	Offset
IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_3	8000_0614h

11.3.5.1.175.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LWR_TARGET_RW_OUTBOUND															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LWR_TARGET_RW_OUTBOUND															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.175.3 Fields

Field	Function
31-0 LWR_TARGET_RW_OUTBOUND	When HEADER_SUBSTITUTE_EN in IATU_REGION_CTRL_2_OFF_OUTBOUND_ is '0' (normal operation): - LWR_TARGET_RW[31:n] forms MSB's of the Lower Target part of the new address of the translated region; - LWR_TARGET_RW[n-1:0] are not used. (The start address must be aligned to the Minimum Size of iATU Region kB boundary, so the lower bits of the start address of the new address of the translated region (bits n-1:0) are always '0'). - n is log2(Minimum Size of iATU Region). When

Field	Function
	HEADER_SUBSTITUTE_EN in IATU_REGION_CTRL_2_OFF_OUTBOUND_i is '1': - LWR_TARGET_RW[31:0] forms bytes 8-11 (for 3 dword header) or bytes 12-15 (for 4 dword header) of the outbound TLP header. Usage scenarios include the transmission of Vendor Defined Messages where the controller determines the content of bytes 12 to 15 of the TLP header. Note: This register field is sticky.

11.3.5.1.176 iATU Upper Target Address Register. (IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_3)

11.3.5.1.176.1 Offset

Register	Offset
IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_3	8000_0618h

11.3.5.1.176.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UPPER_TARGET_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	UPPER_TARGET_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.176.3 Fields

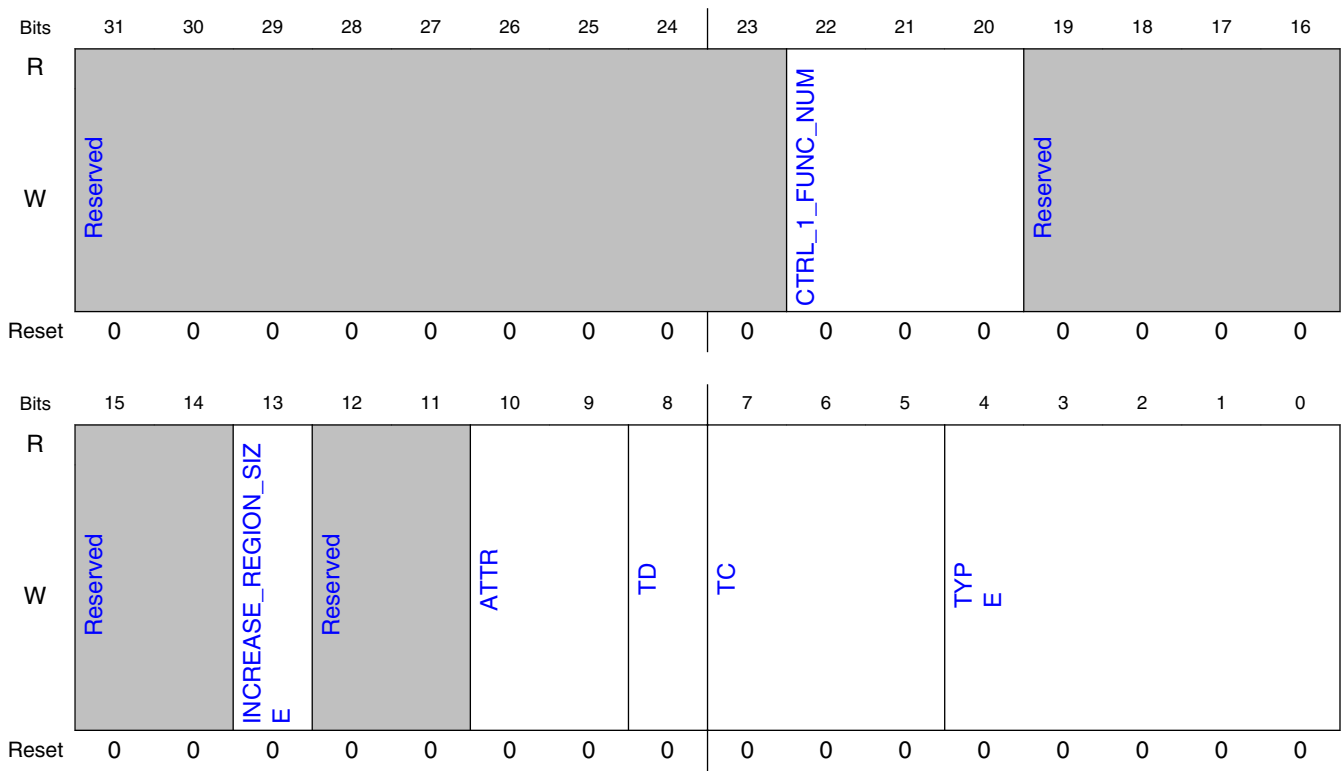
Field	Function
31-0 UPPER_TARGET_RW	Forms bits [63:32] of the start address (Upper Target part) of the new address of the translated region. Note: This register field is sticky.

11.3.5.1.177 iATU Region Control 1 Register. (IATU_REGION_CTRL_1_OFF_INBOUND_3)

11.3.5.1.177.1 Offset

Register	Offset
IATU_REGION_CTRL_1_OFF_INBOUND_3	8000_0700h

11.3.5.1.177.2 Diagram



11.3.5.1.177.3 Fields

Field	Function
31-23 —	Reserved.
22-20 CTRL_1_FUNC_NUM	Function Number. - MEM-I/O: When the Address and BAR matching logic in the controller indicate that a MEM-I/O transaction matches a BAR in the function corresponding to this value, then address translation proceeds. This check is only performed if the "Function Number Match Enable" bit of the "iATU Control 2 Register" is set. - CFG0/CFG1: When the destination function number as specified in the routing ID of the TLP header matches the function, then address translation proceeds. This check is only performed if the

Table continues on the next page...

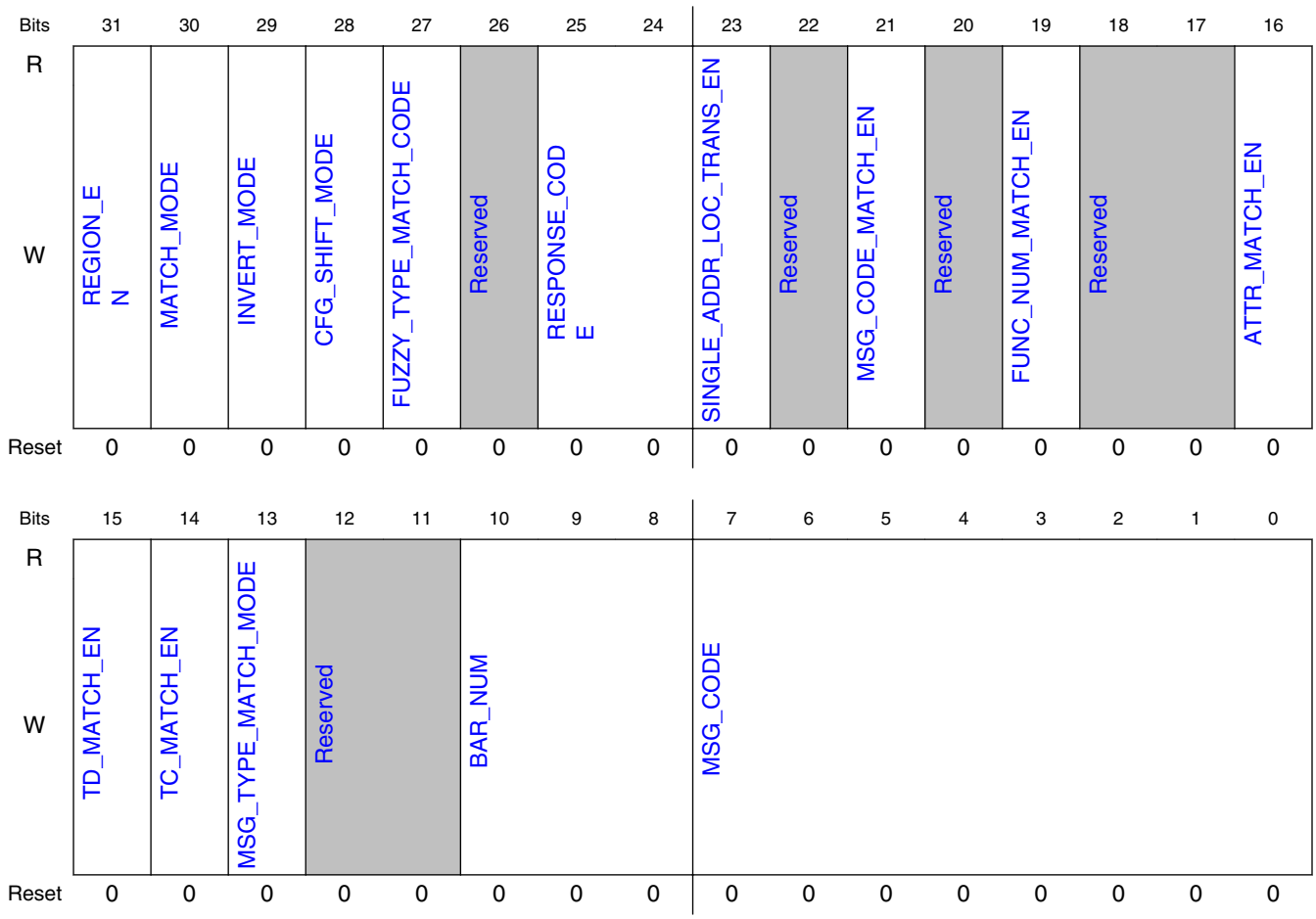
Field	Function
	"Function Number Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
19-14 —	Reserved.
13 INCREASE_REGION_SIZE	Increase the maximum ATU Region size. When set, the size is determined by the Maximum Size of iATU Region. When clear, the maximum ATU Region size is 4 GB (default). Note: This register field is sticky.
12-11 —	Reserved.
10-9 ATTR	When the ATTR field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "ATTR Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
8 TD	When the TD field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "TD Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
7-5 TC	When the TC field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "TC Match Enable" bit of the "iATU Control 2 Register" is set. Note: This register field is sticky.
4-0 TYPE	When the TYPE field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). Note: This register field is sticky.

11.3.5.1.178 iATU Region Control 2 Register. (IATU_REGION_CTRL_2_OFF_INBOUND_3)

11.3.5.1.178.1 Offset

Register	Offset
IATU_REGION_CTRL_2_OFF_INBOUND_3	8000_0704h

11.3.5.1.178.2 Diagram



11.3.5.1.178.3 Fields

Field	Function
31 REGION_EN	Region Enable. This bit must be set to '1' for address translation to take place. Note: This register field is sticky.
30 MATCH_MODE	Match Mode. Determines Inbound matching mode for TLPs. The mode depends on the type of TLP that is received as follows: For MEM-I/O TLPs, this field is interpreted as follows: - 0: Address Match Mode. The iATU operates using addresses as in the outbound direction. The Region Base and Limit Registers must be setup. - 1:BAR Match Mode. BAR matching is used. The "BAR Number" field is relevant. Not used for RC. For CFG0 TLPs, this field is interpreted as follows: - 0: Routing ID Match Mode. The iATU interprets the Routing ID (Bytes 8 to 11 of TLP header) as an address. This corresponds to the upper 16 bits of the address in MEM-I/O transactions. The Routing ID of the TLP must be within the base and limit of the iATU region for matching to proceed. - 1: Accept Mode. The iATU accepts all CFG0 transactions as address matches. The routing ID in the CFG0 TLP is ignored. This is useful as all received CFG0 TLPs should be processed regardless of the Bus number. For MSG/MSGD TLPs, this field is interpreted as follows: - 0: Address Match Mode. The iATU treats the third dword and fourth dword of the inbound MSG/MSGD TLP as an address and it is matched against the Region Base and Limit Registers. - 1: Vendor ID Match Mode. This mode is relevant for ID-routed Vendor Defined Messages. The iATU ignores the Routing ID (Bus, Device, Function) in bits [31:16] of the third dword of the TLP header, but matches

Table continues on the next page...

Field	Function
	against the Vendor ID in bits [15:0] of the third dword of the TLP header. Bits [15:0] of the Region Upper Base register should be programmed with the required Vendor ID. The lower Base and Limit Register should be programmed to translate TLPs based on vendor specific information in the fourth dword of the TLP header. - If SINGLE_ADDRESS_LOCATION_TRANSLATE_EN = 1 AND MSG_TYPE_MATCH_MODE = 1, then Match Mode is ignored. Note: This register field is sticky.
29 INVERT_MODE	Invert Mode. When set the address matching region is inverted. Therefore, an address match occurs when the untranslated address is in the region outside the defined range (Base Address to Limit Address). Note: This register field is sticky.
28 CFG_SHIFT_MODE	CFG Shift Mode. This is useful for CFG transactions where the PCIe configuration mechanism maps bits [27:12] of the address to the bus/device and function number. This allows a CFG configuration space to be located in any 256MB window of your application memory space using a 28-bit effective address. Shifts bits [31:16] of the untranslated address to form bits [27:12] of the translated address. Note: This register field is sticky.
27 FUZZY_TYPE_MATCH_CODE	Fuzzy Type Match Mode. When enabled, the iATU relaxes the matching of the TLP TYPE field against the expected TYPE field so that - CfgRd0 and CfgRd1 TLPs are seen as identical. Similarly with CfgWr0 and CfgWr1. - MWr, MRd and MRdLk TLPs are seen as identical - The Routing field of Msg/MsgD TLPs is ignored - FetchAdd, Swap and CAS are seen as identical. For example, CFG0 in the TYPE field in the "iATU Control 1 Register" matches against an inbound CfgRd0, CfgRd1, CfgWr0 or CfgWr1 TLP. Note: This register field is sticky.
26 —	Reserved.
25-24 RESPONSE_CODE	Response Code. Defines the type of response to give for accesses matching this region. This overrides the normal RADM filter response. Note that this feature is not available for any region where Single Address Location Translate is enabled. - 00 - Normal RADM filter response is used. - 01 - Unsupported request (UR) - 10 - Completer abort (CA) - 11 - Not used / undefined / reserved. Note: This register field is sticky.
23 SINGLE_ADDR_LOC_TRANS_EN	Single Address Location Translate Enable. When enabled, Rx TLPs can be translated to a single address location as determined by the target address register of the iATU region. The main usage scenario is translation of Messages (such as Vendor Defined or ATS Messages) to MWr TLPs when the AXI bridge is enabled. Note: This register field is sticky.
22 —	Reserved.
21 MSG_CODE_MATCH_EN	Message Code Match Enable (Msg TLPS). Ensures that a successful message Code TLP field comparison match (see Message Code field of the "iATU Control 2 Register") occurs (in MSG transactions) for address translation to proceed. ST Match Enable (Mem TLPs). Ensures that a successful ST TLP field comparison match (see ST field of the "iATU Control 2 Register") occurs (in MEM transactions) for address translation to proceed. Note: This register field is sticky.
20 —	Reserved.
19 FUNC_NUM_MATCH_EN	Function Number Match Enable. Ensures that a successful Function Number TLP field comparison match (see Function Number field of the "iATU Control 1 Register") occurs (in MEM-I/O and CFG0/CFG1 transactions) for address translation to proceed. Note: This register field is sticky.
18-17 —	Reserved.
16 ATTR_MATCH_EN	ATTR Match Enable. Ensures that a successful ATTR TLP field comparison match (see ATTR field of the "iATU Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky.

Table continues on the next page...

Field	Function
15 TD_MATCH_EN	TD Match Enable. Ensures that a successful TD TLP field comparison match (see TD field of the "iATU Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky.
14 TC_MATCH_EN	TC Match Enable. Ensures that a successful TC TLP field comparison match (see TC field of the "iATU Control 1 Register") occurs for address translation to proceed. Note: This register field is sticky.
13 MSG_TYPE_MATCH_MODE	Message Type Match Mode. When enabled, and if single address location translate enable is set, then inbound TLPs of type MSG/MSGd which match the type field of the iatu_region_ctrl_1_OFF_inbound register (=>TYPE[4:3]=2'b10) will be translated. Message type match mode overrides any value of MATCH_MODE field in this register. Usage scenarios for this are translation of VDM or ATS messages when AXI bridge is configured on client interface. Note: This register field is sticky.
12-11 —	Reserved.
10-8 BAR_NUM	BAR Number. When the BAR number of an inbound MEM or IO TLP " that is matched by the normal internal BAR address matching mechanism " is the same as this field, address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "Match Mode" bit of the "iATU Control 2 Register" is set. - 000b - BAR0 - 001b - BAR1 - 010b - BAR2 - 011b - BAR3 - 100b - BAR4 - 101b - BAR5 - 110b - ROM - 111b - reserved - IO translation would require either 00100b or 00101b in the inbound TLP TYPE; the BAR Number set in the range 000b - 101b and that BAR configured as an IO BAR. Note: This register field is sticky.
7-0 MSG_CODE	MSG TLPs: (Message Code). When the TYPE field of an inbound Msg/MsgD TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful). This check is only performed if the "Message Code Match Enable" bit of the "iATU Control 2 Register" is set. Memory TLPs: (ST;Steering Tag). When the ST field of an inbound TLP is matched to this value, then address translation proceeds. This check is only performed if the "ST Match Enable" bit of the "iATU Control2 Register" is set. The setting is independent of the setting of the TH field. Note: This register field is sticky.

11.3.5.1.179 iATU Lower Base Address Register. (IATU_LWR_BASE_ADDR_OFF_INBOUND_3)

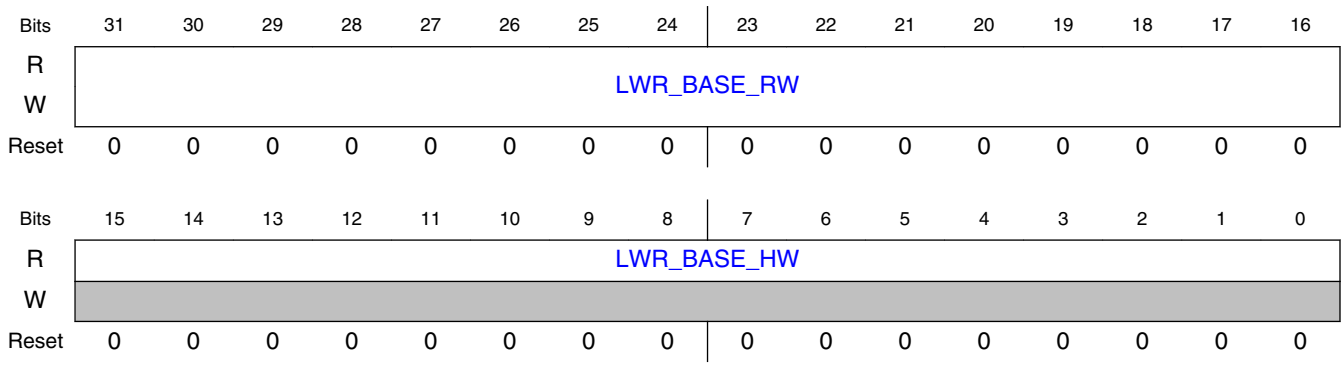
11.3.5.1.179.1 Offset

Register	Offset
IATU_LWR_BASE_ADDR_OFF_INBOUND_3	8000_0708h

11.3.5.1.179.2 Function

iATU Lower Base Address Register. The Minimum Size of iATU Region (Value Range: 4 kB, 8 kB, 16 kB, 32 kB, 64 kB defaults to 64 kB) specifies the minimum size of an address translation region. For example, if set to 64 kB; the lower 16 bits of the Base, Limit and Target registers are zero and all address regions are aligned on 64 kB boundaries. More precisely, the lower log2(Minimum Size of iATU Region) bits are zero.

11.3.5.1.179.3 Diagram



11.3.5.1.179.4 Fields

Field	Function
31-16 LWR_BASE_RW	Forms bits [31:n] of the start address of the address region to be translated. n is log2(Minimum Size of iATU Region) Note: This register field is sticky.
15-0 LWR_BASE_HW	Forms bits [n-1:0] of the start address of the address region to be translated. The start address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. n is log2(Minimum Size of iATU Region)

11.3.5.1.180 iATU Upper Base Address Register. (IATU_UPPER_BASE_ADDR_OFF_INBOUND_3)

11.3.5.1.180.1 Offset

Register	Offset
IATU_UPPER_BASE_ADDR_OFF_INBOUND_3	8000_070Ch

11.3.5.1.180.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	UPPER_BASE_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.180.3 Fields

Field	Function
31-0 UPPER_BASE_RW	Forms bits [63:32] of the start (and end) address of the address region to be translated. Note: This register field is sticky.

11.3.5.1.181 iATU Limit Address Register. (IATU_LIMIT_ADDR_OFF_INBOUND_3)

11.3.5.1.181.1 Offset

Register	Offset
IATU_LIMIT_ADDR_OFF_INBOUND_3	8000_0710h

11.3.5.1.181.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LIMIT_ADDR_RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LIMIT_ADDR_HW															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

11.3.5.1.181.3 Fields

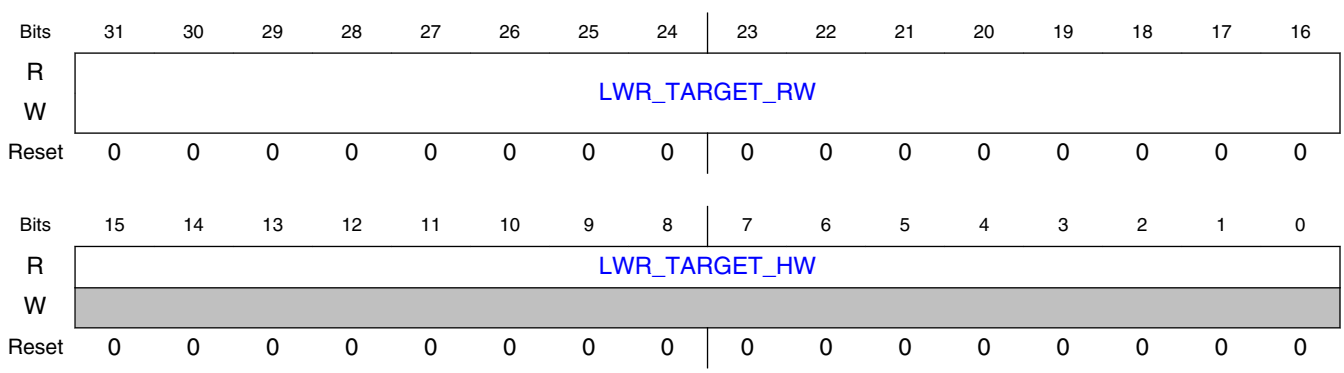
Field	Function
31-16 LIMIT_ADDR_RW	Forms upper bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller. Note: This register field is sticky.
15-0 LIMIT_ADDR_HW	Forms lower bits of the end address of the address region to be translated. The end address must be aligned to the Minimum Size of iATU Region kB boundary, so these bits are always 0. A write to this location is ignored by the PCIe controller.

11.3.5.1.182 iATU Lower Target Address Register. (IATU_LWR_TARGET_ADDR_OFF_INBOUND_3)

11.3.5.1.182.1 Offset

Register	Offset
IATU_LWR_TARGET_ADDR_OFF_INBOUND_3	8000_0714h

11.3.5.1.182.2 Diagram



11.3.5.1.182.3 Fields

Field	Function
31-16 LWR_TARGET_RW	Forms MSB's of the Lower Target part of the new address of the translated region. These bits are always '0'. - Field size depends on log2(Minimum Size of iATU Region) in address match mode. - Field size depends on log2(BAR_MASK+1) in BAR match mode. Note: This register field is sticky.

Table continues on the next page...

Field	Function
15-0 LWR_TARGET_HW	Forms the LSB's of the Lower Target part of the new address of the translated region. The start address must be aligned to the Minimum Size of iATU Region kB boundary (in address match mode); and to the Bar size boundary (in BAR match mode) so that these bits are always '0'. If the BAR is smaller than the iATU region size, then the iATU target address must align to the iATU region size; otherwise it must align to the BAR size. A write to this location is ignored by the PCIe controller. - Field size depends on log2(Minimum Size of iATU Region) in address match mode. - Field size depends on log2(BAR_MASK +1) in BAR match mode.

11.3.5.1.183 DMA Arbitration Scheme for TRGT1 Interface. (DMA_CTRL_DATA_ARB_PRIOR_OFF)

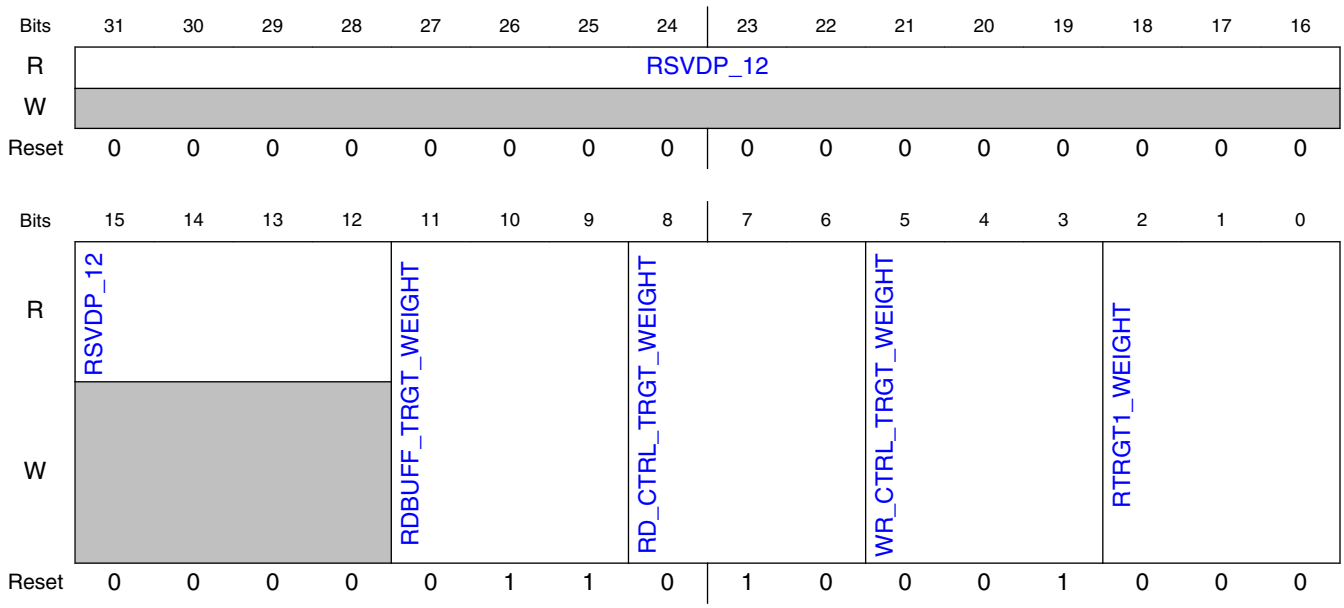
11.3.5.1.183.1 Offset

Register	Offset
DMA_CTRL_DATA_ARB_PRIOR_OFF	8008_0000h

11.3.5.1.183.2 Function

DMA Arbitration Scheme for TRGT1 Interface. This register is used to control traffic priorities among various sources that are delivered to your application through TRGT1 where 0x0 represents the highest priority. - Non-DMA Rx Requests - DMA Write Channel MRd Requests (DMA data requests and LL element/descriptor access) - DMA Read Channel MRd Requests (LL element/descriptor access) - DMA Read Channel MWr Requests Concurrent traffic from channels with same priority are sorted according to Round-Robin arbitration rules. The arbitration priority defaults to Non-DMA requests (highest), Write Channel MRd, Read Channel MRd, Read Channel MWr.

11.3.5.1.183.3 Diagram



11.3.5.1.183.4 Fields

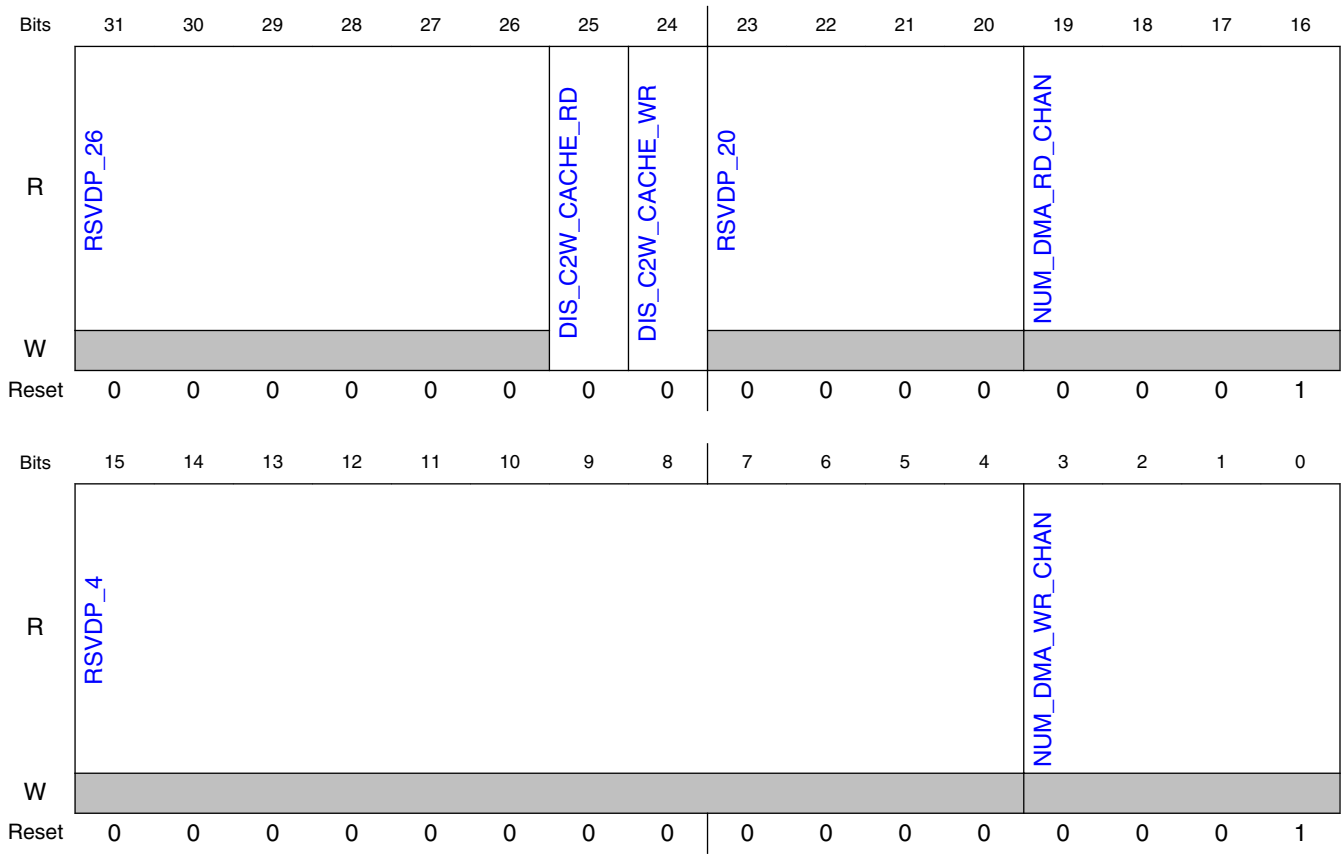
Field	Function
31-12 RSVDP_12	Reserved for future use.
11-9 RDBUFF_TRGT_WEIGHT	DMA Read Channel MWr Requests. Note: The access attributes of this field are as follows: - Dbi: R/W
8-6 RD_CTRL_TRGT_WEIGHT	DMA Read Channel MRd Requests. For LL element/descriptor access. Note: The access attributes of this field are as follows: - Dbi: R/W
5-3 WR_CTRL_TRGT_WEIGHT	DMA Write Channel MRd Requests. For DMA data requests and LL element/descriptor access. Note: The access attributes of this field are as follows: - Dbi: R/W
2-0 RTRGT1_WEIGHT	Non-DMA Rx Requests. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.184 DMA Number of Channels Register. (DMA_CTRL_OFF)

11.3.5.1.184.1 Offset

Register	Offset
DMA_CTRL_OFF	8008_0008h

11.3.5.1.184.2 Diagram



11.3.5.1.184.3 Fields

Field	Function
31-26 RSVDP_26	Reserved for future use.
25 DIS_C2W_CACHE_RD	Disable DMA Read Channels "completion to memory write" context cache pre-fetch function. Note: For internal debugging only. Note: The access attributes of this field are as follows: - Dbi: R/W
24 DIS_C2W_CACHE_WR	Disable DMA Write Channels "completion to memory write" context cache pre-fetch function. Note: For internal debugging only. Note: The access attributes of this field are as follows: - Dbi: R/W

Table continues on the next page...

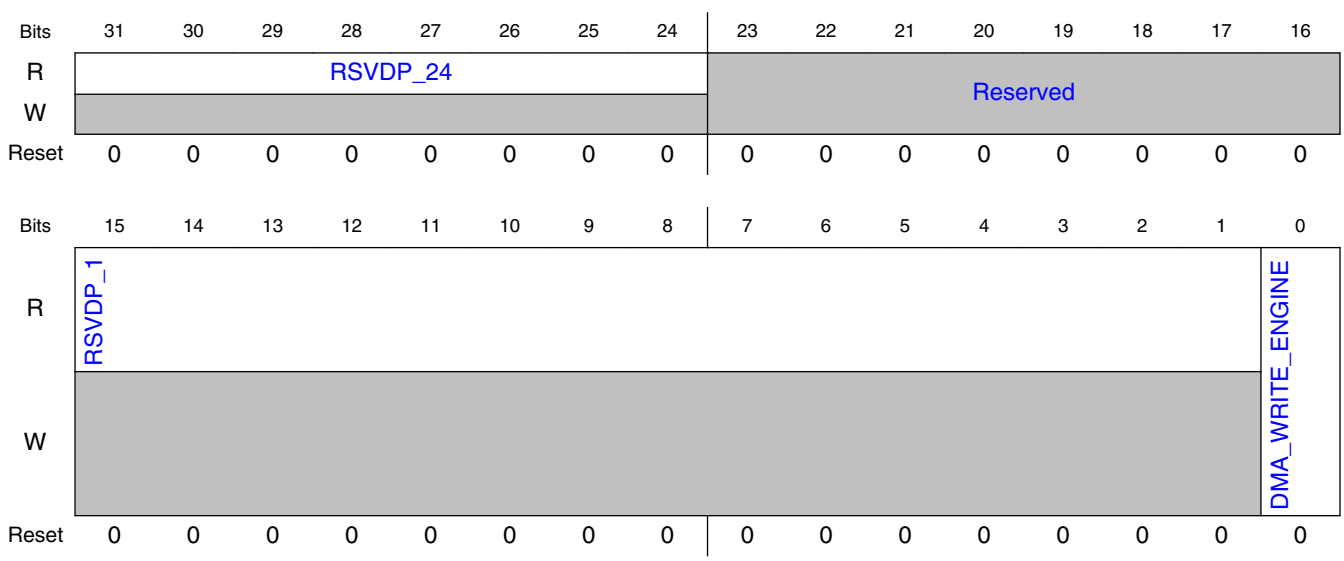
Field	Function
23-20 RSVDP_20	Reserved for future use.
19-16 NUM_DMA_RD_CHAN	Number of Read Channels. You can read this register to determine the number of read channels the DMA controller has been configured to support.
15-4 RSVDP_4	Reserved for future use.
3-0 NUM_DMA_WR_CHAN	Number of Write Channels. You can read this register to determine the number of write channels the DMA controller has been configured to support.

11.3.5.1.185 DMA Write Engine Enable Register. (DMA_WRITE_ENGINE_EN_OFF)

11.3.5.1.185.1 Offset

Register	Offset
DMA_WRITE_ENGINE_EN_OFF	8008_000Ch

11.3.5.1.185.2 Diagram



11.3.5.1.185.3 Fields

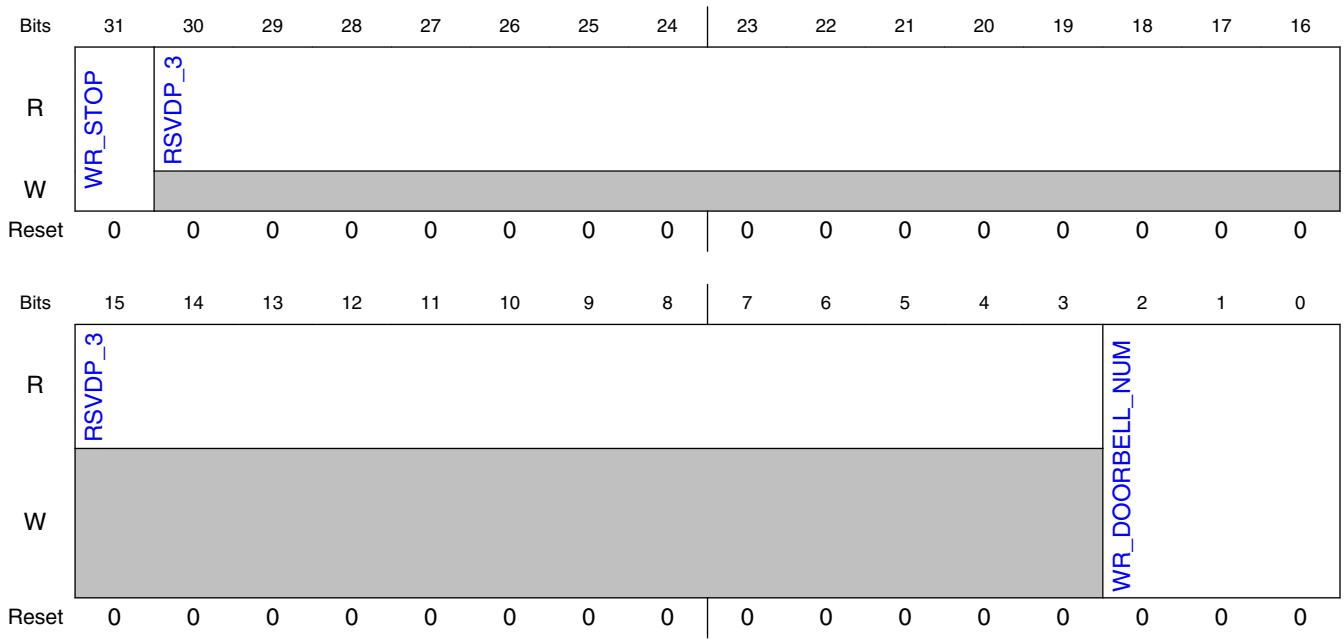
Field	Function
31-24 RSVDP_24	Reserved for future use.
23-16 —	Reserved.
15-1 RSVDP_1	Reserved for future use.
0 DMA_WRITE_ENGINE	DMA Write Engine Enable. - 1: Enable - 0: Disable (Soft Reset) For normal operation, you must initially set this bit to "1", before any other software setup actions. You do not need to toggle or rewrite to this bit during normal operation. You should set this bit to "0" when you want to "Soft Reset" the DMA controller write logic. There are three possible reasons for resetting the DMA controller write logic: - The "Abort Interrupt Status" bit is set (in the "DMA Write Interrupt Status Register" DMA_WRITE_INT_STATUS_OFF), and any of the bits is in the "DMA Write Error Status Register" (DMA_WRITE_ERR_STATUS_OFF) are set. Resetting the DMA controller write logic re-initializes the control logic, ensuring that the next DMA write transfer is executed successfully. - You have executed the procedure outlined in "Stop Bit" , after which, the "Abort Interrupt Status" bit is set and the Channel Status field (CS) of the DMA write "DMA Channel Control 1 Register " (DMA_CH_CONTROL1_OFF_WRCH_0) is set to "Stopped." Resetting the DMA controller write logic re-initializes the control logic ensuring that the next DMA write transfer is executed successfully. - During software development, when you incorrectly program the DMA write engine. To "Soft Reset" the DMA controller write logic, you must: - De-assert the DMA write engine enable bit. - Wait for the DMA to complete any in-progress TLP transfer, by waiting until a read on the DMA write engine enable bit returns a "0". - Assert the DMA write engine enable bit. This "Soft Reset" does not clear the DMA configuration registers. The DMA write transfer does not start until you write to the "DMA Write Doorbell Register" (DMA_WRITE_DOORBELL_OFF). Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.186 DMA Write Doorbell Register. (DMA_WRITE_DOORBELL_OFF)

11.3.5.1.186.1 Offset

Register	Offset
DMA_WRITE_DOORBELL_OFF	8008_0010h

11.3.5.1.186.2 Diagram



11.3.5.1.186.3 Fields

Field	Function
31 WR_STOP	Stop. Set in conjunction with the Doorbell Number field. The DMA write channel stops issuing requests, sets the channel status to "Stopped", and asserts the "Abort" interrupt if it is enabled. Before setting the Stop bit, you must read the channel Status field (CS) of the "DMA Channel Control 1 Register " (DMA_CH_CONTROL1_OFF_WRCH_0) to ensure that the write channel is "Running" (transferring data). For more information, see "Stopping the DMA Transfer (Software Stop)." Note: The access attributes of this field are as follows: - Dbi: R/W
30-3 RSVDP_3	Reserved for future use.
2-0 WR_DOORBELL_NUM	Doorbell Number. You must write the channel number to this register to start the DMA write transfer for that channel. The DMA detects a write to this register field even if the value of this field does not change. You do not need to toggle or write any other value to this register to start a new transfer. The range of this field is 0x0 to 0x7, and 0x0 corresponds to channel 0. Also note that a write to this field triggers the controller to exit L1 substates. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.187 DMA Write Engine Channel Arbitration Weight Low Register. (DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF)

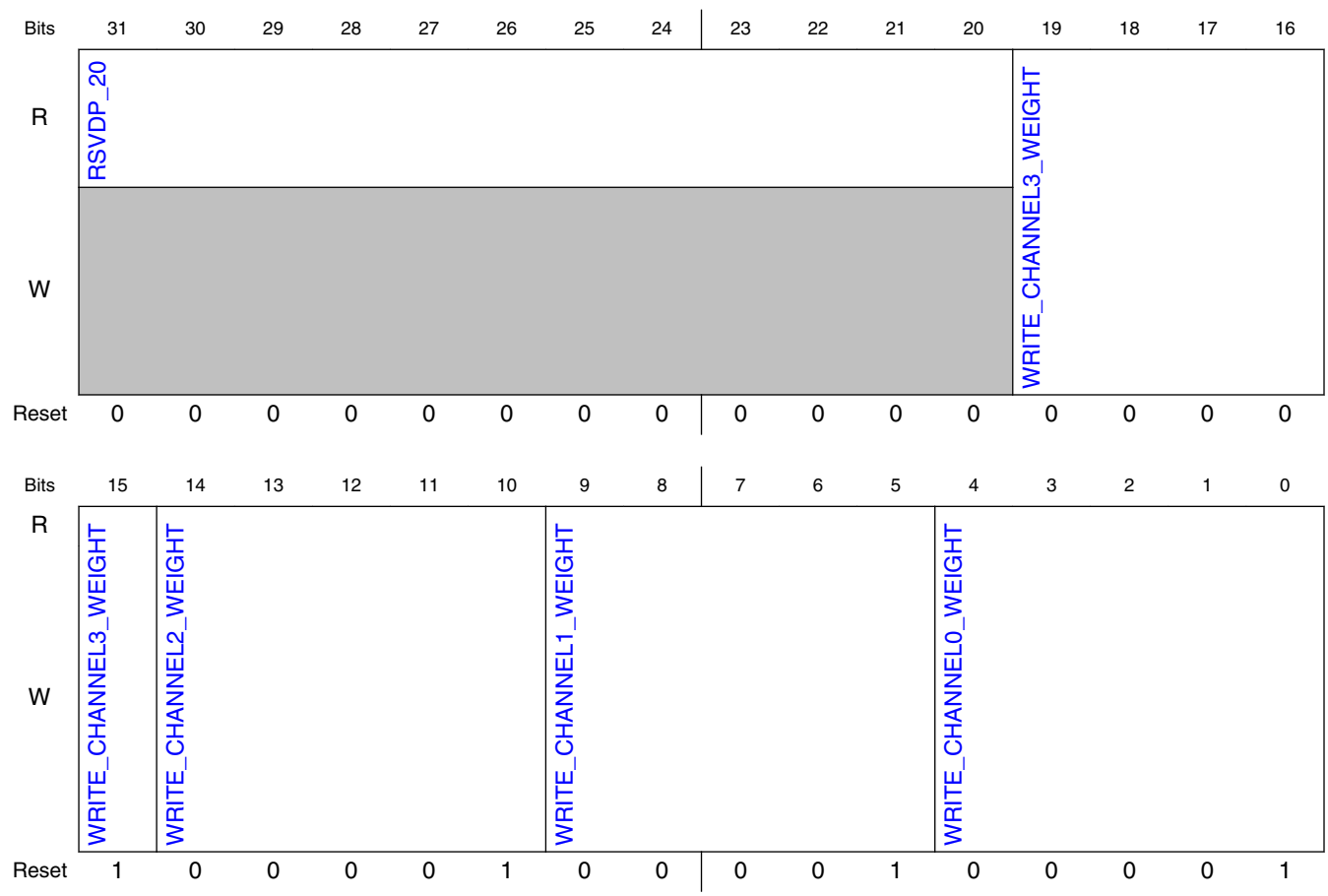
11.3.5.1.187.1 Offset

Register	Offset
DMA_WRITE_CHANNEL_ARB_WEIGHT_LOW_OFF	8008_0018h

11.3.5.1.187.2 Function

DMA Write Engine Channel Arbitration Weight Low Register. The 5-bit channel weight (for write channels 0-3) specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached or DMA channel request transfer size reaches zero, the WWR arbiter selects the next channel to be processed. Your software must initialize this register before ringing the doorbell. For more details, see "Multichannel Arbitration". Value range is (0-0x1F) corresponding to (1-32) transaction requests.

11.3.5.1.187.3 Diagram



11.3.5.1.187.4 Fields

Field	Function
31-20 RSVDP_20	Reserved for future use.
19-15 WRITE_CHANNEL3_WEIGHT	Channel 3 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Dbi: R/W
14-10 WRITE_CHANNEL2_WEIGHT	Channel 2 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Dbi: R/W
9-5 WRITE_CHANNEL1_WEIGHT	Channel 1 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Dbi: R/W
4-0 WRITE_CHANNEL0_WEIGHT	Channel 0 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.188 DMA Write Engine Channel Arbitration Weight High Register. (DMA_WRITE_CHANNEL_ARB_WEIGHT_HIGH_OFF)

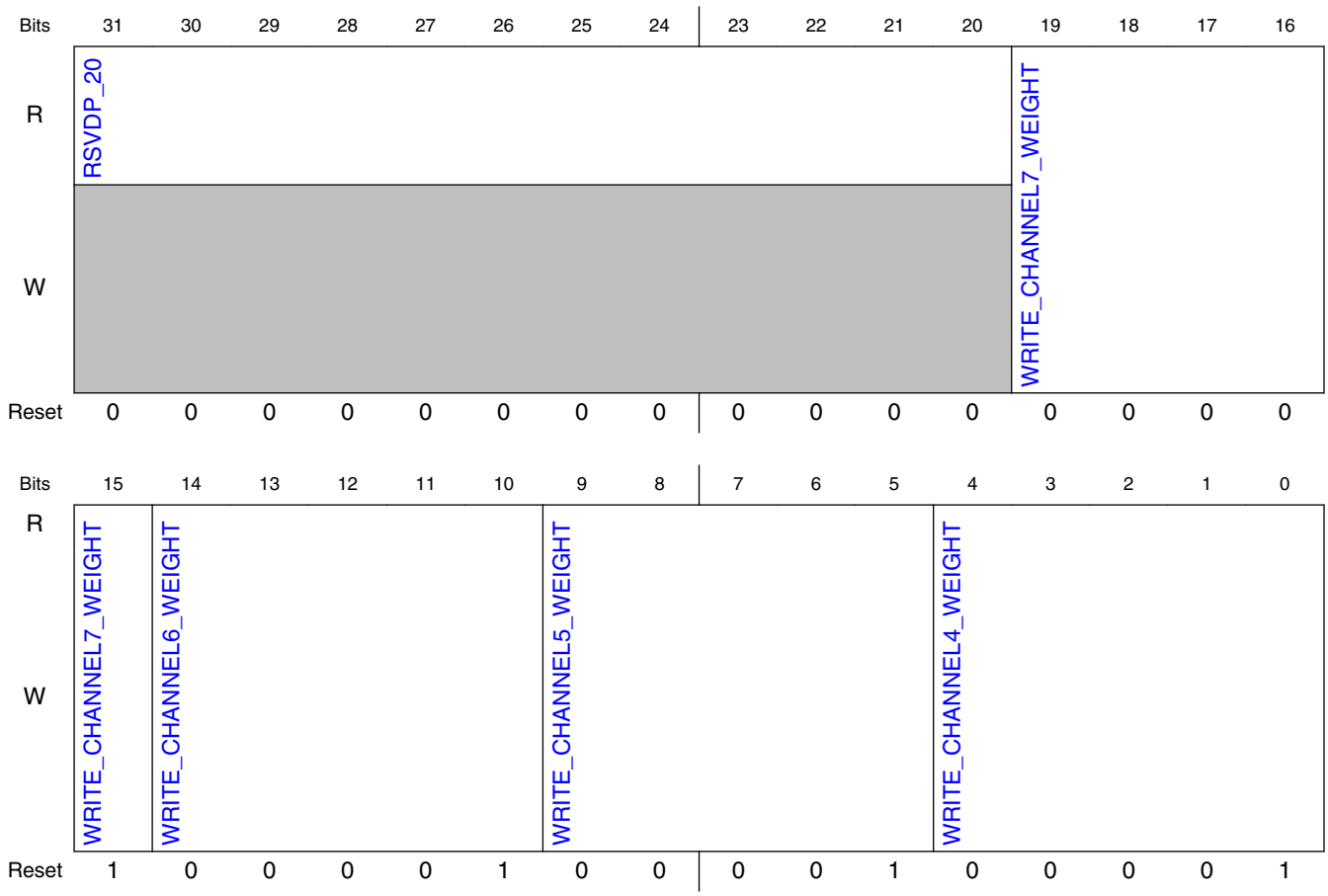
11.3.5.1.188.1 Offset

Register	Offset
DMA_WRITE_CHANNEL_ARB_WEIGHT_HIGH_OFF	8008_001Ch

11.3.5.1.188.2 Function

DMA Write Engine Channel Arbitration Weight High Register. The 5-bit channel weight (for write channels 4-7) specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached or DMA channel request transfer size reaches zero, the WWR arbiter selects the next channel to be processed. Your software must initialize this register before ringing the doorbell. For more details, see "Multichannel Arbitration". Value range is (0-0x1F) corresponding to (1-32) transaction requests.

11.3.5.1.188.3 Diagram



11.3.5.1.188.4 Fields

Field	Function
31-20 RSVDP_20	Reserved for future use.
19-15 WRITE_CHANNEL7_WEIGHT	Channel 7 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Dbi: R/W
14-10 WRITE_CHANNEL6_WEIGHT	Channel 6 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Dbi: R/W
9-5 WRITE_CHANNEL5_WEIGHT	Channel 5 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Dbi: R/W
4-0	Channel 4 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. A value of '0' means

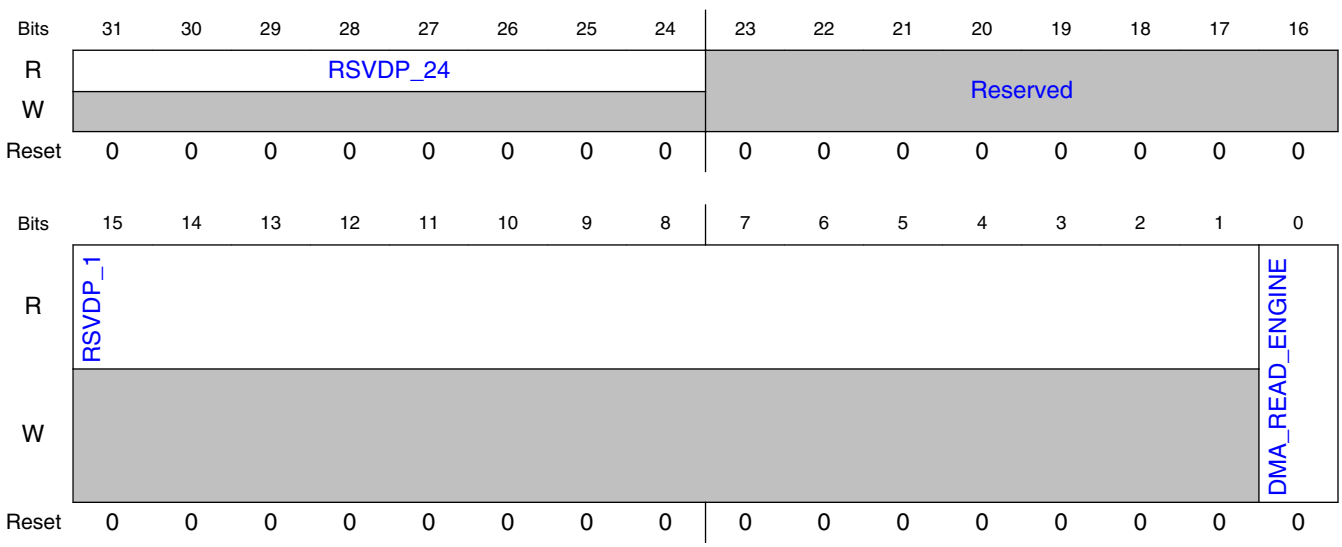
Field	Function
WRITE_CHANNEL4_WEIGHT	that one TLP is issued before moving to the next channel. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.189 DMA Read Engine Enable Register. (DMA_READ_ENGINE_EN_OFF)

11.3.5.1.189.1 Offset

Register	Offset
DMA_READ_ENGINE_EN_OFF	8008_002Ch

11.3.5.1.189.2 Diagram



11.3.5.1.189.3 Fields

Field	Function
31-24 RSVDP_24	Reserved for future use.
23-16 —	Reserved.
15-1 RSVDP_1	Reserved for future use.

Table continues on the next page...

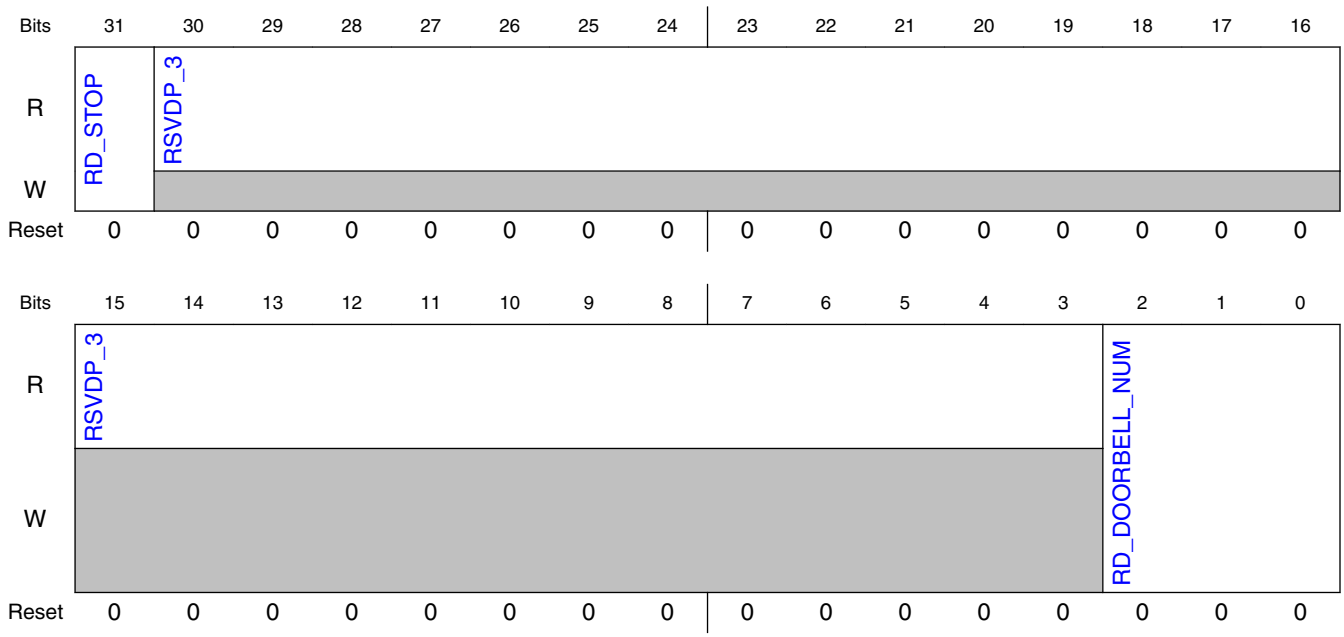
Field	Function
0 DMA_READ_ENGINE	DMA Read Engine Enable. - 1: Enable - 0: Disable (Soft Reset) For normal operation, you must initially set this bit to "1", before any other software setup actions. You do not need to toggle or rewrite to this bit during normal operation. You should set this field to "0" when you want to "Soft Reset" the DMA controller read logic. There are three possible reasons for resetting the DMA controller read logic: - The "Abort Interrupt Status" bit is set (in the "DMA Read Interrupt Status Register" (DMA_READ_INT_STATUS_OFF), and any of the bits in the "DMA Read Error Status Low Register" (DMA_READ_ERR_STATUS_LOW_OFF) is set. Resetting the DMA controller read logic re-initializes the control logic, ensuring that the next DMA read transfer is executed successfully. - You have executed the procedure outlined in "Stop Bit", after which, the "Abort Interrupt Status" bit is set and the channel Status field (CS) of the DMA read "DMA Channel Control 1 Register " (DMA_CH_CONTROL1_OFF_WRCH_0) is set to "Stopped". Resetting the DMA controller read logic re-initializes the control logic ensuring that the next DMA read transfer is executed successfully. - During software development, when you incorrectly program the DMA read engine. To "Soft Reset" the DMA controller read logic, you must: - De-assert the DMA read engine enable bit. - Wait for the DMA to complete any in-progress TLP transfer, by waiting until a read on the DMA read engine enable bit returns a "0". - Assert the DMA read engine enable bit. This "Soft Reset" does not clear the DMA configuration registers. The DMA read transfer does not start until you write to the "DMA Read Doorbell Register" (DMA_READ_DOORBELL_OFF). Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.190 DMA Read Doorbell Register. (DMA_READ_DOORBELL_OFF)

11.3.5.1.190.1 Offset

Register	Offset
DMA_READ_DOORBELL_OFF	8008_0030h

11.3.5.1.190.2 Diagram



11.3.5.1.190.3 Fields

Field	Function
31 RD_STOP	Stop. Set in conjunction with the Doorbell Number field. The DMA read channel stops issuing requests, sets the channel status to "Stopped", and asserts the "Abort" interrupt if it is enabled. Before setting the Stop bit, you must read the channel Status field (CS) of the "DMA Channel Control 1 Register " (DMA_CH_CONTROL1_OFF_RDCH_0) to ensure that the read channel is "Running" (transferring data). For more information, see "Stopping the DMA Transfer (Software Stop)". Note: The access attributes of this field are as follows: - Dbi: R/W
30-3 RSVDP_3	Reserved for future use.
2-0 RD_DOORBELL_NUM	Doorbell Number. You must write 0x0 to this register to start the DMA read transfer for that channel. The DMA detects a write to this register field even if the value of this field does not change. The range of this field is 0x0 to 0x7, and 0x0 corresponds to channel 0. Also note that a write to this field triggers the controller to exit L1 substates. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.191 DMA Read Engine Channel Arbitration Weight Low Register. (DMA_READ_CHANNEL_ARB_WEIGHT_LOW_OFF)

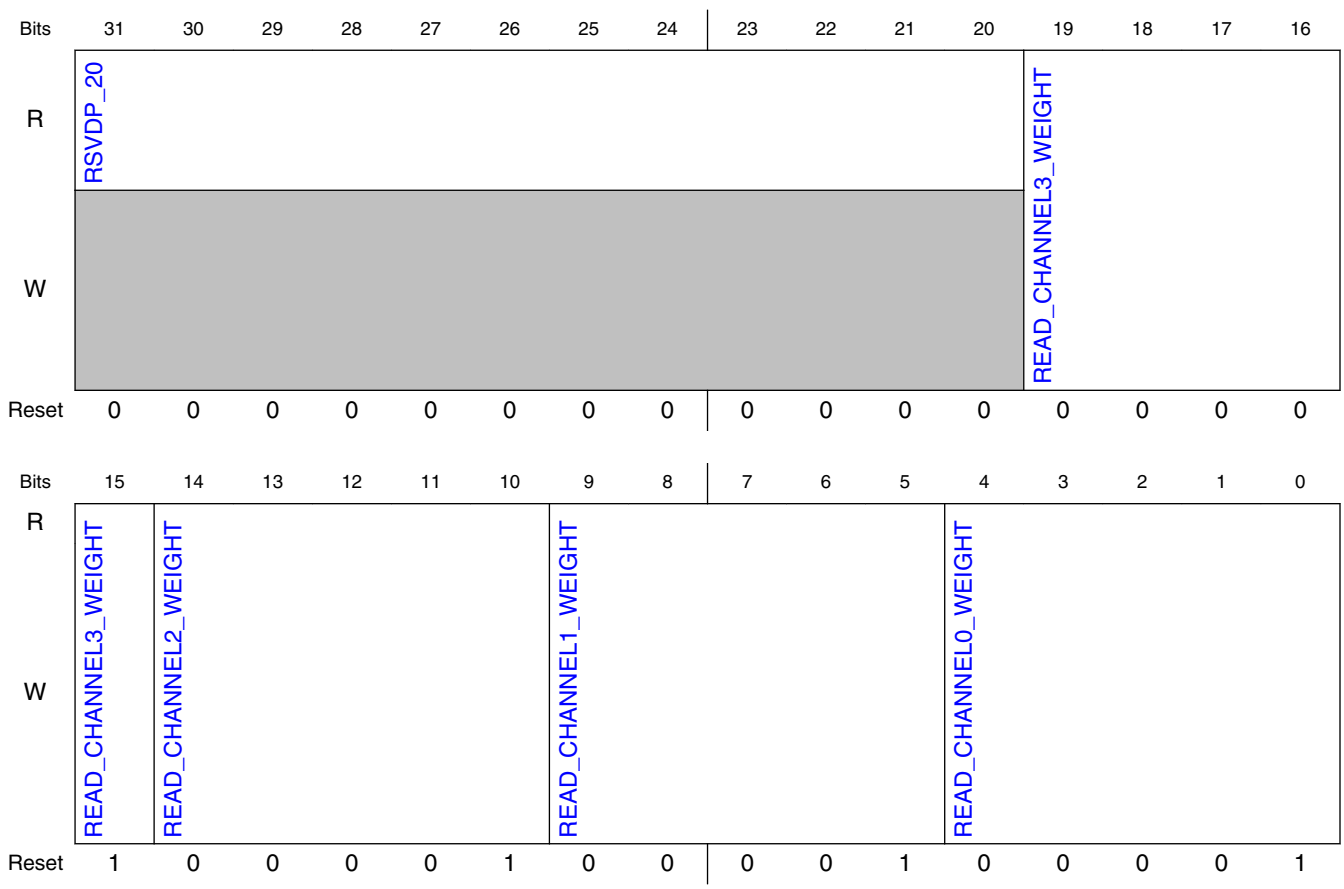
11.3.5.1.191.1 Offset

Register	Offset
DMA_READ_CHANNEL_ARB_WEIGHT_LOW_OFF	8008_0038h

11.3.5.1.191.2 Function

DMA Read Engine Channel Arbitration Weight Low Register. The 5-bit channel weight (for read channels 0-3) specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached or DMA channel request transfer size reaches zero, the WWR arbiter selects the next channel to be processed. Your software must initialize this register before ringing the doorbell. For more details, see "Multichannel Arbitration". Value range is (0-0x1F) corresponding to (1-32) transaction requests.

11.3.5.1.191.3 Diagram



11.3.5.1.191.4 Fields

Field	Function
31-20 RSVDP_20	Reserved for future use.
19-15 READ_CHANNEL3_WEIGHT	Channel 3 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Dbi: R/W
14-10 READ_CHANNEL2_WEIGHT	Channel 2 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Dbi: R/W
9-5 READ_CHANNEL1_WEIGHT	Channel 1 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Dbi: R/W
4-0 READ_CHANNEL0_WEIGHT	Channel 0 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.192 DMA Read Engine Channel Arbitration Weight High Register. (DMA_READ_CHANNEL_ARB_WEIGHT_HIGH_OFF)

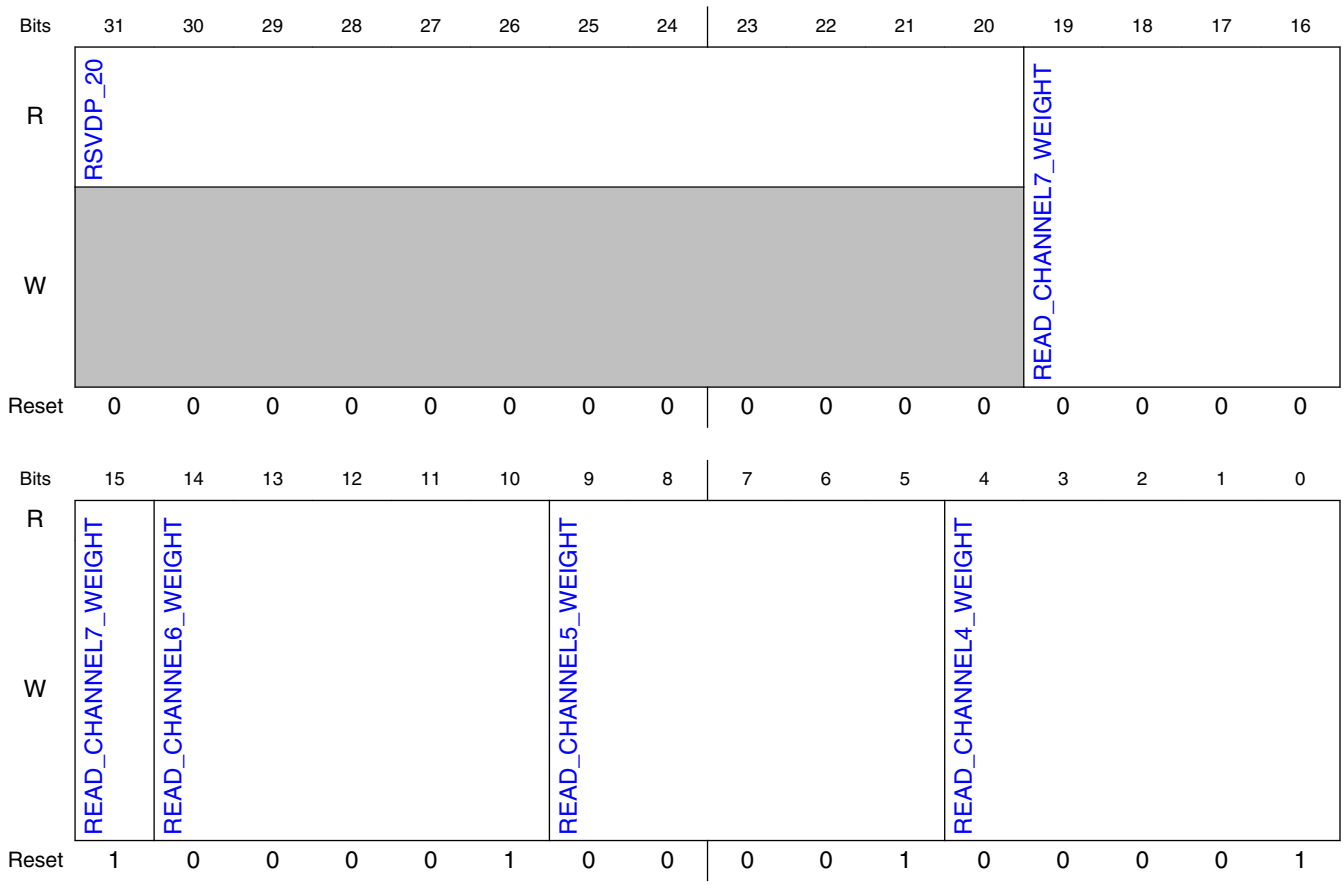
11.3.5.1.192.1 Offset

Register	Offset
DMA_READ_CHANNEL_ARB_WEIGHT_HIGH_OFF	8008_003Ch

11.3.5.1.192.2 Function

DMA Read Engine Channel Arbitration Weight High Register. The 5-bit channel weight (for read channels 4-7) specifies the maximum number of TLP requests that the DMA can issue for that channel before it must return to the arbitration routine. When the channel weight count is reached or DMA channel request transfer size reaches zero, the WWR arbiter selects the next channel to be processed. Your software must initialize this register before ringing the doorbell. For more details, see "Multichannel Arbitration". Value range is (0-0x1F) corresponding to (1-32) transaction requests.

11.3.5.1.192.3 Diagram



11.3.5.1.192.4 Fields

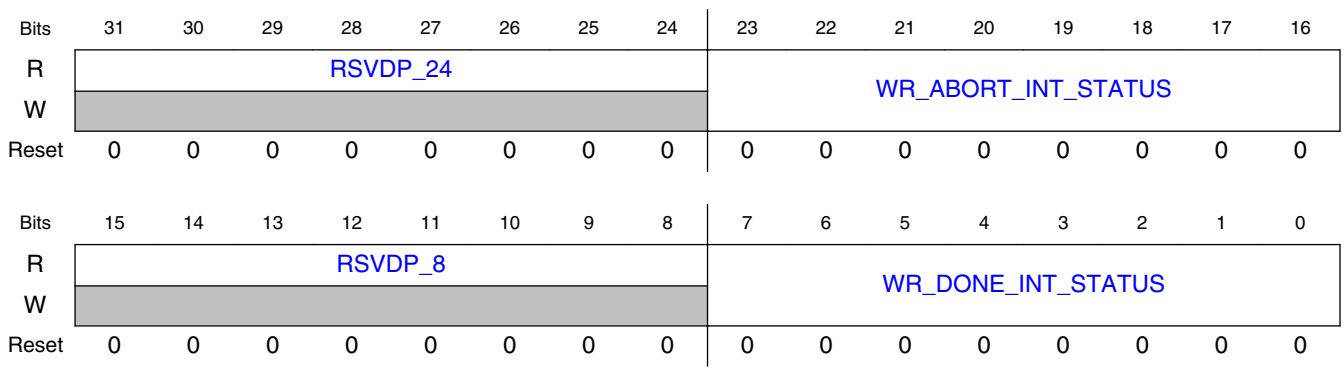
Field	Function
31-20 RSVDP_20	Reserved for future use.
19-15 READ_CHANNEL7_WEIGHT	Channel 7 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Dbi: R/W
14-10 READ_CHANNEL6_WEIGHT	Channel 6 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Dbi: R/W
9-5 READ_CHANNEL5_WEIGHT	Channel 5 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Dbi: R/W
4-0 READ_CHANNEL4_WEIGHT	Channel 4 Weight. The weight is initialized by software before ringing the doorbell. The value is used by the channel weighted round robin arbiter to select the next channel read request. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.193 DMA Write Interrupt Status Register. (DMA_WRITE_INT_STATUS_OFF)

11.3.5.1.193.1 Offset

Register	Offset
DMA_WRITE_INT_STATUS_OFF	8008_004Ch

11.3.5.1.193.2 Diagram



11.3.5.1.193.3 Fields

Field	Function
31-24 RSVDP_24	Reserved for future use.
23-16 WR_ABORT_INT_STATUS	Abort Interrupt Status. The DMA write channel has detected an error, or you manually stopped the transfer as described in "Error Handling Assistance by Remote Software". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. - Enabling: For details, see "Interrupts and Error Handling". - Masking: The DMA write interrupt Mask register has no effect on this register. - Clearing: You must write a 1'b1 to the corresponding channel bit in the DMA write interrupt Clear register to clear this interrupt bit. Note: You can write to this register to emulate interrupt generation, during software or hardware testing. A write to the address triggers an interrupt, but the DMA does not set the Done or Abort bits in this register. Note: The access attributes of this field are as follows: - Dbi: R/W
15-8 RSVDP_8	Reserved for future use.
7-0 WR_DONE_INT_STATUS	Done Interrupt Status. The DMA write channel has successfully completed the DMA transfer. For more details, see "Interrupts and Error Handling". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. - Enabling: For details, see "Interrupts and Error Handling". - Masking: The DMA write interrupt Mask register has no effect on this register. - Clearing: You must write a 1'b1 to the corresponding channel bit in the DMA write interrupt Clear register to clear this interrupt bit. Note: You can write to this register to emulate interrupt generation, during software or hardware testing. A write to

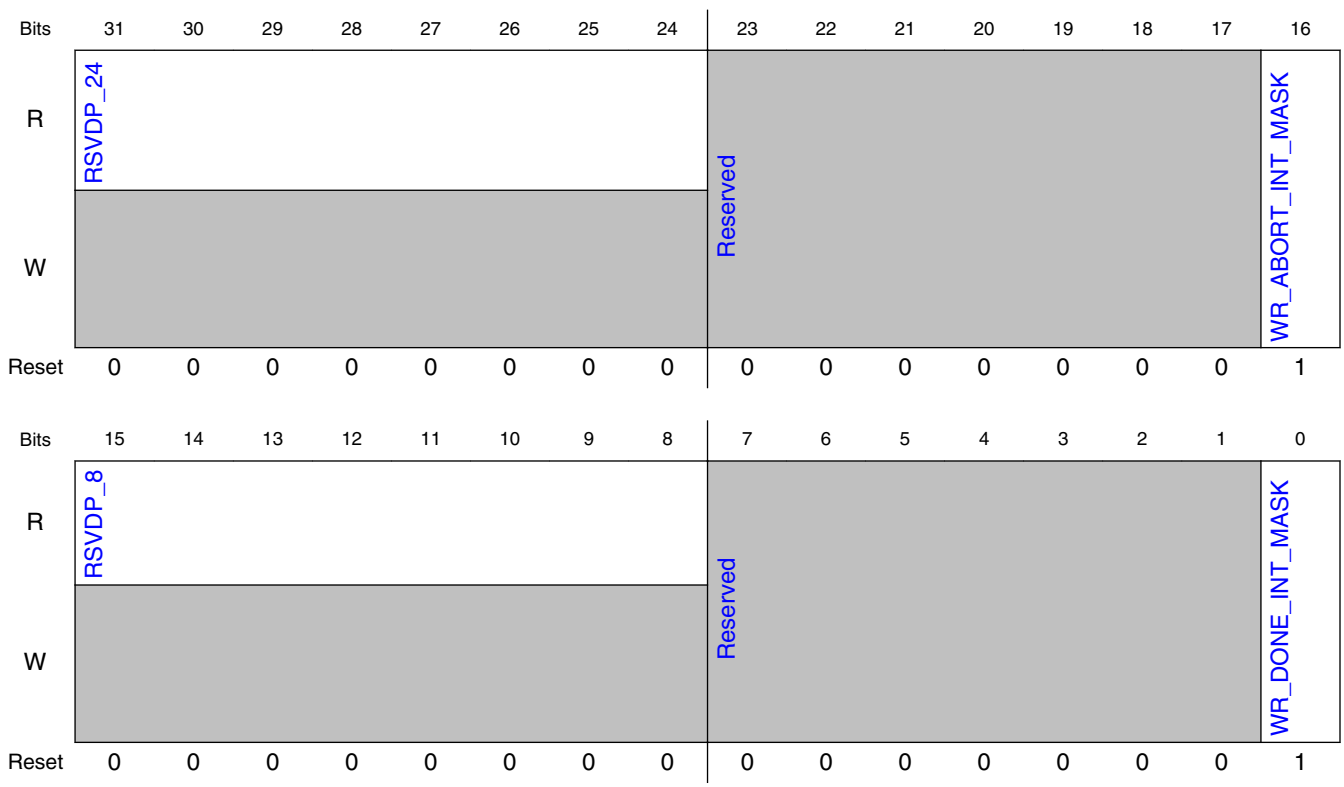
Field	Function
	the address triggers an interrupt, but the DMA does not set the Done or Abort bits in this register. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.194 DMA Write Interrupt Mask Register. (DMA_WRITE_INT_MASK_OFF)

11.3.5.1.194.1 Offset

Register	Offset
DMA_WRITE_INT_MASK_OFF	8008_0054h

11.3.5.1.194.2 Diagram



11.3.5.1.194.3 Fields

Field	Function
31-24	Reserved for future use.

Table continues on the next page...

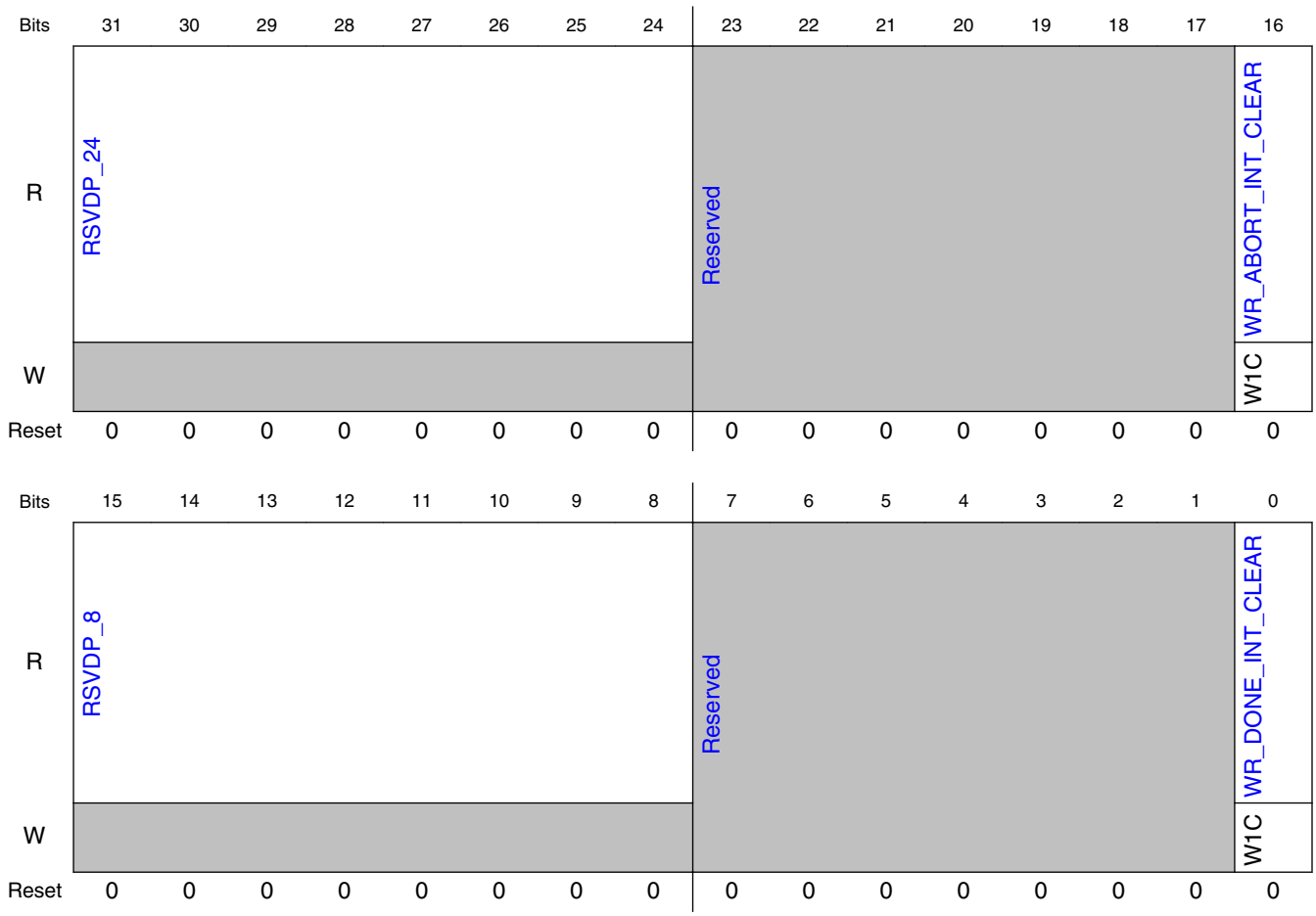
Field	Function
RSVDP_24	
23-17 —	Reserved.
16 WR_ABORT_INT_MASK	Abort Interrupt Mask. Prevents the Abort interrupt status field in the DMA write interrupt status register from asserting the edma_int output. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: The access attributes of this field are as follows: - Dbi: R/W
15-8 RSVDP_8	Reserved for future use.
7-1 —	Reserved.
0 WR_DONE_INT_MASK	Done Interrupt Mask. Prevents the Done interrupt status field in the DMA write interrupt status register from asserting the edma_int output. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.195 DMA Write Interrupt Clear Register. (DMA_WRITE_INT_CLEAR_OFF)

11.3.5.1.195.1 Offset

Register	Offset
DMA_WRITE_INT_CLEAR_OFF	8008_0058h

11.3.5.1.195.2 Diagram



11.3.5.1.195.3 Fields

Field	Function
31-24 RSVDP_24	Reserved for future use.
23-17 —	Reserved.
16 WR_ABORT_INT_CLEAR	Abort Interrupt Clear. You must write a 1'b1 to clear the corresponding bit in the Abort interrupt status field of the DMA write interrupt status register. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: Reading from this self-clearing register field always returns a "0".
15-8 RSVDP_8	Reserved for future use.
7-1 —	Reserved.

Table continues on the next page...

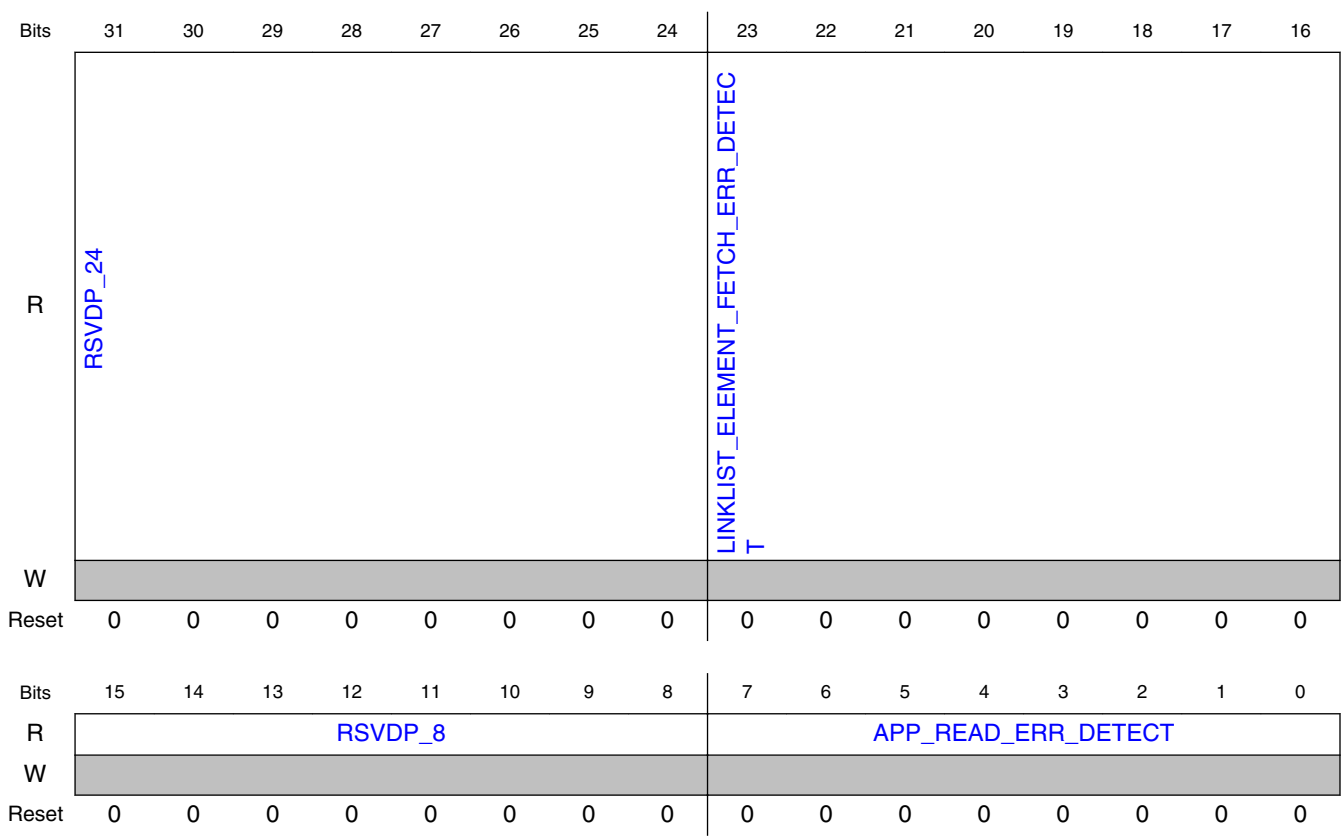
Field	Function
0 WR_DONE_INT_CLEAR	Done Interrupt Clear. You must write a 1'b1 to clear the corresponding bit in the Done interrupt status field of the DMA write interrupt status register. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: Reading from this self-clearing register field always returns a "0".

11.3.5.1.196 DMA Write Error Status Register (DMA_WRITE_ERR_STATUS_OFF)

11.3.5.1.196.1 Offset

Register	Offset
DMA_WRITE_ERR_STATUS_OFF	8008_005Ch

11.3.5.1.196.2 Diagram



11.3.5.1.196.3 Fields

Field	Function
31-24 RSVDP_24	Reserved for future use.
23-16 LINKLIST_ELEMENT_FETCH_ERR_DETECT	Linked List Element Fetch Error Detected. The DMA write channel has received an error response from the AXI bus (or TRGT1 interface when the AXI Bridge is not used) while reading a linked list element from local memory. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. - Enabling: For details, see "Interrupts and Error Handling". - Masking: The DMA write interrupt Mask register has no effect on this register. - Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the "DMA Write Interrupt Clear Register" (DMA_WRITE_INT_CLEAR_OFF) to clear this error bit.
15-8 RSVDP_8	Reserved for future use.
7-0 APP_READ_ERR_DETECT	Application Read Error Detected. The DMA write channel has received an error response from the AXI bus (or TRGT1 interface when the AXI Bridge is not used) while reading data from it. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. - Enabling: For details, see "Interrupts and Error Handling". - Masking: The DMA write interrupt Mask register has no effect on this register. - Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the "DMA Write Interrupt Clear Register" (DMA_WRITE_INT_CLEAR_OFF) to clear this error bit.

11.3.5.1.197 DMA Write Done IMWr Address Low Register. (DMA_WRITE_DONE_IMWR_LOW_OFF)

11.3.5.1.197.1 Offset

Register	Offset
DMA_WRITE_DONE_IMWR_LOW_OFF	8008_0060h

11.3.5.1.197.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_WRITE_DONE_LOW_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_WRITE_DONE_LOW_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.197.3 Fields

Field	Function
31-0 DMA_WRITE_DONE_LOW_REG	The DMA uses this field to generate bits [31:0] of the address field for the Done IMWr TLP. Bits [1:0] must be "00" as this address must be dword aligned. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.198 DMA Write Done IMWr Interrupt Address High Register. (DMA_WRITE_DONE_IMWR_HIGH_OFF)

11.3.5.1.198.1 Offset

Register	Offset
DMA_WRITE_DONE_IMWR_HIGH_OFF	8008_0064h

11.3.5.1.198.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_WRITE_DONE_HIGH_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_WRITE_DONE_HIGH_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.198.3 Fields

Field	Function
31-0 DMA_WRITE_DONE_HIGH_REG	The DMA uses this field to generate bits [63:32] of the address field for the Done IMWr TLP. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.199 DMA Write Abort IMWr Address Low Register. (DMA_WRITE_ABORT_IMWR_LOW_OFF)

11.3.5.1.199.1 Offset

Register	Offset
DMA_WRITE_ABORT_IMWR_LOW_OFF	8008_0068h

11.3.5.1.199.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_WRITE_ABORT_LOW_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_WRITE_ABORT_LOW_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.199.3 Fields

Field	Function
31-0 DMA_WRITE_ABORT_LOW_REG	The DMA uses this field to generate bits [31:0] of the address field for the Abort IMWr TLP it generates. Bits [1:0] must be "00" as this address must be dword aligned. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.200 DMA Write Abort IMWr Address High Register. (DMA_WRITE_ABORT_IMWR_HIGH_OFF)

11.3.5.1.200.1 Offset

Register	Offset
DMA_WRITE_ABORT_IMWR_HIGH_OFF	8008_006Ch

11.3.5.1.200.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_WRITE_ABORT_HIGH_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_WRITE_ABORT_HIGH_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.200.3 Fields

Field	Function
31-0 DMA_WRITE_ABORT_HIGH_REG	The DMA uses this field to generate bits [63:32] of the address field for the Abort IMWr TLP. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.201 DMA Write Channel 1 and 0 IMWr Data Register. (DMA_WRITE_CH01_IMWR_DATA_OFF)

11.3.5.1.201.1 Offset

Register	Offset
DMA_WRITE_CH01_IMWR_DATA_OFF	8008_0070h

11.3.5.1.201.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WR_CHANNEL_1_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WR_CHANNEL_0_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.201.3 Fields

Field	Function
31-16 WR_CHANNEL_1_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 1. Note: The access attributes of this field are as follows: - Dbi: R/W
15-0 WR_CHANNEL_0_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 0. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.202 DMA Write Channel 3 and 2 IMWr Data Register. (DMA_WRITE_CH23_IMWR_DATA_OFF)

11.3.5.1.202.1 Offset

Register	Offset
DMA_WRITE_CH23_IMWR_DATA_OFF	8008_0074h

11.3.5.1.202.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WR_CHANNEL_3_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WR_CHANNEL_2_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.202.3 Fields

Field	Function
31-16 WR_CHANNEL_3_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 3. Note: The access attributes of this field are as follows: - Dbi: R/W

Table continues on the next page...

Field	Function
15-0 WR_CHANNEL_2_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 2. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.203 DMA Write Channel 5 and 4 IMWr Data Register. (DMA_WRITE_CH45_IMWR_DATA_OFF)

11.3.5.1.203.1 Offset

Register	Offset
DMA_WRITE_CH45_IMWR_DATA_OFF	8008_0078h

11.3.5.1.203.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WR_CHANNEL_5_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WR_CHANNEL_4_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.203.3 Fields

Field	Function
31-16 WR_CHANNEL_5_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 5. Note: The access attributes of this field are as follows: - Dbi: R/W
15-0 WR_CHANNEL_4_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 4. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.204 DMA Write Channel 7 and 6 IMWr Data Register. (DMA_WRITE_CH67_IMWR_DATA_OFF)

11.3.5.1.204.1 Offset

Register	Offset
DMA_WRITE_CH67_IMWR_DATA_OFF	8008_007Ch

11.3.5.1.204.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WR_CHANNEL_7_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WR_CHANNEL_6_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.204.3 Fields

Field	Function
31-16 WR_CHANNEL_7_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 7. Note: The access attributes of this field are as follows: - Dbi: R/W
15-0 WR_CHANNEL_6_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for write channel 6. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.205 DMA Write Linked List Error Enable Register. (DMA_WRITE_LINKED_LIST_ERR_EN_OFF)

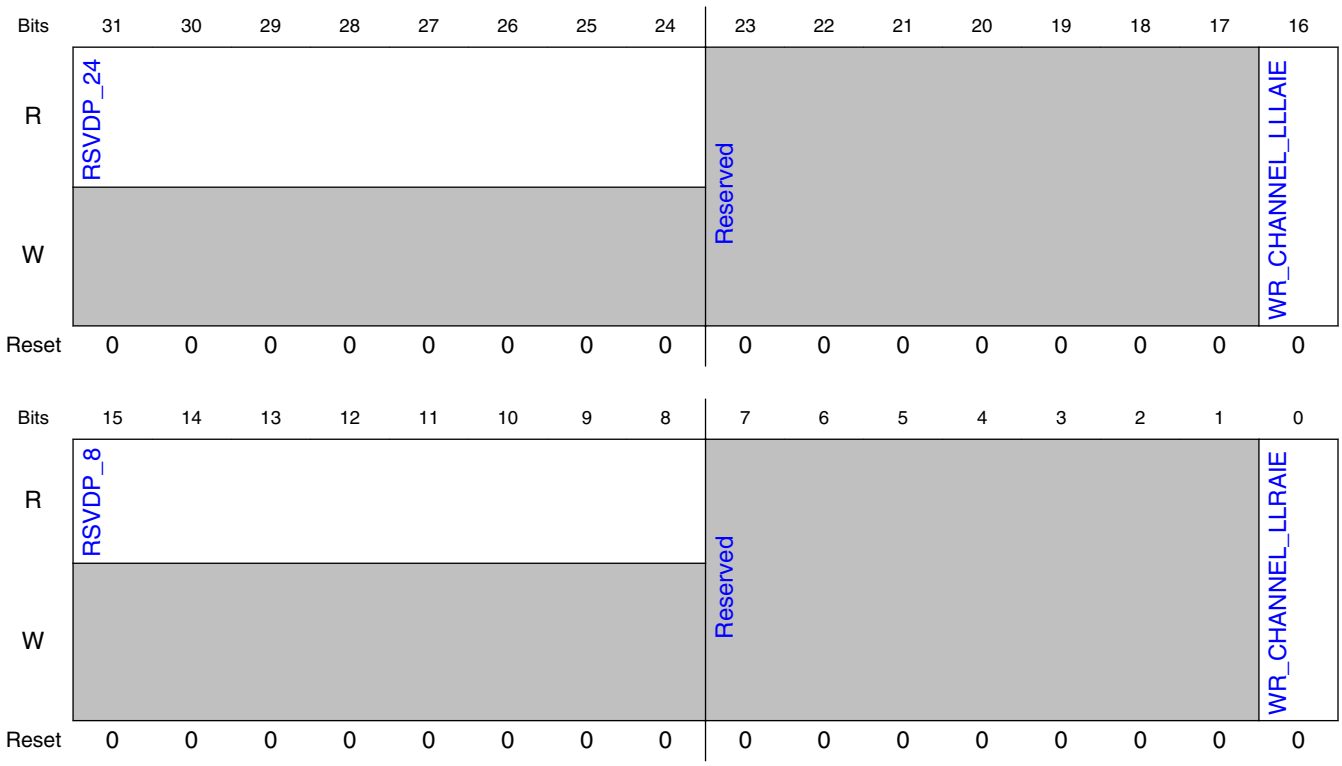
11.3.5.1.205.1 Offset

Register	Offset
DMA_WRITE_LINKED_LIST_ERR_EN_OFF	8008_0090h

11.3.5.1.205.2 Function

DMA Write Linked List Error Enable Register. The LIE and RIE bits in the LL element enable the channel "done" interrupts (local and remote). The LLLAIE and LLRAIE bits enable the channel "abort" interrupts (local and remote).

11.3.5.1.205.3 Diagram



11.3.5.1.205.4 Fields

Field	Function
31-24 RSVDP_24	Reserved for future use.
23-17 —	Reserved.
16 WR_CHANNEL_LLLAIE	Write Channel LL Local Abort Interrupt Enable (LLLAIE). You enable the write channel local abort interrupt through this bit. The LIE and RIE bits in the LL element enable the write channel done interrupts. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Used in linked list mode only. For more details, see "Interrupt Handling". Note: The access attributes of this field are as follows: - Dbi: R/W
15-8	Reserved for future use.

Table continues on the next page...

Field	Function
RSVDP_8	
7-1 —	Reserved.
0 WR_CHANNEL_LLRAIE	Write Channel LL Remote Abort Interrupt Enable (LLRAIE). You enable the write channel remote abort interrupt through this bit. The LIE and RIE bits in the LL element enable the write channel done interrupts. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Used in linked list mode only. For more details, see "Interrupt Handling". Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.206 DMA Read Interrupt Status Register. (DMA_READ_INT_STATUS_OFF)

11.3.5.1.206.1 Offset

Register	Offset
DMA_READ_INT_STATUS_OFF	8008_00A0h

11.3.5.1.206.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVDP_24								RD_ABORT_INT_STATUS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVDP_8								RD_DONE_INT_STATUS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.206.3 Fields

Field	Function
31-24 RSVDP_24	Reserved for future use.
23-16 RD_ABORT_INT_STATUS	Abort Interrupt Status. The DMA read channel has detected an error, or you manually stopped the transfer as described in "Stopping the DMA Transfer (Software Stop)". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. You can read the "DMA Read Error Status Low Register" (DMA_READ_ERR_STATUS_LOW_OFF) and "DMA Read Error Status High Register"

Table continues on the next page...

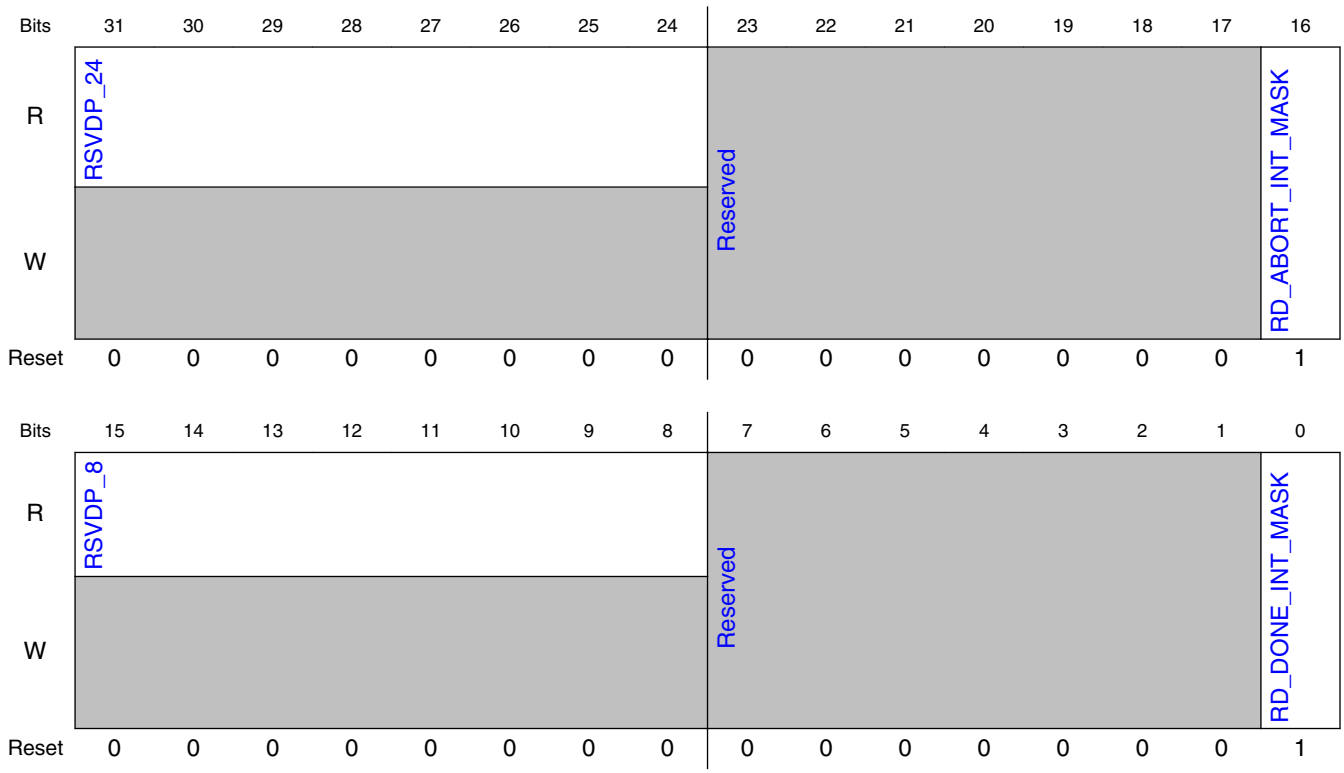
Field	Function
	(DMA_READ_ERR_STATUS_HIGH_OFF) to determine the source of the error. - Enabling: For details, see "Interrupts and Error Handling". - Masking: The DMA read interrupt Mask register has no effect on this register. - Clearing: You must write a 1'b1 to the corresponding channel bit in the DMA read interrupt Clear register to clear this interrupt bit. Note: You can write to this register to emulate interrupt generation, during software or hardware testing. A write to the address triggers an interrupt, but the DMA does not set the Done or Abort bits in this register. Note: The access attributes of this field are as follows: - Dbi: R/W
15-8 RSVDP_8	Reserved for future use.
7-0 RD_DONE_INT_STATUS	Done Interrupt Status. The DMA read channel has successfully completed the DMA read transfer. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. - Enabling: For details, see "Interrupts and Error Handling". - Masking: The DMA read interrupt Mask register has no effect on this register. - Clearing: You must write a 1'b1 to the corresponding channel bit in the DMA read interrupt Clear register to clear this interrupt bit. Note: You can write to this register to emulate interrupt generation, during software or hardware testing. A write to the address triggers an interrupt, but the DMA does not set the Done or Abort bits in this register. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.207 DMA Read Interrupt Mask Register. (DMA_READ_INT_MASK_OFF)

11.3.5.1.207.1 Offset

Register	Offset
DMA_READ_INT_MASK_OFF	8008_00A8h

11.3.5.1.207.2 Diagram



11.3.5.1.207.3 Fields

Field	Function
31-24 RSVDP_24	Reserved for future use.
23-17 —	Reserved.
16 RD_ABORT_INT_MASK	Abort Interrupt Mask. Prevents the Abort interrupt status field in the DMA read interrupt status register from asserting the edma_int output. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: The access attributes of this field are as follows: - Dbi: R/W
15-8 RSVDP_8	Reserved for future use.
7-1 —	Reserved.
0 RD_DONE_INT_MASK	Done Interrupt Mask. Prevents the Done interrupt status field in the DMA read interrupt status register from asserting the edma_int output. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.208 DMA Read Interrupt Clear Register. (DMA_READ_INT_CLEAR_OFF)

11.3.5.1.208.1 Offset

Register	Offset
DMA_READ_INT_CLEAR_OFF	8008_00ACh

11.3.5.1.208.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVDP_24															
W									RD_ABORT_INT_CLEAR							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVDP_8															
W									RD_DONE_INT_CLEAR							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.208.3 Fields

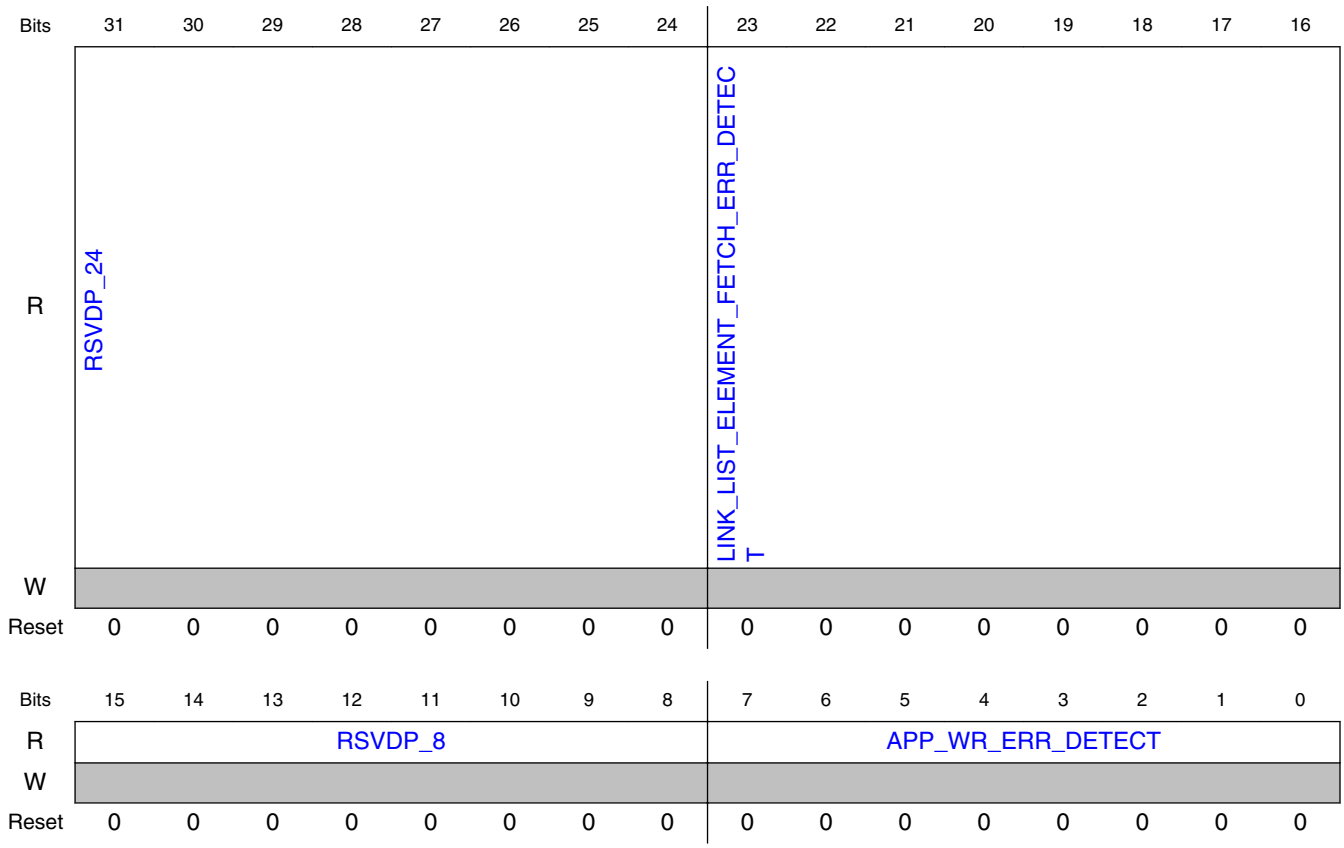
Field	Function
31-24 RSVDP_24	Reserved for future use.
23-16 RD_ABORT_INT_CLEAR	Abort Interrupt Clear. You must write a 1'b1 to clear the corresponding bit in the Abort interrupt status field of the DMA read interrupt status register. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: Reading from this self-clearing register field always returns a "0".
15-8 RSVDP_8	Reserved for future use.
7-0 RD_DONE_INT_CLEAR	Done Interrupt Clear. You must write a 1'b1 to clear the corresponding bit in the Done interrupt status field of the DMA read interrupt status register. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Note: Reading from this self-clearing register field always returns a "0".

11.3.5.1.209 DMA Read Error Status Low Register. (DMA_READ_ERR_STATUS_LOW_OFF)

11.3.5.1.209.1 Offset

Register	Offset
DMA_READ_ERR_STATUS_LOW_OFF	8008_00B4h

11.3.5.1.209.2 Diagram



11.3.5.1.209.3 Fields

Field	Function
31-24 RSVDP_24	Reserved for future use.
23-16 LINK_LIST_ELEMENT_FETCH_ERR_DETECT	Linked List Element Fetch Error Detected. - The DMA read channel has received an error response from the AXI bus while reading a linked list element from local memory. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. - Enabling: For details, see "Interrupts and Error Handling". - Masking: The DMA read interrupt Mask register has no effect on this register. - Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the "DMA Read Interrupt Clear Register" (DMA_READ_INT_CLEAR_OFF) to clear this error bit. Note, this clears all bits in this register, and also the DMA Read Error Status High register (DMA_READ_ERR_STATUS_HIGH_OFF).

Table continues on the next page...

Field	Function
15-8 RSVDP_8	Reserved for future use.
7-0 APP_WR_ERR_DETECT	Application Write Error Detected. The DMA read channel has received an error response from the AXI bus (or TRGT1 interface when the AXI Bridge is not used) while writing data to it. This error is fatal. You must restart the transfer from the beginning, as the channel context is corrupted, and the transfer is not rolled back. For more details, see "Linked List Mode". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. - Enabling: For details, see "Interrupts and Error Handling". - Masking: The DMA read interrupt Mask register has no effect on this register. - Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the "DMA Read Interrupt Clear Register" (DMA_READ_INT_CLEAR_OFF) to clear this error bit. Note, this clears all bits in this register, and also the DMA Read Error Status High register (DMA_READ_ERR_STATUS_HIGH_OFF).

11.3.5.1.210 DMA Read Error Status High Register. (DMA_READ_ERR_STATUS_HIGH_OFF)

11.3.5.1.210.1 Offset

Register	Offset
DMA_READ_ERR_STATUS_HIGH_OFF	8008_00B8h

11.3.5.1.210.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATA_POISIONING								CPL_TIMEOUT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CPL_ABORT								UNSUPPORTED_REQ							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.210.3 Fields

Field	Function
31-24 DATA_POISIONING	Data Poisoning. The DMA read channel has detected data poisoning in the completion from the remote device (in response to the MRd request). The DMA read channel will drop the completion and then be halted. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. - Enabling: For details, see "Interrupts and Error Handling". - Masking: The DMA read interrupt Mask register has no effect on this register. - Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field

Table continues on the next page...

Field	Function
	of the "DMA Read Interrupt Clear Register" (DMA_READ_INT_CLEAR_OFF) to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
23-16 CPL_TIMEOUT	Completion Time Out. The DMA read channel has timed-out while waiting for the remote device to respond to the MRd request, or a malformed CplD has been received. For more details, see "Linked List Mode". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. - Enabling: For details, see "Interrupts and Error Handling" . - Masking: The DMA read interrupt Mask register has no effect on this register. - Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the "DMA Read Interrupt Clear Register" (DMA_READ_INT_CLEAR_OFF) to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
15-8 CPL_ABORT	Completer Abort. The DMA read channel has received a PCIe completer abort completion status from the remote device in response to the MRd request. For more details, see "Linked List Mode". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. - Enabling: For details, see "Interrupts and Error Handling". - Masking: The DMA read interrupt Mask register has no effect on this register. - Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the "DMA Read Interrupt Clear Register" (DMA_READ_INT_CLEAR_OFF) to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.
7-0 UNSUPPORTED_REQ	Unsupported Request. The DMA read channel has received a PCIe unsupported request completion status from the remote device in response to the MRd request. For more details, see "Linked List Mode". Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. - Enabling: For details, see "Interrupts and Error Handling". - Masking: The DMA read interrupt Mask register has no effect on this register. - Clearing: You must write a 1'b1 to the corresponding channel bit in the Abort interrupt field of the "DMA Read Interrupt Clear Register" (DMA_READ_INT_CLEAR_OFF) to clear this error bit. Note, this also clears the other error bits for the same channel in this register and in the DMA Read Error Status Low register.

11.3.5.1.211 DMA Read Linked List Error Enable Register. (DMA_READ_LINKED_LIST_ERR_EN_OFF)

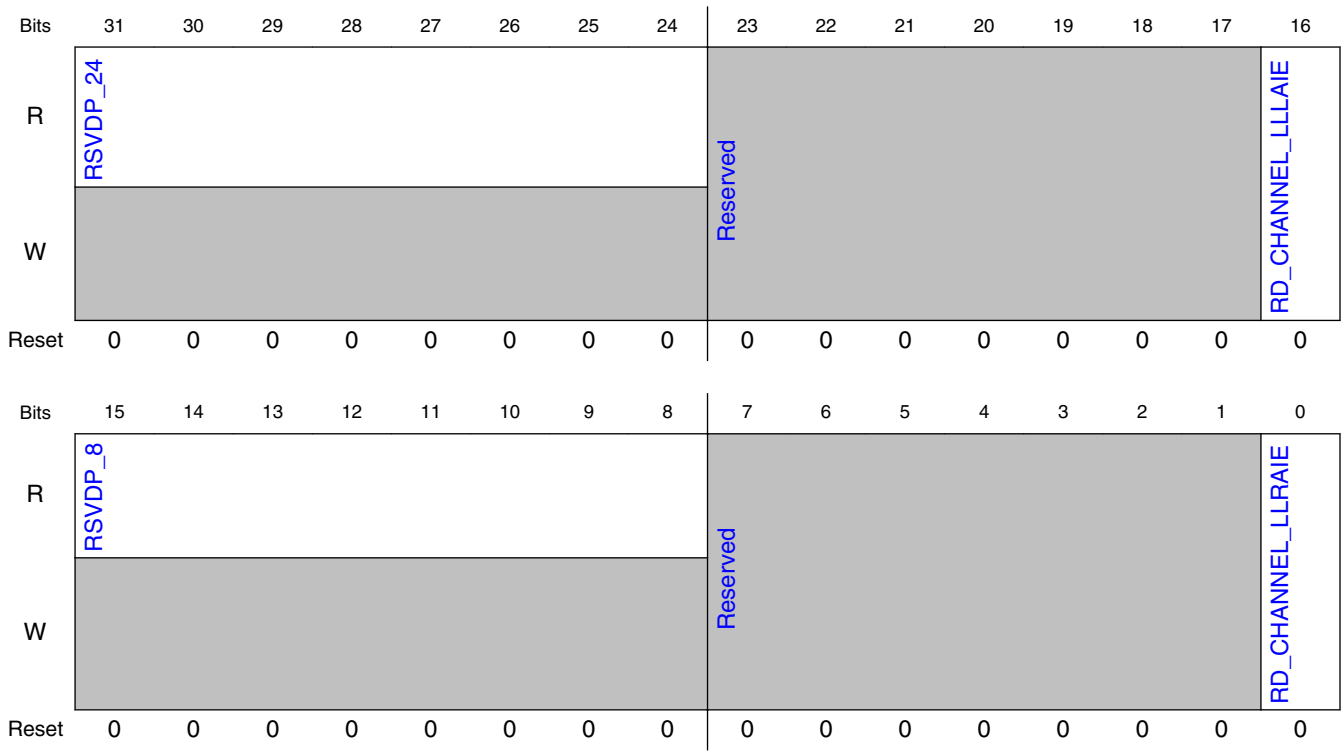
11.3.5.1.211.1 Offset

Register	Offset
DMA_READ_LINKED_LIST_ERR_EN_OFF	8008_00C4h

11.3.5.1.211.2 Function

DMA Read Linked List Error Enable Register. The LIE and RIE bits in the LL element enable the channel "done" interrupts (local and remote). The LLLAIE and LLRAIE bits enable the channel "abort" interrupts (local and remote).

11.3.5.1.211.3 Diagram



11.3.5.1.211.4 Fields

Field	Function
31-24 RSVDP_24	Reserved for future use.
23-17 —	Reserved.
16 RD_CHANNEL_LLLAIE	Read Channel LL Local Abort Interrupt Enable (LLLAIE). You enable the read channel Local Abort interrupt through this bit. The LIE and RIE bits in the LL element enable the read channel done interrupts. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Used in linked list mode only. For more details, see "Interrupt Handling". Note: The access attributes of this field are as follows: - Dbi: R/W
15-8 RSVDP_8	Reserved for future use.
7-1 —	Reserved.
0 RD_CHANNEL_LLRAIE	Read Channel LL Remote Abort Interrupt Enable (LLRAIE). You enable the read channel Remote Abort interrupt through this bit. The LIE and RIE bits in the LL element enable the read channel done interrupts. Each bit corresponds to a DMA channel. Bit [0] corresponds to channel 0. Used in linked list mode only. For more details, see "Interrupt Handling". Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.212 DMA Read Done IMWr Address Low Register. (DMA_READ_DONE_IMWR_LOW_OFF)

11.3.5.1.212.1 Offset

Register	Offset
DMA_READ_DONE_IMWR_LOW_OFF	8008_00CCh

11.3.5.1.212.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_READ_DONE_LOW_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_READ_DONE_LOW_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.212.3 Fields

Field	Function
31-0 DMA_READ_DONE_LOW_REG	The DMA uses this field to generate bits [31:0] of the address field for the Done IMWr TLP. Bits [1:0] must be "00" as this address must be dword aligned. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.213 DMA Read Done IMWr Address High Register. (DMA_READ_DONE_IMWR_HIGH_OFF)

11.3.5.1.213.1 Offset

Register	Offset
DMA_READ_DONE_IMWR_HIGH_OFF	8008_00D0h

11.3.5.1.213.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_READ_DONE_HIGH_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_READ_DONE_HIGH_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.213.3 Fields

Field	Function
31-0 DMA_READ_DONE_HIGH_REG	The DMA uses this field to generate bits [63:32] of the address field for the Done IMWr TLP. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.214 DMA Read Abort IMWr Address Low Register. (DMA_READ_ABORT_IMWR_LOW_OFF)

11.3.5.1.214.1 Offset

Register	Offset
DMA_READ_ABORT_IMWR_LOW_OFF	8008_00D4h

11.3.5.1.214.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_READ_ABORT_LOW_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_READ_ABORT_LOW_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.214.3 Fields

Field	Function
31-0 DMA_READ_ABORT_LOW_REG	The DMA uses this field to generate bits [31:0] of the address field for the Abort IMWr TLP. Bits [1:0] must be "00" as this address must be dword aligned. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.215 DMA Read Abort IMWr Address High Register. (DMA_READ_ABORT_IMWR_HIGH_OFF)

11.3.5.1.215.1 Offset

Register	Offset
DMA_READ_ABORT_IMWR_HIGH_OFF	8008_00D8h

11.3.5.1.215.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_READ_ABORT_HIGH_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_READ_ABORT_HIGH_REG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.215.3 Fields

Field	Function
31-0 DMA_READ_ABORT_HIGH_REG	The DMA uses this field to generate bits [63:32] of the address field for the Abort IMWr TLP. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.216 DMA Read Channel 1 and 0 IMWr Data Register. (DMA_READ_CH01_IMWR_DATA_OFF)

11.3.5.1.216.1 Offset

Register	Offset
DMA_READ_CH01_IMWR_DATA_OFF	8008_00DCh

11.3.5.1.216.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RD_CHANNEL_1_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RD_CHANNEL_0_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.216.3 Fields

Field	Function
31-16 RD_CHANNEL_1_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 1. Note: The access attributes of this field are as follows: - Dbi: R/W
15-0 RD_CHANNEL_0_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 0. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.217 DMA Read Channel 3 and 2 IMWr Data Register. (DMA_READ_CH23_IMWR_DATA_OFF)

11.3.5.1.217.1 Offset

Register	Offset
DMA_READ_CH23_IMWR_DATA_OFF	8008_00E0h

11.3.5.1.217.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RD_CHANNEL_3_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RD_CHANNEL_2_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.217.3 Fields

Field	Function
31-16 RD_CHANNEL_3_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 3. Note: The access attributes of this field are as follows: - Dbi: R/W
15-0 RD_CHANNEL_2_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 2. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.218 DMA Read Channel 5 and 4 IMWr Data Register. (DMA_READ_CH45_IMWR_DATA_OFF)

11.3.5.1.218.1 Offset

Register	Offset
DMA_READ_CH45_IMWR_DATA_OFF	8008_00E4h

11.3.5.1.218.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RD_CHANNEL_5_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RD_CHANNEL_4_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.218.3 Fields

Field	Function
31-16 RD_CHANNEL_5_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 5. Note: The access attributes of this field are as follows: - Dbi: R/W
15-0 RD_CHANNEL_4_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 4. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.219 DMA Read Channel 7 and 6 IMWr Data Register. (DMA_READ_CH67_IMWR_DATA_OFF)

11.3.5.1.219.1 Offset

Register	Offset
DMA_READ_CH67_IMWR_DATA_OFF	8008_00E8h

11.3.5.1.219.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RD_CHANNEL_7_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RD_CHANNEL_6_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.219.3 Fields

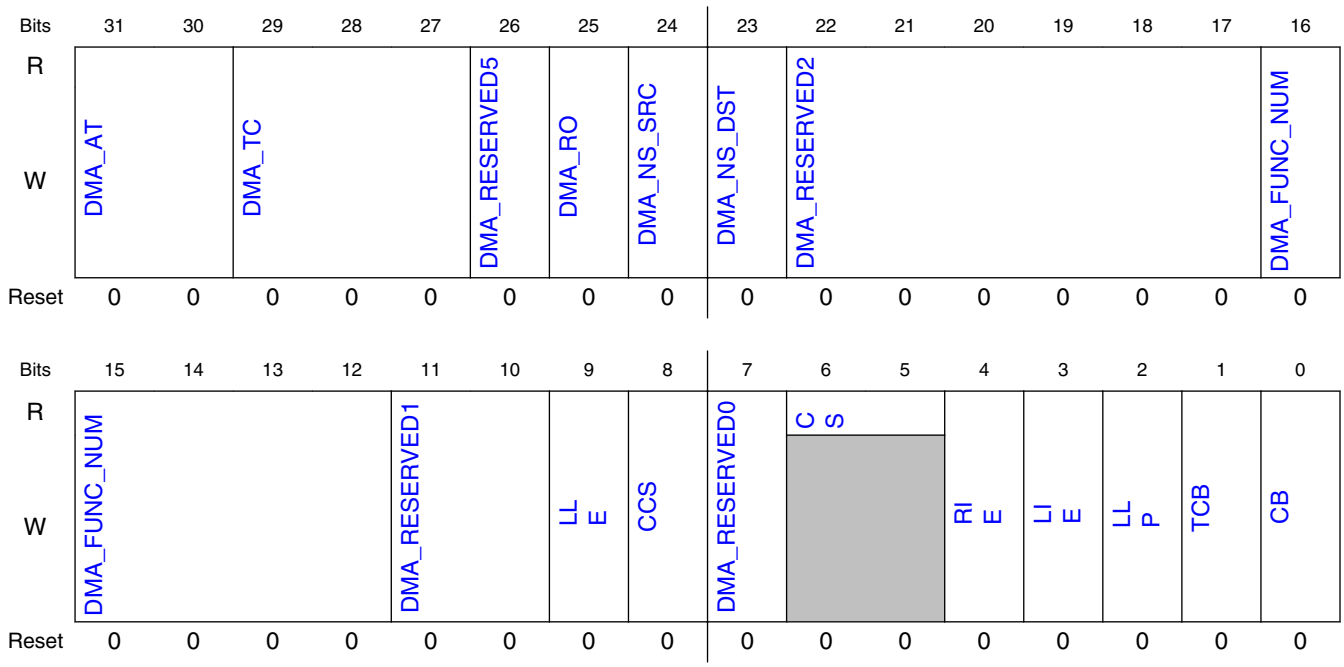
Field	Function
31-16 RD_CHANNEL_7_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 7. Note: The access attributes of this field are as follows: - Dbi: R/W
15-0 RD_CHANNEL_6_DATA	The DMA uses this field to generate the data field for the Done or Abort IMWr TLPs it generates for read channel 6. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.220 DMA Write Channel Control 1 Register. (DMA_CH_CONTROL1_OFF_WRCH_0)

11.3.5.1.220.1 Offset

Register	Offset
DMA_CH_CONTROL1_OFF_WRCH_0	8008_0200h

11.3.5.1.220.2 Diagram



11.3.5.1.220.3 Fields

Field	Function
31-30 DMA_AT	Address Translation TLP Header Bit (AT) The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Dbi: R/W
29-27 DMA_TC	Traffic Class TLP Header Bit (TC) The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Dbi: R/W
26 DMA_RESERVED5	Reserved. Note: The access attributes of this field are as follows: - Dbi: R/W
25 DMA_RO	Relaxed Ordering TLP Header Bit (RO) The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Dbi: R/W
24 DMA_NS_SRC	Source No Snoop TLP Header Bit (DMA_NS_SRC). The DMA uses this TLP header field when generating MRd (SAR addressing space) (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Dbi: R/W
23 DMA_NS_DST	Destination No Snoop TLP Header Bit (DMA_NS_DST). The DMA uses this TLP header field when generating MWr (DAR addressing space) (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Dbi: R/W
22-17 DMA_RESERVED2	Reserved. Note: The access attributes of this field are as follows: - Dbi: R/W
16-12	Function Number (FN). The controller uses this field when generating the requester ID for the MRd/MWr DMA TLP. When you have enabled SR-IOV, then this field is ignored if you have set the VFE field in the

Table continues on the next page...

Field	Function
DMA_FUNC_NUM	"DMA Write Channel Control 2 Register" (DMA_CH_CONTROL2_OFF_WRCH_0). Note: The access attributes of this field are as follows: - Dbi: R/W
11-10 DMA_RESERVED1	Reserved. Note: The access attributes of this field are as follows: - Dbi: R/W
9 LLE	Linked List Enable (LLE). - 0: Disable linked list operation - 1: Enable linked list operation Note: The access attributes of this field are as follows: - Dbi: R/W
8 CCS	Consumer Cycle State (CCS). Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". You must initialize this bit. The DMA updates this bit during linked list operation. Note: The access attributes of this field are as follows: - Dbi: R/W
7 DMA_RESERVED0	Reserved. Note: The access attributes of this field are as follows: - Dbi: R/W
6-5 CS	Channel Status (CS). The channel status bits identify the current operational state of the DMA channel. The operation state encoding for each DMA channel is a s follows: - 00: Reserved - 01: Running. This channel is active and transferring data. - 10: Halted. An error condition has been detected, and the DMA has stopped this channel. - 11: Stopped. The DMA has transferred all data for this channel, or you have prematurely stopped this channel by writing to the Stop field of the "DMA Write Doorbell Register" (DMA_WRITE_DOORBELL_OFF) or "DMA Read Doorbell Register" (DMA_READ_DOORBELL_OFF).
4 RIE	Remote Interrupt Enable (RIE). You must set this bit to enable the generation of the Done or Abort Remote interrupts. For more details, see "Interrupts and Error Handling". In LL mode, the DMA overwrites this with the RIE of the LL element. The RIE bit in a LL element only enables the Done interrupt. In non-LL mode, the RIE bit enables the Done and Abort interrupts. This field is not defined in a link LL element. Note: The access attributes of this field are as follows: - Dbi: R/W
3 LIE	Local Interrupt Enable (LIE). You must set this bit to enable the generation of the Done or Abort Local interrupts. For more details, see "Interrupts and Error Handling". In LL mode, the DMA overwrites this with the LIE of the LL element. The LIE bit in a LL element only enables the Done interrupt. In non-LL mode, the LIE bit enables the Done and Abort interrupts. This field is not defined in a link LL element. Note: The access attributes of this field are as follows: - Dbi: R/W
2 LLP	Load Link Pointer (LLP). Used in linked list mode only. Indicates that this linked list element is a link element, and its LL element pointer dwords are pointing to the next (non-contiguous) element. The DMA loads this field with the LLP of the linked list element. Note: The access attributes of this field are as follows: - Dbi: R/W
1 TCB	Toggle Cycle Bit (TCB). Indicates to the DMA to toggle its interpretation of the CB. Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". The DMA loads this field with the TCB of the linked list element. this field is not defined in a data LL element. Note: The access attributes of this field are as follows: - Dbi: R/W
0 CB	Cycle Bit (CB). Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". The DMA loads this field with the CB of the linked list element. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.221 DMA Write Transfer Size Register. (DMA_TRANSFER_SIZE_OFF_WRCH_0)

11.3.5.1.221.1 Offset

Register	Offset
DMA_TRANSFER_SIZE_OFF_WRCH_0	8008_0208h

11.3.5.1.221.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_TRANSFER_SIZE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_TRANSFER_SIZE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.221.3 Fields

Field	Function
31-0 DMA_TRANSFER_SIZE	DMA Transfer Size. You program this register with the size of the DMA transfer. The maximum DMA transfer size is 4Gbytes. The minimum transfer size is one byte (0x1). This field is automatically decremented by the DMA as the DMA write channel transfer progresses. This field indicates the number bytes remaining to be transferred. When all bytes are successfully transferred the current transfer size is zero. In LL mode, the DMA overwrites this register with the corresponding dword of the LL element. You can read this register to monitor the transfer progress, however in some scenarios this register is updated after a delay. For example, when less than 3 channels are doorbelled, this register is updated only after a descriptor finishes(linked list mode), or the transfer ends (non-linked list mode). Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.222 DMA Write SAR Low Register. (DMA_SAR_LOW_OFF_WRCH_0)

11.3.5.1.222.1 Offset

Register	Offset
DMA_SAR_LOW_OFF_WRCH_0	8008_020Ch

11.3.5.1.222.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SRC_ADDR_REG_LOW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SRC_ADDR_REG_LOW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.222.3 Fields

Field	Function
31-0 SRC_ADDR_REG_LOW	Source Address Register (Lower 32 bits). Indicates the next address to be read from. The DMA increments the SAR as the DMA transfer progresses. In LL mode, the DMA overwrites this with the corresponding dword of the LL element. - DMA Read: The SAR is the address of the remote memory. - DMA Write: The SAR is the address of the local memory. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.223 DMA Write SAR High Register. (DMA_SAR_HIGH_OFF_WRCH_0)

11.3.5.1.223.1 Offset

Register	Offset
DMA_SAR_HIGH_OFF_WRCH_0	8008_0210h

11.3.5.1.223.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SRC_ADDR_REG_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SRC_ADDR_REG_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.223.3 Fields

Field	Function
31-0 SRC_ADDR_REG_HIGH	Source Address Register (Higher 32 bits). In LL mode, the DMA overwrites this with the corresponding dword of the LL element. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.224 DMA Write DAR Low Register. (DMA_DAR_LOW_OFF_WRCH_0)

11.3.5.1.224.1 Offset

Register	Offset
DMA_DAR_LOW_OFF_WRCH_0	8008_0214h

11.3.5.1.224.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DST_ADDR_REG_LOW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DST_ADDR_REG_LOW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.224.3 Fields

Field	Function
31-0 DST_ADDR_REG_LOW	Destination Address Register (Lower 32 bits). Indicates the next address to be written to. The DMA increments the DAR as the DMA transfer progresses. In LL mode, the DMA overwrites this with the corresponding dword of the LL element. - DMA Read: The DAR is the address of the local memory. - DMA Write: The DAR is the address of the remote memory. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.225 DMA Write DAR High Register. (DMA_DAR_HIGH_OFF_WRCH_0)

11.3.5.1.225.1 Offset

Register	Offset
DMA_DAR_HIGH_OFF_WRCH_0	8008_0218h

11.3.5.1.225.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DST_ADDR_REG_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DST_ADDR_REG_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.225.3 Fields

Field	Function
31-0 DST_ADDR_REG_HIGH	Destination Address Register (Higher 32 bits). In LL mode, the DMA overwrites this with the corresponding dword of the LL element. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.226 DMA Write Linked List Pointer Low Register. (DMA_LLP_LOW_OFF_WRCH_0)

11.3.5.1.226.1 Offset

Register	Offset
DMA_LLP_LOW_OFF_WRCH_0	8008_021Ch

11.3.5.1.226.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LLP_LOW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LLP_LOW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.226.3 Fields

Field	Function
31-0 LLP_LOW	Lower bits of the address of the linked list transfer list in local memory. Used in linked list mode only. Updated by the DMA to point to the next element in the transfer list after the previous element is consumed. - When the current element is a data element; this field is incremented by 6. - When the current element is a link element; this field is overwritten by the LL Element Pointer of the element. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.227 DMA Write Linked List Pointer High Register. (DMA_LLP_HIGH_OFF_WRCH_0)

11.3.5.1.227.1 Offset

Register	Offset
DMA_LLP_HIGH_OFF_WRCH_0	8008_0220h

11.3.5.1.227.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LLP_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LLP_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.227.3 Fields

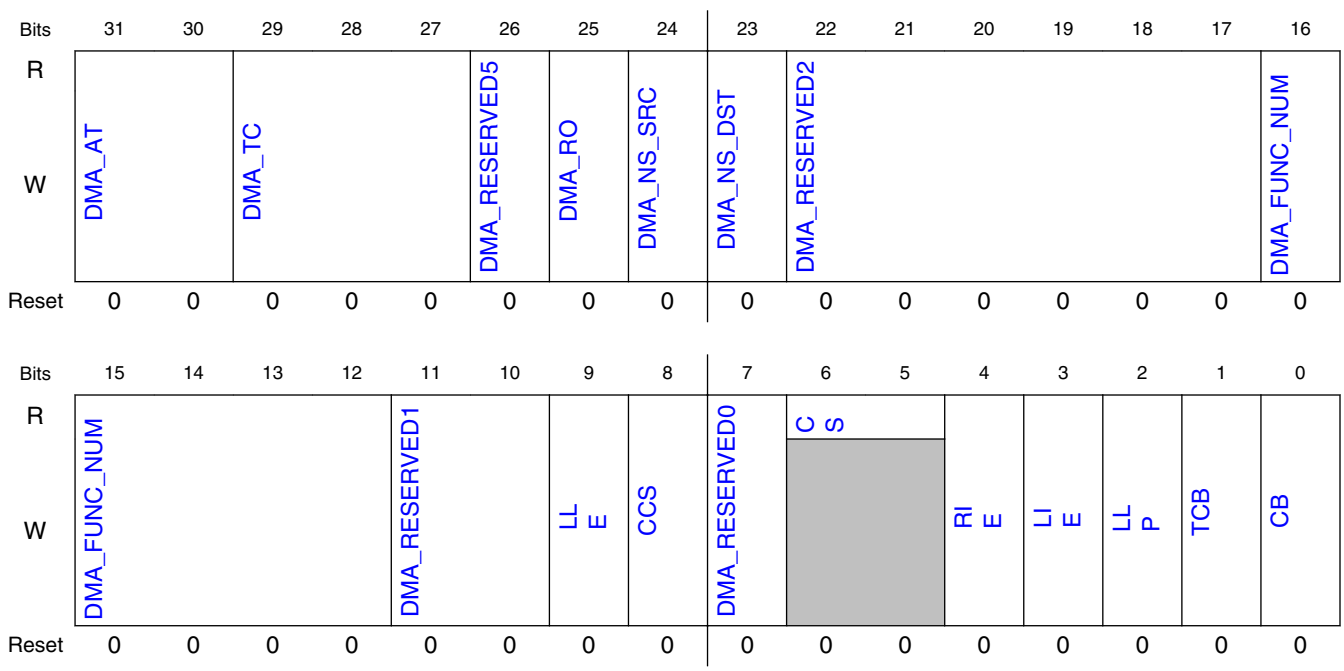
Field	Function
31-0 LLP_HIGH	Upper 32 bits of the address of the linked list transfer list in local memory. Used in linked list mode only. Updated by the DMA to point to the next element in the transfer list as elements are consumed. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.228 DMA Read Channel Control 1 Register. (DMA_CH_CONTROL1_OFF_RDCH_0)

11.3.5.1.228.1 Offset

Register	Offset
DMA_CH_CONTROL1_OFF_RDCH_0	8008_0300h

11.3.5.1.228.2 Diagram



11.3.5.1.228.3 Fields

Field	Function
31-30 DMA_AT	Address Translation TLP Header Bit (AT) The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Dbi: R/W
29-27 DMA_TC	Traffic Class TLP Header Bit (TC) The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Dbi: R/W
26 DMA_RESERVED5	Reserved. Note: The access attributes of this field are as follows: - Dbi: R/W
25 DMA_RO	Relaxed Ordering TLP Header Bit (RO) The DMA uses this TLP header field when generating MRd/MWr (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Dbi: R/W
24 DMA_NS_SRC	Source No Snoop TLP Header Bit (DMA_NS_SRC). The DMA uses this TLP header field when generating MRd (SAR addressing space) (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Dbi: R/W
23 DMA_NS_DST	Destination No Snoop TLP Header Bit (DMA_NS_DST). The DMA uses this TLP header field when generating MWr (DAR addressing space) (not IMWr) TLPs. Note: The access attributes of this field are as follows: - Dbi: R/W
22-17 DMA_RESERVED2	Reserved. Note: The access attributes of this field are as follows: - Dbi: R/W
16-12 DMA_FUNC_NUM	Function Number (FN). The controller uses this field when generating the requester ID for the MRd/MWr DMA TLP. When you have enabled SR-IOV, then this field is ignored if you have set the VFE field in the "DMA Read Channel Control 2 Register" (DMA_CH_CONTROL2_OFF_RDCH_0). Note: The access attributes of this field are as follows: - Dbi: R/W
11-10 DMA_RESERVED1	Reserved. Note: The access attributes of this field are as follows: - Dbi: R/W
9 LLE	Linked List Enable (LLE). - 0: Disable linked list operation - 1: Enable linked list operation Note: The access attributes of this field are as follows: - Dbi: R/W
8 CCS	Consumer Cycle State (CCS). Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". You must initialize this bit. The DMA updates this bit during linked list operation. Note: The access attributes of this field are as follows: - Dbi: R/W
7 DMA_RESERVED0	Reserved. Note: The access attributes of this field are as follows: - Dbi: R/W
6-5 CS	Channel Status (CS). The channel status bits identify the current operational state of the DMA channel. The operation state encoding for each DMA channel is a s follows: - 00: Reserved - 01: Running. This channel is active and transferring data. - 10: Halted. An error condition has been detected, and the DMA has stopped this channel. - 11: Stopped. The DMA has transferred all data for this channel, or you have prematurely stopped this channel by writing to the Stop field of the "DMA Read Doorbell Register" (DMA_WRITE_DOORBELL_OFF) or "DMA Read Doorbell Register" (DMA_READ_DOORBELL_OFF).
4 RIE	Remote Interrupt Enable (RIE). You must set this bit to enable the generation of the Done or Abort Remote interrupts. For more details, see "Interrupts and Error Handling". In LL mode, the DMA overwrites this with the RIE of the LL element. The RIE bit in a LL element only enables the Done interrupt. In non-LL mode, the RIE bit enables the Done and Abort interrupts. This field is not defined in a link LL element. Note: The access attributes of this field are as follows: - Dbi: R/W

Table continues on the next page...

Field	Function
3 LIE	Local Interrupt Enable (LIE). You must set this bit to enable the generation of the Done or Abort Local interrupts. For more details, see "Interrupts and Error Handling". In LL mode, the DMA overwrites this with the LIE of the LL element. The LIE bit in a LL element only enables the Done interrupt. In non-LL mode, the LIE bit enables the Done and Abort interrupts. This field is not defined in a link LL element. Note: The access attributes of this field are as follows: - Dbi: R/W
2 LLP	Load Link Pointer (LLP). Used in linked list mode only. Indicates that this linked list element is a link element, and its LL element pointer dwords are pointing to the next (non-contiguous) element. The DMA loads this field with the LLP of the linked list element. Note: The access attributes of this field are as follows: - Dbi: R/W
1 TCB	Toggle Cycle Bit (TCB). Indicates to the DMA to toggle its interpretation of the CB. Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". The DMA loads this field with the TCB of the linked list element. this field is not defined in a data LL element. Note: The access attributes of this field are as follows: - Dbi: R/W
0 CB	Cycle Bit (CB). Used in linked list mode only. It is used to synchronize the producer (software) and the consumer (DMA). For more details, see "PCS-CCS-CB-TCB Producer-Consumer Synchronization". The DMA loads this field with the CB of the linked list element. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.229 DMA Read Transfer Size Register. (DMA_TRANSFER_SIZE_OFF_RDCH_0)

11.3.5.1.229.1 Offset

Register	Offset
DMA_TRANSFER_SIZE_OFF_RDCH_0	8008_0308h

11.3.5.1.229.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_TRANSFER_SIZE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_TRANSFER_SIZE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.229.3 Fields

Field	Function
31-0 DMA_TRANSFER_SIZE	DMA Transfer Size. You program this register with the size of the DMA transfer. The maximum DMA transfer size is 4Gbytes. The minimum transfer size is one byte (0x1). This field is automatically decremented by the DMA as the DMA read channel transfer progresses. This field indicates the number bytes remaining to be transferred. When all bytes are successfully transferred the current transfer size is zero. In LL mode, the DMA overwrites this register with the corresponding dword of the LL element. You can read this register to monitor the transfer progress, however in some scenarios this register is updated after a delay. For example, when less than 3 channels are doorbelled, this register is updated only after a descriptor finishes(linked list mode), or the transfer ends (non-linked list mode). Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.230 DMA Read SAR Low Register. (DMA_SAR_LOW_OFF_RDCH_0)

11.3.5.1.230.1 Offset

Register	Offset
DMA_SAR_LOW_OFF_RDCH_0	8008_030Ch

11.3.5.1.230.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SRC_ADDR_REG_LOW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SRC_ADDR_REG_LOW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.230.3 Fields

Field	Function
31-0 SRC_ADDR_REG_LOW	Source Address Register (Lower 32 bits). Indicates the next address to be read from. The DMA increments the SAR as the DMA transfer progresses. In LL mode, the DMA overwrites this with the corresponding dword of the LL element. - DMA Read: The SAR is the address of the remote memory. - DMA Read: The SAR is the address of the local memory. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.231 DMA Read SAR High Register. (DMA_SAR_HIGH_OFF_RDCH_0)

11.3.5.1.231.1 Offset

Register	Offset
DMA_SAR_HIGH_OFF_RDCH_0	8008_0310h

11.3.5.1.231.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SRC_ADDR_REG_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SRC_ADDR_REG_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.231.3 Fields

Field	Function
31-0 SRC_ADDR_REG_HIGH	Source Address Register (Higher 32 bits). In LL mode, the DMA overwrites this with the corresponding dword of the LL element. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.232 DMA Read DAR Low Register. (DMA_DAR_LOW_OFF_RDCH_0)

11.3.5.1.232.1 Offset

Register	Offset
DMA_DAR_LOW_OFF_RDCH_0	8008_0314h

11.3.5.1.232.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DST_ADDR_REG_LOW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DST_ADDR_REG_LOW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.232.3 Fields

Field	Function
31-0 DST_ADDR_REG_LOW	Destination Address Register (Lower 32 bits). Indicates the next address to be written to. The DMA increments the DAR as the DMA transfer progresses. In LL mode, the DMA overwrites this with the corresponding dword of the LL element. - DMA Read: The DAR is the address of the local memory. - DMA Read: The DAR is the address of the remote memory. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.233 DMA Read DAR High Register. (DMA_DAR_HIGH_OFF_RDCH_0)

11.3.5.1.233.1 Offset

Register	Offset
DMA_DAR_HIGH_OFF_RDCH_0	8008_0318h

11.3.5.1.233.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DST_ADDR_REG_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DST_ADDR_REG_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.233.3 Fields

Field	Function
31-0 DST_ADDR_REG_HIGH	Destination Address Register (Higher 32 bits). In LL mode, the DMA overwrites this with the corresponding dword of the LL element. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.234 DMA Read Linked List Pointer Low Register. (DMA_LLP_LOW_OFF_RDCH_0)

11.3.5.1.234.1 Offset

Register	Offset
DMA_LLP_LOW_OFF_RDCH_0	8008_031Ch

11.3.5.1.234.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LLP_LOW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LLP_LOW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.234.3 Fields

Field	Function
31-0 LLP_LOW	Lower bits of the address of the linked list transfer list in local memory. Used in linked list mode only. Updated by the DMA to point to the next element in the transfer list after the previous element is consumed. - When the current element is a data element; this field is incremented by 6. - When the current element is a link element; this field is overwritten by the LL Element Pointer of the element. Note: The access attributes of this field are as follows: - Dbi: R/W

11.3.5.1.235 DMA Read Linked List Pointer High Register. (DMA_LLP_HIGH_OFF_RDCH_0)

11.3.5.1.235.1 Offset

Register	Offset
DMA_LLP_HIGH_OFF_RDCH_0	8008_0320h

11.3.5.1.235.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LLP_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LLP_HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.3.5.1.235.3 Fields

Field	Function
31-0 LLP_HIGH	Upper 32 bits of the address of the linked list transfer list in local memory. Used in linked list mode only. Updated by the DMA to point to the next element in the transfer list as elements are consumed. Note: The access attributes of this field are as follows: - Dbi: R/W

11.4 PCI Express PHY (PCIe_PHY)

11.4.1 Overview

The PCIe Gen4 PHY IP core is used for PCI-Express (PCIe) applications. The transceiver in the core performs these functions:

- Serializes the 8b/10b encoded data for transmission for Gen1 and Gen2 operation, and 128b/130b encoded data for Gen3 and Gen4
- De-serializes the received code groups

When transmitting the data, the transceiver performs these functions:

- Accepts four 10-bit 8b/10b encoded transmit characters or four 8-bit 128b/130b encoded data
- Attaches and serializes the data to the TXP/TXN differential outputs to a maximum value of 16.0 Gb/s

When receiving the data, the transceiver performs these functions:

- Samples the received serial data on the RXP/RXN differential inputs
- Deserializes it into four 10-bit(or 8b-bit) received characters

PCIe Gen4 PHY core has on-chip PLL circuitry for synthesis of the baud-rate transmitting clocks, and extraction of the retimed clocks from the received serial stream.

11.4.1.1 Features

The supported features in a Samsung PCIe Gen4 PHY IP core are:

- PCI Express Base Specification 4.0 compliance
- 2.5Gb/s, 5.0Gb/s, 8.0Gb/s and 16Gb/s Serializer/Deserializer
- PHY Interface for the PCI Express Architecture, Version 4.2 compliance
- Receiver Detection
- Spread Spectrum Clocking in Transmitter and Receiver
- Separate Refclk Independent SSC (SRIS) Architecture
- Continuous-Time Linear Equalizer and 5-tap adaptive Decision-Feedback Equalizer

11.4.1.2 Block Diagram

The figure below illustrates the block diagram of a 4-lane PCIe Gen4 PHY.

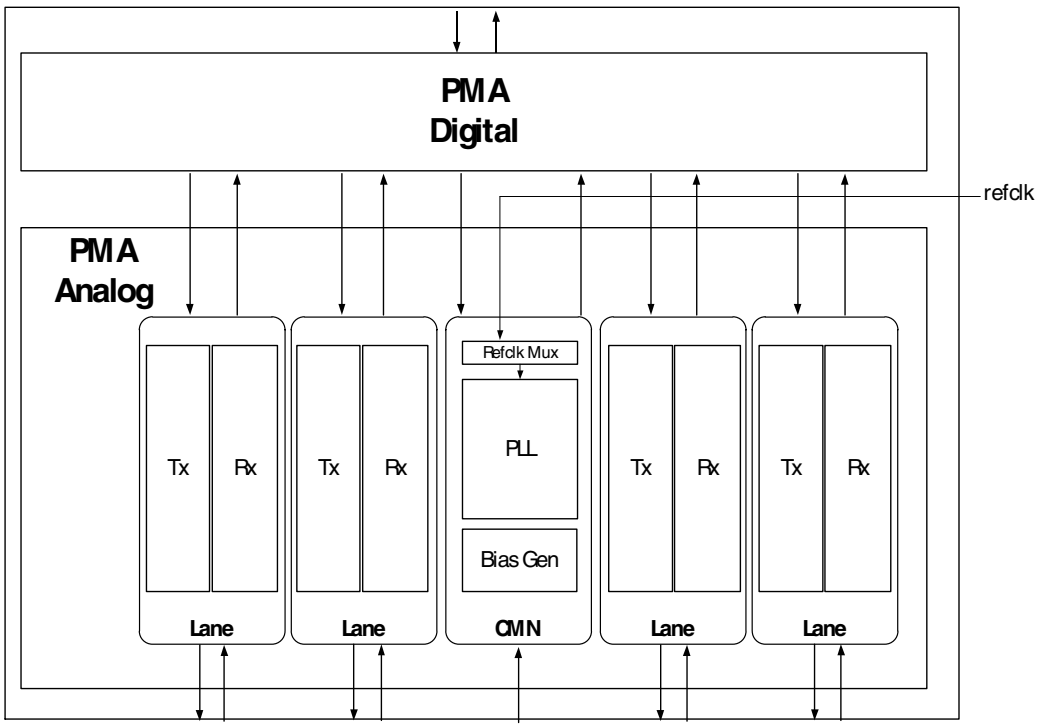


Figure 11-67. Block Diagram of PCIe Gen4 PH

The PCIe Gen4 PHY comprises of one Common, four lanes of transceiver, one external reference clock in/output buffer, and PMA digital.

11.4.2 Functional Description

The PCIe Gen4 PHY comprises of a PMA analog and PMA digital. The PMA consists of:

- One CMN block
- Four lane blocks including a full-duplex transceiver,
- One external reference clock input/output (bidirectional) buffer.

In the transmitter side, the PMA Analog executes these actions:

- PMA Analog receives parallel data from PMA digital.
- The TX driver characteristics, such as amplitude and 3-tap FIR filter, are both presettable and user- controllable.

In the receiver side, the PMA executes these actions:

- Distortion on received signal due to channel loss is compensated by Continuous Time Linear Equalizer (CTLE) and Decision Feedback Equalizer (DFE).
- Clock and Data Recovery (CDR) generates frequency- and phase-aligned clock and equalized signal is deserialized to parallel data which is delivered to the PHY digital.

11.4.2.1 Common (CMN)

The CMN consists of a band-gap reference (BGR), bias generator, PLL, and RX RCAL. Detailed information on each block is described in the following sections

11.4.2.1.1 BGR and bias generator

The BGR circuit generates high-accuracy reference voltage across PVT variation.

A bias generator makes two kinds of current, external- and internal-resistor (RMRES)-referred current, and these currents are distributed to each block in CMN and lanes. The external resistor-referred (REXT) current is generated using an 8.2KΩ resistor which is connected between the rext pad and ground outside the chip. On the other hand, the internal resistor-referred current is generated using internally integrated replica resistor to compensate resistance error across PVT variation.

11.4.2.1.2 Phase Locked Loop (PLL)

The PLL in the CMN synthesizes high-speed clock, which is used for TX serializer and RX CDR lock, from a reference clock. The reference clock can be selected from two clock sources; internal SoC reference clock and external differential reference clock.

In the PCIe Gen4 PHY, the high frequency clock from PLL is used in the serialization of TX data and CDR lock acquisition and maintenance. The following are the main features of the PLL:

- Supports wide-range LC VCO from 8GHz to 10GHz
- Supports pre-defined and programmable divider setting for each PCIe specification
- Supports spread spectrum clocking (SSC) with sigma-delta modulated fractional divider.
- Supports Automatic Frequency Calibrated (AFC) oscillator setting for the desired frequency of operation
- Supports programmable charge pump current and loop filter resistance allowing for wide range of programmable loop bandwidth and peaking

The PLL is based on a second-order system in dual-path (integral and proportional) structure.

There is a controllable third pole with an additional ripple suppression capacitor in the loop filter.

In initial phase, the frequency of the VCO is roughly calibrated by AFC function. During AFC, the loop is disconnected and control voltage of the VCO is tied to DC voltage. The VCO frequency is measured by a digital counter for every AFC code and an AFC logic stops when the measured digital number reaches a target number. After the AFC is completed, the loop is closed again and tries to lock frequency and phase.

The PLL has two gain paths, proportional and integral paths, to control the frequency and phase of VCO clock. Each path is driven by proportional or integral charge pump. The bandwidth of the PLL is mainly affected by proportional path gain, especially by the current of proportional charge-pump, the resistance in loop filter and the VCO gain.

11.4.2.1.3 RX resistance calibration (RCAL)

In order to meet the return-loss specification across the PVT variation, the resistance of termination circuit is automatically calibrated by a programmed logic sequence, which is called resistance calibration (RCAL)

Unlike TX RCAL, RX RCAL is implemented in the CMN and the RX in each lane shares the calibration code.

The PCIe Gen4 PHY performs RX RCAL at the start of power-up sequence in default. Although the result of RX RCAL is commonly used in each lane, the RX RTERM in each lane can be tuned with offset value.

11.4.2.2 Transmitter

The main features of the TX are provided below:

- Supports high voltage swing with auxiliary current-mode driver.
- Supports 3-tap FIR filter with resolution of 1/48.
- Supports 10 FIR presets (P0 to P9)
- Supports automatic TX termination resistance calibration.
- Supports electrical idle mode with either fast or low-power common-mode keep.
- Supports receiver detection.

11.4.2.2.1 Serializer

The serializer accepts 20-bit or 32-bit data from the PMA digital via 40-bit bus which is synchronized to TX byte clock (TBC). The serializer has 40-bit data input port in order to support bit-duplication (This is not used in normal operation). The parallel data is

converted to serial according to input bit-width and TBC frequency for each specification as shown in the following table. Note that the LSB in the input parallel data is transmitted first and the MSB the last.

Table 11-100. TX serializer input bit width and frequency

Protocol	Input bit width	TBC frequency
Gen1	40	62.5MHz
Gen2	40	125MHz
Gen3	32	250MHz
Gen4	32	500MHz

In order to support 3-tap FIR filtering in TX driver, the serializer gives not only the present bit but also both 1-bit preceding and 1-bit delayed bits to the pre-driver.

11.4.3 Memory Map and register definition

This section includes the PCIE_PHY module memory map and descriptions of the registers.

11.4.3.1 PCIe PHY register descriptions

11.4.3.1.1 PCIe_PHY Memory map

PCIE_PHY base address: 32F0_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	(CMN_REG000)	8	RW	00h
4h	(CMN_REG001)	8	RW	0Ah
8h	(CMN_REG002)	8	RW	00h
Ch	(CMN_REG003)	8	RW	15h
10h	(CMN_REG004)	8	RW	01h
14h	(CMN_REG005)	8	RW	F0h
18h	(CMN_REG006)	8	RW	08h
1Ch	(CMN_REG007)	8	RW	83h
20h	(CMN_REG008)	8	RW	06h
24h	(CMN_REG009)	8	RW	20h
28h	(CMN_REG00A)	8	RW	A9h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
2Ch	(CMN_REG00B)	8	RW	0Fh
30h	(CMN_REG00C)	8	RW	46h
34h	(CMN_REG00D)	8	RW	46h
38h	(CMN_REG00E)	8	RW	46h
3Ch	(CMN_REG00F)	8	RW	46h
40h	(CMN_REG010)	8	RW	3Fh
44h	(CMN_REG011)	8	RW	3Fh
48h	(CMN_REG012)	8	RW	24h
4Ch	(CMN_REG013)	8	RW	24h
50h	(CMN_REG014)	8	RW	FFh
54h	(CMN_REG015)	8	RW	FFh
58h	(CMN_REG016)	8	RW	77h
5Ch	(CMN_REG017)	8	RW	77h
60h	(CMN_REG018)	8	RW	24h
64h	(CMN_REG019)	8	RW	24h
68h	(CMN_REG01A)	8	RW	00h
6Ch	(CMN_REG01B)	8	RW	00h
70h	(CMN_REG01C)	8	RW	3Fh
74h	(CMN_REG01D)	8	RW	3Fh
78h	(CMN_REG01E)	8	RW	DDh
7Ch	(CMN_REG01F)	8	RW	AAh
80h	(CMN_REG020)	8	RW	11h
84h	(CMN_REG021)	8	RW	11h
88h	(CMN_REG022)	8	RW	1Eh
8Ch	(CMN_REG023)	8	RW	A8h
90h	(CMN_REG024)	8	RW	00h
94h	(CMN_REG025)	8	RW	00h
98h	(CMN_REG026)	8	RW	72h
9Ch	(CMN_REG027)	8	RW	00h
A0h	(CMN_REG028)	8	RW	00h
A4h	(CMN_REG029)	8	RW	FFh
A8h	(CMN_REG02A)	8	RW	FFh
ACh	(CMN_REG02B)	8	RW	00h
B0h	(CMN_REG02C)	8	RW	32h
B4h	(CMN_REG02D)	8	RW	32h
B8h	(CMN_REG02E)	8	RW	28h
BCh	(CMN_REG02F)	8	RW	28h
C0h	(CMN_REG030)	8	RW	32h
C4h	(CMN_REG031)	8	RW	32h
C8h	(CMN_REG032)	8	RW	28h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
CCh	(CMN_REG033)	8	RW	28h
D0h	(CMN_REG034)	8	RW	01h
D4h	(CMN_REG035)	8	RW	04h
D8h	(CMN_REG036)	8	RW	32h
DCh	(CMN_REG037)	8	RW	10h
E0h	(CMN_REG038)	8	RW	05h
E4h	(CMN_REG039)	8	RW	02h
E8h	(CMN_REG03A)	8	RW	00h
ECh	(CMN_REG03B)	8	RW	00h
F0h	(CMN_REG03C)	8	RW	00h
F4h	(CMN_REG03D)	8	RW	00h
F8h	(CMN_REG03E)	8	RW	00h
FCh	(CMN_REG03F)	8	RW	00h
100h	(CMN_REG040)	8	RW	00h
104h	(CMN_REG041)	8	RW	00h
108h	(CMN_REG042)	8	RW	00h
10Ch	(CMN_REG043)	8	RW	00h
110h	(CMN_REG044)	8	RW	00h
114h	(CMN_REG045)	8	RW	00h
118h	(CMN_REG046)	8	RW	00h
11Ch	(CMN_REG047)	8	RW	00h
120h	(CMN_REG048)	8	RW	00h
124h	(CMN_REG049)	8	RW	00h
128h	(CMN_REG04A)	8	RW	00h
12Ch	(CMN_REG04B)	8	RW	00h
130h	(CMN_REG04C)	8	RW	00h
134h	(CMN_REG04D)	8	RW	00h
138h	(CMN_REG04E)	8	RW	00h
13Ch	(CMN_REG04F)	8	RW	00h
140h	(CMN_REG050)	8	RW	00h
144h	(CMN_REG051)	8	RW	00h
148h	(CMN_REG052)	8	RW	00h
14Ch	(CMN_REG053)	8	RW	30h
150h	(CMN_REG054)	8	RW	30h
154h	(CMN_REG055)	8	RW	30h
158h	(CMN_REG056)	8	RW	30h
15Ch	(CMN_REG057)	8	RW	1Fh
160h	(CMN_REG058)	8	RW	1Fh
164h	(CMN_REG059)	8	RW	1Fh
168h	(CMN_REG05A)	8	RW	1Fh

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
16Ch	(CMN_REG05B)	8	RW	00h
170h	(CMN_REG05C)	8	RW	10h
174h	(CMN_REG05D)	8	RW	1Bh
178h	(CMN_REG05E)	8	RW	0Eh
17Ch	(CMN_REG05F)	8	RW	11h
180h	(CMN_REG060)	8	RW	23h
184h	(CMN_REG061)	8	RW	01h
188h	(CMN_REG062)	8	RW	00h
18Ch	(CMN_REG063)	8	RW	00h
190h	(CMN_REG064)	8	RW	0Ah
194h	(CMN_REG065)	8	RW	99h
198h	(CMN_REG066)	8	RW	C2h
19Ch	(CMN_REG067)	8	RW	00h
1A0h	(CMN_REG068)	8	RW	00h
1A4h	(CMN_REG069)	8	RW	00h
1A8h	(CMN_REG06A)	8	RW	05h
1ACh	(CMN_REG06B)	8	RW	10h
1B0h	(CMN_REG06C)	8	RW	14h
1B4h	(CMN_REG06D)	8	RW	00h
1B8h	(CMN_REG06E)	8	RW	00h
1BCh	(CMN_REG06F)	8	RW	00h
1C0h	(CMN_REG070)	8	RW	00h
1C4h	(CMN_REG071)	8	RW	00h
1C8h	(CMN_REG072)	8	RW	00h
1CCh	(CMN_REG073)	8	RW	00h
1D0h	(CMN_REG074)	8	RW	00h
1D4h	(CMN_REG075)	8	RW	00h
200h	(CMN_REG076)	8	RW	00h
204h	(CMN_REG077)	8	RW	1Fh
208h	(CMN_REG078)	8	RW	55h
20Ch	(CMN_REG079)	8	RW	00h
210h	(CMN_REG080)	8	RW	00h
214h	(CMN_REG081)	8	RW	18h
218h	(CMN_REG082)	8	RW	00h
400h	(TRSV_REG000)	8	RW	00h
404h	(TRSV_REG001)	8	RW	1Fh
408h	(TRSV_REG002)	8	RW	1Fh
40Ch	(TRSV_REG003)	8	RW	1Fh
410h	(TRSV_REG004)	8	RW	1Fh
414h	(TRSV_REG005)	8	RW	00h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
418h	(TRSV_REG006)	8	RW	00h
41Ch	(TRSV_REG007)	8	RW	00h
420h	(TRSV_REG008)	8	RW	00h
424h	(TRSV_REG009)	8	RW	00h
428h	(TRSV_REG00A)	8	RW	00h
42Ch	(TRSV_REG00B)	8	RW	05h
430h	(TRSV_REG00C)	8	RW	93h
434h	(TRSV_REG00D)	8	RW	00h
438h	(TRSV_REG00E)	8	RW	55h
43Ch	(TRSV_REG00F)	8	RW	00h
440h	(TRSV_REG010)	8	RW	00h
444h	(TRSV_REG011)	8	RW	00h
448h	(TRSV_REG012)	8	RW	01h
44Ch	(TRSV_REG013)	8	RW	77h
450h	(TRSV_REG014)	8	RW	77h
454h	(TRSV_REG015)	8	RW	77h
458h	(TRSV_REG016)	8	RW	77h
45Ch	(TRSV_REG017)	8	RW	00h
460h	(TRSV_REG018)	8	RW	30h
464h	(TRSV_REG019)	8	RW	00h
468h	(TRSV_REG01A)	8	RW	3Ch
46Ch	(TRSV_REG01B)	8	RW	30h
470h	(TRSV_REG01C)	8	RW	00h
474h	(TRSV_REG01D)	8	RW	10h
478h	(TRSV_REG01E)	8	RW	80h
47Ch	(TRSV_REG01F)	8	RW	00h
480h	(TRSV_REG020)	8	RW	01h
484h	(TRSV_REG021)	8	RW	37h
488h	(TRSV_REG022)	8	RW	00h
48Ch	(TRSV_REG023)	8	RW	00h
490h	(TRSV_REG024)	8	RW	33h
494h	(TRSV_REG025)	8	RW	37h
498h	(TRSV_REG026)	8	RW	31h
49Ch	(TRSV_REG027)	8	RW	00h
4A0h	(TRSV_REG028)	8	RW	04h
4A4h	(TRSV_REG029)	8	RW	6Dh
4A8h	(TRSV_REG02A)	8	RW	48h
4ACh	(TRSV_REG02B)	8	RW	00h
4B0h	(TRSV_REG02C)	8	RW	01h
4B4h	(TRSV_REG02D)	8	RW	1Bh

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
4B8h	(TRSV_REG02E)	8	RW	70h
4BCh	(TRSV_REG02F)	8	RW	0Fh
4C0h	(TRSV_REG030)	8	RW	5Ch
4C4h	(TRSV_REG031)	8	RW	5Ch
4C8h	(TRSV_REG032)	8	RW	7Fh
4CCh	(TRSV_REG033)	8	RW	7Fh
4D0h	(TRSV_REG034)	8	RW	00h
4D4h	(TRSV_REG035)	8	RW	00h
4D8h	(TRSV_REG036)	8	RW	03h
4DCh	(TRSV_REG037)	8	RW	0Fh
4E0h	(TRSV_REG038)	8	RW	0Fh
4E4h	(TRSV_REG039)	8	RW	0Fh
4E8h	(TRSV_REG03A)	8	RW	0Fh
4ECh	(TRSV_REG03B)	8	RW	00h
4F0h	(TRSV_REG03C)	8	RW	02h
4F4h	(TRSV_REG03D)	8	RW	04h
4F8h	(TRSV_REG03E)	8	RW	00h
4FCh	(TRSV_REG03F)	8	RW	02h
500h	(TRSV_REG040)	8	RW	02h
504h	(TRSV_REG041)	8	RW	00h
508h	(TRSV_REG042)	8	RW	11h
50Ch	(TRSV_REG043)	8	RW	00h
510h	(TRSV_REG044)	8	RW	5Fh
514h	(TRSV_REG045)	8	RW	00h
518h	(TRSV_REG046)	8	RW	EEh
51Ch	(TRSV_REG047)	8	RW	FFh
520h	(TRSV_REG048)	8	RW	00h
524h	(TRSV_REG049)	8	RW	08h
528h	(TRSV_REG04A)	8	RW	01h
52Ch	(TRSV_REG04B)	8	RW	00h
530h	(TRSV_REG04C)	8	RW	00h
534h	(TRSV_REG04D)	8	RW	00h
538h	(TRSV_REG04E)	8	RW	00h
53Ch	(TRSV_REG04F)	8	RW	E0h
540h	(TRSV_REG050)	8	RW	00h
544h	(TRSV_REG051)	8	RW	05h
548h	(TRSV_REG052)	8	RW	00h
54Ch	(TRSV_REG053)	8	RW	24h
550h	(TRSV_REG054)	8	RW	00h
554h	(TRSV_REG055)	8	RW	00h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
558h	(TRSV_REG056)	8	RW	00h
55Ch	(TRSV_REG057)	8	RW	00h
560h	(TRSV_REG058)	8	RW	00h
564h	(TRSV_REG059)	8	RW	00h
568h	(TRSV_REG05A)	8	RW	00h
56Ch	(TRSV_REG05B)	8	RW	00h
570h	(TRSV_REG05C)	8	RW	00h
574h	(TRSV_REG05D)	8	RW	00h
578h	(TRSV_REG05E)	8	RW	00h
57Ch	(TRSV_REG05F)	8	RW	00h
580h	(TRSV_REG060)	8	RW	00h
584h	(TRSV_REG061)	8	RW	00h
588h	(TRSV_REG062)	8	RW	00h
58Ch	(TRSV_REG063)	8	RW	00h
590h	(TRSV_REG064)	8	RW	00h
594h	(TRSV_REG065)	8	RW	01h
598h	(TRSV_REG066)	8	RW	00h
59Ch	(TRSV_REG067)	8	RW	00h
5A0h	(TRSV_REG068)	8	RW	3Fh
5A4h	(TRSV_REG069)	8	RW	04h
5A8h	(TRSV_REG06A)	8	RW	0Ch
5ACh	(TRSV_REG06B)	8	RW	18h
5B0h	(TRSV_REG06C)	8	RW	00h
5B4h	(TRSV_REG06D)	8	RW	04h
5B8h	(TRSV_REG06E)	8	RW	02h
5BCh	(TRSV_REG06F)	8	RW	56h
5C0h	(TRSV_REG070)	8	RW	00h
5C4h	(TRSV_REG071)	8	RW	00h
5C8h	(TRSV_REG072)	8	RW	00h
5CCh	(TRSV_REG073)	8	RW	00h
5D0h	(TRSV_REG074)	8	RW	00h
5D4h	(TRSV_REG075)	8	RW	00h
5D8h	(TRSV_REG076)	8	RW	10h
5DCh	(TRSV_REG077)	8	RW	00h
5E0h	(TRSV_REG078)	8	RW	00h
5E4h	(TRSV_REG079)	8	RW	00h
5E8h	(TRSV_REG07A)	8	RW	00h
5ECh	(TRSV_REG07B)	8	RW	21h
5F0h	(TRSV_REG07C)	8	RW	00h
5F4h	(TRSV_REG07D)	8	RW	00h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
5F8h	(TRSV_REG07E)	8	RW	C0h
5FCh	(TRSV_REG07F)	8	RW	00h
600h	(TRSV_REG080)	8	RW	00h
604h	(TRSV_REG081)	8	RW	00h
608h	(TRSV_REG082)	8	RW	00h
60Ch	(TRSV_REG083)	8	RW	00h
610h	(TRSV_REG084)	8	RW	00h
614h	(TRSV_REG085)	8	RW	17h
618h	(TRSV_REG086)	8	RW	FDh
61Ch	(TRSV_REG087)	8	RW	30h
620h	(TRSV_REG088)	8	RW	00h
624h	(TRSV_REG089)	8	RW	C8h
628h	(TRSV_REG08A)	8	RW	00h
62Ch	(TRSV_REG08B)	8	RW	C8h
630h	(TRSV_REG08C)	8	RW	00h
634h	(TRSV_REG08D)	8	RW	A0h
638h	(TRSV_REG08E)	8	RW	01h
63Ch	(TRSV_REG08F)	8	RW	40h
640h	(TRSV_REG090)	8	RW	03h
644h	(TRSV_REG091)	8	RW	03h
648h	(TRSV_REG092)	8	RW	03h
64Ch	(TRSV_REG093)	8	RW	05h
650h	(TRSV_REG094)	8	RW	0Ah
654h	(TRSV_REG095)	8	RW	10h
658h	(TRSV_REG096)	8	RW	10h
65Ch	(TRSV_REG097)	8	RW	05h
660h	(TRSV_REG098)	8	RW	00h
664h	(TRSV_REG099)	8	RW	06h
668h	(TRSV_REG09A)	8	RW	FFh
66Ch	(TRSV_REG09B)	8	RW	FFh
670h	(TRSV_REG09C)	8	RW	FFh
674h	(TRSV_REG09D)	8	RW	CCh
678h	(TRSV_REG09E)	8	RW	F7h
67Ch	(TRSV_REG09F)	8	RW	44h
680h	(TRSV_REG0A0)	8	RW	FFh
684h	(TRSV_REG0A1)	8	RW	FFh
688h	(TRSV_REG0A2)	8	RW	FFh
68Ch	(TRSV_REG0A3)	8	RW	A4h
690h	(TRSV_REG0A4)	8	RW	F5h
694h	(TRSV_REG0A5)	8	RW	20h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
698h	(TRSV_REG0A6)	8	RW	03h
69Ch	(TRSV_REG0A7)	8	RW	02h
6A0h	(TRSV_REG0A8)	8	RW	25h
6A4h	(TRSV_REG0A9)	8	RW	03h
6A8h	(TRSV_REG0AA)	8	RW	13h
6ACh	(TRSV_REG0AB)	8	RW	24h
6B0h	(TRSV_REG0AC)	8	RW	40h
6B4h	(TRSV_REG0AD)	8	RW	80h
6B8h	(TRSV_REG0AE)	8	RW	33h
6BCh	(TRSV_REG0AF)	8	RW	01h
6C0h	(TRSV_REG0B0)	8	RW	00h
6C4h	(TRSV_REG0B1)	8	RW	00h
6C8h	(TRSV_REG0B2)	8	RW	00h
6CCh	(TRSV_REG0B3)	8	RW	00h
6D0h	(TRSV_REG0B4)	8	RW	00h
6D4h	(TRSV_REG0B5)	8	RW	07h
6D8h	(TRSV_REG0B6)	8	RW	04h
6DCh	(TRSV_REG0B7)	8	RW	00h
6E0h	(TRSV_REG0B8)	8	RW	00h
6E4h	(TRSV_REG0B9)	8	RW	00h
6E8h	(TRSV_REG0BA)	8	RW	00h
6ECh	(TRSV_REG0BB)	8	RW	00h
6F0h	(TRSV_REG0BC)	8	RW	08h
6F4h	(TRSV_REG0BD)	8	RW	00h
6F8h	(TRSV_REG0BE)	8	RW	00h
6FCh	(TRSV_REG0BF)	8	RW	00h
700h	(TRSV_REG0C0)	8	RW	00h
704h	(TRSV_REG0C1)	8	RW	00h
708h	(TRSV_REG0C2)	8	RW	00h
70Ch	(TRSV_REG0C3)	8	RW	00h
710h	(TRSV_REG0C4)	8	RW	00h
714h	(TRSV_REG0C5)	8	RW	00h
718h	(TRSV_REG0C6)	8	RW	00h
71Ch	(TRSV_REG0C7)	8	RW	00h
720h	(TRSV_REG0C8)	8	RW	00h
724h	(TRSV_REG0C9)	8	RW	00h
728h	(TRSV_REG0CA)	8	RW	00h
72Ch	(TRSV_REG0CB)	8	RW	00h
730h	(TRSV_REG0CC)	8	RW	00h
734h	(TRSV_REG0CD)	8	RW	00h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
738h	(TRSV_REG0CE)	8	RW	31h
73Ch	(TRSV_REG0CF)	8	RW	00h
740h	(TRSV_REG0D0)	8	RW	00h
744h	(TRSV_REG0D1)	8	RW	01h
748h	(TRSV_REG0D2)	8	RW	00h
74Ch	(TRSV_REG0D3)	8	RW	00h
750h	(TRSV_REG0D4)	8	RW	00h
754h	(TRSV_REG0D5)	8	RW	00h
758h	(TRSV_REG0D6)	8	RW	00h
75Ch	(TRSV_REG0D7)	8	RW	00h
760h	(TRSV_REG0D8)	8	RW	00h
764h	(TRSV_REG0D9)	8	RW	00h
768h	(TRSV_REG0DA)	8	RW	00h
76Ch	(TRSV_REG0DB)	8	RW	00h
770h	(TRSV_REG0DC)	8	RW	00h
774h	(TRSV_REG0DD)	8	RW	00h
778h	(TRSV_REG0DE)	8	RW	00h
77Ch	(TRSV_REG0DF)	8	RW	00h
780h	(TRSV_REG0E0)	8	RW	00h
784h	(TRSV_REG0E1)	8	RW	00h
788h	(TRSV_REG0E2)	8	RW	00h
78Ch	(TRSV_REG0E3)	8	RW	00h
790h	(TRSV_REG0E4)	8	RW	00h
794h	(TRSV_REG0E5)	8	RW	00h
798h	(TRSV_REG0E6)	8	RW	00h
79Ch	(TRSV_REG0E7)	8	RW	00h
7A0h	(TRSV_REG0E8)	8	RW	00h
7A4h	(TRSV_REG0E9)	8	RW	00h
7A8h	(TRSV_REG0EA)	8	RW	00h
7ACh	(TRSV_REG0EB)	8	RW	00h
7B0h	(TRSV_REG0EC)	8	RW	00h
7B4h	(TRSV_REG0ED)	8	RW	00h
7B8h	(TRSV_REG0EE)	8	RW	00h
7BCh	(TRSV_REG0EF)	8	RW	00h
7C0h	(TRSV_REG0F0)	8	RW	00h
7C4h	(TRSV_REG0F1)	8	RW	00h
7C8h	(TRSV_REG0F2)	8	RW	00h
7CCh	(TRSV_REG0F3)	8	RW	00h
7D0h	(TRSV_REG0F4)	8	RW	00h
7D4h	(TRSV_REG0F5)	8	RW	00h

Table continues on the next page...

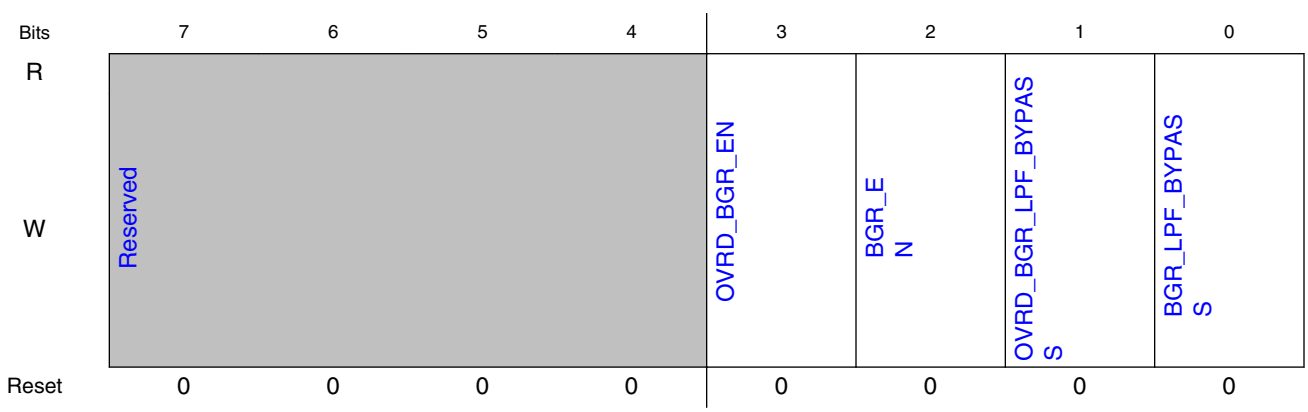
Offset	Register	Width (In bits)	Access	Reset value
7D8h	(TRSV_REG0F6)	8	RW	00h
7DCh	(TRSV_REG0F7)	8	RW	00h
7E0h	(TRSV_REG0F8)	8	RW	00h
7E4h	(TRSV_REG0F9)	8	RW	00h
7E8h	(TRSV_REG0FA)	8	RW	00h
7ECh	(TRSV_REG0FB)	8	RW	00h
7F0h	(TRSV_REG0FC)	8	RW	00h
7F4h	(TRSV_REG0FD)	8	RW	00h
7F8h	(TRSV_REG0FE)	8	RW	1Fh
7FCh	(TRSV_REG0FF)	8	RW	18h

11.4.3.1.2 (CMN_REG000)

11.4.3.1.2.1 Offset

Register	Offset
CMN_REG000	0h

11.4.3.1.2.2 Diagram



11.4.3.1.2.3 Fields

Field	Function
7-4	Reserved

Table continues on the next page...

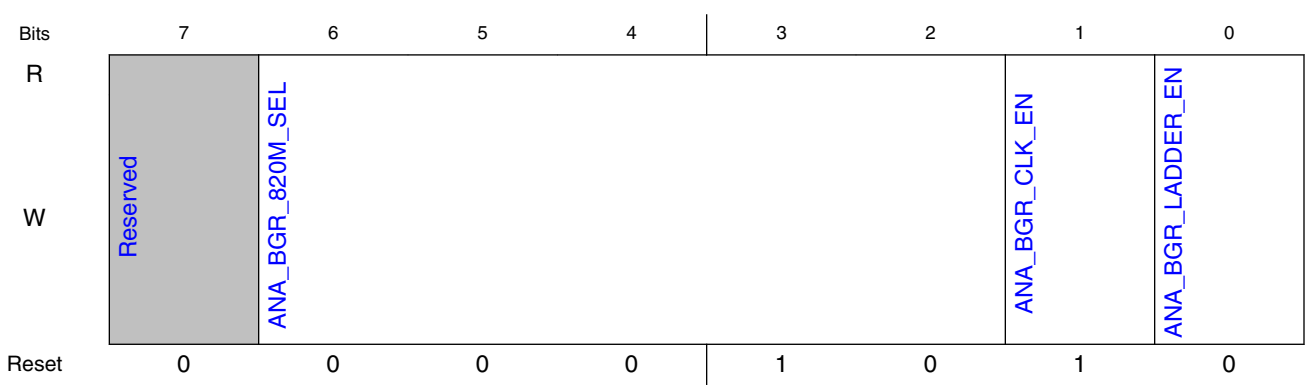
Field	Function
—	
3 OVRD_BGR_EN	Override enable for bgr_en
2 BGR_EN	BGR enable 0: Disable, 1: Enable
1 OVRD_BGR_LPF_BYPASS	Override enable for bgr_lpf_bypass
0 BGR_LPF_BYPASS	BGR LPF bypass to reduce BGR settle time 0: LPF enable, 1: LPF bypass

11.4.3.1.3 (CMN_REG001)

11.4.3.1.3.1 Offset

Register	Offset
CMN_REG001	4h

11.4.3.1.3.2 Diagram



11.4.3.1.3.3 Fields

Field	Function
7 —	Reserved

Table continues on the next page...

Field	Function
6-2 ANA_BGR_820M_SEL	BGR 820mV selection (for current bias) 00001: 800mV 00010: 820mV 00100: 840mV 01000: 860mV 10000: 900mV
1 ANA_BGR_CLK_EN	BGR chopper clock enable 0: Disable, 1: Enable
0 ANA_BGR_LADDER_EN	BGR output voltage selection 0: BGR output, 1: Resistor ladder output

11.4.3.1.4 (CMN_REG002)

11.4.3.1.4.1 Offset

Register	Offset
CMN_REG002	8h

11.4.3.1.4.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	ANA_BGR_LADDER_SEL			ANA_BGR_ATB_SEL	OVRD_BIAS_EN	BIAS_EN	OVRD_BIAS_RCAL_EN	BIAS_RCAL_EN
Reset	0	0	0	0	0	0	0	0

11.4.3.1.4.3 Fields

Field	Function
7-5	Resistor ladder voltage selection 000: 1000mV, 001: 960mV, 010: 920mV, 011: 880mV,

Table continues on the next page...

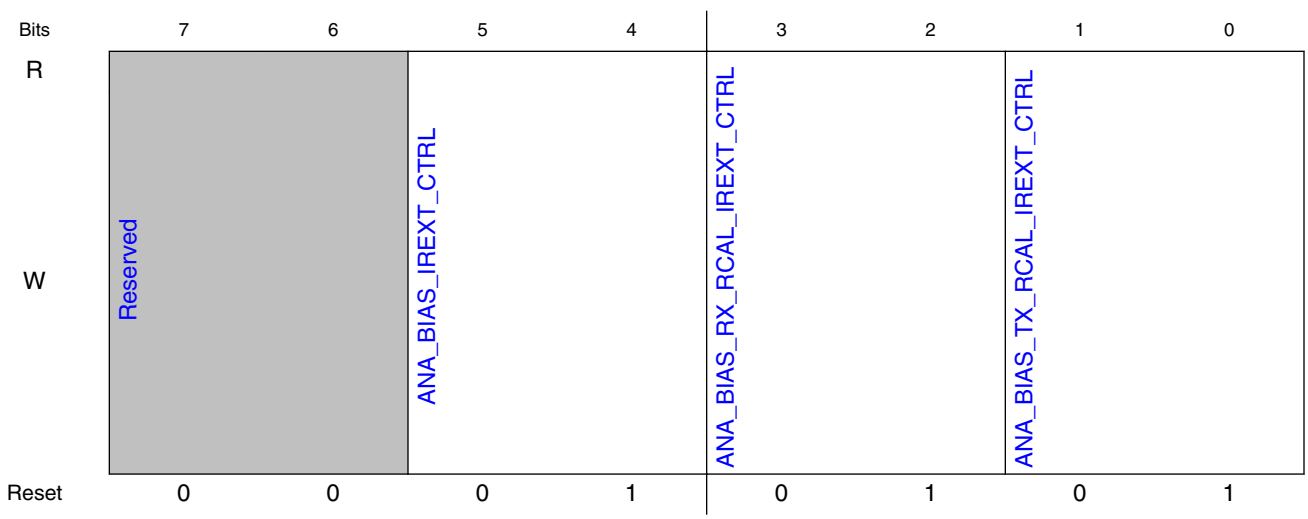
Field	Function
ANA_BGR_LADDER_SEL	100: 840mV, 101: 800mV, 110: 760mV, 111: 720mV
4 ANA_BGR_ATB_SEL	BGR ATB select
3 OVRD_BIAS_EN	Override enable for bias_en
2 BIAS_EN	Bias current enable 0: Disable, 1: Enable
1 OVRD_BIAS_RCAL_EN	Override enable for bias_rcal_en
0 BIAS_RCAL_EN	RX RCAL bias current enable 0: Disable, 1: Enable

11.4.3.1.5 (CMN_REG003)

11.4.3.1.5.1 Offset

Register	Offset
CMN_REG003	Ch

11.4.3.1.5.2 Diagram



11.4.3.1.5.3 Fields

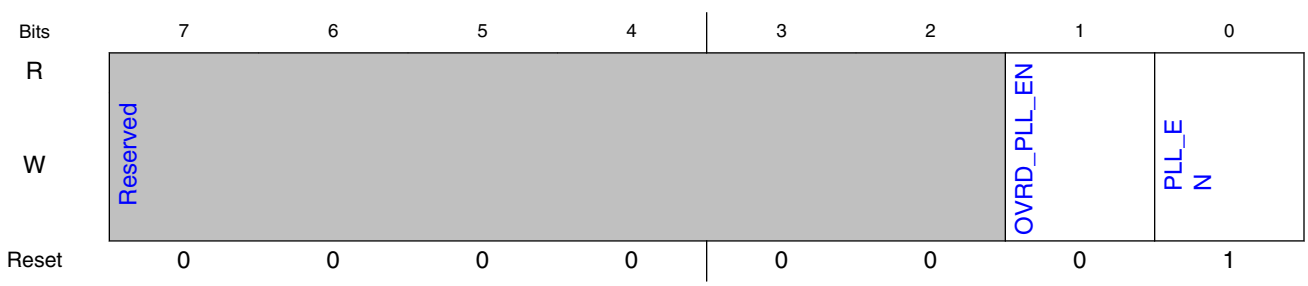
Field	Function
7-6 —	Reserved
5-4 ANA_BIAS_IREXT_CTRL	REXT-refered bias current controll for overall IP 00: 80uA 01: 100uA 10: 132uA
3-2 ANA_BIAS_RX_RCAL_IREXT_CTRL	REXT-refered bias current controll for RX RCAL 00: 85uA 01: 100uA 11: 115uA
1-0 ANA_BIAS_TX_RCAL_IREXT_CTRL	REXT-refered bias current control MSB for TX RCAL

11.4.3.1.6 (CMN_REG004)

11.4.3.1.6.1 Offset

Register	Offset
CMN_REG004	10h

11.4.3.1.6.2 Diagram



11.4.3.1.6.3 Fields

Field	Function
7-2	Reserved

Table continues on the next page...

Field	Function
—	
1 OVRD_PLL_EN	Override enable for pll_en
0 PLL_EN	PLL enable 0: Disable, 1: Enable

11.4.3.1.7 (CMN_REG005)

11.4.3.1.7.1 Offset

Register	Offset
CMN_REG005	14h

11.4.3.1.7.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	PLL_VCO_MODE_G1	PLL_VCO_MODE_G2	PLL_VCO_MODE_G3	PLL_VCO_MODE_G4	OVRD_PLL_AFC_INIT_RSTN	PLL_AFC_INIT_RSTN	OVRD_PLL_AFC_RSTN	PLL_AFC_RSTN
W								
Reset	1	1	1	1	0	0	0	0

11.4.3.1.7.3 Fields

Field	Function
7 PLL_VCO_MODE_G1	[GEN1] PLL VCO selection 0: Ring VCO, 1: LC VCO
6 PLL_VCO_MODE_G2	[GEN2]
5	[GEN3]

Table continues on the next page...

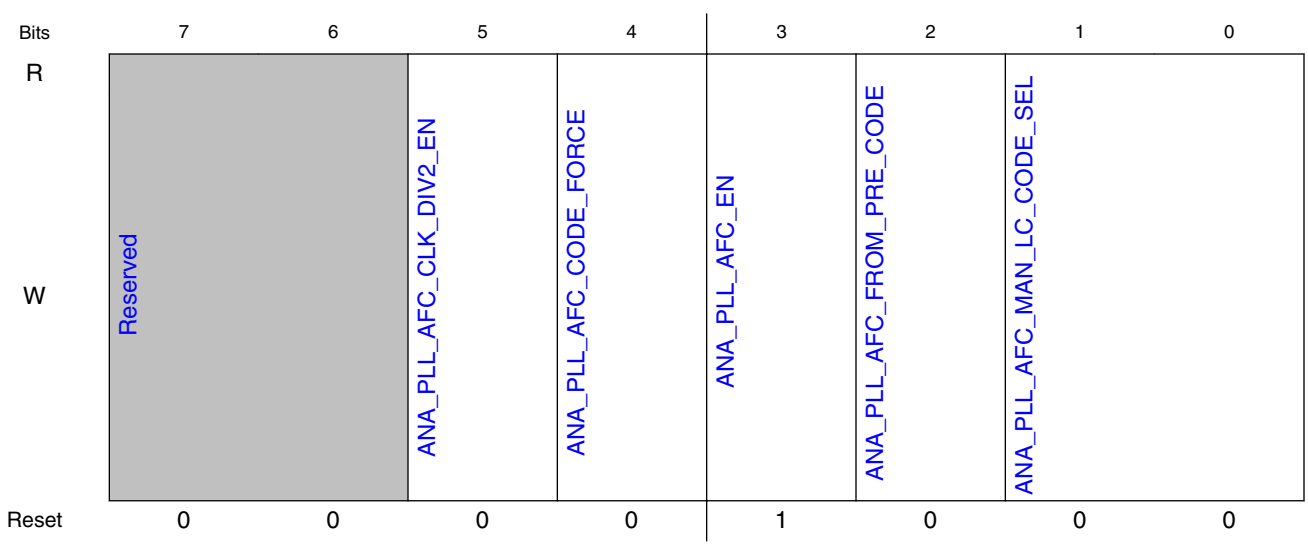
Field	Function
PLL_VCO_MODE_G3	
4 PLL_VCO_MODE_G4	[GEN4]
3 OVRD_PLL_AFC_INIT_RSTN	Override enable for pll_afc_init_rstn
2 PLL_AFC_INIT_RSTN	PLL AFC initial reset. When initial reset is asserted, the previous AFC result is reset. When initial reset is released, AFC starts from the initial AFC code given in i_rx_cdr_afc_sel_logic[3:0]. 0: Reset, 1: Released
1 OVRD_PLL_AFC_RSTN	Override enable for pll_afc_rstn
0 PLL_AFC_RSTN	PLL AFC reset. When AFC reset is asserted, the previous AFC result is held. When AFC reset is released, AFC starts from the previous AFC code stored in internal memory. 0: Reset, 1: Released

11.4.3.1.8 (CMN_REG006)

11.4.3.1.8.1 Offset

Register	Offset
CMN_REG006	18h

11.4.3.1.8.2 Diagram



11.4.3.1.8.3 Fields

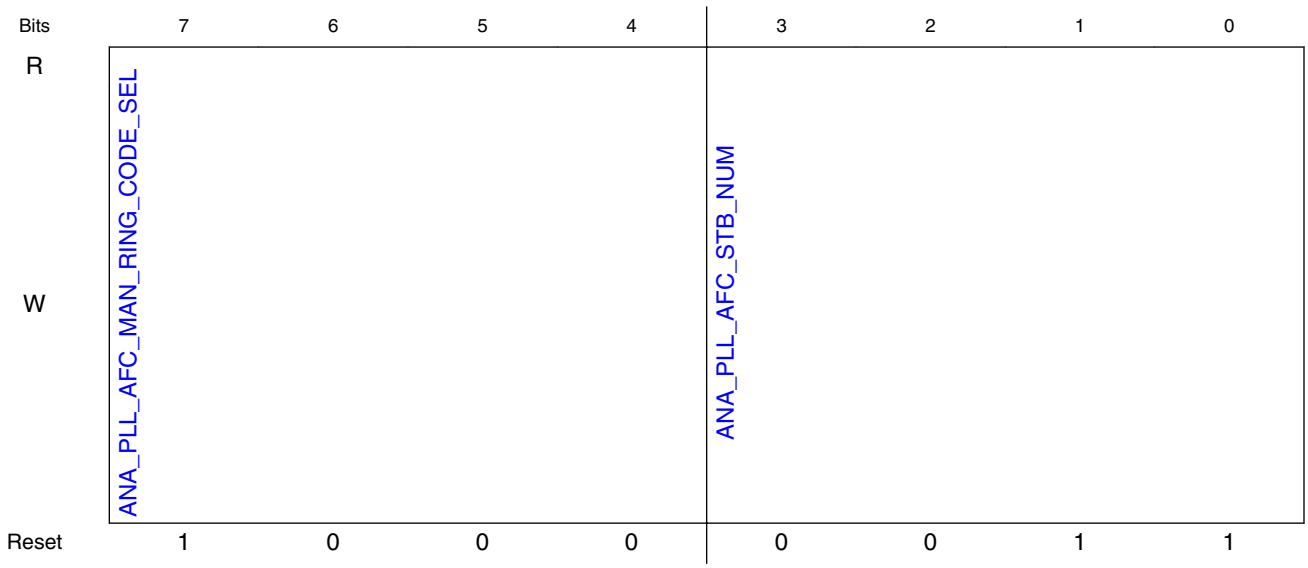
Field	Function
7-6 —	Reserved
5 ANA_PLL_AFC_CLK_DIV2_EN	PLL AFC clock frequency selection 0: fVCO, 1: fVCO/2
4 ANA_PLL_AFC_CODE_FORCE	PLL AFC code manual selection enable 0: AFC result, 1: Manual selection
3 ANA_PLL_AFC_EN	PLL AFC enable; if enabled, VCO frequency is automatically calibrated. If disabled, VCO starts to oscillate with fixed AFC code of i_pll_man_bsel_m and _l. 0: Disable, 1: Enable
2 ANA_PLL_AFC_FROM_PRE_CODE	PLL AFC option in restart case 0: Restart from initial AFC code, 1: Restart from previous AFC code
1-0 ANA_PLL_AFC_MAN_LC_CODE_SEL	Manual PLL AFC code selection (MSB)

11.4.3.1.9 (CMN_REG007)

11.4.3.1.9.1 Offset

Register	Offset
CMN_REG007	1Ch

11.4.3.1.9.2 Diagram



11.4.3.1.9.3 Fields

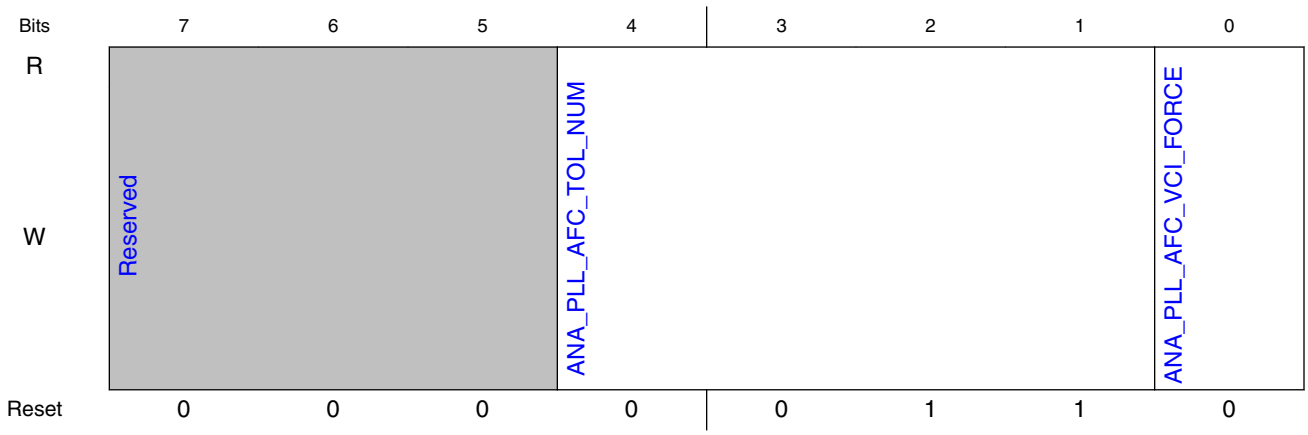
Field	Function
7-4 ANA_PLL_AFC_MAN_RING_CODE_SEL	Manual PLL AFC code selection (LSB)
3-0 ANA_PLL_AFC_STB_NUM	Number of reference clock cycle to check VCO stabilization during PLL AFC start

11.4.3.1.10 (CMN_REG008)

11.4.3.1.10.1 Offset

Register	Offset
CMN_REG008	20h

11.4.3.1.10.2 Diagram



11.4.3.1.10.3 Fields

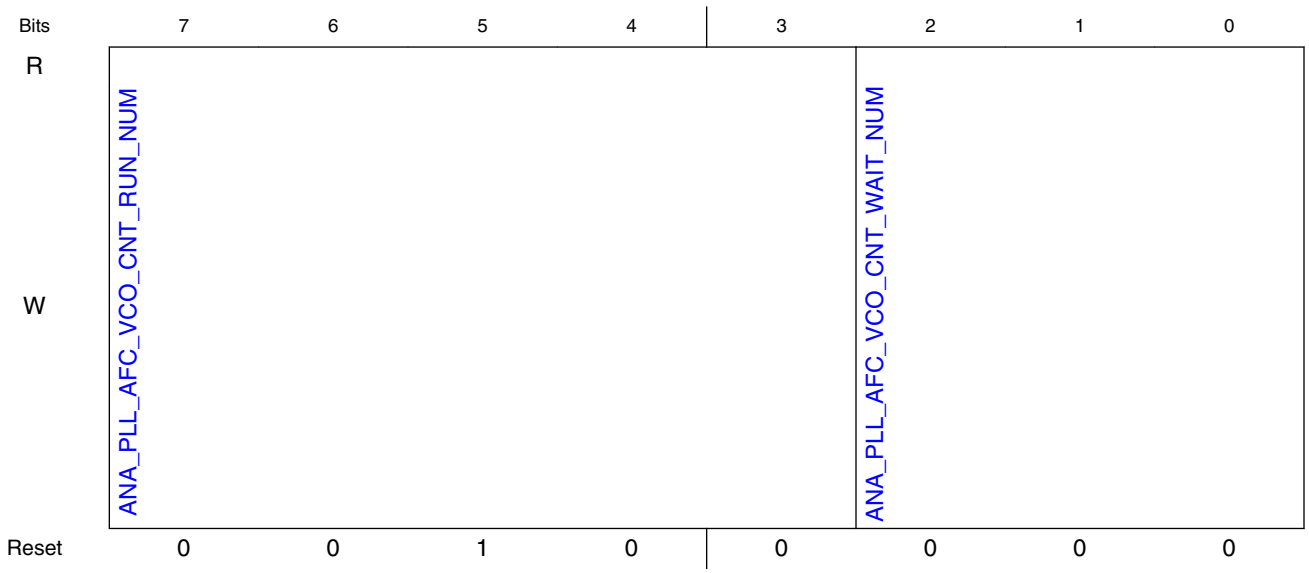
Field	Function
7-5 —	Reserved
4-1 ANA_PLL_AFC_TOL_NUM	PLL VCO stabilization tolerance; VCO is considered as settled-down if |counter difference|<i_pll_afc_tol during i_pll_afc_stb_num
0 ANA_PLL_AFC_VCI_FORCE	PLL control voltage force for open-loop test purpose 0: Released, 1: Forced

11.4.3.1.11 (CMN_REG009)

11.4.3.1.11.1 Offset

Register	Offset
CMN_REG009	24h

11.4.3.1.11.2 Diagram



11.4.3.1.11.3 Fields

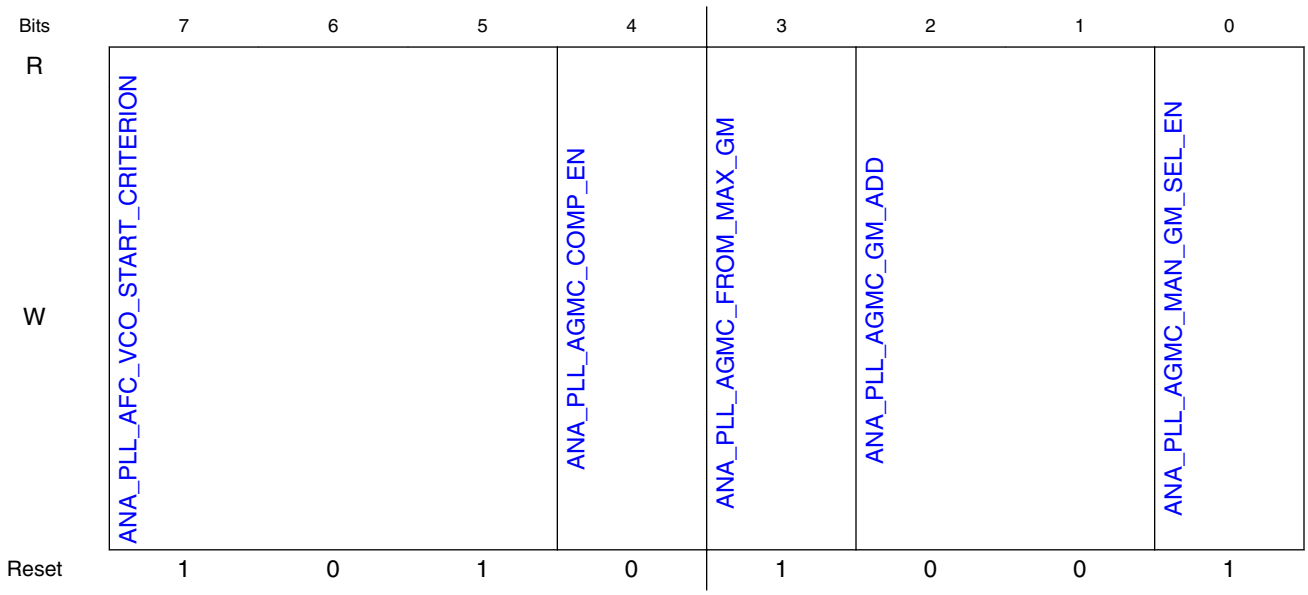
Field	Function
7-3 ANA_PLL_AFC_VCO_CNT_RUN_NUM	Number of reference clock cycle to wait VCO clock during AFC
2-0 ANA_PLL_AFC_VCO_CNT_WAIT_NUM	Number of reference clock cycle to count VCO clock during AFC

11.4.3.1.12 (CMN_REG00A)

11.4.3.1.12.1 Offset

Register	Offset
CMN_REG00A	28h

11.4.3.1.12.2 Diagram



11.4.3.1.12.3 Fields

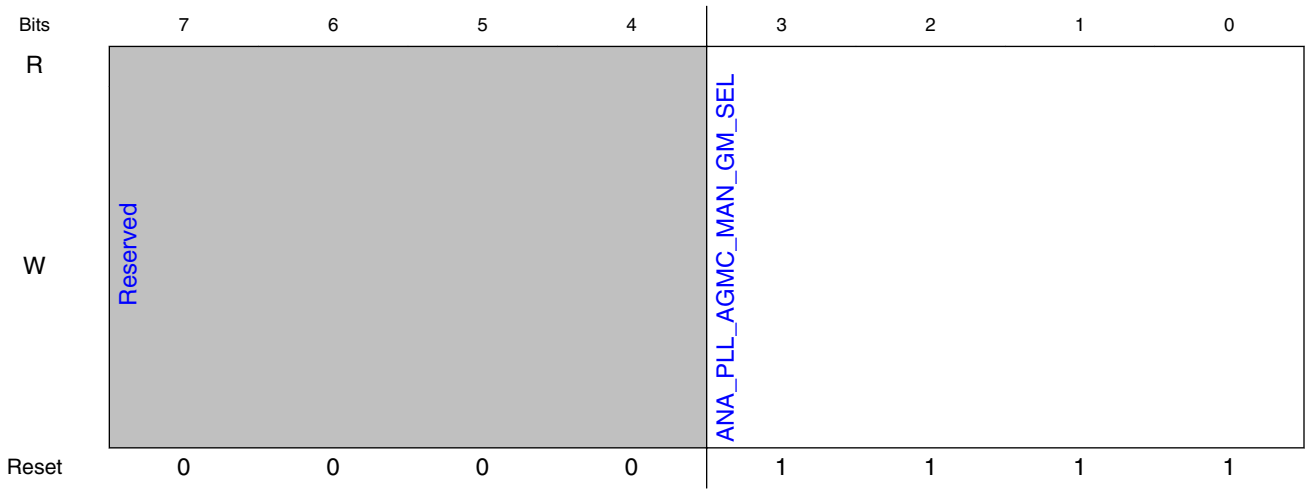
Field	Function
7-5 ANA_PLL_AFC_VCO_START_CRITERION	Minimum PLL VCO counter value to start AFC (Criterion to suppose PLL VCO successfully start to oscillate)
4 ANA_PLL_AGMC_COMP_EN	Comparator enable for PLL LC VCO automatic gm search 0: Disable, 1: Enable
3 ANA_PLL_AGMC_FROM_MAX_GM	PLL LC VCO automatic gm search initial condition 0: Start from min gm condition, 1:Start from max gm condition
2-1 ANA_PLL_AGMC_GM_ADD	Offset code to be added to final gm code 00: Min, … , 11: Max
0 ANA_PLL_AGMC_MAN_GM_SEL_EN	PLL LC VCO GM code selection 0: AGMC result, 1: Manual selection

11.4.3.1.13 (CMN_REG00B)

11.4.3.1.13.1 Offset

Register	Offset
CMN_REG00B	2Ch

11.4.3.1.13.2 Diagram



11.4.3.1.13.3 Fields

Field	Function
7-4 —	Reserved
3-0 ANA_PLL_AGMC_MAN_GM_SEL	Manual GM code selection for LC VCO 0000: GM min (ampitude min), …, 1111: GM max (amplitude max)

11.4.3.1.14 (CMN_REG00C)

11.4.3.1.14.1 Offset

Register	Offset
CMN_REG00C	30h

11.4.3.1.14.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_AGMC_TG_CODE_G1							
Reset	0	1	0	0	0	1	1	0

11.4.3.1.14.3 Fields

Field	Function
7-0 PLL_AGMC_TG_CODE_G1	[GEN1] Target counter value for automatic GM search of LC VCO 4GHz:8'd40 or 8'b00101000 or 8'h28 (Default) 7GHz: 8'd70 or 8'b1000110 or 8'h46

11.4.3.1.15 (CMN_REG00D)

11.4.3.1.15.1 Offset

Register	Offset
CMN_REG00D	34h

11.4.3.1.15.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_AGMC_TG_CODE_G2							
Reset	0	1	0	0	0	1	1	0

11.4.3.1.15.3 Fields

Field	Function
7-0 PLL_AGMC_TG_CODE_G2	[GEN2]

11.4.3.1.16 (CMN_REG00E)

11.4.3.1.16.1 Offset

Register	Offset
CMN_REG00E	38h

11.4.3.1.16.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_AGMC_TG_CODE_G3							
Reset	0	1	0	0	0	1	1	0

11.4.3.1.16.3 Fields

Field	Function
7-0 PLL_AGMC_TG_CODE_G3	[GEN3]

11.4.3.1.17 (CMN_REG00F)

11.4.3.1.17.1 Offset

Register	Offset
CMN_REG00F	3Ch

11.4.3.1.17.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_AGMC_TG_CODE_G4							
Reset	0	1	0	0	0	1	1	0

11.4.3.1.17.3 Fields

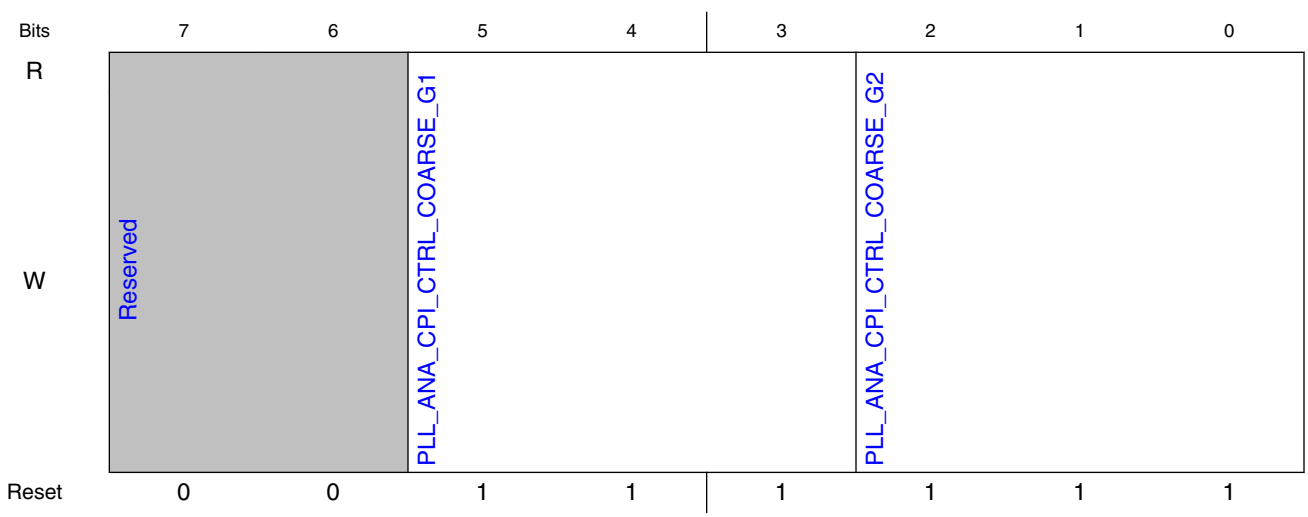
Field	Function
7-0 PLL_AGMC_TG_CODE_G4	[GEN4]

11.4.3.1.18 (CMN_REG010)

11.4.3.1.18.1 Offset

Register	Offset
CMN_REG010	40h

11.4.3.1.18.2 Diagram



11.4.3.1.18.3 Fields

Field	Function
7-6 —	Reserved
5-3 PLL_ANA_CPI_CTRL_COARSE_G1	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA

Table continues on the next page...

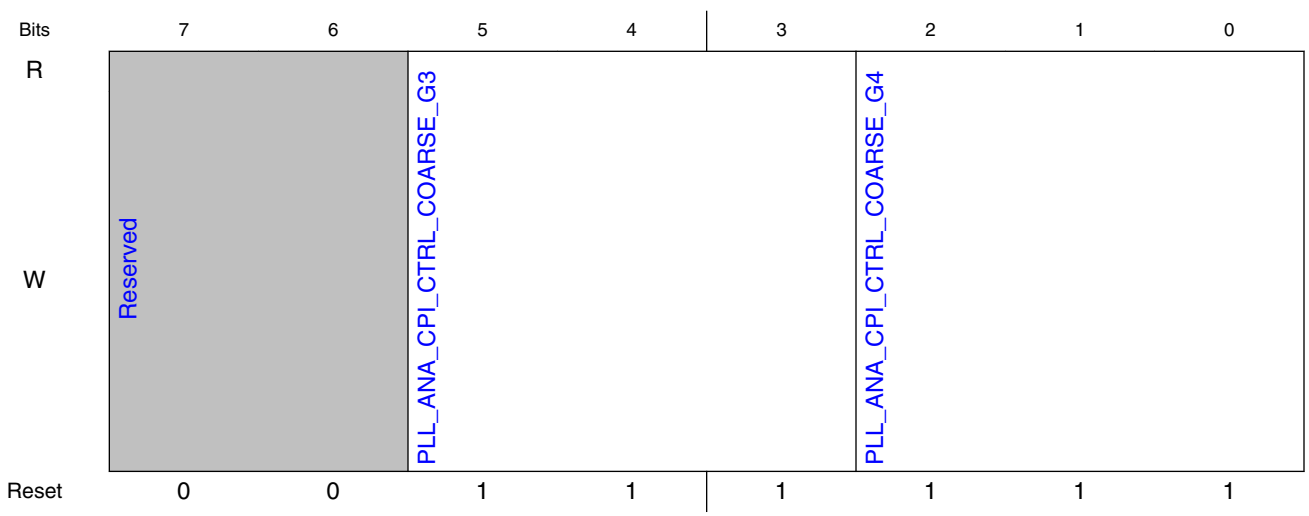
Field	Function
2-0 PLL_ANA_CPI_CTRL_COARSE_G2	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA

11.4.3.1.19 (CMN_REG011)

11.4.3.1.19.1 Offset

Register	Offset
CMN_REG011	44h

11.4.3.1.19.2 Diagram



11.4.3.1.19.3 Fields

Field	Function
7-6 —	Reserved
5-3 PLL_ANA_CPI_CTRL_COARSE_G3	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA
2-0	[GEN1] PLL integral path charge-pump current contorl

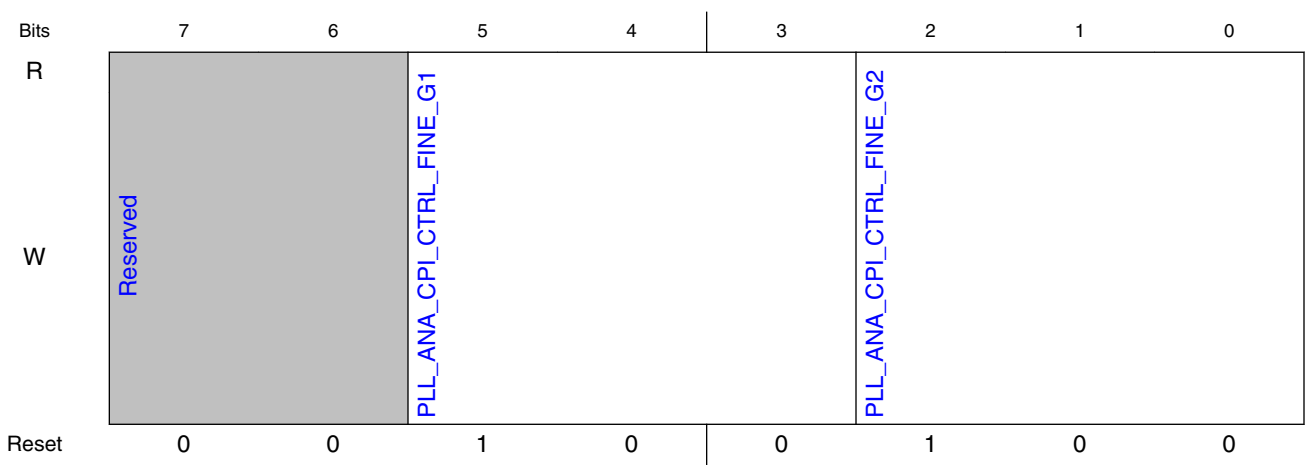
Field	Function
PLL_ANA_CPI_CTRL_COARSE_G4	0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA

11.4.3.1.20 (CMN_REG012)

11.4.3.1.20.1 Offset

Register	Offset
CMN_REG012	48h

11.4.3.1.20.2 Diagram



11.4.3.1.20.3 Fields

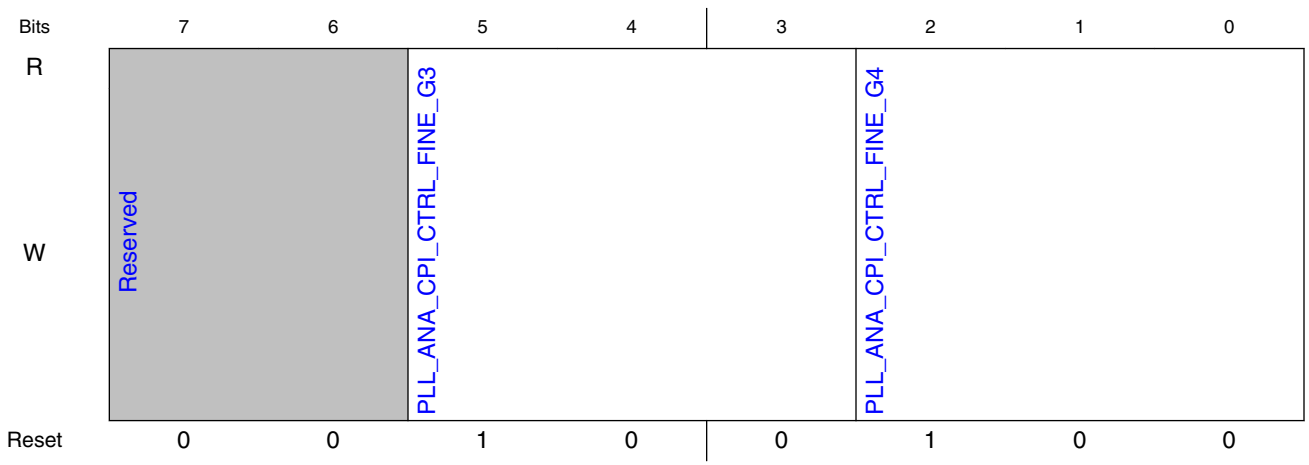
Field	Function
7-6 —	Reserved
5-3 PLL_ANA_CPI_CTRL_FINE_G1	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA
2-0 PLL_ANA_CPI_CTRL_FINE_G2	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA

11.4.3.1.21 (CMN_REG013)

11.4.3.1.21.1 Offset

Register	Offset
CMN_REG013	4Ch

11.4.3.1.21.2 Diagram



11.4.3.1.21.3 Fields

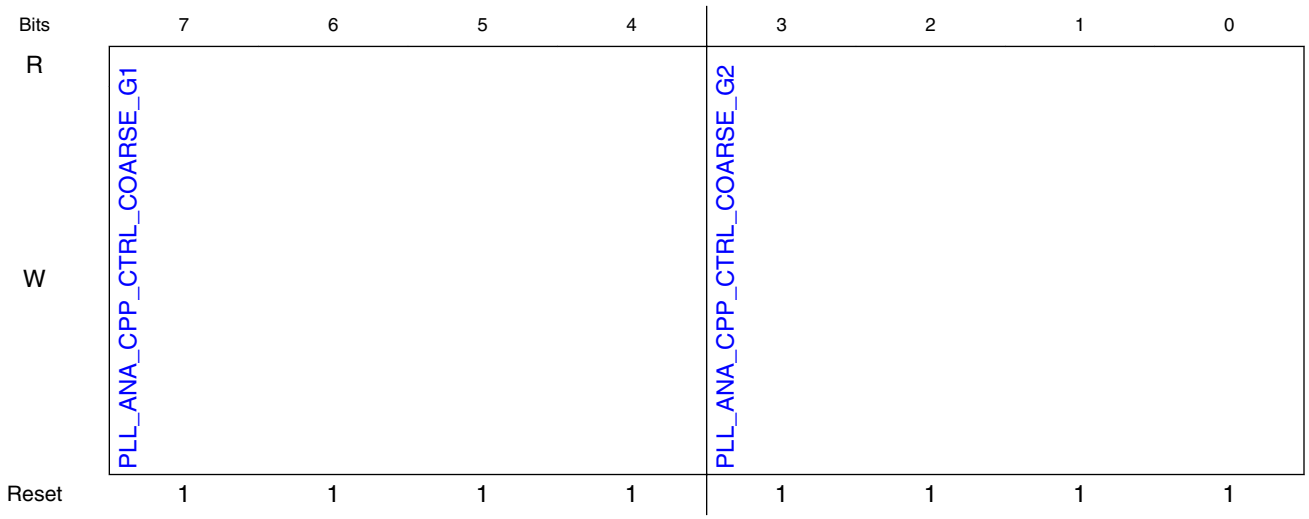
Field	Function
7-6 —	Reserved
5-3 PLL_ANA_CPI_CTRL_FINE_G3	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA
2-0 PLL_ANA_CPI_CTRL_FINE_G4	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA

11.4.3.1.22 (CMN_REG014)

11.4.3.1.22.1 Offset

Register	Offset
CMN_REG014	50h

11.4.3.1.22.2 Diagram



11.4.3.1.22.3 Fields

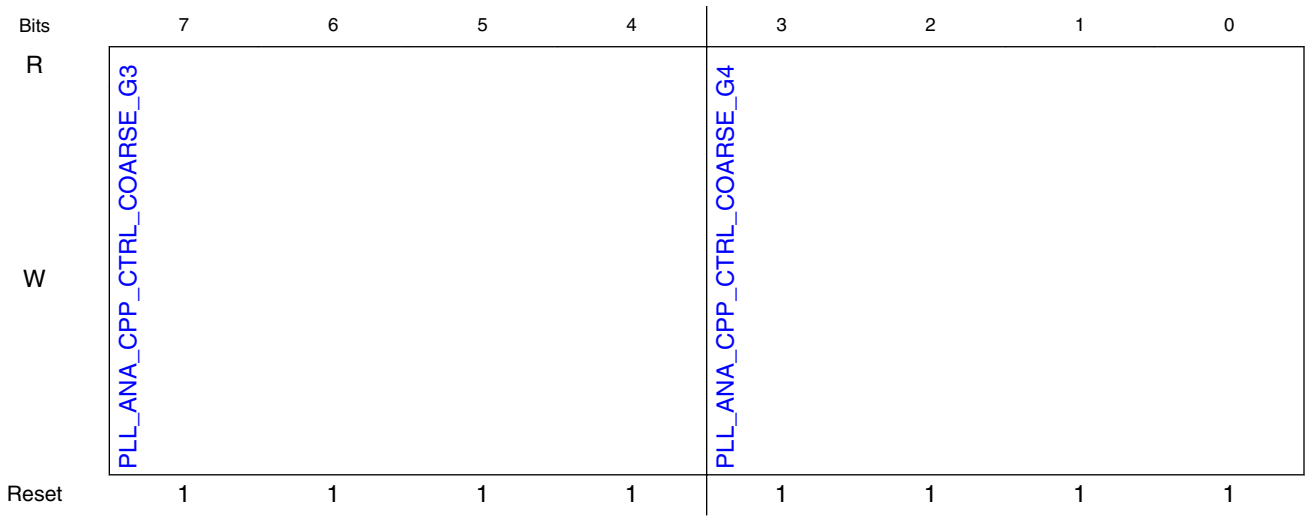
Field	Function
7-4 PLL_ANA_CPP_CTRL_COARSE_G1	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA
3-0 PLL_ANA_CPP_CTRL_COARSE_G2	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA

11.4.3.1.23 (CMN_REG015)

11.4.3.1.23.1 Offset

Register	Offset
CMN_REG015	54h

11.4.3.1.23.2 Diagram



11.4.3.1.23.3 Fields

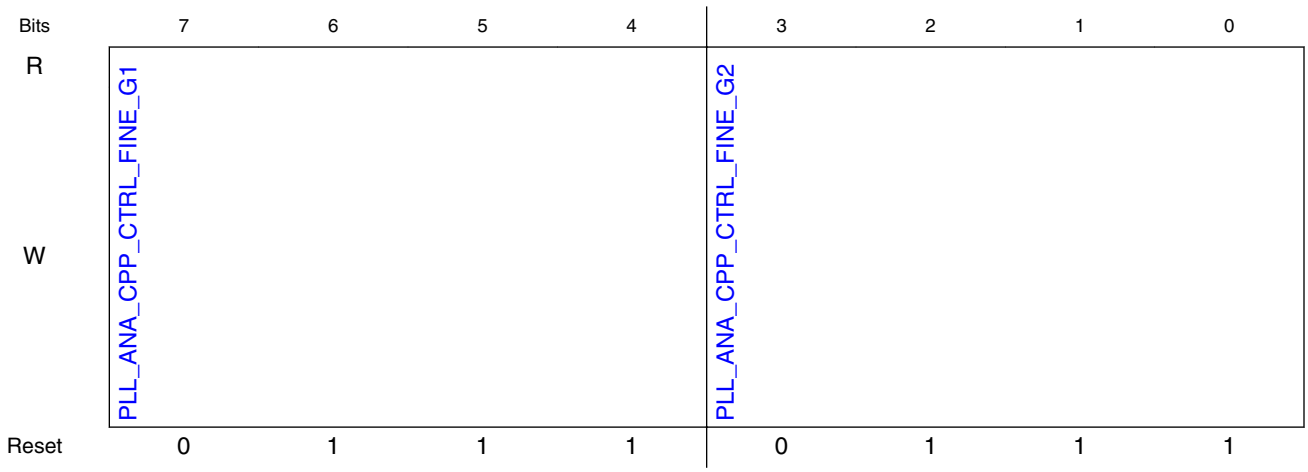
Field	Function
7-4 PLL_ANA_CPP_CTRL_COARSE_G3	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA
3-0 PLL_ANA_CPP_CTRL_COARSE_G4	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA

11.4.3.1.24 (CMN_REG016)

11.4.3.1.24.1 Offset

Register	Offset
CMN_REG016	58h

11.4.3.1.24.2 Diagram



11.4.3.1.24.3 Fields

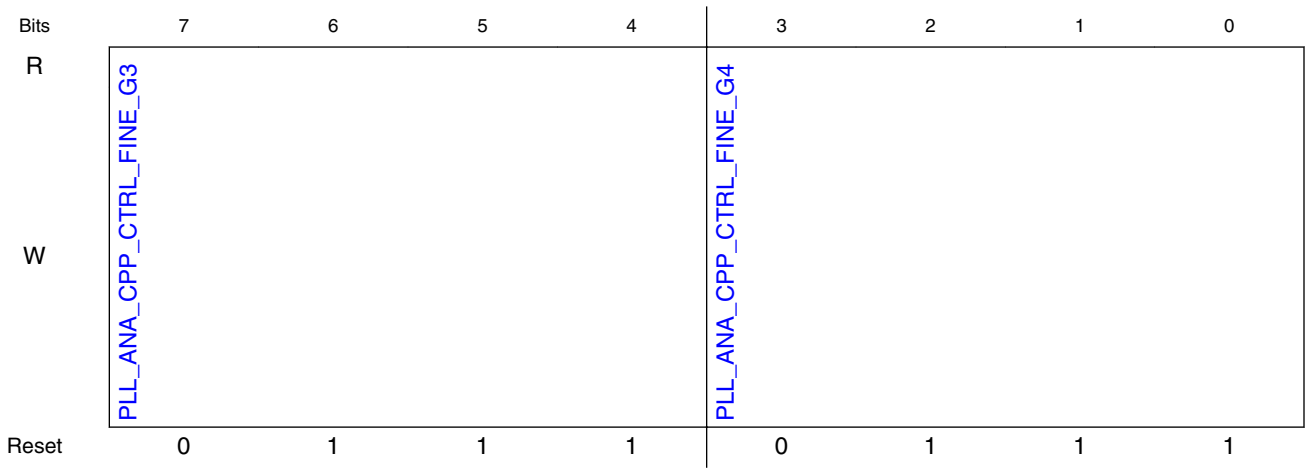
Field	Function
7-4 PLL_ANA_CPP_CTRL_FINE_G1	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA
3-0 PLL_ANA_CPP_CTRL_FINE_G2	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA

11.4.3.1.25 (CMN_REG017)

11.4.3.1.25.1 Offset

Register	Offset
CMN_REG017	5Ch

11.4.3.1.25.2 Diagram



11.4.3.1.25.3 Fields

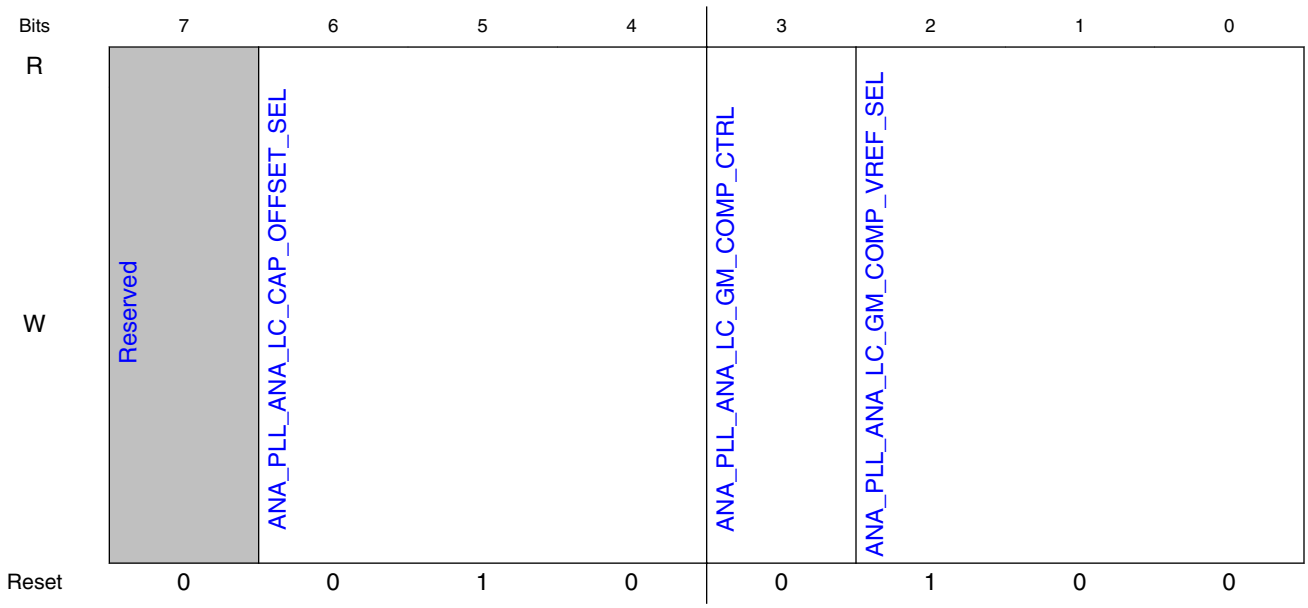
Field	Function
7-4 PLL_ANA_CPP_CTRL_FINE_G3	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA
3-0 PLL_ANA_CPP_CTRL_FINE_G4	[GEN1] PLL integral path charge-pump current contorl 0xx: half current (not used) 100: 1.25uA, 101:1.5625uA, 110: 20833uA, 111: 3.125uA

11.4.3.1.26 (CMN_REG018)

11.4.3.1.26.1 Offset

Register	Offset
CMN_REG018	60h

11.4.3.1.26.2 Diagram



11.4.3.1.26.3 Fields

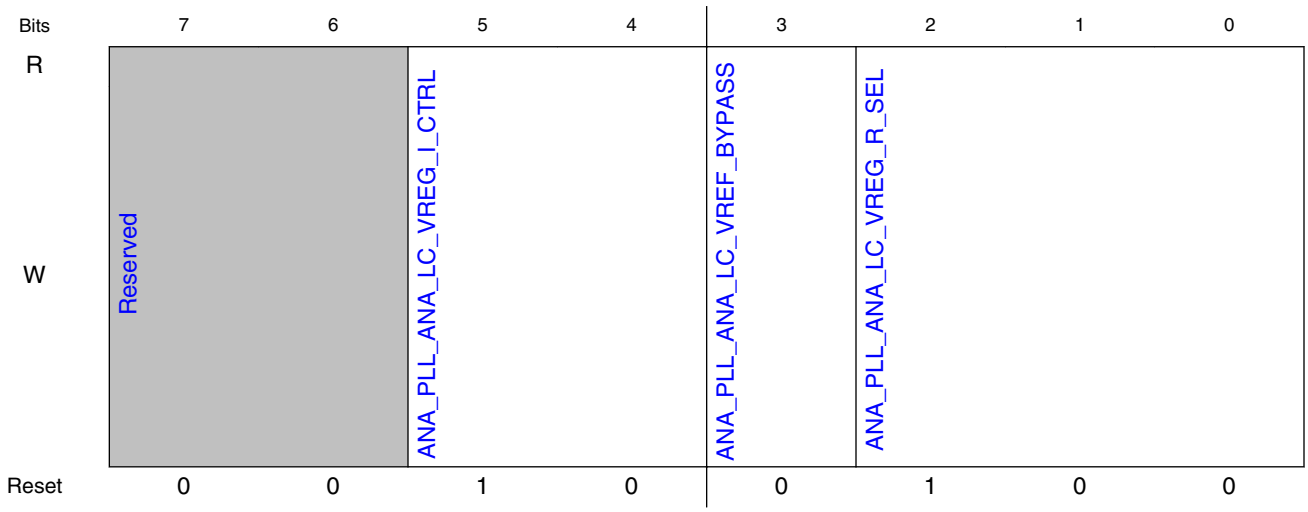
Field	Function
7 —	Reserved
6-4 ANA_PLL_ANA_LC_CAP_OFFSET_SEL	LC VCO varactor bias voltage selection 000: 1.3V, 001: 1.2V, 010: 1.1V, 011: 1.0V, 100: 0.9V, 101: 0.8V, 110: 0.7V, 111: 0.6V:
3 ANA_PLL_ANA_LC_GM_COMP_CTRL	
2-0 ANA_PLL_ANA_LC_GM_COMP_VREF_SEL	PLL GM comparator reference voltage selection 000: 769mV, 001: 724mV, 010: 679mV, 011: 634mV, 100: 588mV, 101: 543mV, 110: 498mV, 111: 454mV

11.4.3.1.27 (CMN_REG019)

11.4.3.1.27.1 Offset

Register	Offset
CMN_REG019	64h

11.4.3.1.27.2 Diagram



11.4.3.1.27.3 Fields

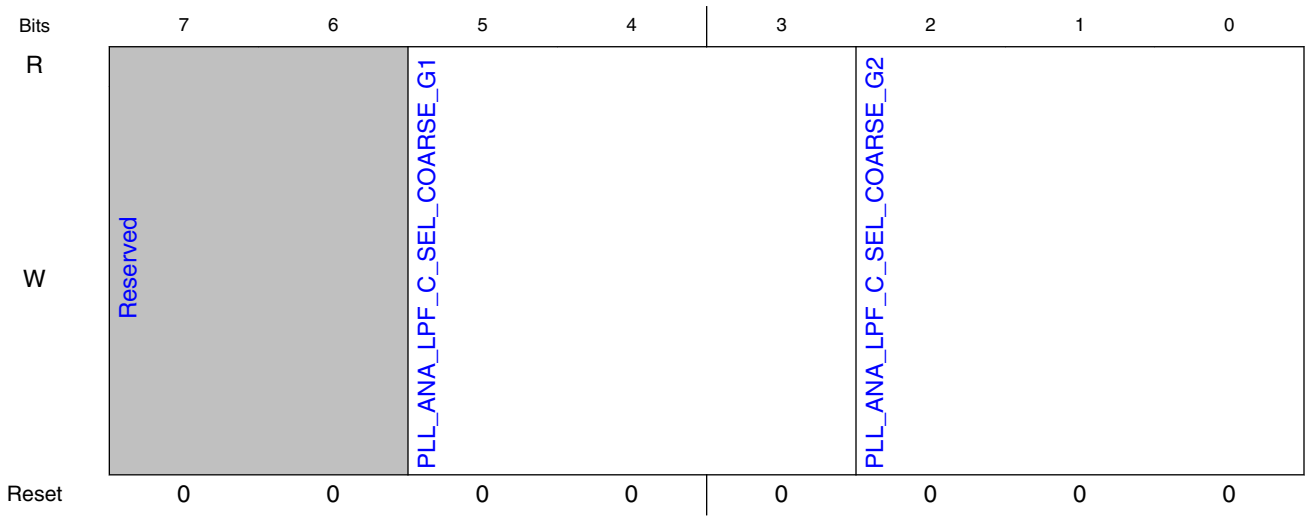
Field	Function
7-6 —	Reserved
5-4 ANA_PLL_ANA_LC_VREG_I_CTRL	LC VCO Vreg current branch enable (1+1/20 or 1+ 1/33)
3 ANA_PLL_ANA_LC_VREF_BYPASS	LPF on reference voltage for PLL LC VCO bypass 0: LPF enable, 1: LPF bypass
2-0 ANA_PLL_ANA_LC_VREG_R_SEL	LC VCO voltage regulator output control 000: 853mV, 001: 863mV, 010: 878mV, 011: 888mV 100: 911mV, 101: 921mV, 110: 936mV, 111: 946mV

11.4.3.1.28 (CMN_REG01A)

11.4.3.1.28.1 Offset

Register	Offset
CMN_REG01A	68h

11.4.3.1.28.2 Diagram



11.4.3.1.28.3 Fields

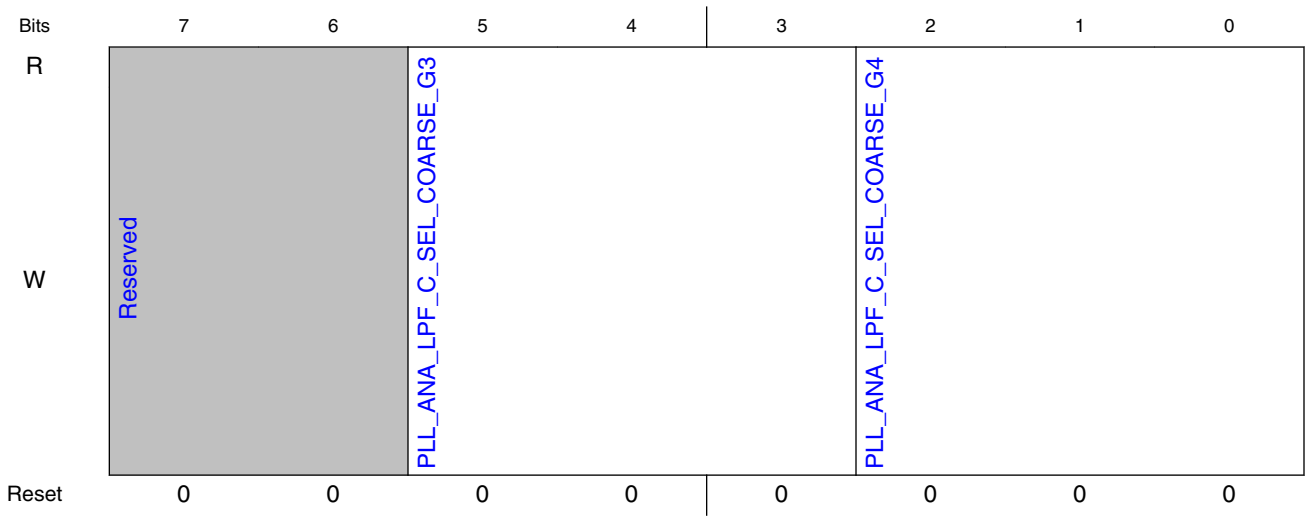
Field	Function
7-6 —	Reserved
5-3 PLL_ANA_LPF_C_SEL_COARSE_G1	[GEN1] [Coarse Mode] PLL loop filter capacitor control 000: 0pF, 001: 1pF, 010: 2pF, 011: 3pF, 100: 4pF, 101: 5pF,110: 6pF, 111: 7pF
2-0 PLL_ANA_LPF_C_SEL_COARSE_G2	[GEN2] [Coarse Mode]

11.4.3.1.29 (CMN_REG01B)

11.4.3.1.29.1 Offset

Register	Offset
CMN_REG01B	6Ch

11.4.3.1.29.2 Diagram



11.4.3.1.29.3 Fields

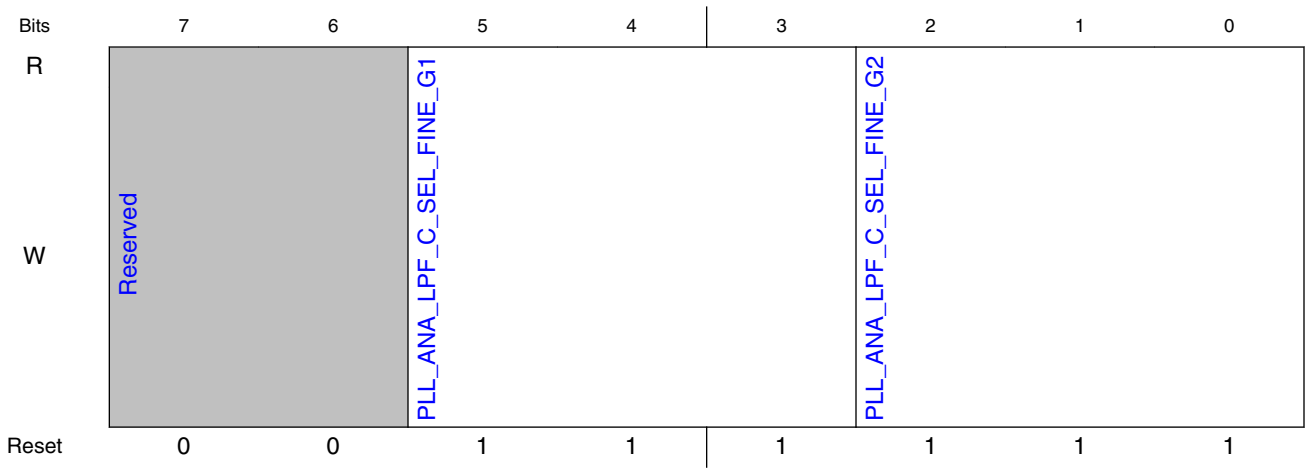
Field	Function
7-6 —	Reserved
5-3 PLL_ANA_LPF_C_SEL_COARSE_G3	[GEN3] [Coarse Mode]
2-0 PLL_ANA_LPF_C_SEL_COARSE_G4	[GEN4] [Coarse Mode]

11.4.3.1.30 (CMN_REG01C)

11.4.3.1.30.1 Offset

Register	Offset
CMN_REG01C	70h

11.4.3.1.30.2 Diagram



11.4.3.1.30.3 Fields

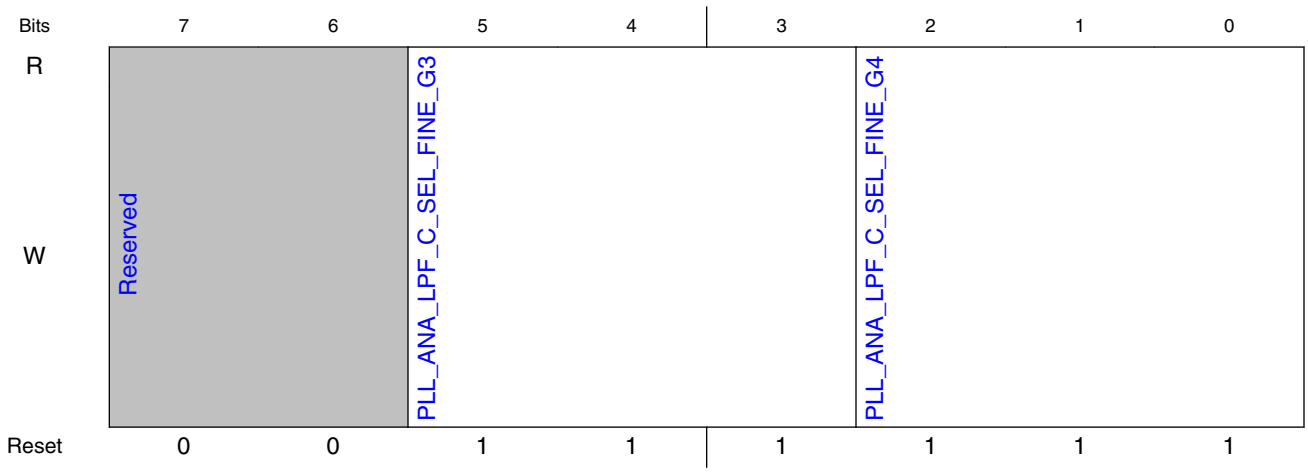
Field	Function
7-6 —	Reserved
5-3 PLL_ANA_LPF_C_SEL_FINE_G1	[GEN1] [Fine Mode] PLL loop filter capacitor control 000: Min, … 111: Max
2-0 PLL_ANA_LPF_C_SEL_FINE_G2	[GEN2] [Fine Mode]

11.4.3.1.31 (CMN_REG01D)

11.4.3.1.31.1 Offset

Register	Offset
CMN_REG01D	74h

11.4.3.1.31.2 Diagram



11.4.3.1.31.3 Fields

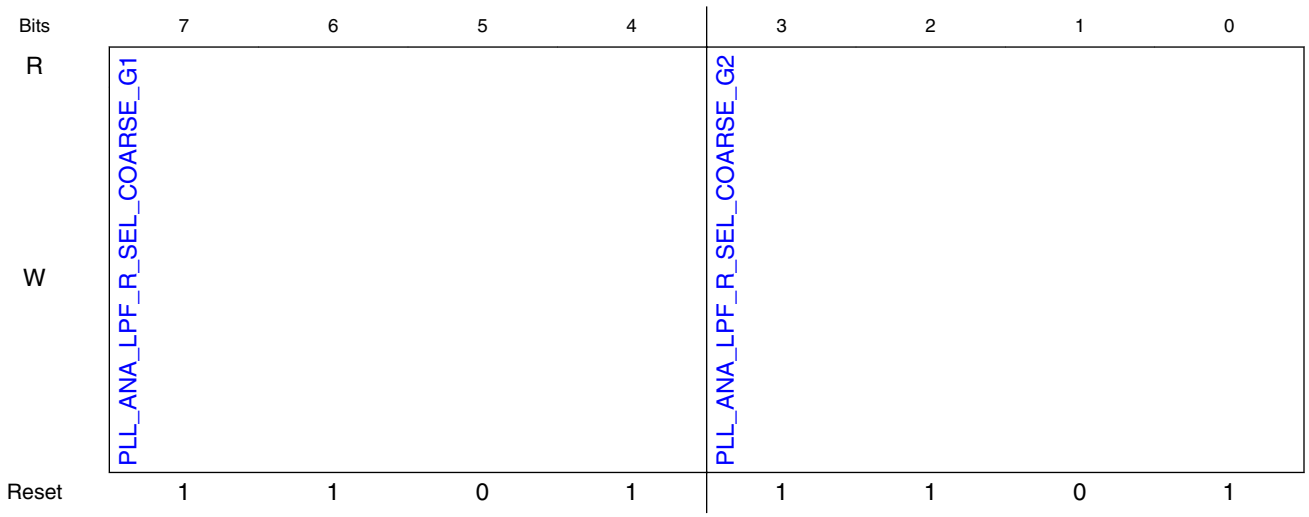
Field	Function
7-6 —	Reserved
5-3 PLL_ANA_LPF_C_SEL_FINE_G3	[GEN3] [Fine Mode]
2-0 PLL_ANA_LPF_C_SEL_FINE_G4	[GEN4] [Fine Mode]

11.4.3.1.32 (CMN_REG01E)

11.4.3.1.32.1 Offset

Register	Offset
CMN_REG01E	78h

11.4.3.1.32.2 Diagram



11.4.3.1.32.3 Fields

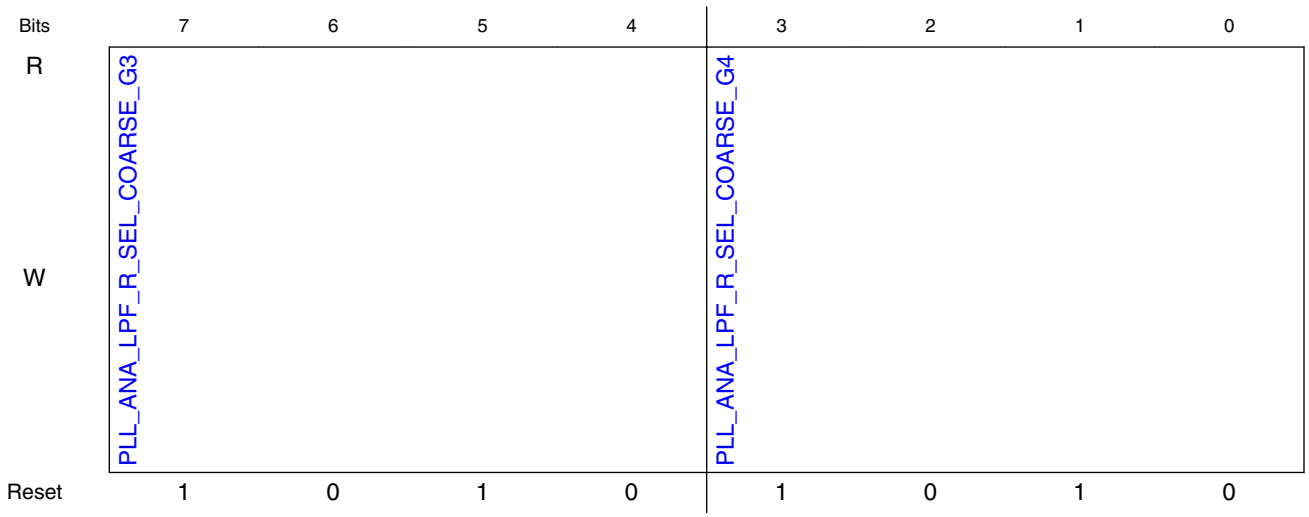
Field	Function
7-4 PLL_ANA_LPF_R_SEL_COARSE_G1	[GEN1] [Coarse Mode] PLL loop filter resistor control 0000: Min, … 1111: Max
3-0 PLL_ANA_LPF_R_SEL_COARSE_G2	[GEN2] [Coarse Mode]

11.4.3.1.33 (CMN_REG01F)

11.4.3.1.33.1 Offset

Register	Offset
CMN_REG01F	7Ch

11.4.3.1.33.2 Diagram



11.4.3.1.33.3 Fields

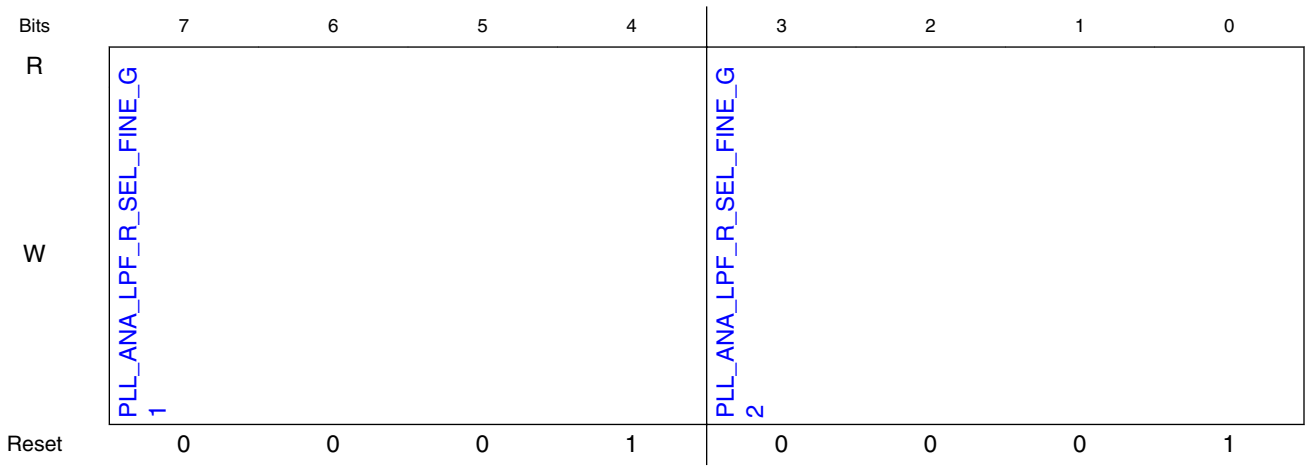
Field	Function
7-4 PLL_ANA_LPF_R_SEL_COARSE_G3	[GEN3] [Coarse Mode]
3-0 PLL_ANA_LPF_R_SEL_COARSE_G4	[GEN4] [Coarse Mode]

11.4.3.1.34 (CMN_REG020)

11.4.3.1.34.1 Offset

Register	Offset
CMN_REG020	80h

11.4.3.1.34.2 Diagram



11.4.3.1.34.3 Fields

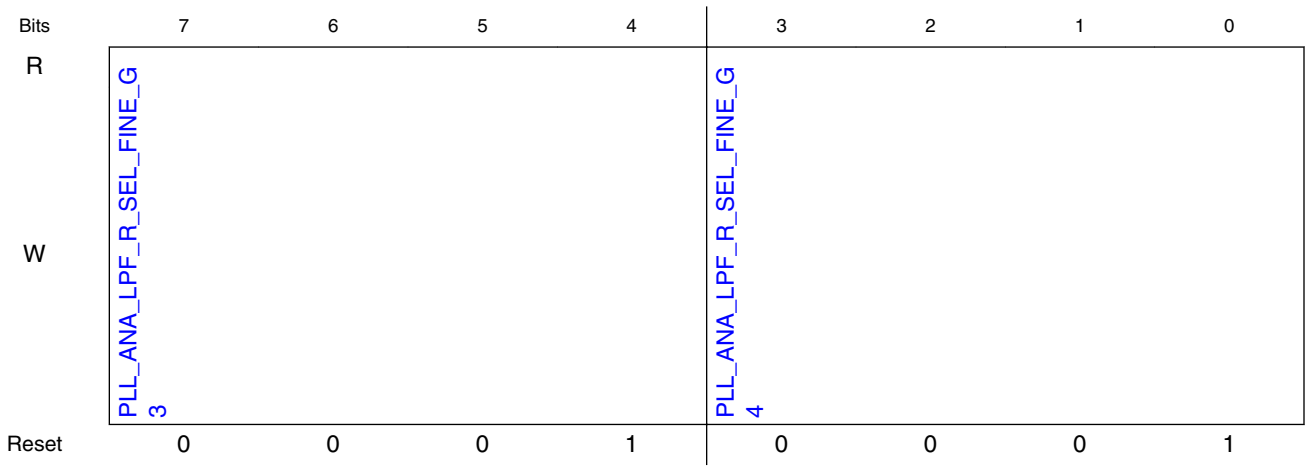
Field	Function
7-4 PLL_ANA_LPF_R_SEL_FINE_G1	[GEN1] [Fine Mode] PLL loop filter resistor control 0000: Min, … 1111: Max
3-0 PLL_ANA_LPF_R_SEL_FINE_G2	[GEN2] [Fine Mode]

11.4.3.1.35 (CMN_REG021)

11.4.3.1.35.1 Offset

Register	Offset
CMN_REG021	84h

11.4.3.1.35.2 Diagram



11.4.3.1.35.3 Fields

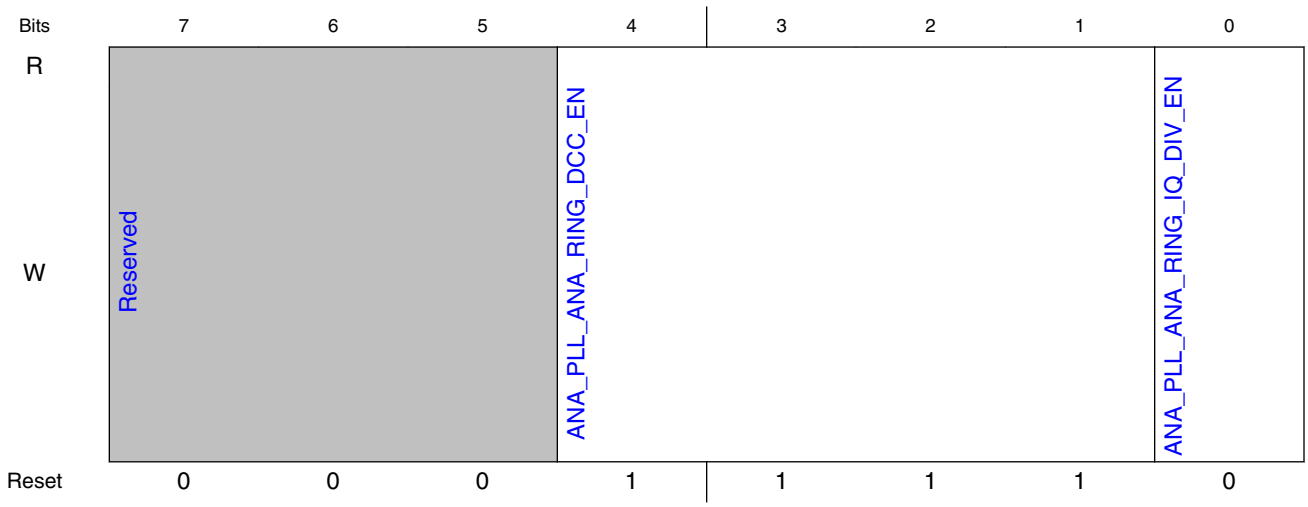
Field	Function
7-4 PLL_ANA_LPF_R_SEL_FINE_G3	[GEN3] [Fine Mode]
3-0 PLL_ANA_LPF_R_SEL_FINE_G4	[GEN4] [Fine Mode]

11.4.3.1.36 (CMN_REG022)

11.4.3.1.36.1 Offset

Register	Offset
CMN_REG022	88h

11.4.3.1.36.2 Diagram



11.4.3.1.36.3 Fields

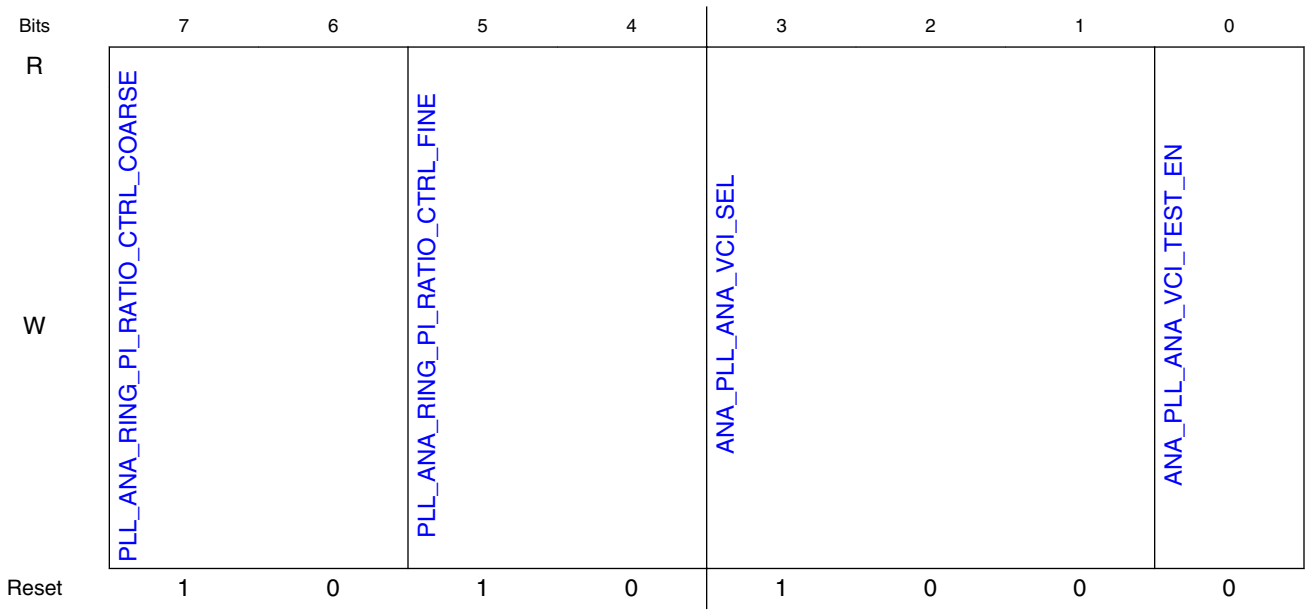
Field	Function
7-5 —	Reserved
4-1 ANA_PLL_ANA_RING_DCC_EN	PLL ring VCO DCC enable for each phase 0: Disable, 1: Enable
0 ANA_PLL_ANA_RING_IQ_DIV_EN	I/Q divider enable for PLL ring VCO clock 0: Disable, 1: Enable

11.4.3.1.37 (CMN_REG023)

11.4.3.1.37.1 Offset

Register	Offset
CMN_REG023	8Ch

11.4.3.1.37.2 Diagram



11.4.3.1.37.3 Fields

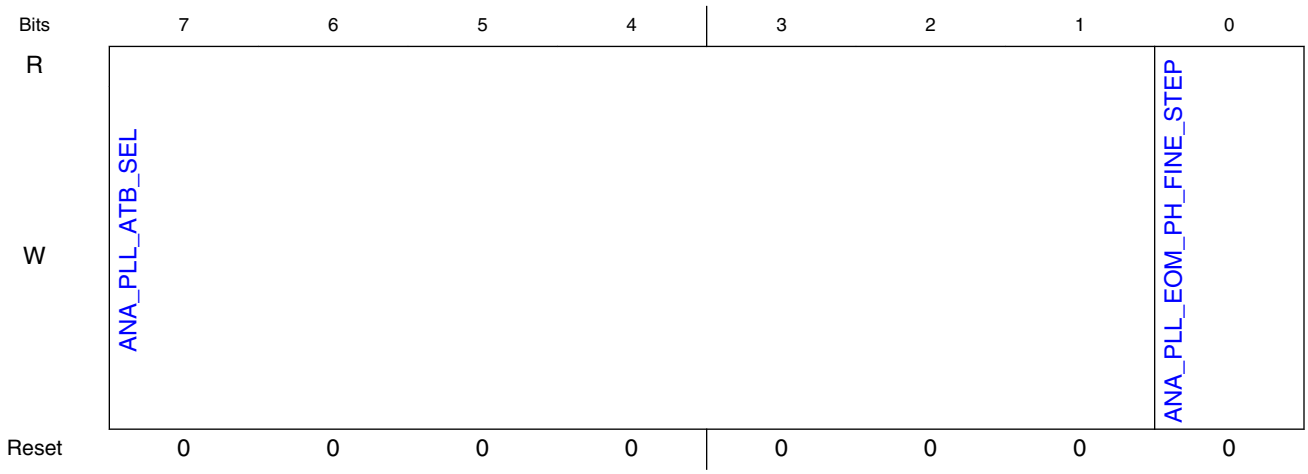
Field	Function
7-6 PLL_ANA_RING_PI_RATIO_CTRL_COARSE	Ratio between proportional and integral path gain 00: N/A, 01: 1/20, 10: 1/33, 11: N/A
5-4 PLL_ANA_RING_PI_RATIO_CTRL_FINE	Ratio between proportional and integral path gain 00: N/A, 01: 1/20, 10: 1/33, 11: N/A
3-1 ANA_PLL_ANA_VCI_SEL	PLL control voltage selection in AFC or open-loop test 000: 1.69V, 001: 1.59V, 010: 1.48V, 011: 1.38V, 100: 1.27V, 101: 1.16V, 110: 1.06V, 111: 0.95V
0 ANA_PLL_ANA_VCI_TEST_EN	PLL VCO test mode enable 0: Disable, 1: Enable

11.4.3.1.38 (CMN_REG024)

11.4.3.1.38.1 Offset

Register	Offset
CMN_REG024	90h

11.4.3.1.38.2 Diagram



11.4.3.1.38.3 Fields

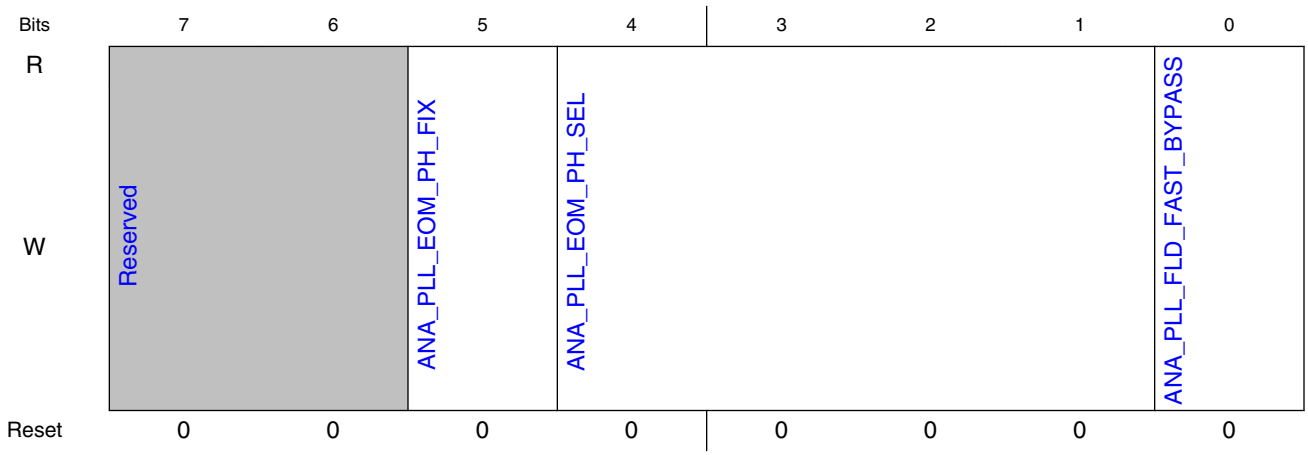
Field	Function
7-1 ANA_PLL_ATB_SEL	Select PLL ATB 0000000: No ATB (OPEN) 0000001: VCO_VCI 0000010: LF_VCI_BUF 0000100: GM_VCI 0001000: CAP_OFFSET_VCI 0010000: LC_MON_VDD_LDO 0100000: VDDH 1000000: RING_MON_VCM
0 ANA_PLL_EOM_PH_FINE_STEP	EOM phase resolution enhancement 0: Disable, 1: Double resolution

11.4.3.1.39 (CMN_REG025)

11.4.3.1.39.1 Offset

Register	Offset
CMN_REG025	94h

11.4.3.1.39.2 Diagram



11.4.3.1.39.3 Fields

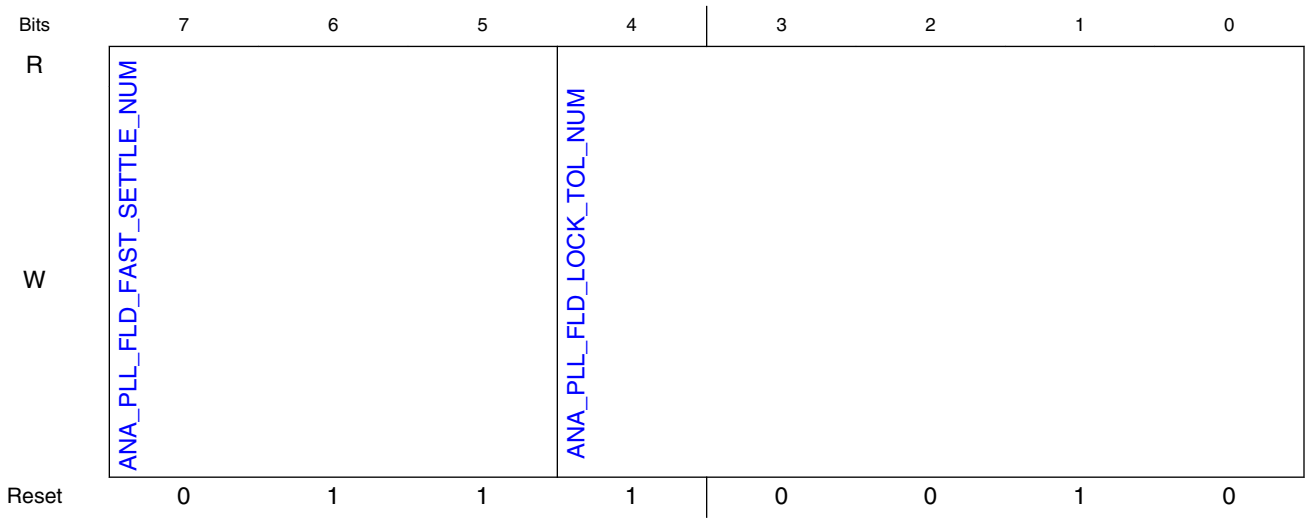
Field	Function
7-6 —	Reserved
5 ANA_PLL_EOM_PH_FIX	Phase shift enable for EOM 0: Disable, 1: Enable
4-1 ANA_PLL_EOM_PH_SEL	EOM phase selection 0000: 0°, 0001: 22.5°, 0010: 45°, 0011: 67.5° … , 1110: 315°, 1111: 337.5°
0 ANA_PLL_FLD_FAST_BYPASS	PLL fast frequency lock detection bypass 0: no bypass, 1: bypass

11.4.3.1.40 (CMN_REG026)

11.4.3.1.40.1 Offset

Register	Offset
CMN_REG026	98h

11.4.3.1.40.2 Diagram



11.4.3.1.40.3 Fields

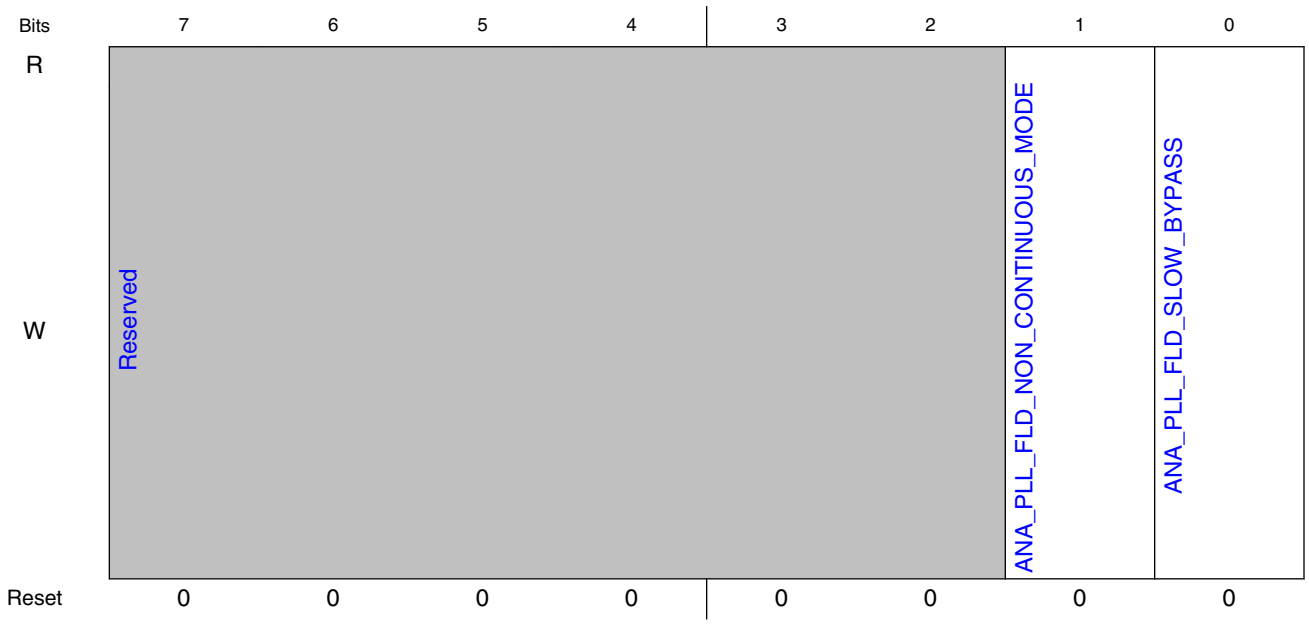
Field	Function
7-5 ANA_PLL_FLD_FAST_SETTLE_NUM	Number of reference clock cycle to check VCO stabilization in fast FLD
4-0 ANA_PLL_FLD_LOCK_TOL_NUM	FLD lock tolerance setting Lock detector activates lock signal when VCO counter value is smaller than i_pll_fld_ppm_set/(Min(1024<(i_pll_pms_m* 2^n)<2048)) * 1000000 where n=1~10, in fast FLD, and i_pll_fld_ppm_set/1024) * 1000000, In slow FLD, respectively.

11.4.3.1.41 (CMN_REG027)

11.4.3.1.41.1 Offset

Register	Offset
CMN_REG027	9Ch

11.4.3.1.41.2 Diagram



11.4.3.1.41.3 Fields

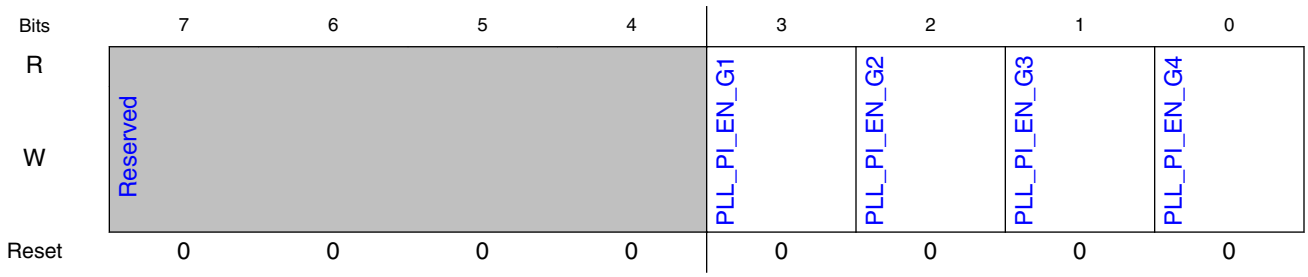
Field	Function
7-2 —	Reserved
1 ANA_PLL_FLD_NON_CONTINUOUS_MODE	Check frequency lock detection of PLL 0: once, 1: continuously
0 ANA_PLL_FLD_SLOW_BYPASS	PLL slow frequency lock detection bypass 0: no bypass, 1: bypass

11.4.3.1.42 (CMN_REG028)

11.4.3.1.42.1 Offset

Register	Offset
CMN_REG028	A0h

11.4.3.1.42.2 Diagram



11.4.3.1.42.3 Fields

Field	Function
7-4 —	Reserved
3 PLL_PI_EN_G1	[GEN1] PLL phase interpolator enable 0: Disable, 1: Enable
2 PLL_PI_EN_G2	[GEN2]
1 PLL_PI_EN_G3	[GEN3]
0 PLL_PI_EN_G4	[GEN4]

11.4.3.1.43 (CMN_REG029)

11.4.3.1.43.1 Offset

Register	Offset
CMN_REG029	A4h

11.4.3.1.43.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	PLL_PI_STR_G1				PLL_PI_STR_G2			
W								
Reset	1	1	1	1	1	1	1	1

11.4.3.1.43.3 Fields

Field	Function
7-4 PLL_PI_STR_G1	[GEN1] PLL phase interpolator input buffer strength control for Gen3 and Gen4 (for 8GHz VCO) 0000: Min. strength …. 1111: Max. strength
3-0 PLL_PI_STR_G2	[GEN2]

11.4.3.1.44 (CMN_REG02A)

11.4.3.1.44.1 Offset

Register	Offset
CMN_REG02A	A8h

11.4.3.1.44.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	PLL_PI_STR_G3				PLL_PI_STR_G4			
W								
Reset	1	1	1	1	1	1	1	1

11.4.3.1.44.3 Fields

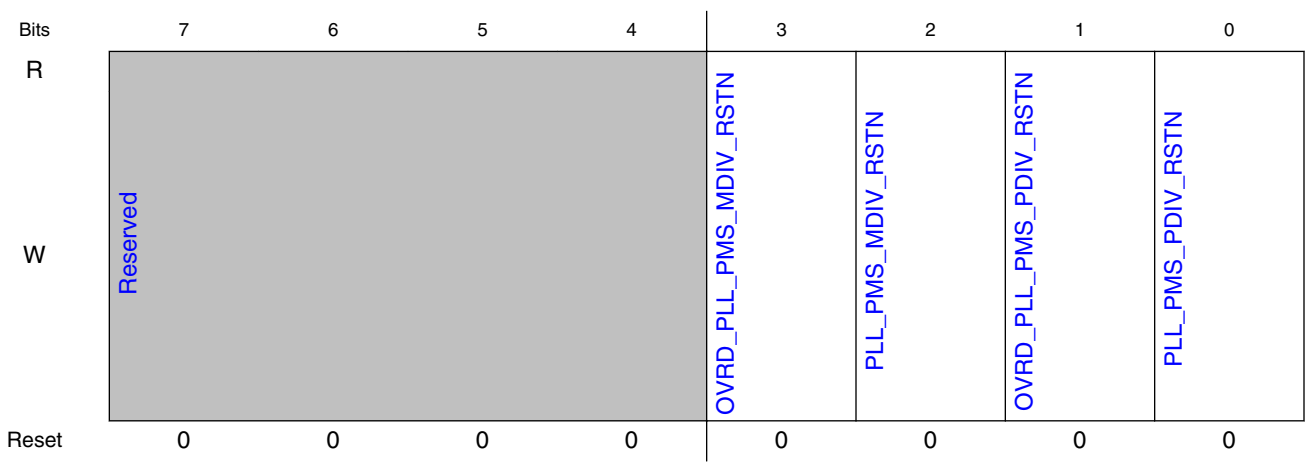
Field	Function
7-4 PLL_PI_STR_G3	[GEN3]
3-0 PLL_PI_STR_G4	[GEN4]

11.4.3.1.45 (CMN_REG02B)

11.4.3.1.45.1 Offset

Register	Offset
CMN_REG02B	ACh

11.4.3.1.45.2 Diagram



11.4.3.1.45.3 Fields

Field	Function
7-4 —	Reserved
3	Override enable for pll_pms_mdiv_rstn

Table continues on the next page...

Field	Function
OVRD_PLL_PMS_MDIV_RSTN	
2 PLL_PMS_MDIV_RSTN	PLL main divider reset 0: Reset, 1: Released
1 OVRD_PLL_PMS_PDIV_RSTN	Override enable for pll_pms_pdiv_rstn
0 PLL_PMS_PDIV_RSTN	PLL pre-divider reset 0: Reset, 1: Released

11.4.3.1.46 (CMN_REG02C)

11.4.3.1.46.1 Offset

Register	Offset
CMN_REG02C	B0h

11.4.3.1.46.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_PMS_MDIV_AFC_G1							
Reset	0	0	1	1	0	0	1	0

11.4.3.1.46.3 Fields

Field	Function
7-0 PLL_PMS_MDIV_AFC_G1	[GEN1] PLL AFC target value (fVCO/fREF) setting (Same as i_pll_pms_mdiv)

11.4.3.1.47 (CMN_REG02D)

11.4.3.1.47.1 Offset

Register	Offset
CMN_REG02D	B4h

11.4.3.1.47.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_PMS_MDIV_AFC_G2							
Reset	0	0	1	1	0	0	1	0

11.4.3.1.47.3 Fields

Field	Function
7-0 PLL_PMS_MDIV_AFC_G2	[GEN2]

11.4.3.1.48 (CMN_REG02E)

11.4.3.1.48.1 Offset

Register	Offset
CMN_REG02E	B8h

11.4.3.1.48.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_PMS_MDIV_AFC_G3							
Reset	0	0	1	0	1	0	0	0

11.4.3.1.48.3 Fields

Field	Function
7-0 PLL_PMS_MDIV_AFC_G3	[GEN3]

11.4.3.1.49 (CMN_REG02F)

11.4.3.1.49.1 Offset

Register	Offset
CMN_REG02F	BCh

11.4.3.1.49.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_PMS_MDIV_AFC_G4							
Reset	0	0	1	0	1	0	0	0

11.4.3.1.49.3 Fields

Field	Function
7-0 PLL_PMS_MDIV_AFC_G4	[GEN4]

11.4.3.1.50 (CMN_REG030)

11.4.3.1.50.1 Offset

Register	Offset
CMN_REG030	C0h

11.4.3.1.50.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_PMS_MDIV_G1							
Reset	0	0	1	1	0	0	1	0

11.4.3.1.50.3 Fields

Field	Function
7-0 PLL_PMS_MDIV_G1	[GEN1] PLL main divider setting Divider value = Setting code

11.4.3.1.51 (CMN_REG031)

11.4.3.1.51.1 Offset

Register	Offset
CMN_REG031	C4h

11.4.3.1.51.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_PMS_MDIV_G2							
Reset	0	0	1	1	0	0	1	0

11.4.3.1.51.3 Fields

Field	Function
7-0 PLL_PMS_MDIV_G2	[GEN2]

11.4.3.1.52 (CMN_REG032)

11.4.3.1.52.1 Offset

Register	Offset
CMN_REG032	C8h

11.4.3.1.52.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_PMS_MDIV_G3							
Reset	0	0	1	0	1	0	0	0

11.4.3.1.52.3 Fields

Field	Function
7-0 PLL_PMS_MDIV_G3	[GEN3]

11.4.3.1.53 (CMN_REG033)

11.4.3.1.53.1 Offset

Register	Offset
CMN_REG033	CCh

11.4.3.1.53.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_PMS_MDIV_G4							
Reset	0	0	1	0	1	0	0	0

11.4.3.1.53.3 Fields

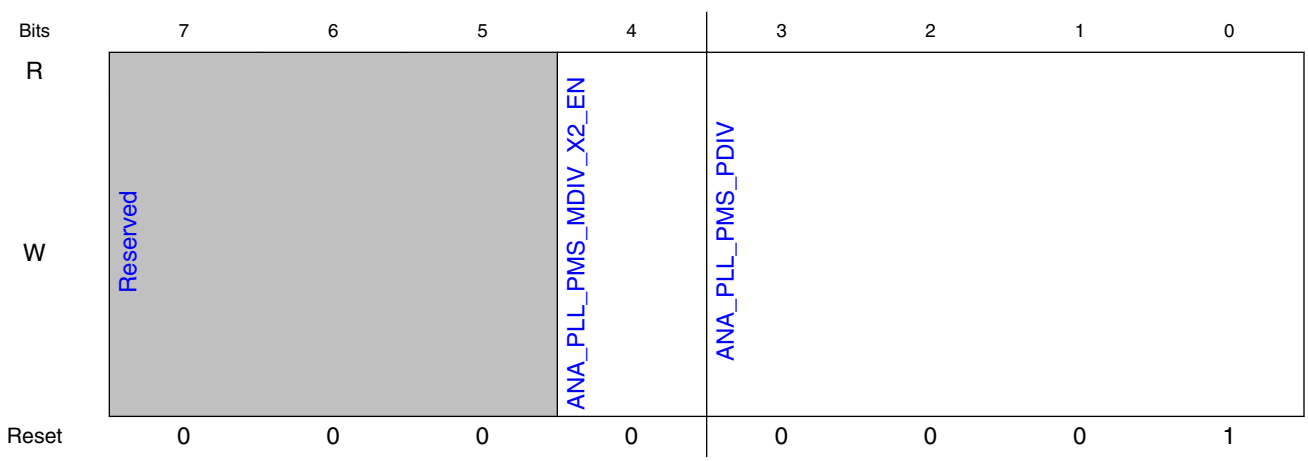
Field	Function
7-0 PLL_PMS_MDIV_G4	[GEN4]

11.4.3.1.54 (CMN_REG034)

11.4.3.1.54.1 Offset

Register	Offset
CMN_REG034	D0h

11.4.3.1.54.2 Diagram



11.4.3.1.54.3 Fields

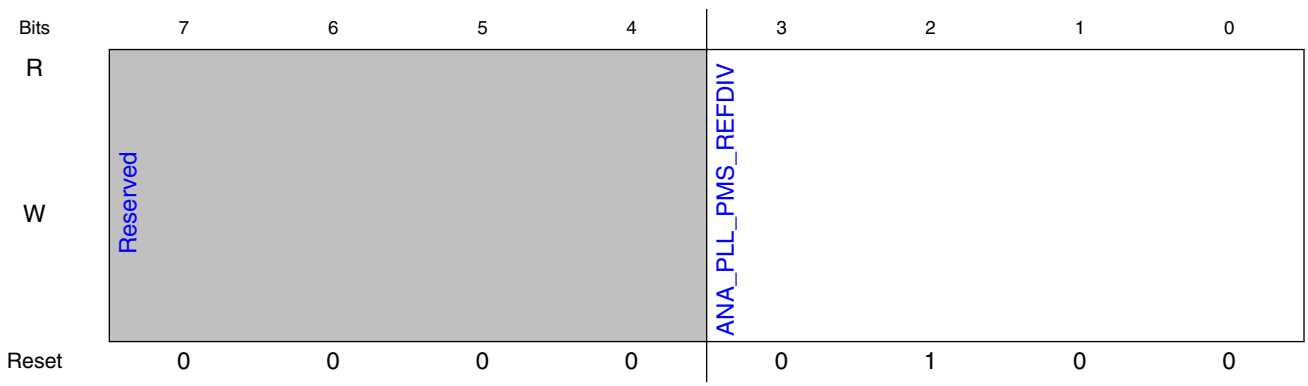
Field	Function
7-5 —	Reserved
4 ANA_PLL_PMS_MDIV_X2_EN	PLL main divider extra X2 enable 0: /1, 1: /2
3-0 ANA_PLL_PMS_PDIV	PLL pre-divider setting 0000: N/A, 0001: /1, 0010: /2, 0011: /3, … 1111: /15

11.4.3.1.55 (CMN_REG035)

11.4.3.1.55.1 Offset

Register	Offset
CMN_REG035	D4h

11.4.3.1.55.2 Diagram



11.4.3.1.55.3 Fields

Field	Function
7-4 —	Reserved
3-0 ANA_PLL_PMS_REFDIV	PLL reference clock divider setting 0000: N/A, 0001: /1, 0010: /2, 0011: /3, … 1111: /15

11.4.3.1.56 (CMN_REG036)

11.4.3.1.56.1 Offset

Register	Offset
CMN_REG036	D8h

11.4.3.1.56.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	PLL_PMS_SDIV_G1				PLL_PMS_SDIV_G2			
W								
Reset	0	0	1	1	0	0	1	0

11.4.3.1.56.3 Fields

Field	Function
7-4 PLL_PMS_SDIV_G1	[GEN1] PLL post divider setting 0000: /1, 0001: /2, 0010: /4, 0011: /8
3-0 PLL_PMS_SDIV_G2	[GEN2]

11.4.3.1.57 (CMN_REG037)

11.4.3.1.57.1 Offset

Register	Offset
CMN_REG037	DCh

11.4.3.1.57.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	PLL_PMS_SDIV_G3				PLL_PMS_SDIV_G4			
W								
Reset	0	0	0	1	0	0	0	0

11.4.3.1.57.3 Fields

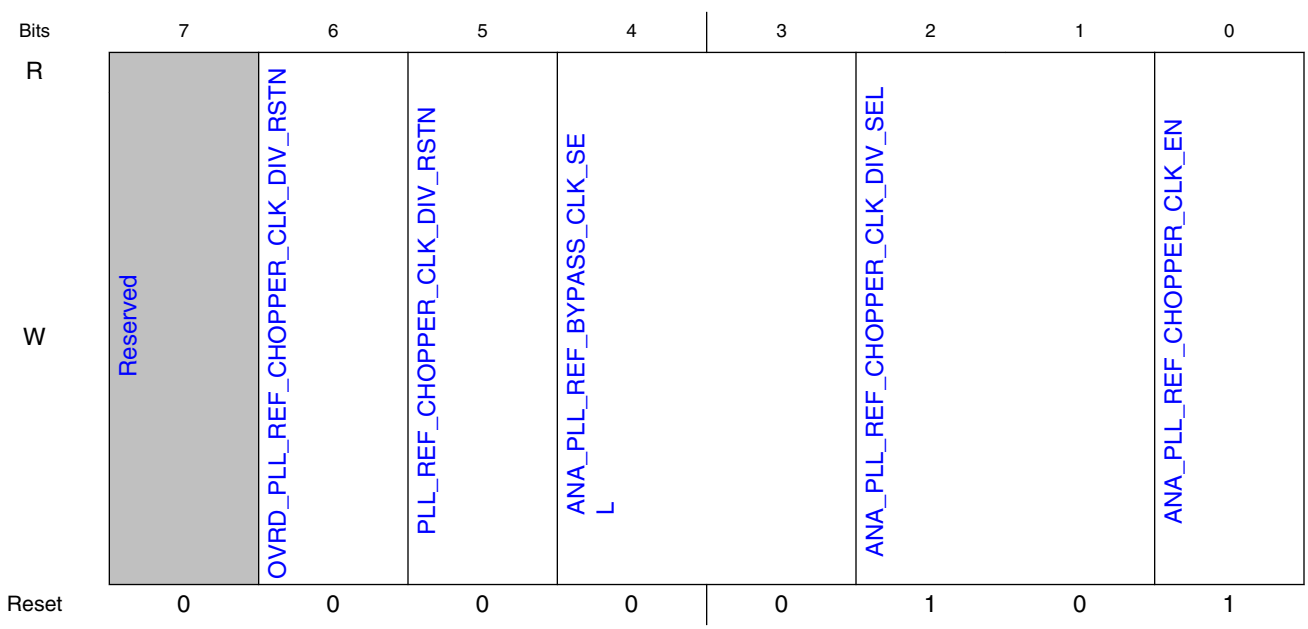
Field	Function
7-4 PLL_PMS_SDIV_G3	[GEN3]
3-0 PLL_PMS_SDIV_G4	[GEN4]

11.4.3.1.58 (CMN_REG038)

11.4.3.1.58.1 Offset

Register	Offset
CMN_REG038	E0h

11.4.3.1.58.2 Diagram



11.4.3.1.58.3 Fields

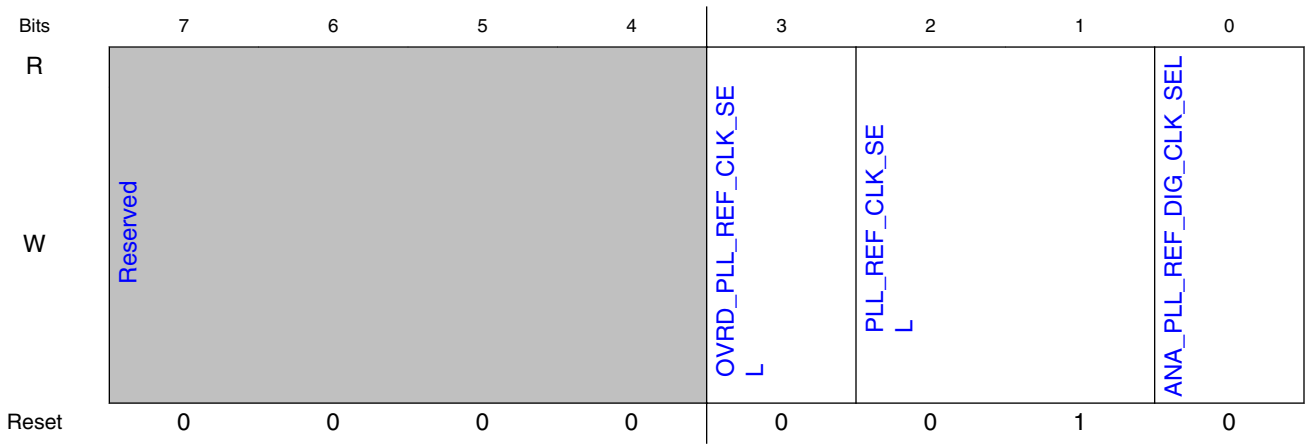
Field	Function
7 —	Reserved
6 OVRD_PLL_REF_CHOPPER_CLK_DIV_RSTN	Override enable for pll_ref_chopper_clk_div_rstn
5 PLL_REF_CHOPPER_CLK_DIV_RSTN	Chopper clk divider reset 0: Reset, 1: Released
4-3 ANA_PLL_REF_BYPASS_CLK_SEL	PLL Bypass clock selection 00: VSS 01: XO 10: External reference clock 11: System PLL clock
2-1 ANA_PLL_REF_CHOPPER_CLK_DIV_SEL	Chopper clk divider value 00: /20 (5MHz) , 01: /40 (2.5MHz), 10: /50 (2MHz), 11: /100 (1MHz)
0 ANA_PLL_REF_CHOPPER_CLK_EN	Chopper clk enable 0: Disable, 1: Enable

11.4.3.1.59 (CMN_REG039)

11.4.3.1.59.1 Offset

Register	Offset
CMN_REG039	E4h

11.4.3.1.59.2 Diagram



11.4.3.1.59.3 Fields

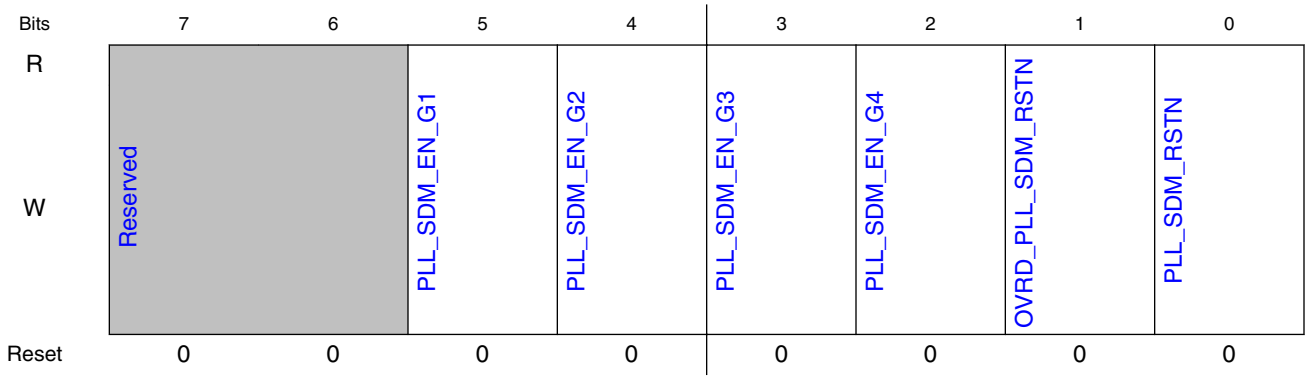
Field	Function
7-4 —	Reserved
3 OVRD_PLL_REF_CLK_SEL	Override enable for pll_ref_clk_sel
2-1 PLL_REF_CLK_SEL	PLL reference clock selection 00: VSS 01: XO 10: External reference clock 11: System PLL clock
0 ANA_PLL_REF_DIG_CLK_SEL	

11.4.3.1.60 (CMN_REG03A)

11.4.3.1.60.1 Offset

Register	Offset
CMN_REG03A	E8h

11.4.3.1.60.2 Diagram



11.4.3.1.60.3 Fields

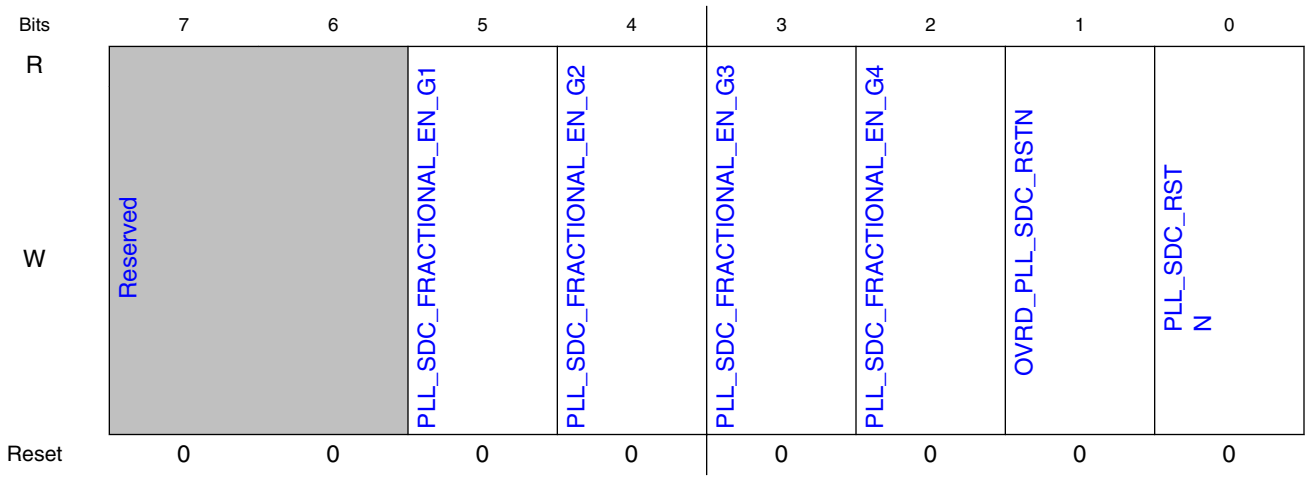
Field	Function
7-6 —	Reserved
5 PLL_SDM_EN_G1	[GEN1] PLL SDM enable 0: Disable, 1: Enable
4 PLL_SDM_EN_G2	[GEN2]
3 PLL_SDM_EN_G3	[GEN3]
2 PLL_SDM_EN_G4	[GEN4]
1 OVRD_PLL_SDM_RSTN	Override enable for pll_sdm_rstn
0 PLL_SDM_RSTN	PLL SDM reset 0: reset, 1: released

11.4.3.1.61 (CMN_REG03B)

11.4.3.1.61.1 Offset

Register	Offset
CMN_REG03B	ECh

11.4.3.1.61.2 Diagram



11.4.3.1.61.3 Fields

Field	Function
7-6 —	Reserved
5 PLL_SDC_FRACTIONAL_EN_G1	[GEN1] Fractional clock divide in SDM clock generation 0: Integer division, 1: Fractional division
4 PLL_SDC_FRACTIONAL_EN_G2	[GEN2]
3 PLL_SDC_FRACTIONAL_EN_G3	[GEN3]
2 PLL_SDC_FRACTIONAL_EN_G4	[GEN4]
1	Override enable for pll_sdc_rstn

Table continues on the next page...

Field	Function
OVRD_PLL_SDC_RSTN	
0 PLL_SDC_RSTN	PLL SDM clock generation (SDC) reset 0: Reset, 1: Release

11.4.3.1.62 (CMN_REG03C)

11.4.3.1.62.1 Offset

Register	Offset
CMN_REG03C	F0h

11.4.3.1.62.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_SDM_DENOMINATOR_G1							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.62.3 Fields

Field	Function
7-0 PLL_SDM_DENOMINATOR_G1	[GEN1] Denominator of SDM (Max. 255) Constraint: i_pll_sdm_lc > \signed i_pll_sdm_k-i_pll_ssc_fm_deviation-i_pll_ssc_fm_freq-1\

11.4.3.1.63 (CMN_REG03D)

11.4.3.1.63.1 Offset

Register	Offset
CMN_REG03D	F4h

11.4.3.1.63.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_SDM_DENOMINATOR_G2							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.63.3 Fields

Field	Function
7-0 PLL_SDM_DENOMINATOR_G2	[GEN2]

11.4.3.1.64 (CMN_REG03E)

11.4.3.1.64.1 Offset

Register	Offset
CMN_REG03E	F8h

11.4.3.1.64.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_SDM_DENOMINATOR_G3							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.64.3 Fields

Field	Function
7-0 PLL_SDM_DENOMINATOR_G3	[GEN3]

11.4.3.1.65 (CMN_REG03F)

11.4.3.1.65.1 Offset

Register	Offset
CMN_REG03F	FCh

11.4.3.1.65.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_SDM_DENOMINATOR_G4							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.65.3 Fields

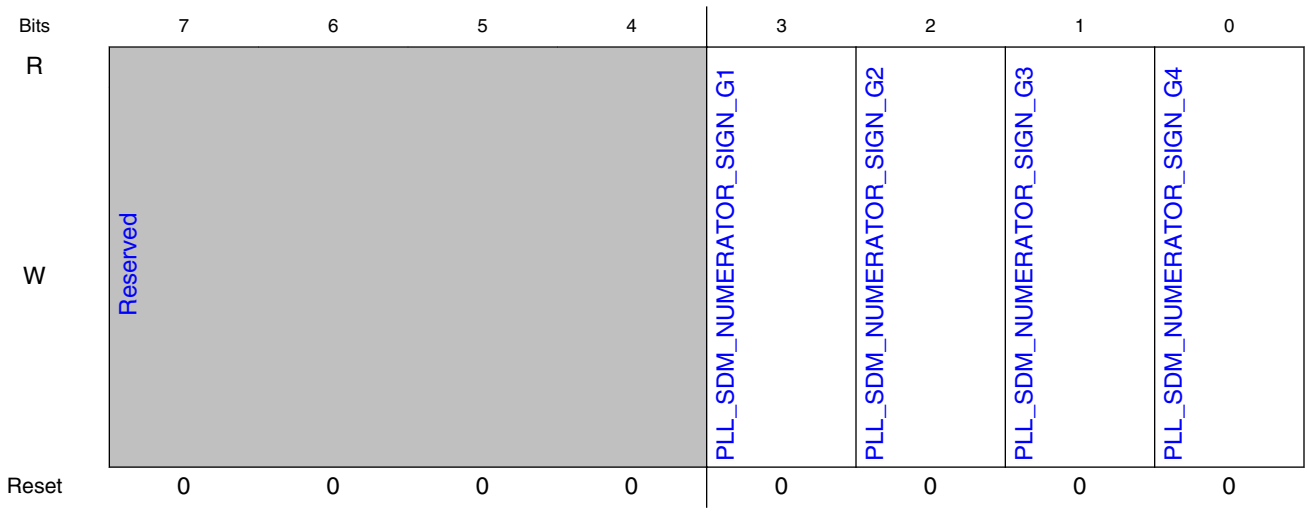
Field	Function
7-0 PLL_SDM_DENOMINATOR_G4	[GEN4]

11.4.3.1.66 (CMN_REG040)

11.4.3.1.66.1 Offset

Register	Offset
CMN_REG040	100h

11.4.3.1.66.2 Diagram



11.4.3.1.66.3 Fields

Field	Function
7-4 —	Reserved
3 PLL_SDM_NUMERATOR_SIGN_G1	[GEN1] Sign of SDM numerator 0: positive, 1: negative
2 PLL_SDM_NUMERATOR_SIGN_G2	[GEN2]
1 PLL_SDM_NUMERATOR_SIGN_G3	[GEN3]
0 PLL_SDM_NUMERATOR_SIGN_G4	[GEN4]

11.4.3.1.67 (CMN_REG041)

11.4.3.1.67.1 Offset

Register	Offset
CMN_REG041	104h

11.4.3.1.67.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_SDM_NUMERATOR_G1							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.67.3 Fields

Field	Function
7-0 PLL_SDM_NUMERATOR_G1	[GEN1] Numerator of SDM with i_pll_sdm_k_sign (-255~255)

11.4.3.1.68 (CMN_REG042)

11.4.3.1.68.1 Offset

Register	Offset
CMN_REG042	108h

11.4.3.1.68.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_SDM_NUMERATOR_G2							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.68.3 Fields

Field	Function
7-0 PLL_SDM_NUMERATOR_G2	[GEN2]

11.4.3.1.69 (CMN_REG043)

11.4.3.1.69.1 Offset

Register	Offset
CMN_REG043	10Ch

11.4.3.1.69.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_SDM_NUMERATOR_G3							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.69.3 Fields

Field	Function
7-0 PLL_SDM_NUMERATOR_G3	[GEN3]

11.4.3.1.70 (CMN_REG044)

11.4.3.1.70.1 Offset

Register	Offset
CMN_REG044	110h

11.4.3.1.70.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PLL_SDM_NUMERATOR_G4							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.70.3 Fields

Field	Function
7-0 PLL_SDM_NUMERATOR_G4	[GEN4]

11.4.3.1.71 (CMN_REG045)

11.4.3.1.71.1 Offset

Register	Offset
CMN_REG045	114h

11.4.3.1.71.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	Reserved				PLL_SDM_PH_NUM_SEL_G1	PLL_SDM_PH_NUM_SEL_G2	PLL_SDM_PH_NUM_SEL_G3	PLL_SDM_PH_NUM_SEL_G4
Reset	0	0	0	0	0	0	0	0

11.4.3.1.71.3 Fields

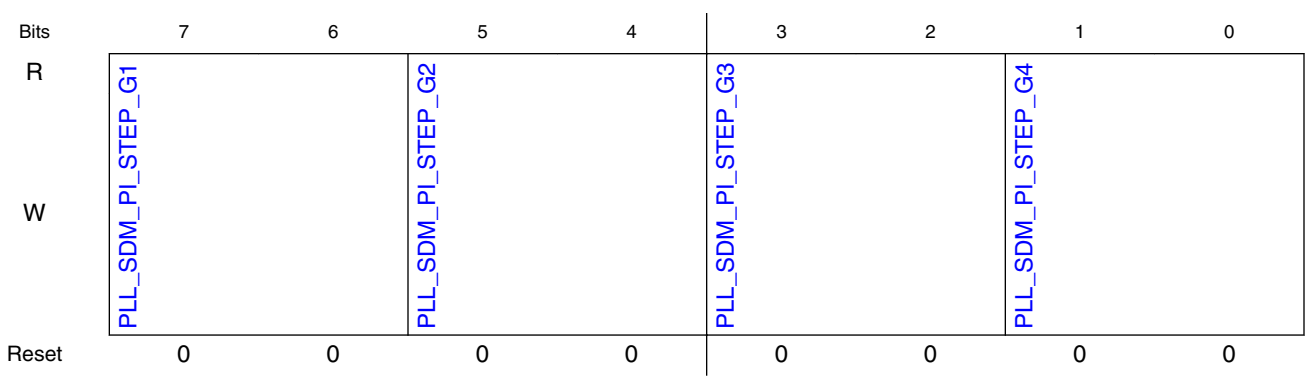
Field	Function
7-4 —	Reserved
3 PLL_SDM_PH_NUM_SEL_G1	[GEN1] PLL PI input clock phase number 0: 8-phase, 1: 4-phase
2 PLL_SDM_PH_NUM_SEL_G2	[GEN2]
1 PLL_SDM_PH_NUM_SEL_G3	[GEN3]
0 PLL_SDM_PH_NUM_SEL_G4	[GEN4]

11.4.3.1.72 (CMN_REG046)

11.4.3.1.72.1 Offset

Register	Offset
CMN_REG046	118h

11.4.3.1.72.2 Diagram



11.4.3.1.72.3 Fields

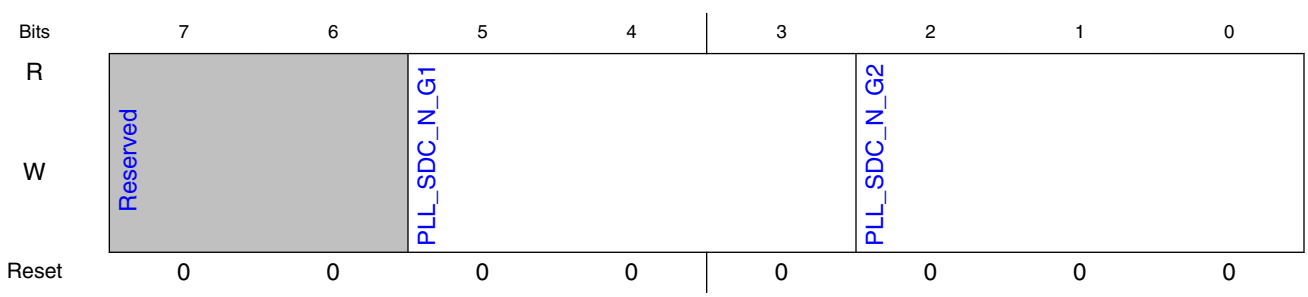
Field	Function
7-6 PLL_SDM_PI_STEP_G1	[GEN1] PLL phase interpolstor step 00: 1-step (16-phase), 01: 2-step (8-phase), 1x: 0.5-step
5-4 PLL_SDM_PI_STEP_G2	[GEN2]
3-2 PLL_SDM_PI_STEP_G3	[GEN3]
1-0 PLL_SDM_PI_STEP_G4	[GEN4]

11.4.3.1.73 (CMN_REG047)

11.4.3.1.73.1 Offset

Register	Offset
CMN_REG047	11Ch

11.4.3.1.73.2 Diagram



11.4.3.1.73.3 Fields

Field	Function
7-6 —	Reserved

Table continues on the next page...

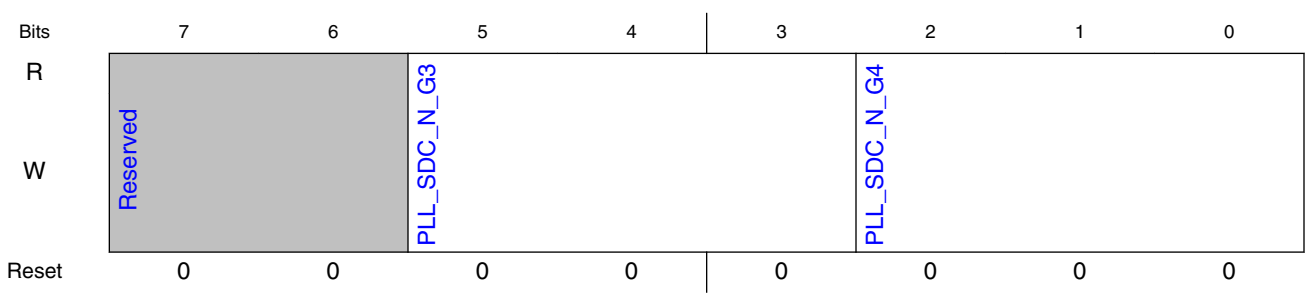
Field	Function
5-3 PLL_SDC_N_G1	[GEN1] PLL SDC divide-ratio selection 000: /3, 001: /4, 010: /5, 011: /6, 100: /7, 101: /8
2-0 PLL_SDC_N_G2	[GEN2]

11.4.3.1.74 (CMN_REG048)

11.4.3.1.74.1 Offset

Register	Offset
CMN_REG048	120h

11.4.3.1.74.2 Diagram



11.4.3.1.74.3 Fields

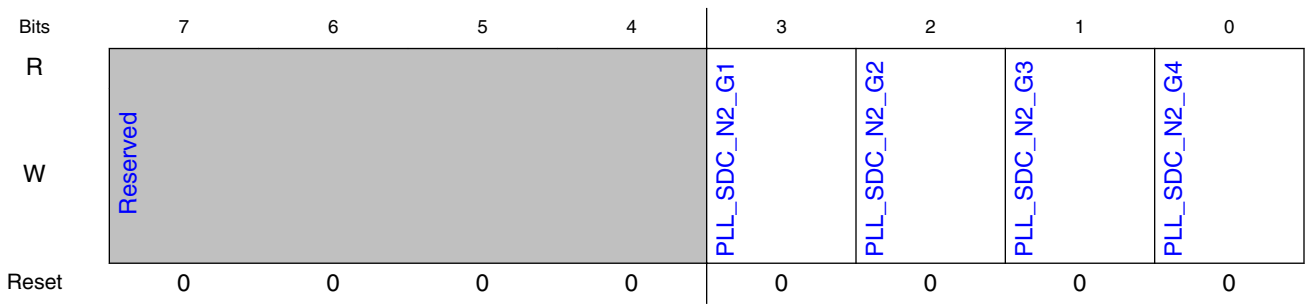
Field	Function
7-6 —	Reserved
5-3 PLL_SDC_N_G3	[GEN3]
2-0 PLL_SDC_N_G4	[GEN4]

11.4.3.1.75 (CMN_REG049)

11.4.3.1.75.1 Offset

Register	Offset
CMN_REG049	124h

11.4.3.1.75.2 Diagram



11.4.3.1.75.3 Fields

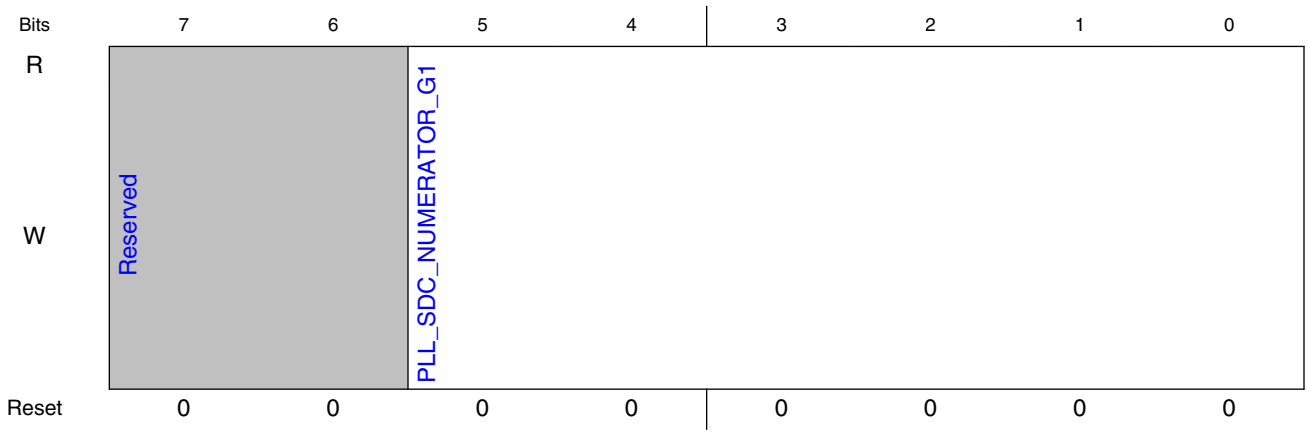
Field	Function
7-4 —	Reserved
3 PLL_SDC_N2_G1	[GEN1] PLL SDC divide-ratio selection 0: No additional /2, 1: Additional /2
2 PLL_SDC_N2_G2	[GEN2]
1 PLL_SDC_N2_G3	[GEN3]
0 PLL_SDC_N2_G4	[GEN4]

11.4.3.1.76 (CMN_REG04A)

11.4.3.1.76.1 Offset

Register	Offset
CMN_REG04A	128h

11.4.3.1.76.2 Diagram



11.4.3.1.76.3 Fields

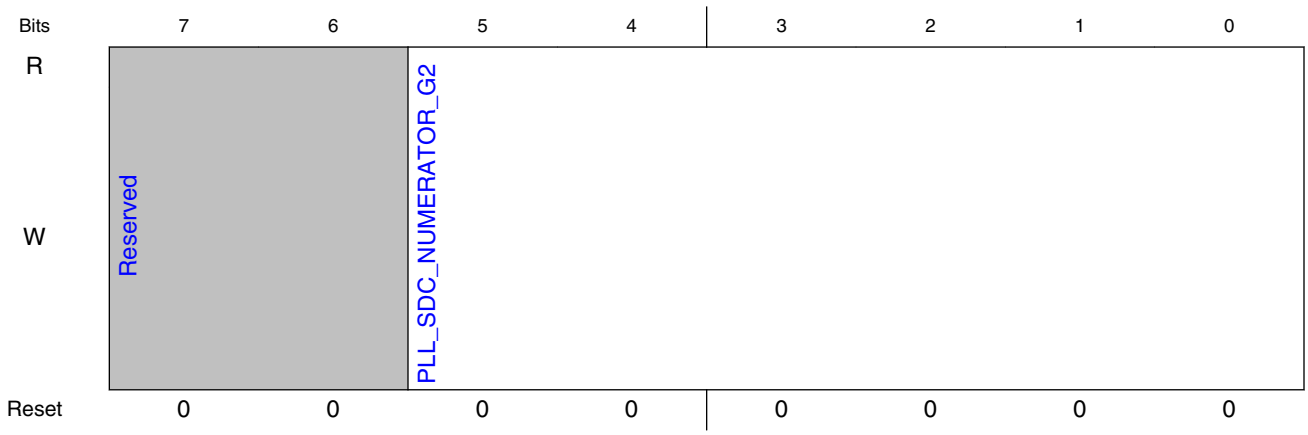
Field	Function
7-6 —	Reserved
5-0 PLL_SDC_NUMERATOR_G1	[GEN1] Numerator of SDC (Max 65)

11.4.3.1.77 (CMN_REG04B)

11.4.3.1.77.1 Offset

Register	Offset
CMN_REG04B	12Ch

11.4.3.1.77.2 Diagram



11.4.3.1.77.3 Fields

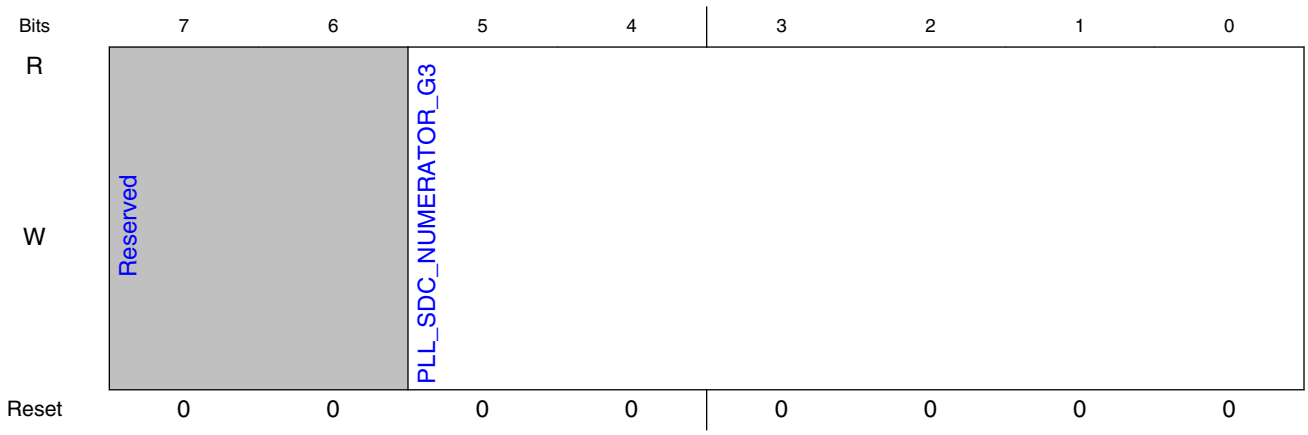
Field	Function
7-6 —	Reserved
5-0 PLL_SDC_NUMERATOR_G2	[GEN2]

11.4.3.1.78 (CMN_REG04C)

11.4.3.1.78.1 Offset

Register	Offset
CMN_REG04C	130h

11.4.3.1.78.2 Diagram



11.4.3.1.78.3 Fields

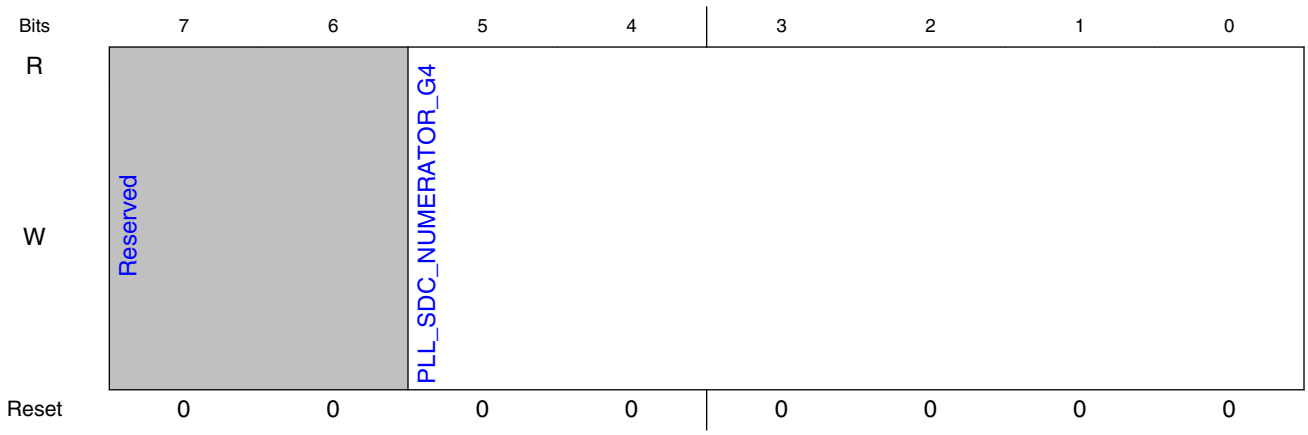
Field	Function
7-6 —	Reserved
5-0 PLL_SDC_NUMERATOR_G3	[GEN3]

11.4.3.1.79 (CMN_REG04D)

11.4.3.1.79.1 Offset

Register	Offset
CMN_REG04D	134h

11.4.3.1.79.2 Diagram



11.4.3.1.79.3 Fields

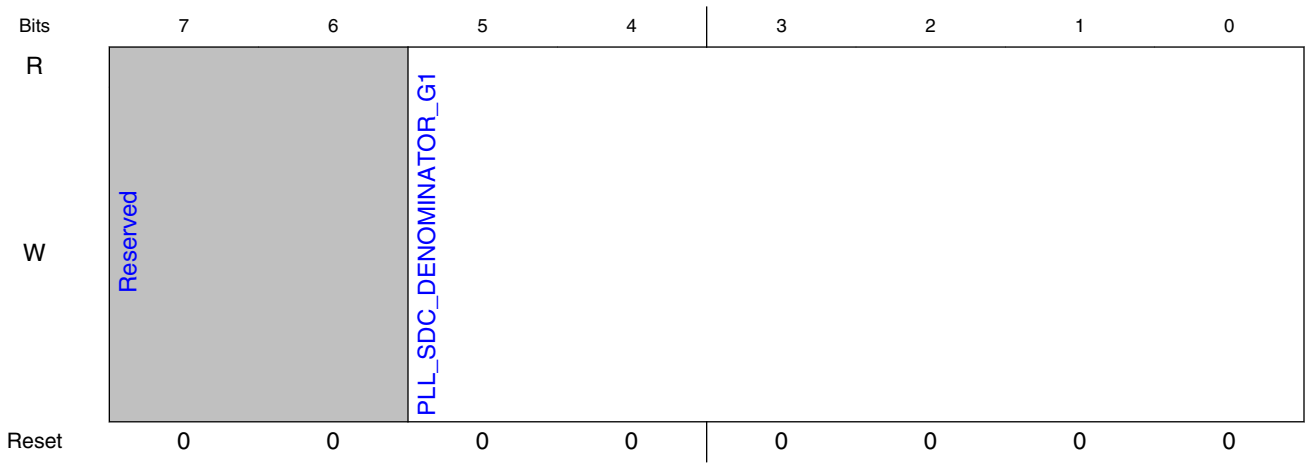
Field	Function
7-6 —	Reserved
5-0 PLL_SDC_NUMERATOR_G4	[GEN4]

11.4.3.1.80 (CMN_REG04E)

11.4.3.1.80.1 Offset

Register	Offset
CMN_REG04E	138h

11.4.3.1.80.2 Diagram



11.4.3.1.80.3 Fields

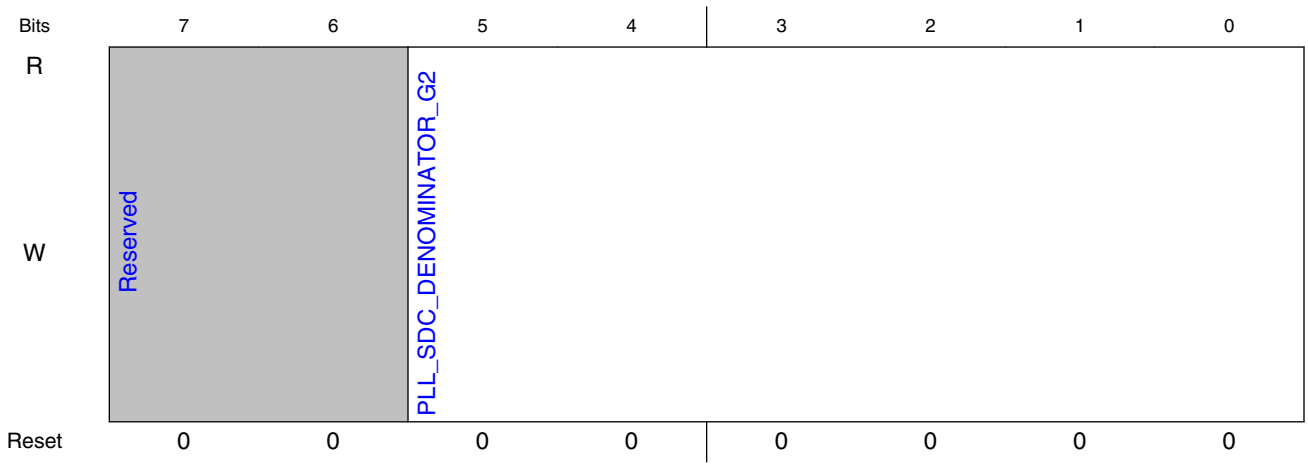
Field	Function
7-6 —	Reserved
5-0 PLL_SDC_DENOMINATOR_G1	[GEN1] Denominator of SDC (Max 65)

11.4.3.1.81 (CMN_REG04F)

11.4.3.1.81.1 Offset

Register	Offset
CMN_REG04F	13Ch

11.4.3.1.81.2 Diagram



11.4.3.1.81.3 Fields

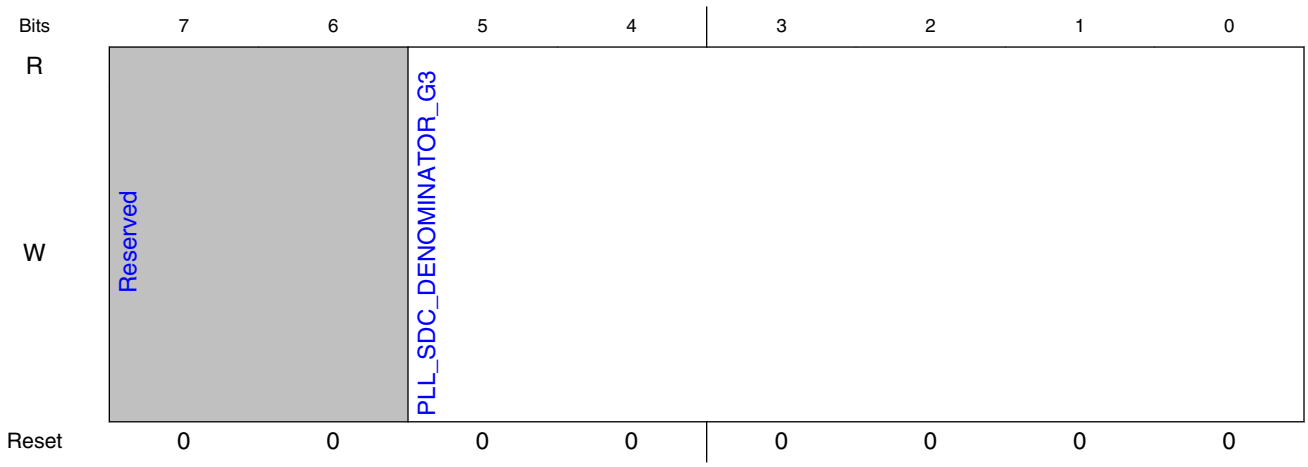
Field	Function
7-6 —	Reserved
5-0 PLL_SDC_DENOMINATOR_G2	[GEN2]

11.4.3.1.82 (CMN_REG050)

11.4.3.1.82.1 Offset

Register	Offset
CMN_REG050	140h

11.4.3.1.82.2 Diagram



11.4.3.1.82.3 Fields

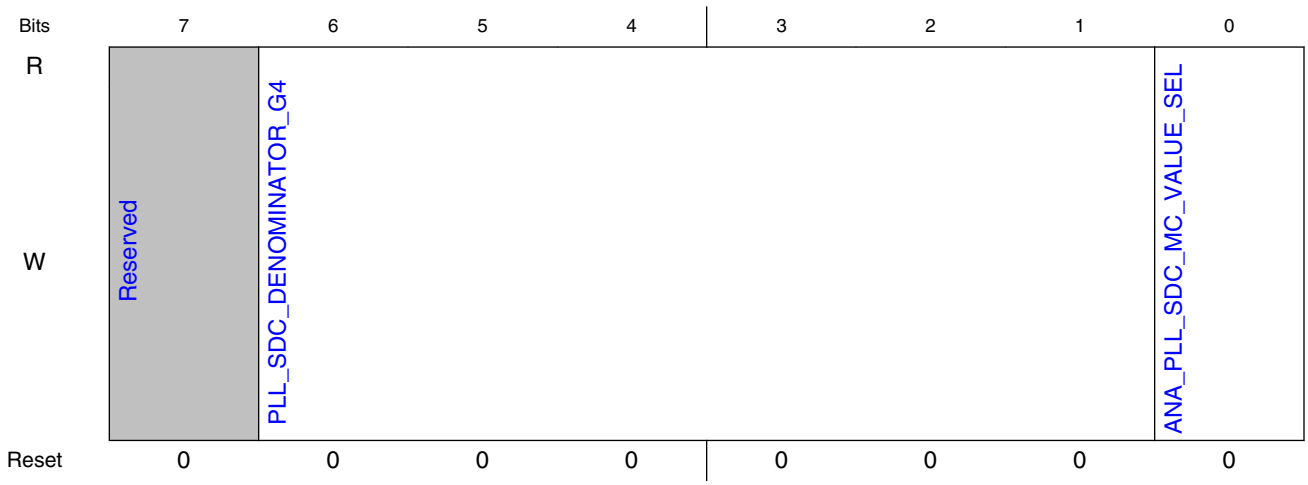
Field	Function
7-6 —	Reserved
5-0 PLL_SDC_DENOMINATOR_G3	[GEN3]

11.4.3.1.83 (CMN_REG051)

11.4.3.1.83.1 Offset

Register	Offset
CMN_REG051	144h

11.4.3.1.83.2 Diagram



11.4.3.1.83.3 Fields

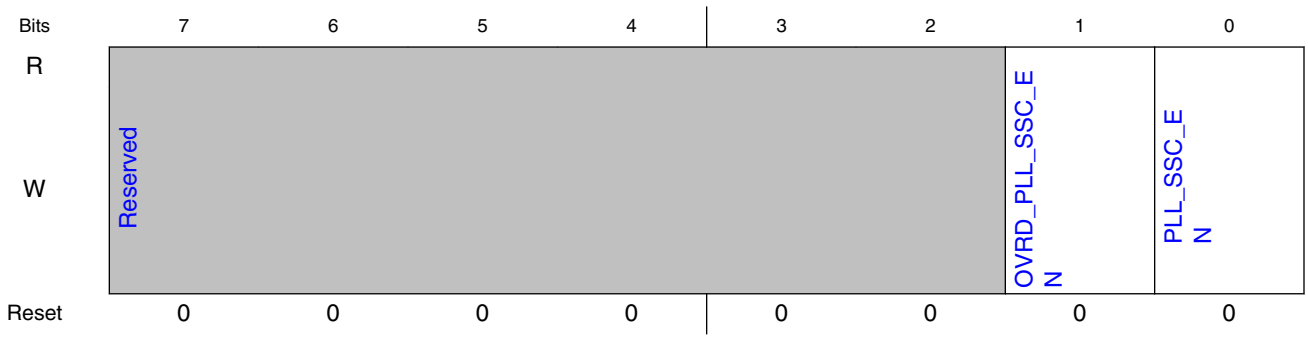
Field	Function
7 —	Reserved
6-1 PLL_SDC_DENOMINATOR_G4	[GEN4]
0 ANA_PLL_SDC_MC_VALUE_SEL	PLL SDC value force 0: SDC MC = sdc_n, 1: SDC_MC = sdc_n-1

11.4.3.1.84 (CMN_REG052)

11.4.3.1.84.1 Offset

Register	Offset
CMN_REG052	148h

11.4.3.1.84.2 Diagram



11.4.3.1.84.3 Fields

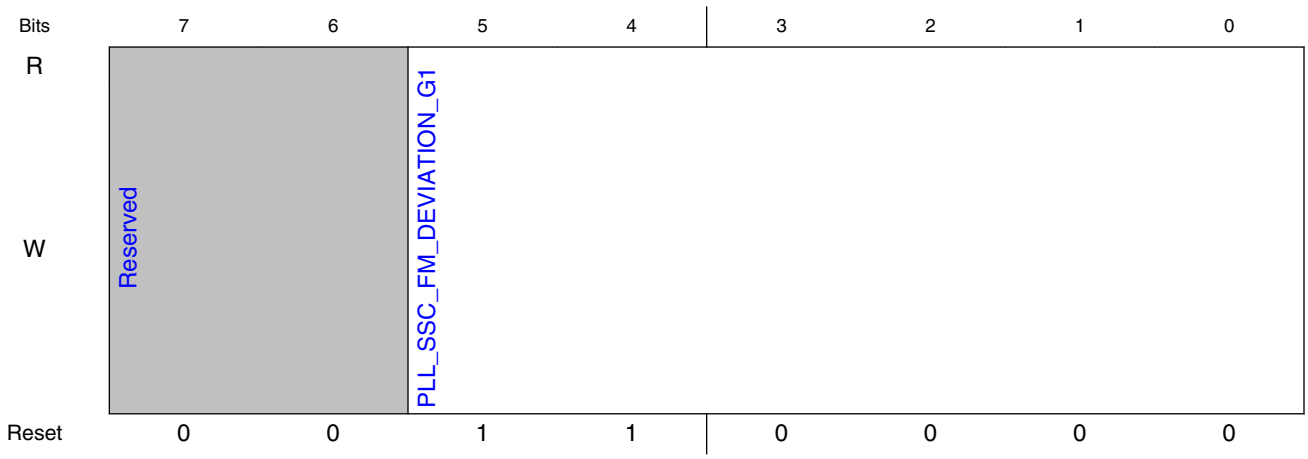
Field	Function
7-2 —	Reserved
1 OVRD_PLL_SSC_EN	Override enable for pll_ssc_en
0 PLL_SSC_EN	PLL SSC enable 0: Disable, 1: Enable

11.4.3.1.85 (CMN_REG053)

11.4.3.1.85.1 Offset

Register	Offset
CMN_REG053	14Ch

11.4.3.1.85.2 Diagram



11.4.3.1.85.3 Fields

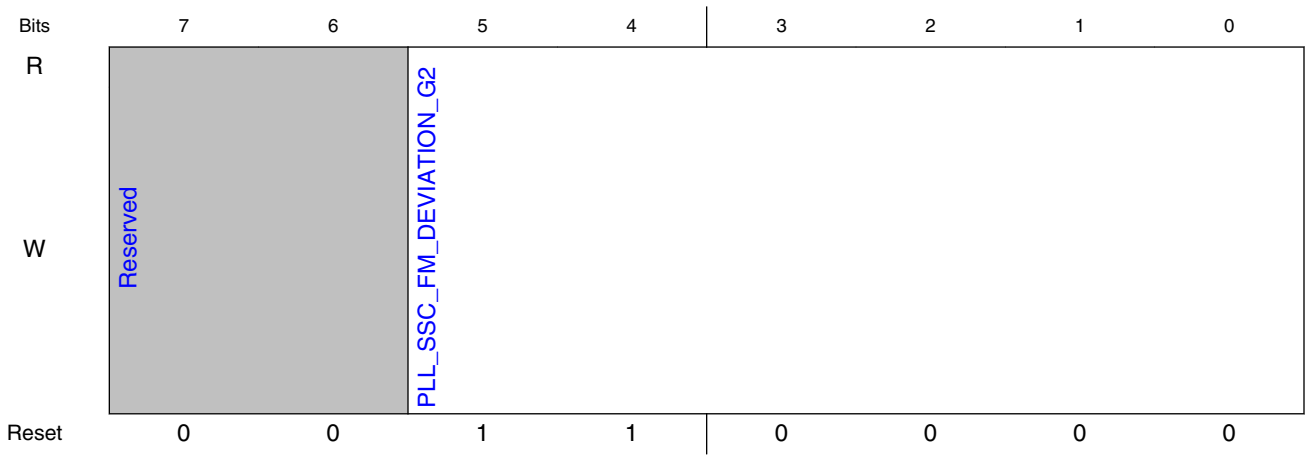
Field	Function
7-6 —	Reserved
5-0 PLL_SSC_FM_DEVIATION_G1	[GEN1] PLL SSC modulation deviation

11.4.3.1.86 (CMN_REG054)

11.4.3.1.86.1 Offset

Register	Offset
CMN_REG054	150h

11.4.3.1.86.2 Diagram



11.4.3.1.86.3 Fields

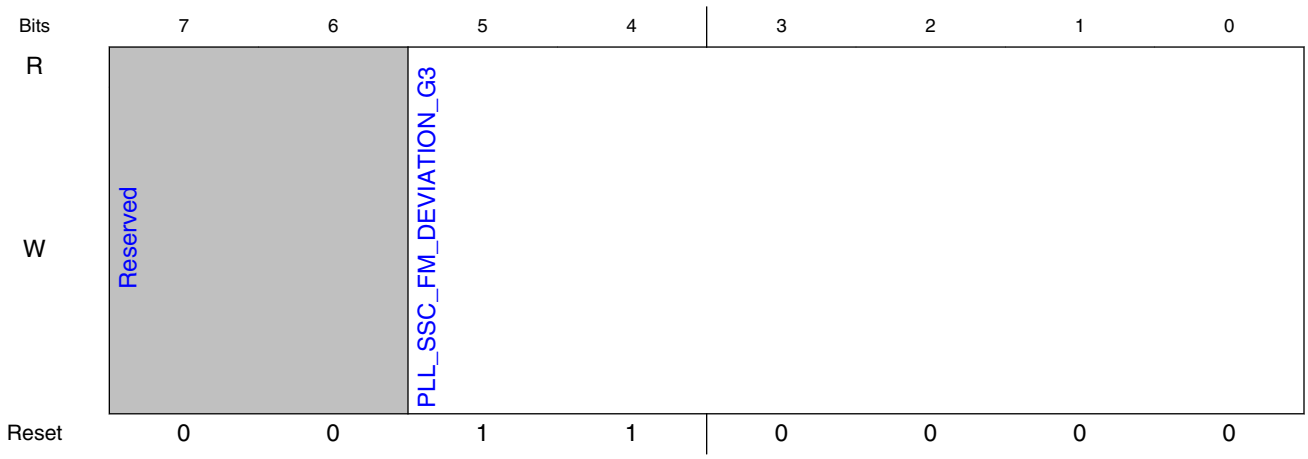
Field	Function
7-6 —	Reserved
5-0 PLL_SSC_FM_DEVIATION_G2	[GEN2]

11.4.3.1.87 (CMN_REG055)

11.4.3.1.87.1 Offset

Register	Offset
CMN_REG055	154h

11.4.3.1.87.2 Diagram



11.4.3.1.87.3 Fields

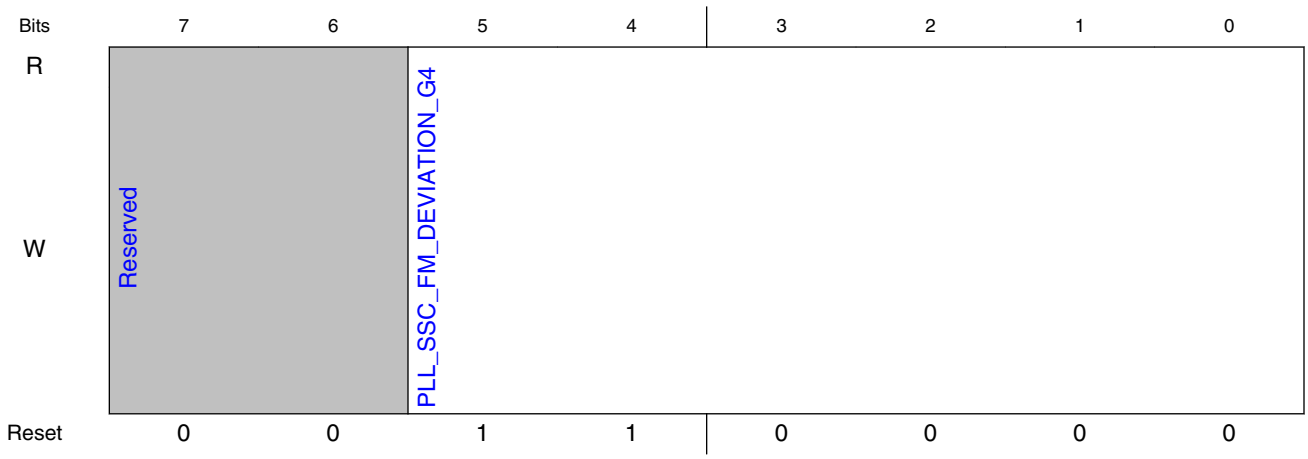
Field	Function
7-6 —	Reserved
5-0 PLL_SSC_FM_DEVIATION_G3	[GEN3]

11.4.3.1.88 (CMN_REG056)

11.4.3.1.88.1 Offset

Register	Offset
CMN_REG056	158h

11.4.3.1.88.2 Diagram



11.4.3.1.88.3 Fields

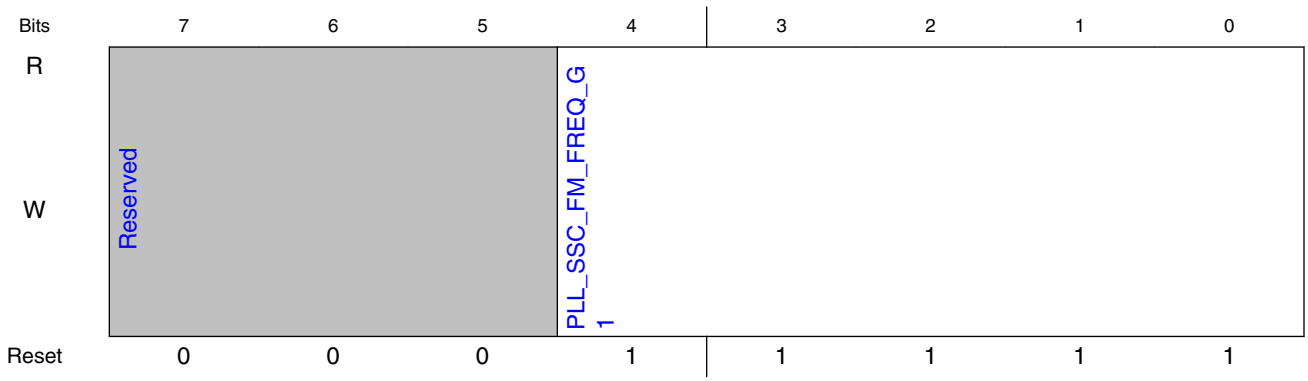
Field	Function
7-6 —	Reserved
5-0 PLL_SSC_FM_DEVIATION_G4	[GEN4]

11.4.3.1.89 (CMN_REG057)

11.4.3.1.89.1 Offset

Register	Offset
CMN_REG057	15Ch

11.4.3.1.89.2 Diagram



11.4.3.1.89.3 Fields

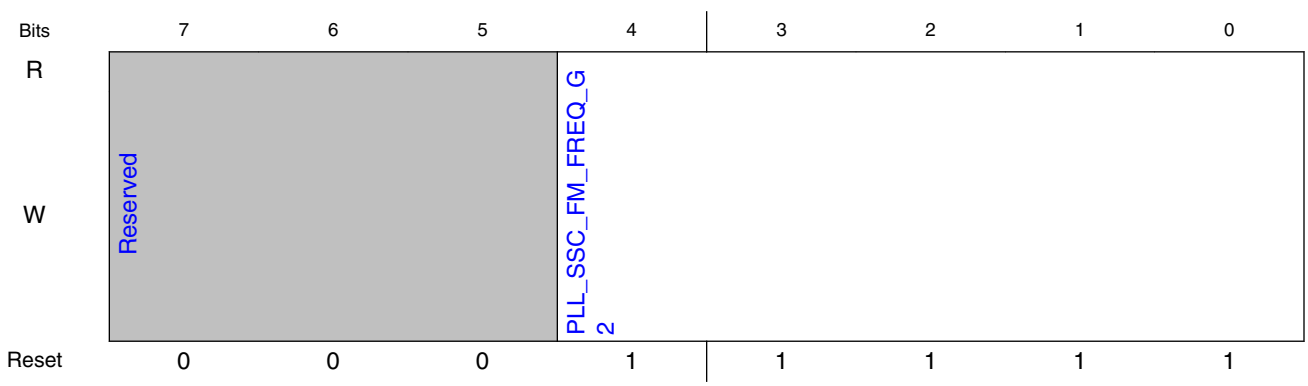
Field	Function
7-5 —	Reserved
4-0 PLL_SSC_FM_FREQ_G1	[GEN1] PLL SSC modulation frequency

11.4.3.1.90 (CMN_REG058)

11.4.3.1.90.1 Offset

Register	Offset
CMN_REG058	160h

11.4.3.1.90.2 Diagram



11.4.3.1.90.3 Fields

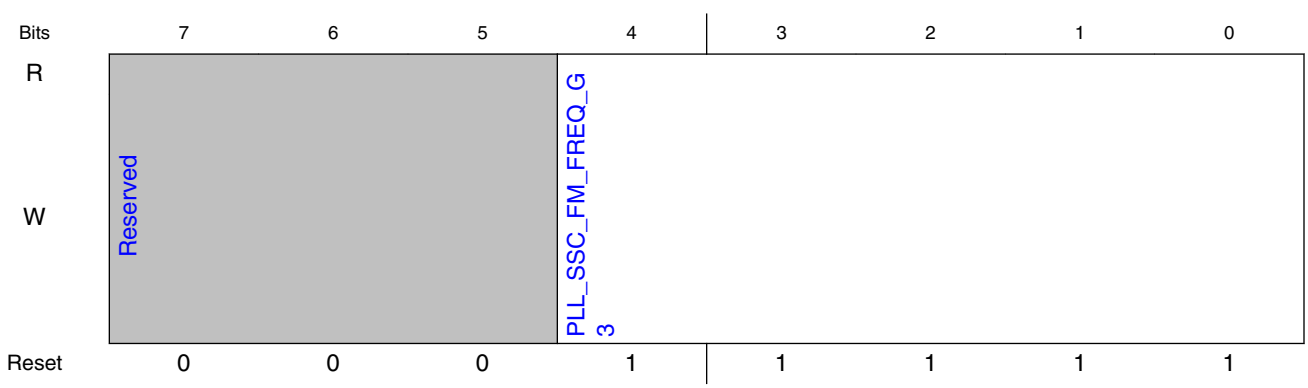
Field	Function
7-5 —	Reserved
4-0 PLL_SSC_FM_FREQ_G2	[GEN2]

11.4.3.1.91 (CMN_REG059)

11.4.3.1.91.1 Offset

Register	Offset
CMN_REG059	164h

11.4.3.1.91.2 Diagram



11.4.3.1.91.3 Fields

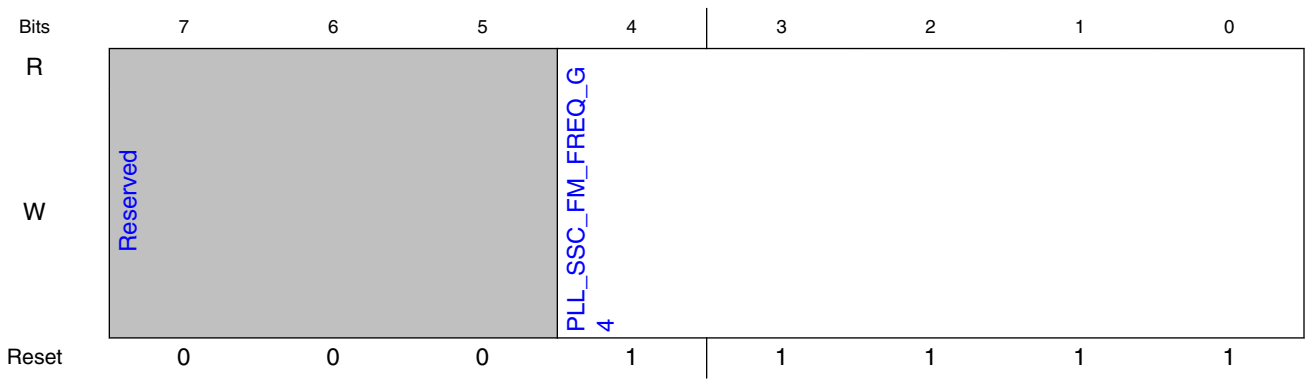
Field	Function
7-5 —	Reserved
4-0 PLL_SSC_FM_FREQ_G3	[GEN3]

11.4.3.1.92 (CMN_REG05A)

11.4.3.1.92.1 Offset

Register	Offset
CMN_REG05A	168h

11.4.3.1.92.2 Diagram



11.4.3.1.92.3 Fields

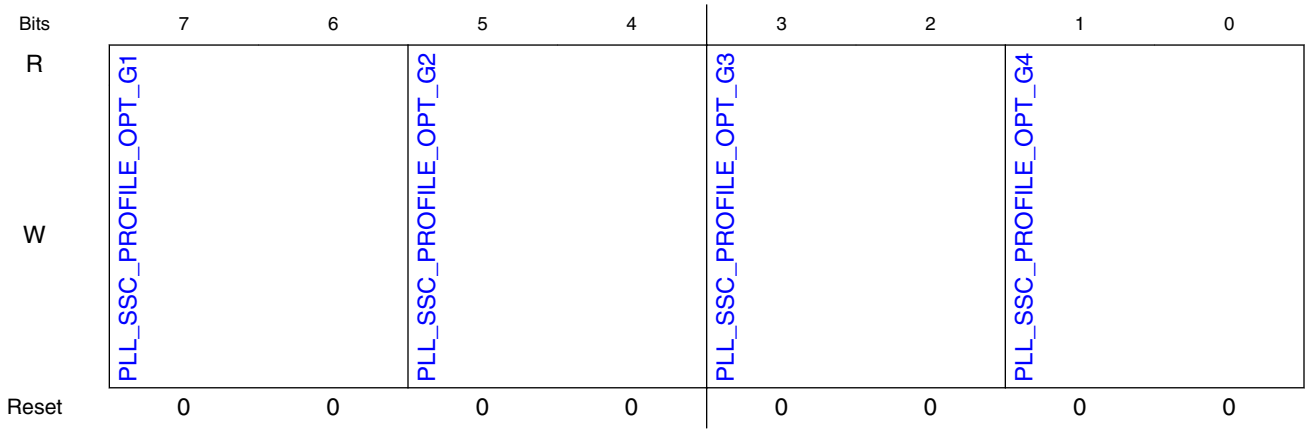
Field	Function
7-5 —	Reserved
4-0 PLL_SSC_FM_FREQ_G4	[GEN4]

11.4.3.1.93 (CMN_REG05B)

11.4.3.1.93.1 Offset

Register	Offset
CMN_REG05B	16Ch

11.4.3.1.93.2 Diagram



11.4.3.1.93.3 Fields

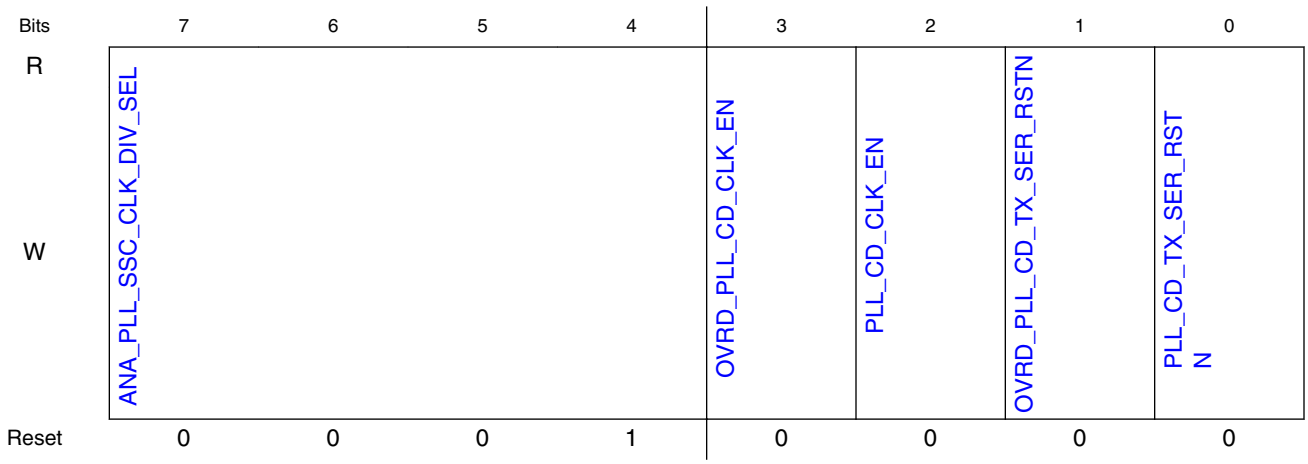
Field	Function
7-6 PLL_SSC_PROFILE_OPT_G1	[GEN1] PLL SSC modulation profile shape control 00: No jump, x1:2-level jump high edge , 1x: 2-level jump low edge
5-4 PLL_SSC_PROFILE_OPT_G2	[GEN2]
3-2 PLL_SSC_PROFILE_OPT_G3	[GEN3]
1-0 PLL_SSC_PROFILE_OPT_G4	[GEN4]

11.4.3.1.94 (CMN_REG05C)

11.4.3.1.94.1 Offset

Register	Offset
CMN_REG05C	170h

11.4.3.1.94.2 Diagram



11.4.3.1.94.3 Fields

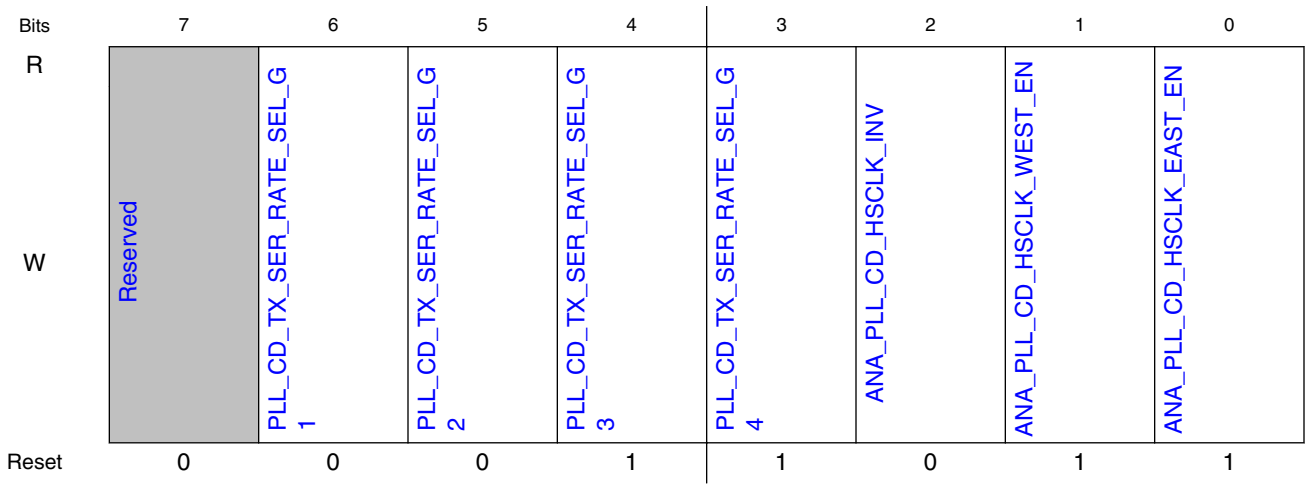
Field	Function
7-4 ANA_PLL_SSC_CLK_DIV_SEL	PLL SSC clock divide ratio 0000: N/A, 0001: /1, 0010: /2, …, 1111: /15
3 OVRD_PLL_CD_CLK_EN	Override enable for pll_cd_clk_en
2 PLL_CD_CLK_EN	CD enable 0: Disable, 1: Enable
1 OVRD_PLL_CD_TX_SER_RSTN	Override enable for pll_cd_tx_ser_rstn
0 PLL_CD_TX_SER_RSTN	TX_SER resetn

11.4.3.1.95 (CMN_REG05D)

11.4.3.1.95.1 Offset

Register	Offset
CMN_REG05D	174h

11.4.3.1.95.2 Diagram



11.4.3.1.95.3 Fields

Field	Function
7 —	Reserved
6 PLL_CD_TX_SER_RATE_SEL_G1	[GEN1] TX serializer data rate selection for Gen4 (Need to be controlled with i_tx_en_40bit) 0: 20/40-bit 1: 16/32-bit
5 PLL_CD_TX_SER_RATE_SEL_G2	[GEN2]
4 PLL_CD_TX_SER_RATE_SEL_G3	[GEN3]
3 PLL_CD_TX_SER_RATE_SEL_G4	[GEN4]
2 ANA_PLL_CD_HSCLK_INV	CD output clock polarity inversion 0: No swap, 1: P/N swap
1 ANA_PLL_CD_HSCLK_WEST_EN	CD driver nmos strength control N/A

Table continues on the next page...

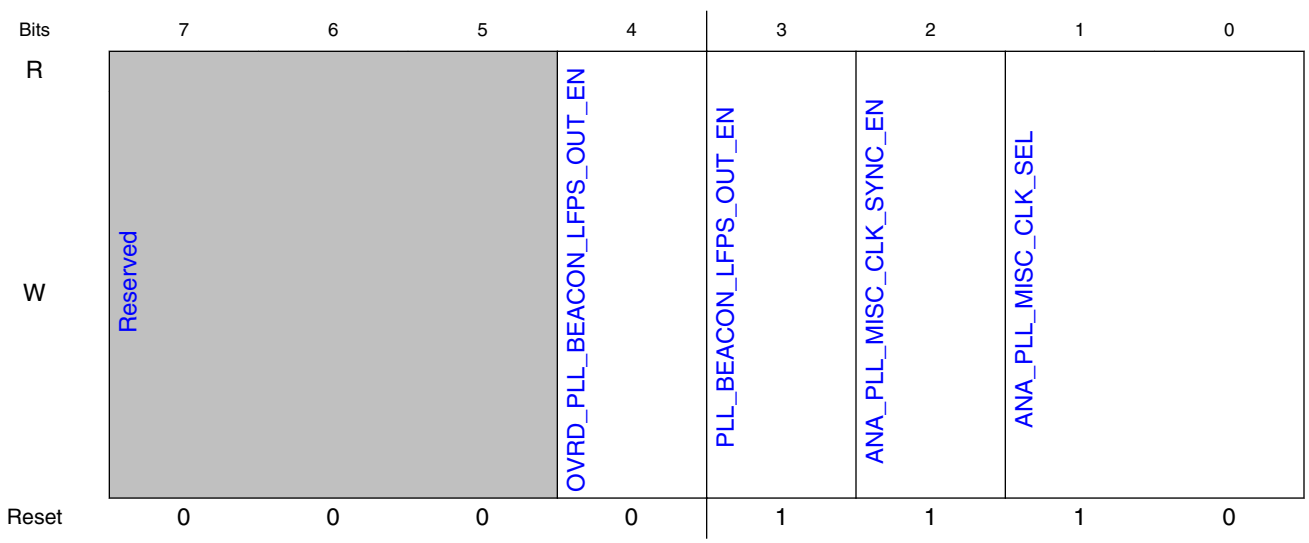
Field	Function
0 ANA_PLL_CD_HSCLK_EAST_EN	CD driver pmos strength control N/A

11.4.3.1.96 (CMN_REG05E)

11.4.3.1.96.1 Offset

Register	Offset
CMN_REG05E	178h

11.4.3.1.96.2 Diagram



11.4.3.1.96.3 Fields

Field	Function
7-5 —	Reserved
4 OVRD_PLL_BEACON_LFPS_OUT_EN	Override enable for pll_beacon_lfps_out_en

Table continues on the next page...

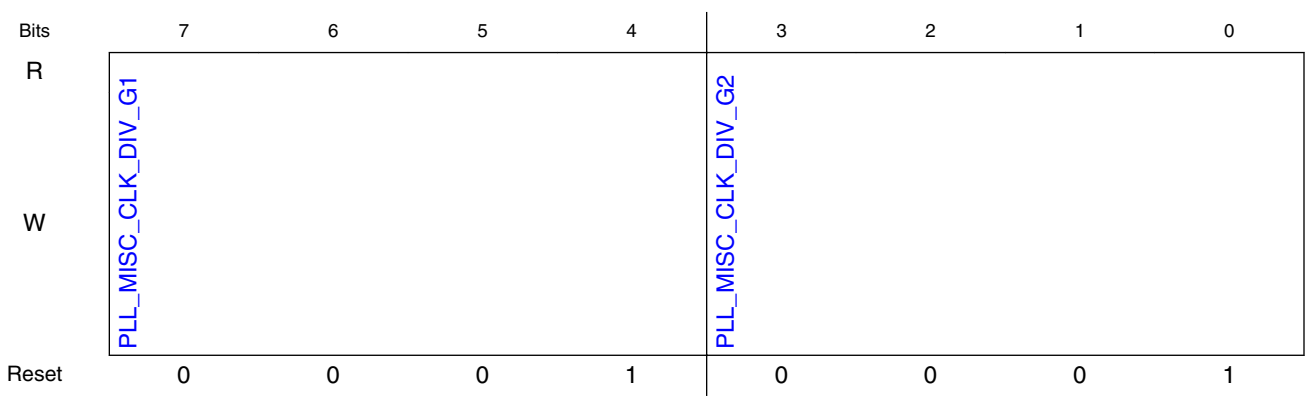
Field	Function
3 PLL_BEACON_LFPS_OUT_EN	TX beacon clock enable 0: Disable, 1: Enable
2 ANA_PLL_MISC_CLK_SYNC_EN	PLL miscellaneous clock synchronization enable If enabled, PLL misc clock output is self-synchronized so that clock shape is not affedted by LFPS/ beacon enable timing. 0: Disable (async), 1: Enalbe (sync)
1-0 ANA_PLL_MISC_CLK_SEL	PLL low-frequency clock output source selection 00: Reference clock, 01: Config clock from SoC, 10: Internal oscillator clock, 11: Divided PLL clock

11.4.3.1.97 (CMN_REG05F)

11.4.3.1.97.1 Offset

Register	Offset
CMN_REG05F	17Ch

11.4.3.1.97.2 Diagram



11.4.3.1.97.3 Fields

Field	Function
7-4 PLL_MISC_CLK_DIV_G1	[GEN1] PLL miscellaneous clock divider ratio 0000:
3-0	[GEN2]

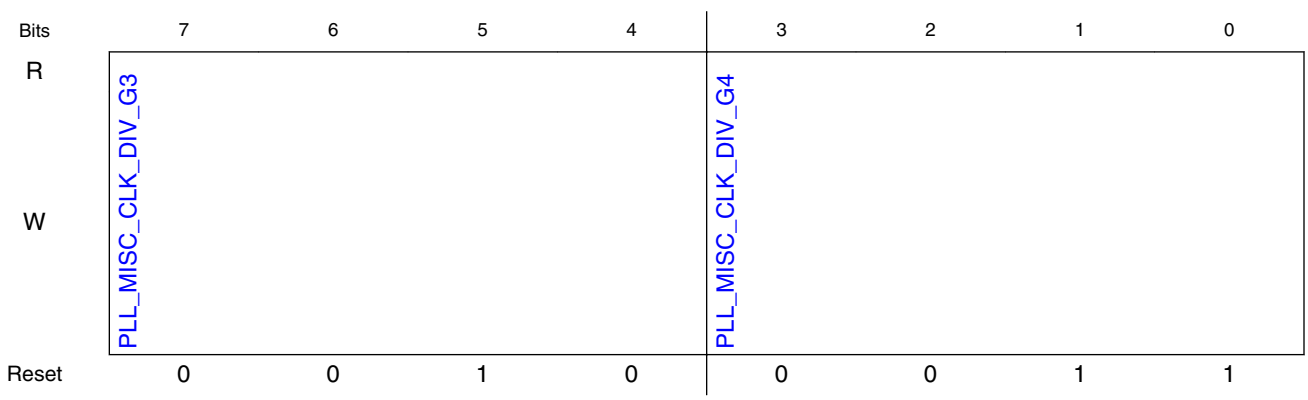
Field	Function
PLL_MISC_CLK_DIV_G2	

11.4.3.1.98 (CMN_REG060)

11.4.3.1.98.1 Offset

Register	Offset
CMN_REG060	180h

11.4.3.1.98.2 Diagram



11.4.3.1.98.3 Fields

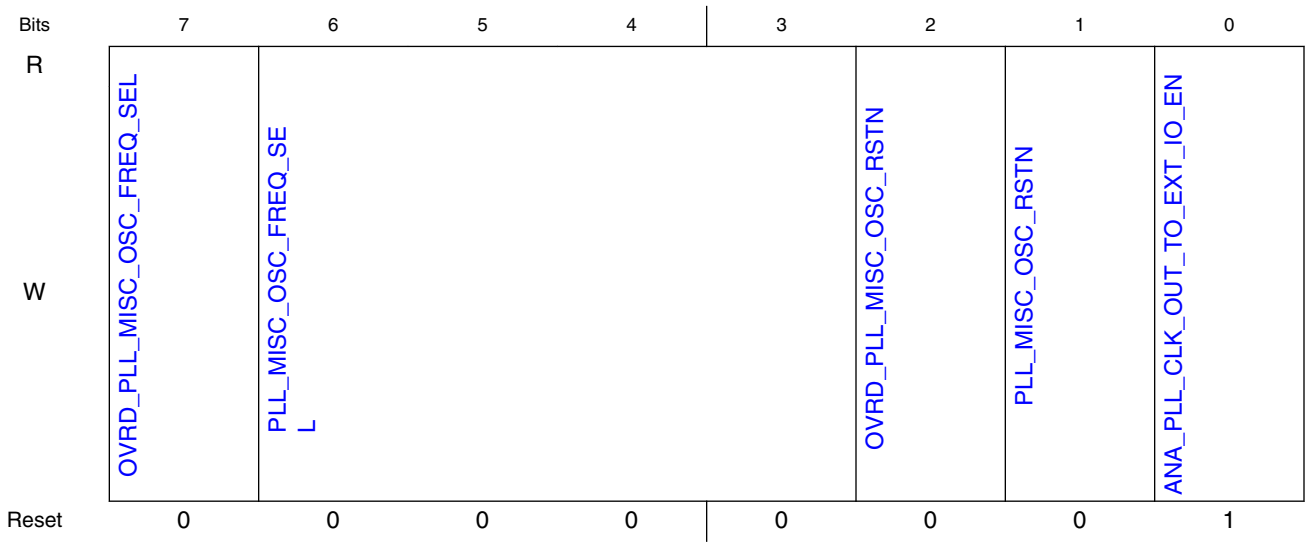
Field	Function
7-4 PLL_MISC_CLK_DIV_G3	[GEN3]
3-0 PLL_MISC_CLK_DIV_G4	[GEN4]

11.4.3.1.99 (CMN_REG061)

11.4.3.1.99.1 Offset

Register	Offset
CMN_REG061	184h

11.4.3.1.99.2 Diagram



11.4.3.1.99.3 Fields

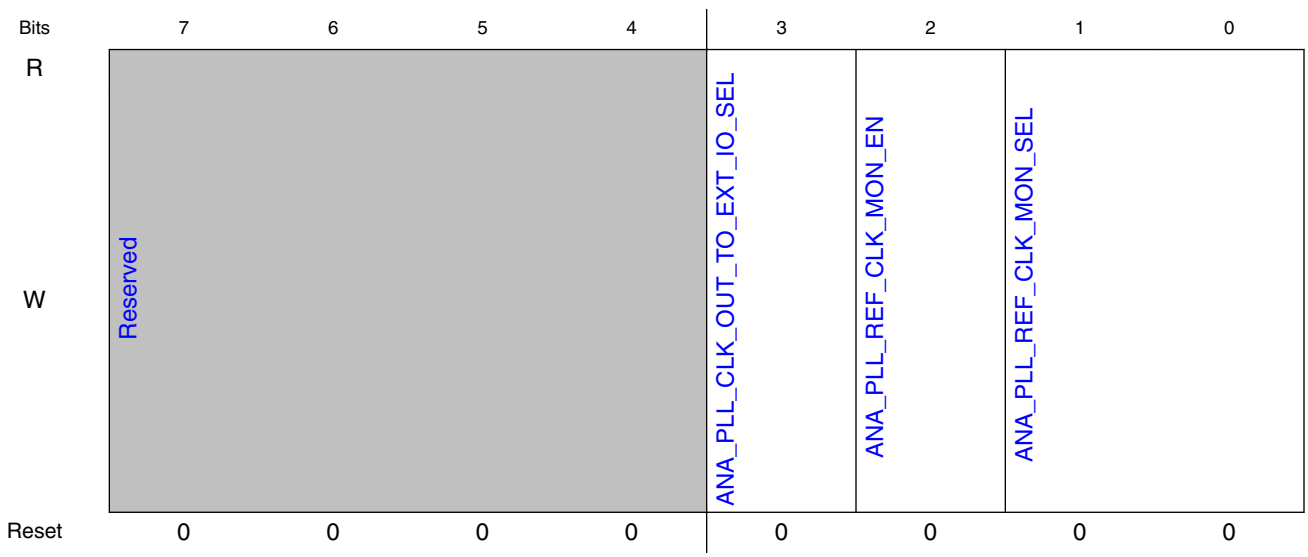
Field	Function
7 OVRD_PLL_MISC_OSC_FREQ_SEL	Override enable for pll_misc_osc_freq_sel
6-3 PLL_MISC_OSC_FREQ_SEL	PLL miscellaneous clock frequency selection Since it is usually automatically determined by oscillator calibration, this is used to force the frequency w/o calibration. 0000: lowest frequency, 1111: highest frequency
2 OVRD_PLL_MISC_OSC_RSTN	Override enable for pll_misc_osc_rstn
1 PLL_MISC_OSC_RSTN	PLL miscellaneous clock oscillator reset 0: Reset, 1: Released
0 ANA_PLL_CLK_OUT_TO_EXT_IO_EN	PLL low-frequency clock output to external I/O enable 0: Disable, 1: Enable

11.4.3.1.100 (CMN_REG062)

11.4.3.1.100.1 Offset

Register	Offset
CMN_REG062	188h

11.4.3.1.100.2 Diagram



11.4.3.1.100.3 Fields

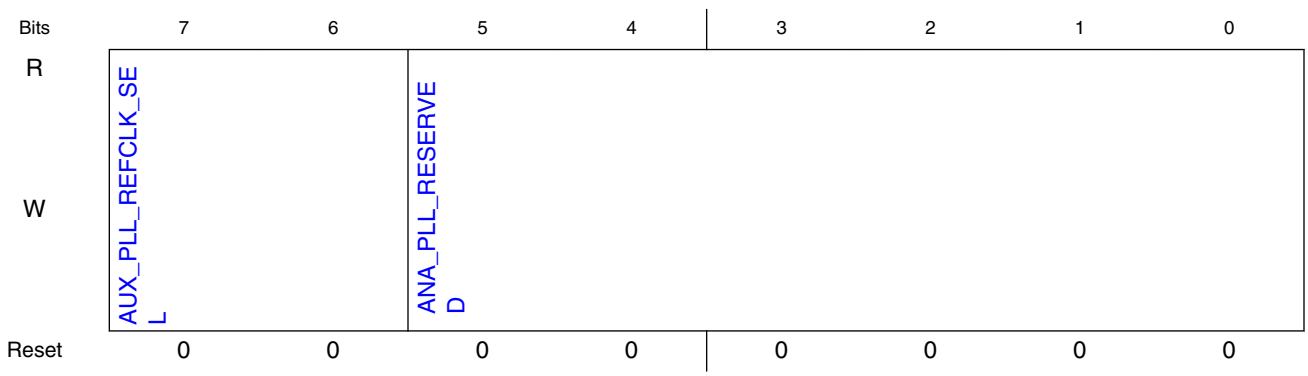
Field	Function
7-4 —	Reserved
3 ANA_PLL_CLK_OUT_TO_EXT_IO_SEL	PLL low-frequency clock output to external I/O source selection 0: 100MHz clock generated by PLL (for PCIe), 1: Reference clock bypass (for MPHY)
2 ANA_PLL_REF_CLK_MON_EN	PLL reference clock monitor enable 0: Disable, 1: Enable
1-0 ANA_PLL_REF_CLK_MON_SEL	PLL reference clock selection for monitor 00: FB clock 01: REF clock 10: Chopper clock 11: Bypass clock

11.4.3.1.101 (CMN_REG063)

11.4.3.1.101.1 Offset

Register	Offset
CMN_REG063	18Ch

11.4.3.1.101.2 Diagram



11.4.3.1.101.3 Fields

Field	Function
7-6 AUX_PLL_REFCLK_SEL	0X: AUX_IN (PLL clock) 10: I_PLL_REFCLK_FROM_IO 11: I_PLL_REFCLK_FROM_SYSPLL
5-0 ANA_PLL_RESERVED	PLL Reserved pins

11.4.3.1.102 (CMN_REG064)

11.4.3.1.102.1 Offset

Register	Offset
CMN_REG064	190h

11.4.3.1.102.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	ANA_AUX_RX_TX_SEL	OVRD_AUX_EN	AUX_EN	ANA_AUX_RX_CAP_BYPASS	ANA_AUX_RX_TERM_GND_EN	ANA_AUX_TX_TERM		
W								
Reset	0	0	0	0	1	0	1	0

11.4.3.1.102.3 Fields

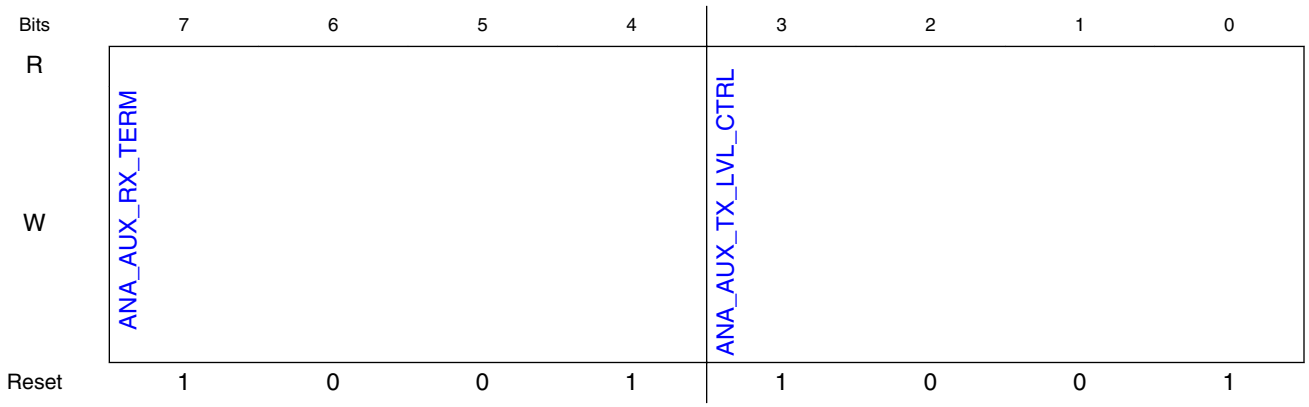
Field	Function
7 ANA_AUX_RX_TX_SEL	Select mode (TX or RX) 0: RX, 1:TX
6 OVRD_AUX_EN	Override enable for aux_en
5 AUX_EN	AUX Enable 0:Disable 1:Enable
4 ANA_AUX_RX_CAP_BYPASS	External reference clock I/O AC-coupling capacitor bypass enable 0: Bypass AC coupling cap, 1: Use AC coupling cap
3 ANA_AUX_RX_TERM_GND_EN	External reference clock I/O termination to ground 0: Disable termination to GND , 1: Enable termination to GND
2-0 ANA_AUX_TX_TERM	TX termination resistor control. Default code : 010, 50.7Ω 30Ω (000) ~ 79Ω (111)

11.4.3.1.103 (CMN_REG065)

11.4.3.1.103.1 Offset

Register	Offset
CMN_REG065	194h

11.4.3.1.103.2 Diagram



11.4.3.1.103.3 Fields

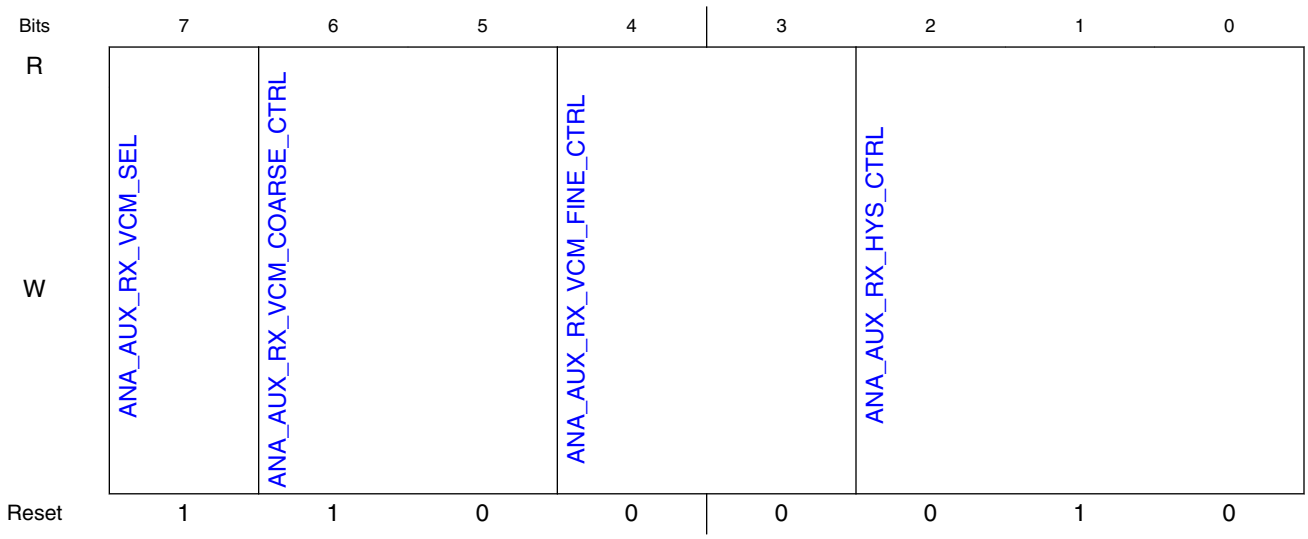
Field	Function
7-4 ANA_AUX_RX_TERM	RX termination resistor control. Default code : 1001, 99.6Ω 77Ω (0000) ~ 140Ω (1111)
3-0 ANA_AUX_TX_LVL_CTRL	TX Amplitude resistor control. Default code : 101, 375mVpp 187.5mVpp (000) ~ 750mVpp (111)

11.4.3.1.104 (CMN_REG066)

11.4.3.1.104.1 Offset

Register	Offset
CMN_REG066	198h

11.4.3.1.104.2 Diagram



11.4.3.1.104.3 Fields

Field	Function
7 ANA_AUX_RX_VCM_SEL	Input common mode voltage control. 0: Disalbe, 1: Enable
6-5 ANA_AUX_RX_VCM_COARSE_CTRL	VCM of RX control. 171mV ~ 680mV at typical condition. Default code : 10 (637mV)
4-3 ANA_AUX_RX_VCM_FINE_CTRL	VCM of RX control. 171mV ~ 680mV at typical condition. Default code : 00 (637mV)
2-0 ANA_AUX_RX_HYS_CTRL	Hysteresis for RX noise blocking control. 0mv ~ 103.6mv (must sense more than 140mv)

11.4.3.1.105 (CMN_REG067)

11.4.3.1.105.1 Offset

Register	Offset
CMN_REG067	19Ch

11.4.3.1.105.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	ANA_AUX_RESERVED							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.105.3 Fields

Field	Function
7-0 ANA_AUX_RESERVED	Reserved port

11.4.3.1.106 (CMN_REG068)

11.4.3.1.106.1 Offset

Register	Offset
CMN_REG068	1A0h

11.4.3.1.106.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	Reserved		OVRD_BGR_SET_DONE	BGR_SET_DONE	OVRD_PLL_AFC_DONE	PLL_AFC_DONE	OVRD_PLL_LOCK_DONE	PLL_LOCK_DONE
Reset	0	0	0	0	0	0	0	0

11.4.3.1.106.3 Fields

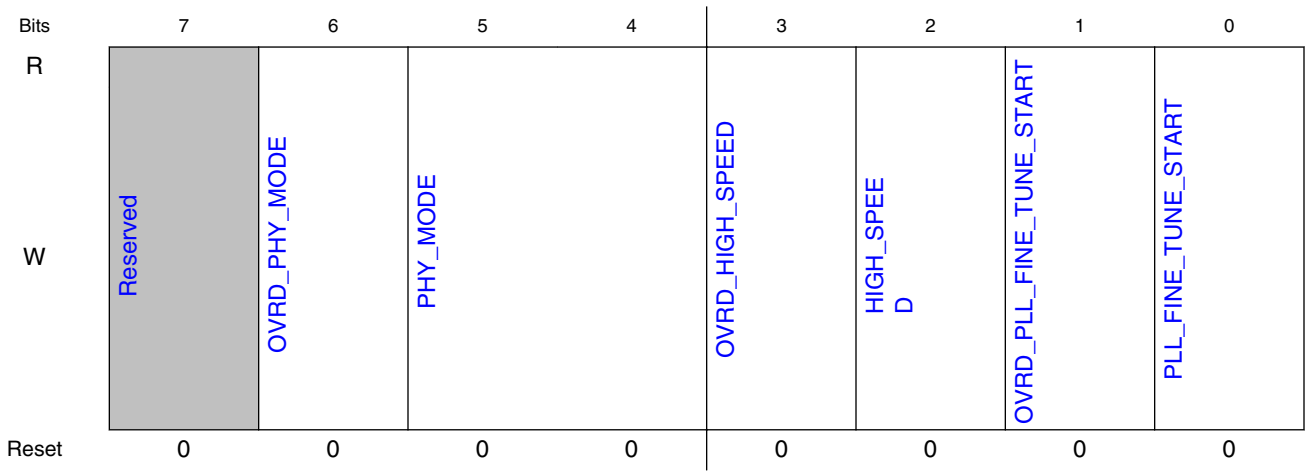
Field	Function
7-6 —	Reserved
5 OVRD_BGR_SET_DONE	Override enable for bgr_set_done
4 BGR_SET_DONE	BGR set done
3 OVRD_PLL_AFC_DONE	Override enable for pll_afc_done
2 PLL_AFC_DONE	PLL AFC done overide value
1 OVRD_PLL_LOCK_DONE	Override enable for pll_lock_done
0 PLL_LOCK_DONE	PLL lock done overide value

11.4.3.1.107 (CMN_REG069)

11.4.3.1.107.1 Offset

Register	Offset
CMN_REG069	1A4h

11.4.3.1.107.2 Diagram



11.4.3.1.107.3 Fields

Field	Function
7 —	Reserved
6 OVRD_PHY_MODE	Override enable for phy_mode
5-4 PHY_MODE	
3 OVRD_HIGH_SPEED	Override enable for high_speed
2 HIGH_SPEED	HIGH SPEED indicator by operating LC VCO 0: Disable, 1: Enable
1 OVRD_PLL_FINE_TUNE_START	Override enable for pll_fine_tune_start
0 PLL_FINE_TUNE_START	

11.4.3.1.108 (CMN_REG06A)

11.4.3.1.108.1 Offset

Register	Offset
CMN_REG06A	1A8h

11.4.3.1.108.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	OVRD_CMN_RATE	CMN_RATE		CMN_TIMER_SEL	TG_BGR_FAST_PULSE_TIME			
W								
Reset	0	0	0	0	0	1	0	1

11.4.3.1.108.3 Fields

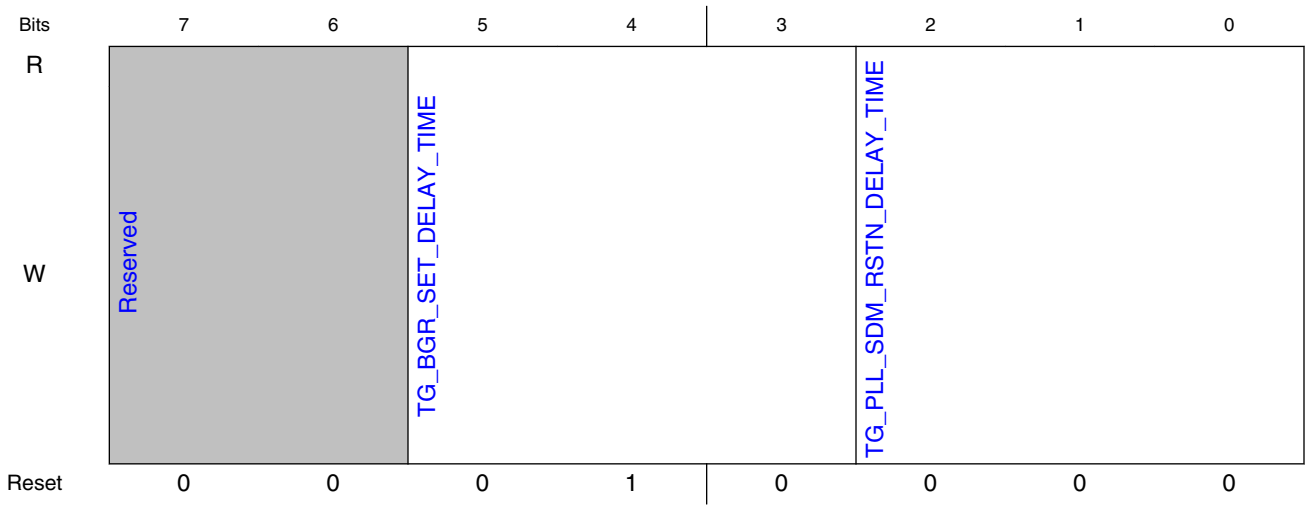
Field	Function
7 OVRD_CMN_RATE	Override enable for cmn_rate
6-5 CMN_RATE	TX Data Rate manual setting 00: Gen1, 01: Gen2, 10: Gen3, 11:Gen4
4 CMN_TIMER_SEL	
3-0 TG_BGR_FAST_PULSE_TIME	BGR LPF bypass duration after BGR_EN = 1

11.4.3.1.109 (CMN_REG06B)

11.4.3.1.109.1 Offset

Register	Offset
CMN_REG06B	1ACh

11.4.3.1.109.2 Diagram



11.4.3.1.109.3 Fields

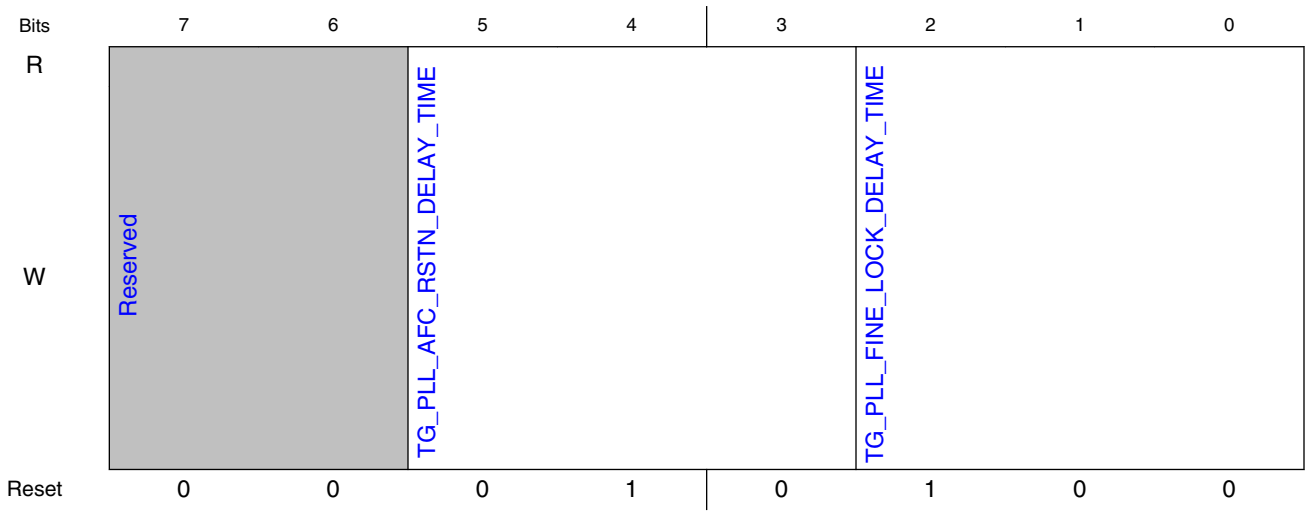
Field	Function
7-6 —	Reserved
5-3 TG_BGR_SET_DELAY_TIME	
2-0 TG_PLL_SDM_RSTN_DELAY_TIME	PLL SDM start delay after PLL integer-mode lock(PLL lock)

11.4.3.1.110 (CMN_REG06C)

11.4.3.1.110.1 Offset

Register	Offset
CMN_REG06C	1B0h

11.4.3.1.110.2 Diagram



11.4.3.1.110.3 Fields

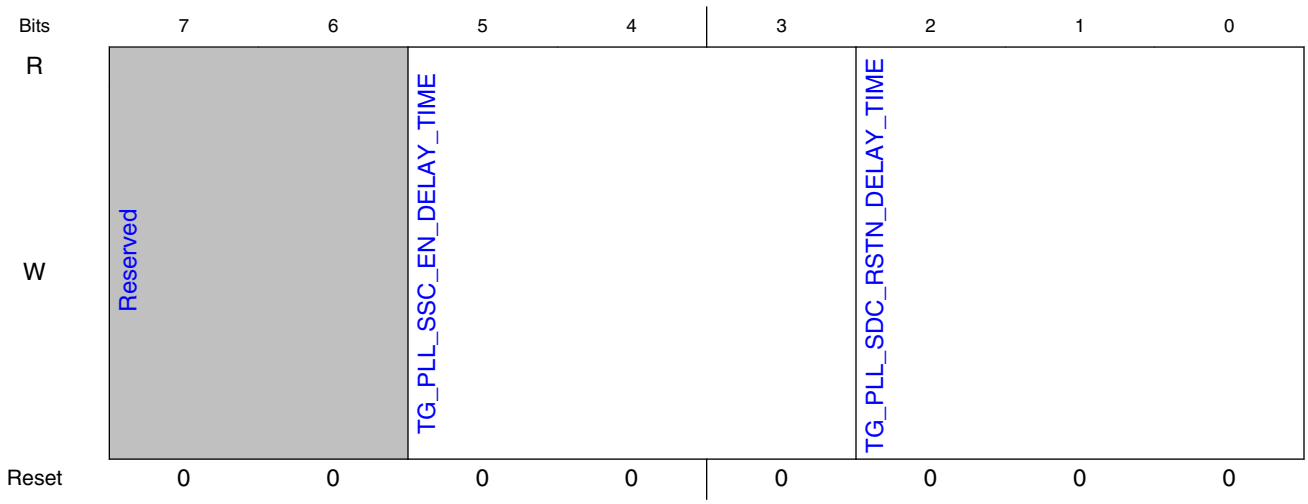
Field	Function
7-6 —	Reserved
5-3 TG_PLL_AFC_RSTN_DELAY_TIME	PLL AFC reset delay time after bypassing BGR LPF
2-0 TG_PLL_FINE_LOCK_DELAY_TIME	PLL Fine LOCK DLY CODE

11.4.3.1.111 (CMN_REG06D)

11.4.3.1.111.1 Offset

Register	Offset
CMN_REG06D	1B4h

11.4.3.1.111.2 Diagram



11.4.3.1.111.3 Fields

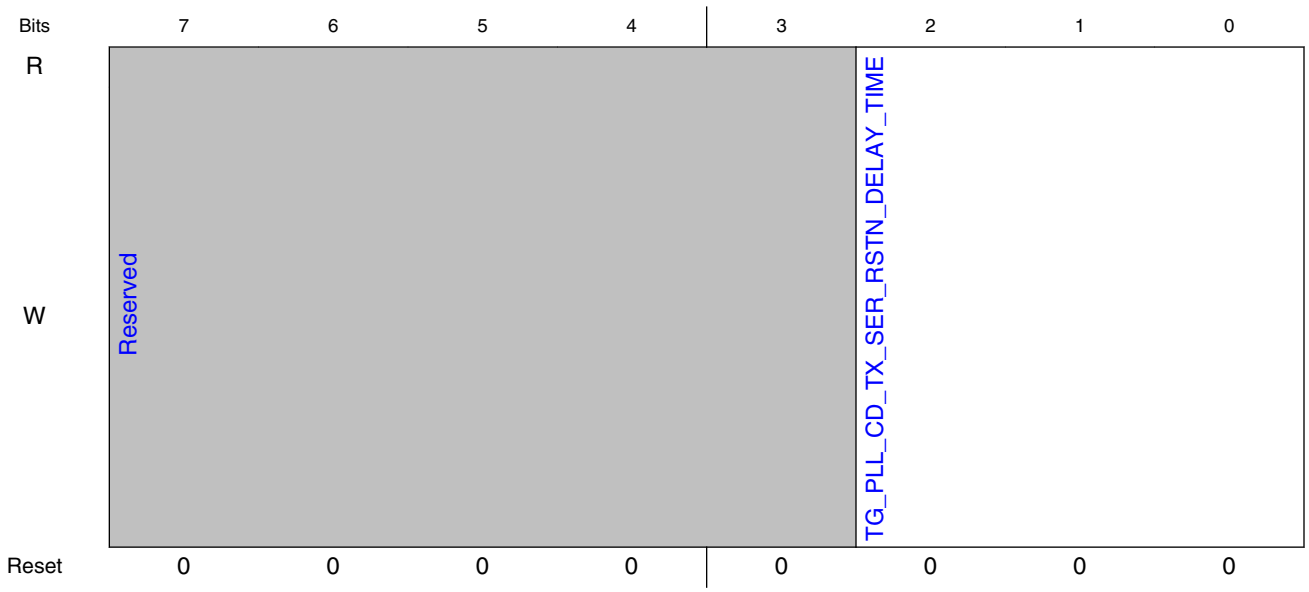
Field	Function
7-6 —	Reserved
5-3 TG_PLL_SSC_EN_DELAY_TIME	PLL SSC start delay time after
2-0 TG_PLL_SDC_RSTN_DELAY_TIME	PLL SDM RESET STABLE DLY CODE

11.4.3.1.112 (CMN_REG06E)

11.4.3.1.112.1 Offset

Register	Offset
CMN_REG06E	1B8h

11.4.3.1.112.2 Diagram



11.4.3.1.112.3 Fields

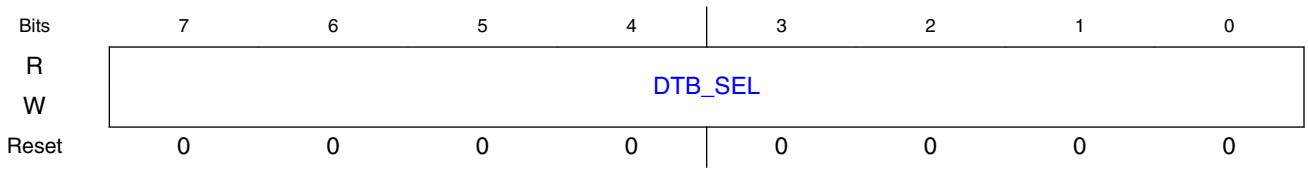
Field	Function
7-3 —	Reserved
2-0 TG_PLL_CD_TX_SER_RSTN_DELAY_TIME	

11.4.3.1.113 (CMN_REG06F)

11.4.3.1.113.1 Offset

Register	Offset
CMN_REG06F	1BCh

11.4.3.1.113.2 Diagram



11.4.3.1.113.3 Fields

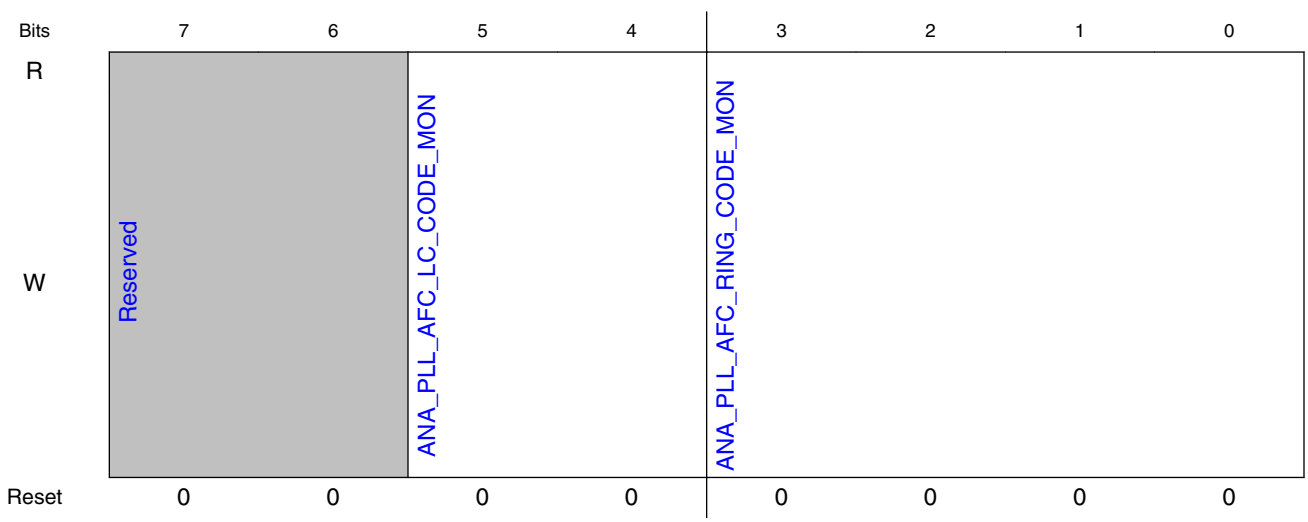
Field	Function
7-0 DTB_SEL	Digital Test Bus (DTB) selection

11.4.3.1.114 (CMN_REG070)

11.4.3.1.114.1 Offset

Register	Offset
CMN_REG070	1C0h

11.4.3.1.114.2 Diagram



11.4.3.1.114.3 Fields

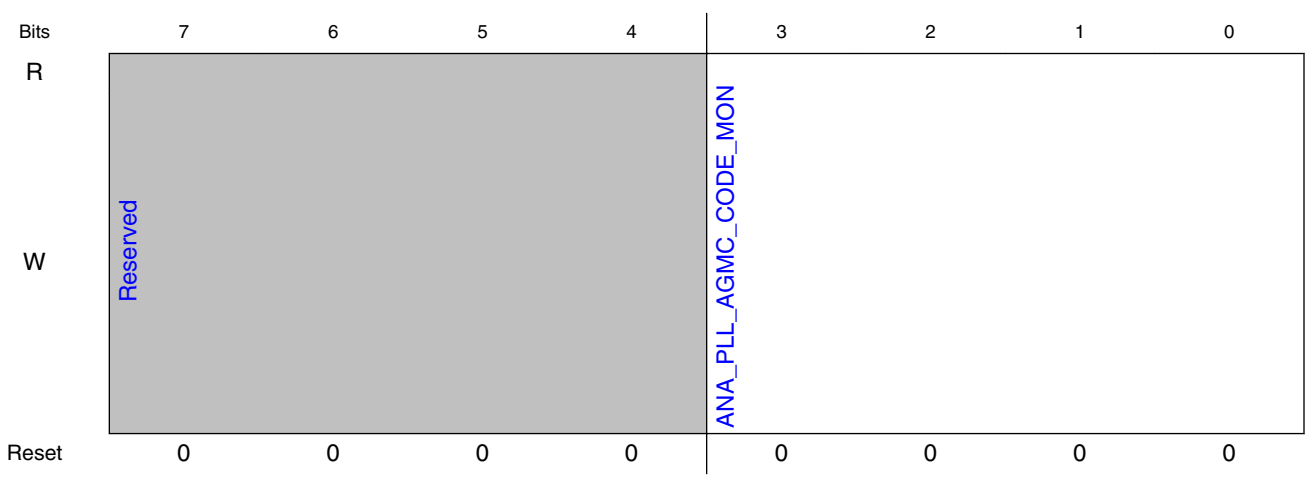
Field	Function
7-6 —	Reserved
5-4 ANA_PLL_AFC_LC_CODE_MON	LC AFC code MSB monitor
3-0 ANA_PLL_AFC_RING_CODE_MON	LC AFC code MSB monitor

11.4.3.1.115 (CMN_REG071)

11.4.3.1.115.1 Offset

Register	Offset
CMN_REG071	1C4h

11.4.3.1.115.2 Diagram



11.4.3.1.115.3 Fields

Field	Function
7-4	Reserved

Table continues on the next page...

Field	Function
—	
3-0 ANA_PLL_AGMC_CODE_MON	LC VCO GM code monitor

11.4.3.1.116 (CMN_REG072)

11.4.3.1.116.1 Offset

Register	Offset
CMN_REG072	1C8h

11.4.3.1.116.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	Reserved				MON_CMN_STATE			
Reset	0	0	0	0	0	0	0	0

11.4.3.1.116.3 Fields

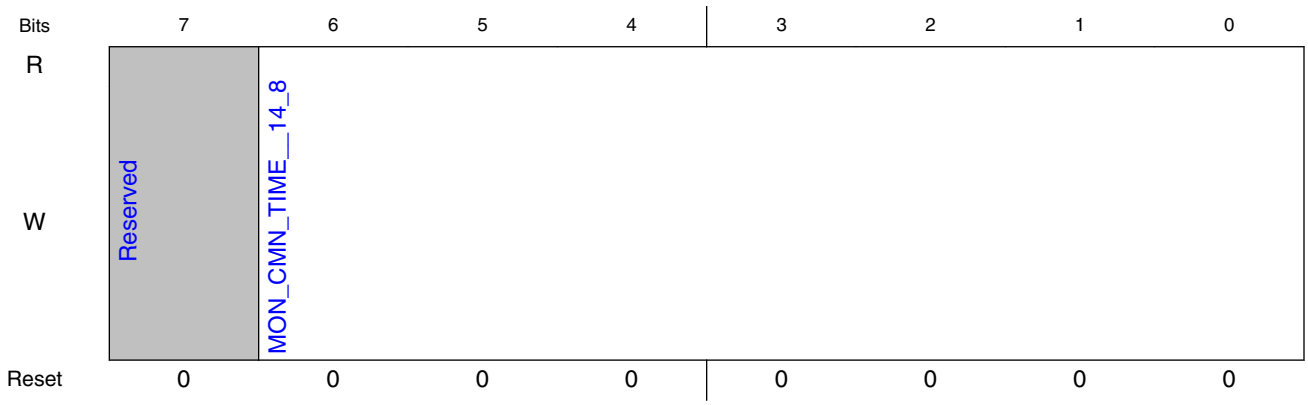
Field	Function
7-5 —	Reserved
4-0 MON_CMN_STATE	CMN state monitor

11.4.3.1.117 (CMN_REG073)

11.4.3.1.117.1 Offset

Register	Offset
CMN_REG073	1CCh

11.4.3.1.117.2 Diagram



11.4.3.1.117.3 Fields

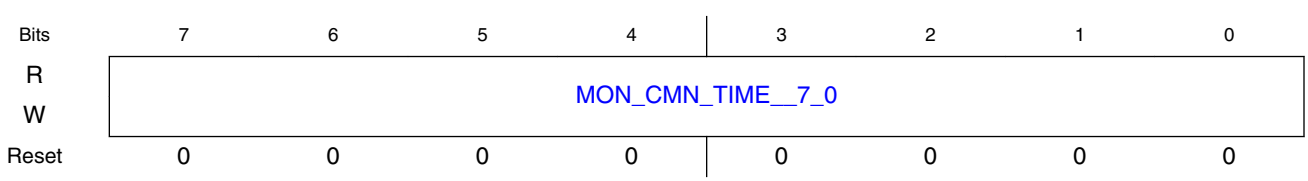
Field	Function
7 —	Reserved
6-0 MON_CMN_TIME__14_8	CMN timer monitor

11.4.3.1.118 (CMN_REG074)

11.4.3.1.118.1 Offset

Register	Offset
CMN_REG074	1D0h

11.4.3.1.118.2 Diagram



11.4.3.1.118.3 Fields

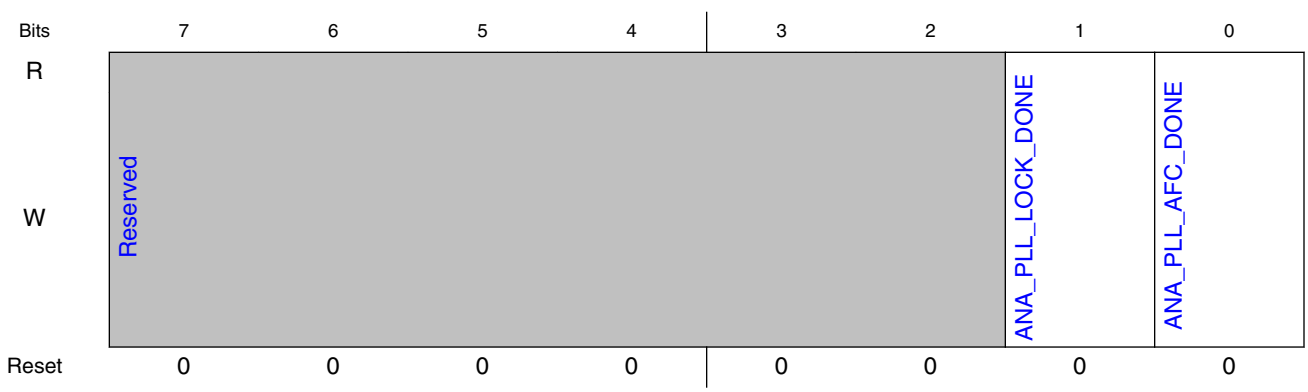
Field	Function
7-0 MON_CMN_TIME__7_0	CMN timer monitor

11.4.3.1.119 (CMN_REG075)

11.4.3.1.119.1 Offset

Register	Offset
CMN_REG075	1D4h

11.4.3.1.119.2 Diagram



11.4.3.1.119.3 Fields

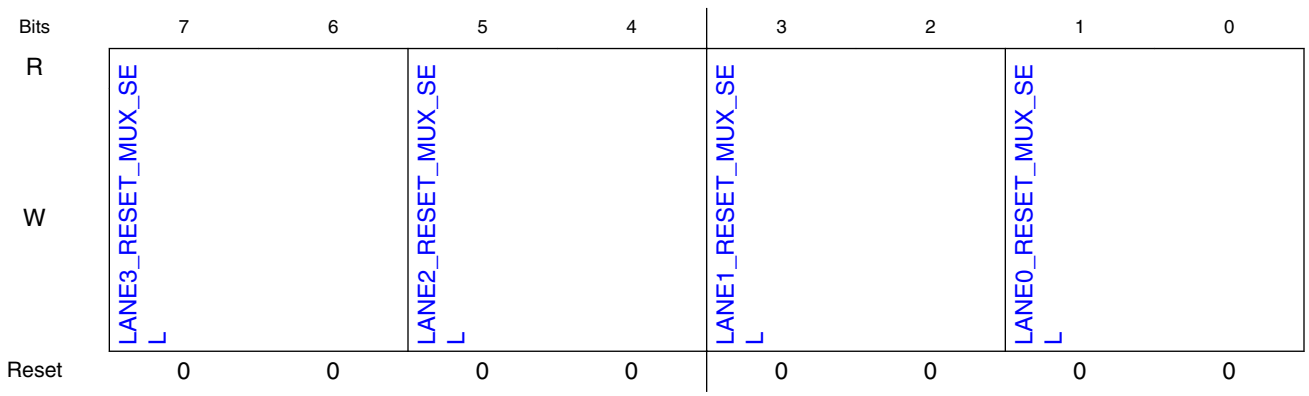
Field	Function
7-2 —	Reserved
1 ANA_PLL_LOCK_DONE	PLL Lock Done
0 ANA_PLL_AFC_DONE	PLL AFC Done

11.4.3.1.120 (CMN_REG076)

11.4.3.1.120.1 Offset

Register	Offset
CMN_REG076	200h

11.4.3.1.120.2 Diagram



11.4.3.1.120.3 Fields

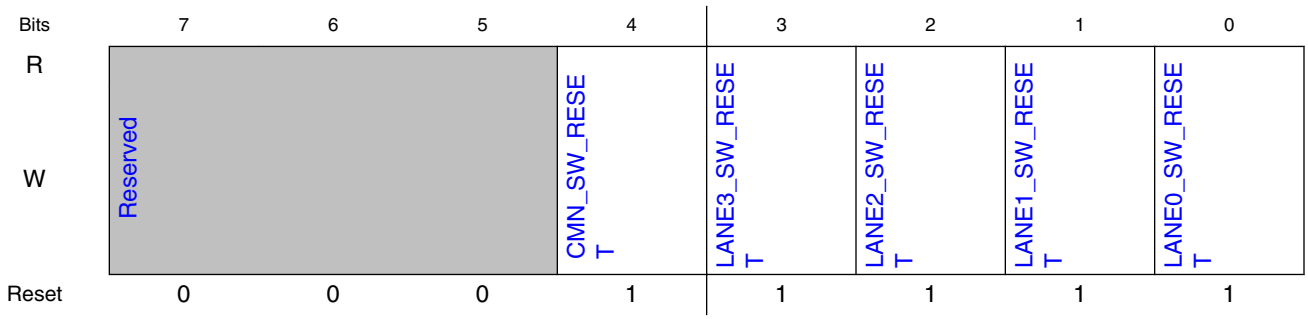
Field	Function
7-6 LANE3_RESET_MUX_SEL	0x0 : Lane3 reset signal from Port 0 0x1 : Lane3 reset signal from Port 1
5-4 LANE2_RESET_MUX_SEL	0x0 : Lane2 reset signal from Port 0 0x1 : Lane2 reset signal from Port 1
3-2 LANE1_RESET_MUX_SEL	0x0 : Lane1 reset signal from Port 0 0x1 : Lane1 reset signal from Port 1
1-0 LANE0_RESET_MUX_SEL	0x0 : Lane0 reset signal from Port 0 0x1 : Lane0 reset signal from Port 1

11.4.3.1.121 (CMN_REG077)

11.4.3.1.121.1 Offset

Register	Offset
CMN_REG077	204h

11.4.3.1.121.2 Diagram



11.4.3.1.121.3 Fields

Field	Function
7-5 —	Reserved
4 CMN_SW_RESET	0x0 : cmn reset 0x1 : cmn reset release
3 LANE3_SW_RESET	0x0 : Lane3 reset 0x1 : Lane3 reset release
2 LANE2_SW_RESET	0x0 : Lane2 reset 0x1 : Lane2 reset release
1 LANE1_SW_RESET	0x0 : Lane1 reset 0x1 : Lane1 reset release
0 LANE0_SW_RESET	0x0 : Lane0 reset 0x1 : Lane0 reset release

11.4.3.1.122 (CMN_REG078)

11.4.3.1.122.1 Offset

Register	Offset
CMN_REG078	208h

11.4.3.1.122.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LANE3_TX_DATA_CLK_MUX_SEL		LANE2_TX_DATA_CLK_MUX_SEL		LANE1_TX_DATA_CLK_MUX_SEL		LANE0_TX_DATA_CLK_MUX_SEL	
W								
Reset	0	1	0	1	0	1	0	1

11.4.3.1.122.3 Fields

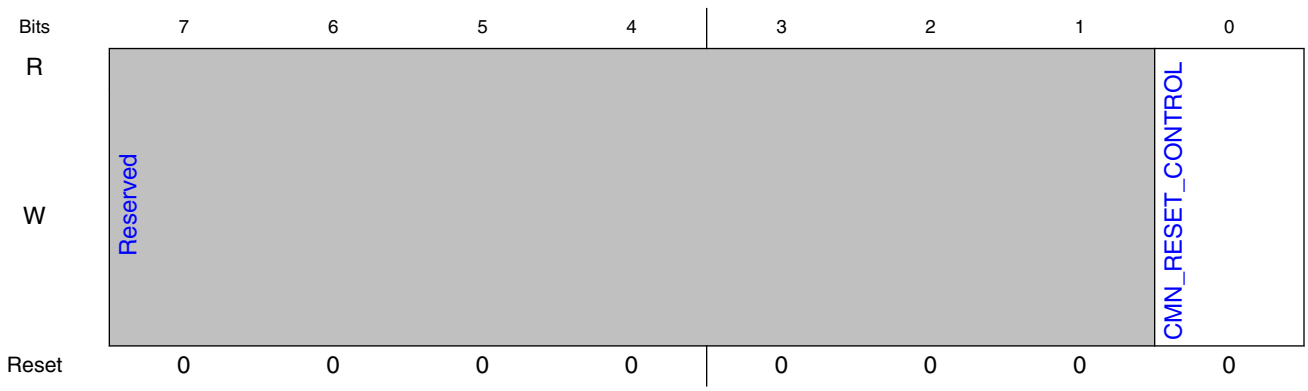
Field	Function
7-6 LANE3_TX_DATA_CLK_MUX_SEL	0x0 : tx data clock from lane0 0x1 : tx data clock from lane1 0x2 : tx data clock from lane2 0x3 : tx data clock from lane3
5-4 LANE2_TX_DATA_CLK_MUX_SEL	
3-2 LANE1_TX_DATA_CLK_MUX_SEL	
1-0 LANE0_TX_DATA_CLK_MUX_SEL	

11.4.3.1.123 (CMN_REG079)

11.4.3.1.123.1 Offset

Register	Offset
CMN_REG079	20Ch

11.4.3.1.123.2 Diagram



11.4.3.1.123.3 Fields

Field	Function
7-1 —	Reserved
0 CMN_RESET_CONTROL	

11.4.3.1.124 (CMN_REG080)

11.4.3.1.124.1 Offset

Register	Offset
CMN_REG080	210h

11.4.3.1.124.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	RATE_CHANGE_DELAY							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.124.3 Fields

Field	Function
7-0 RATE_CHANGE_DELAY	

11.4.3.1.125 (CMN_REG081)

11.4.3.1.125.1 Offset

Register	Offset
CMN_REG081	214h

11.4.3.1.125.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	RX_EFOM_ERROR_TH_7_0							
Reset	0	0	0	1	1	0	0	0

11.4.3.1.125.3 Fields

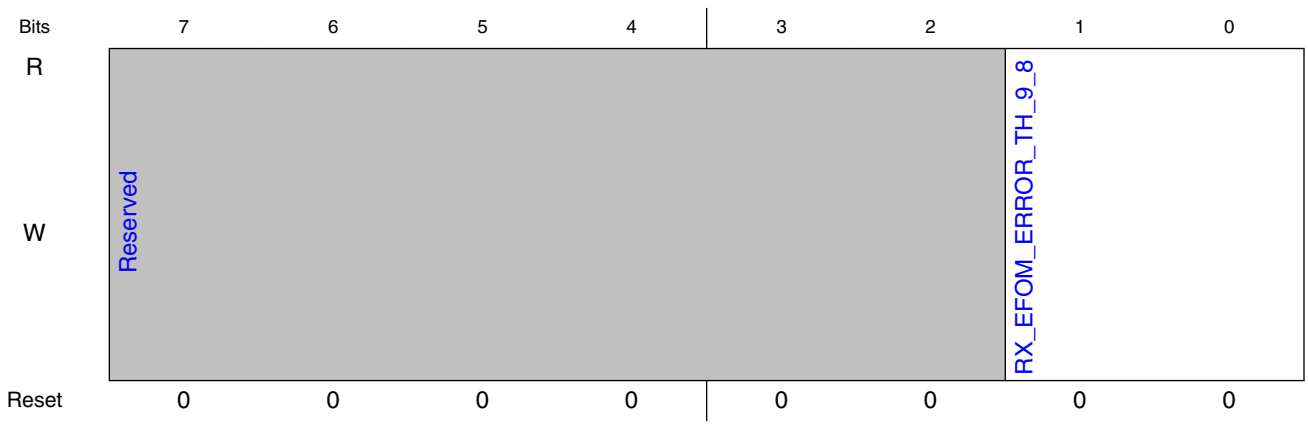
Field	Function
7-0 RX_EFOM_ERROR_TH_7_0	

11.4.3.1.126 (CMN_REG082)

11.4.3.1.126.1 Offset

Register	Offset
CMN_REG082	218h

11.4.3.1.126.2 Diagram



11.4.3.1.126.3 Fields

Field	Function
7-2 —	Reserved
1-0 RX_EFOM_ERROR_TH_9_8	

11.4.3.1.127 (TRSV_REG000)

11.4.3.1.127.1 Offset

Register	Offset
TRSV_REG000	400h

11.4.3.1.127.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_OVRD_TX_DRV_EN	LN0_TX_DRV_EN	LN0_OVRD_TX_DRV_BEACON_LFPS_OUT_EN	LN0_TX_DRV_BEACON_LFPS_OUT_EN	LN0_OVRD_TX_DRV_CM_KEEPER_EN	LN0_TX_DRV_CM_KEEPER_EN	LN0_OVRD_TX_DRV_EI_EN	LN0_TX_DRV_EI_EN
W								
Reset	0	0	0	0	0	0	0	0

11.4.3.1.127.3 Fields

Field	Function
7 LN0_OVRD_TX_DRV_EN	Override enable for tx_drv_en
6 LN0_TX_DRV_EN	TX driver enable 0: Disable, 1: Enable
5 LN0_OVRD_TX_DRV_BEACON_LFPS_OUT_EN	Override enable for tx_drv_beacon_lfps_out_en
4 LN0_TX_DRV_BEACON_LFPS_OUT_EN	TX beacon or LFPS enable 0: Disable, 1: Enable
3 LN0_OVRD_TX_DRV_CM_KEEPER_EN	Override enable for tx_drv_cm_keeper_en
2	TX driver common-mode keep enable 0: Disable, 1: Common-mode voltage keep

Table continues on the next page...

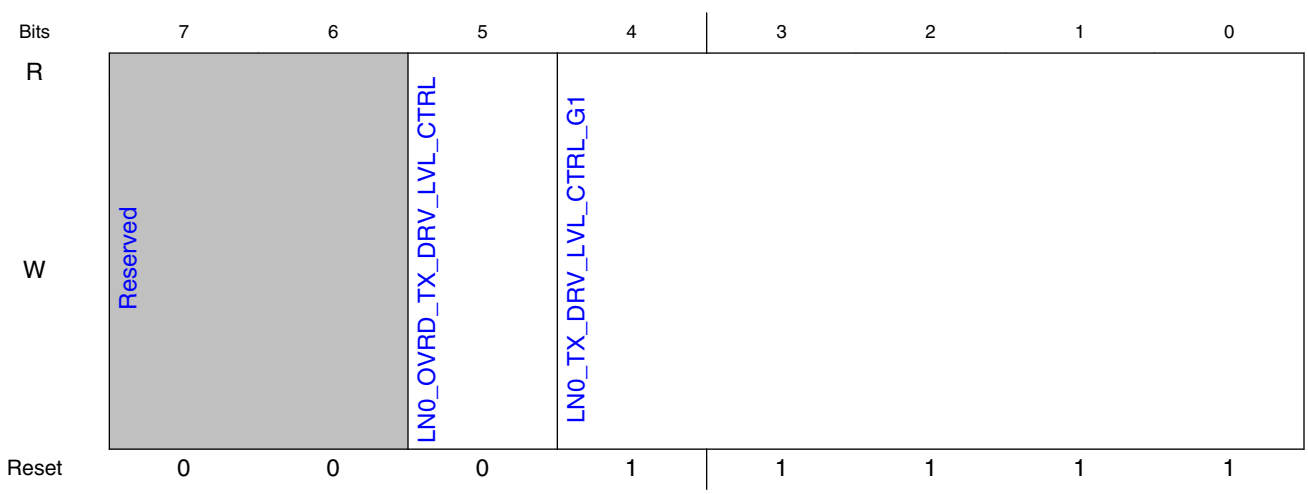
Field	Function
LN0_TX_DRV_CM_KEEPER_EN	
1 LN0_OVRD_TX_DRV_EI_EN	Override enable for tx_drv_ei_en
0 LN0_TX_DRV_EI_EN	TX driver electrical-idle state enable 0: Disable, 1: Electrical-idle enable

11.4.3.1.128 (TRSV_REG001)

11.4.3.1.128.1 Offset

Register	Offset
TRSV_REG001	404h

11.4.3.1.128.2 Diagram



11.4.3.1.128.3 Fields

Field	Function
7-6 —	Reserved
5	Override enable for tx_drv_lvl_ctrl_g1

Table continues on the next page...

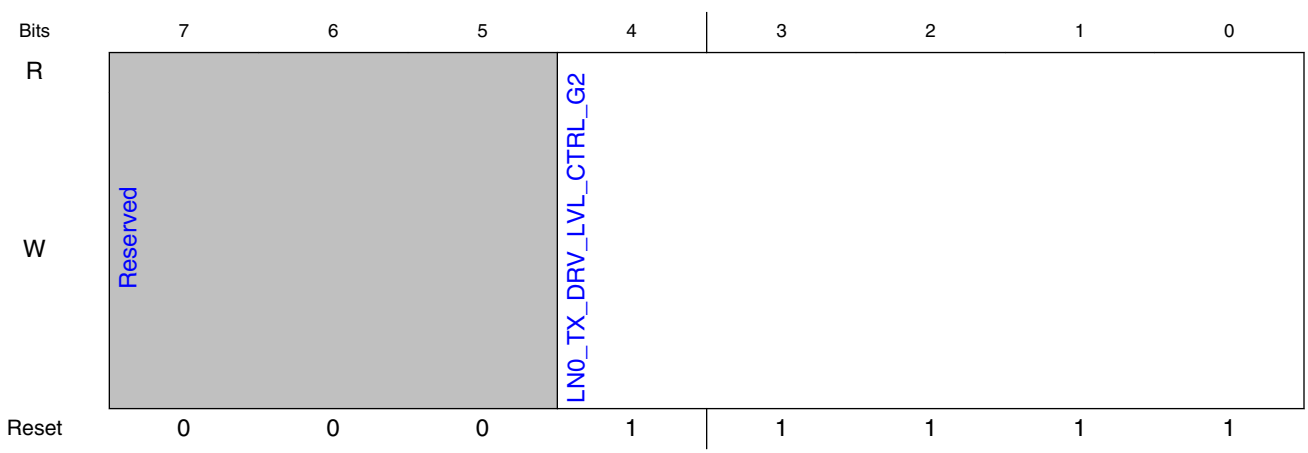
Field	Function
LN0_OVRD_TX_DRV_LVL_CTRL	
4-0 LN0_TX_DRV_LVL_CTRL_G1	[GEN1] TX driver main-tap level 1111: max main-tap level … 0000: min main-tap level Others: N/A

11.4.3.1.129 (TRSV_REG002)

11.4.3.1.129.1 Offset

Register	Offset
TRSV_REG002	408h

11.4.3.1.129.2 Diagram



11.4.3.1.129.3 Fields

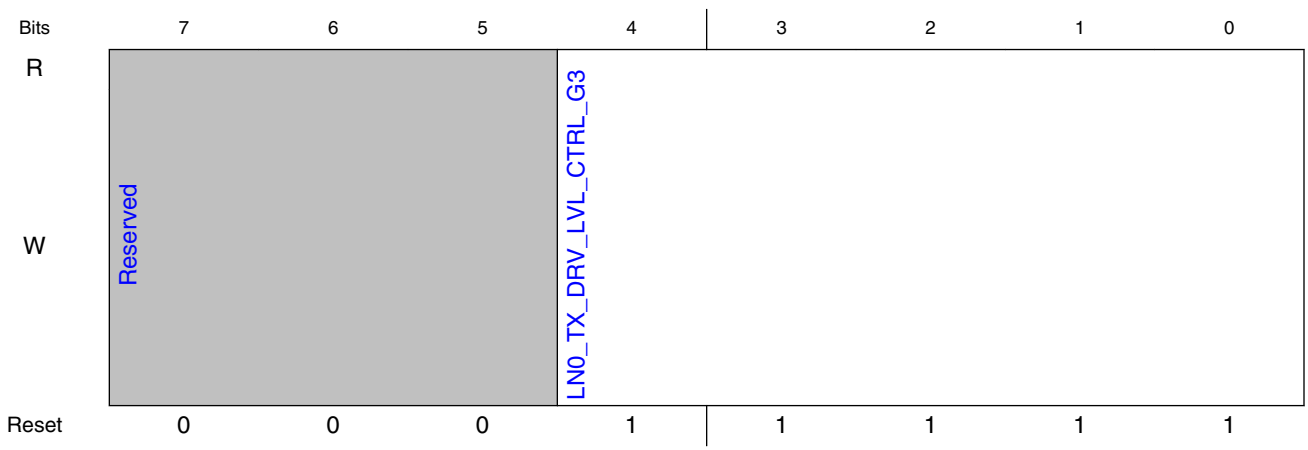
Field	Function
7-5 —	Reserved
4-0 LN0_TX_DRV_LVL_CTRL_G2	[GEN2]

11.4.3.1.130 (TRSV_REG003)

11.4.3.1.130.1 Offset

Register	Offset
TRSV_REG003	40Ch

11.4.3.1.130.2 Diagram



11.4.3.1.130.3 Fields

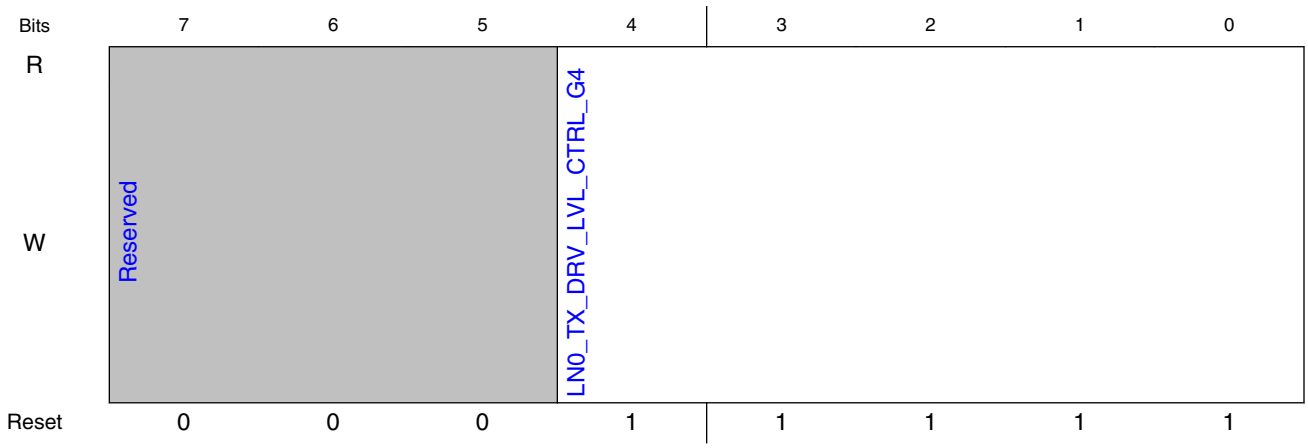
Field	Function
7-5 —	Reserved
4-0 LN0_TX_DRV_LVL_CTRL_G3	[GEN3]

11.4.3.1.131 (TRSV_REG004)

11.4.3.1.131.1 Offset

Register	Offset
TRSV_REG004	410h

11.4.3.1.131.2 Diagram



11.4.3.1.131.3 Fields

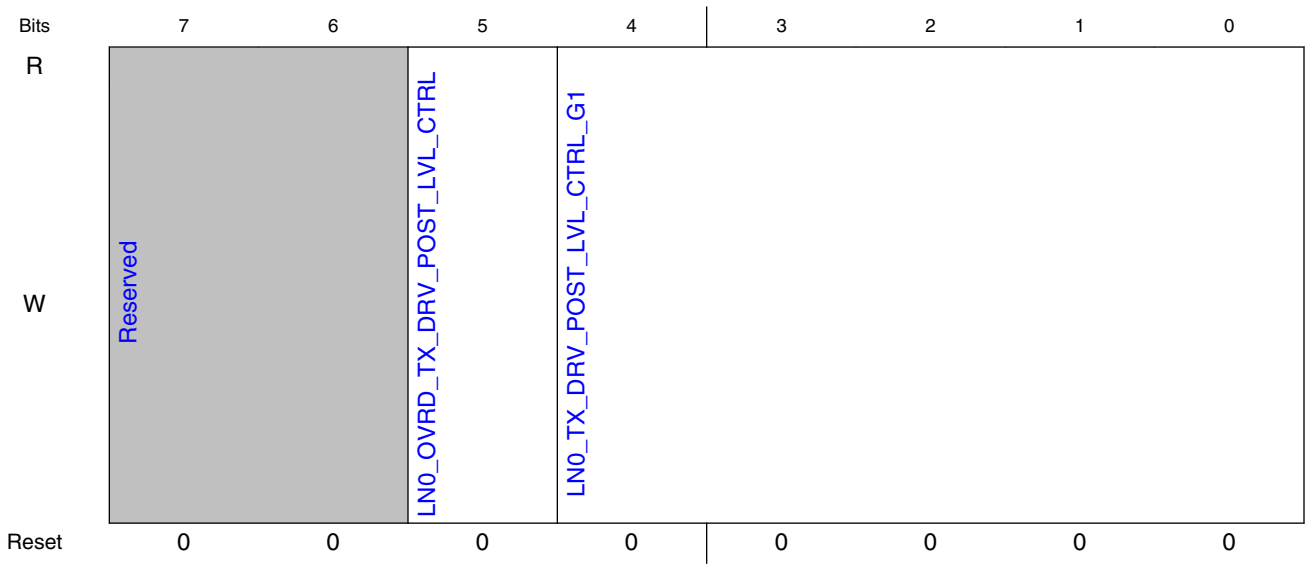
Field	Function
7-5 —	Reserved
4-0 LN0_TX_DRV_LVL_CTRL_G4	[GEN4]

11.4.3.1.132 (TRSV_REG005)

11.4.3.1.132.1 Offset

Register	Offset
TRSV_REG005	414h

11.4.3.1.132.2 Diagram



11.4.3.1.132.3 Fields

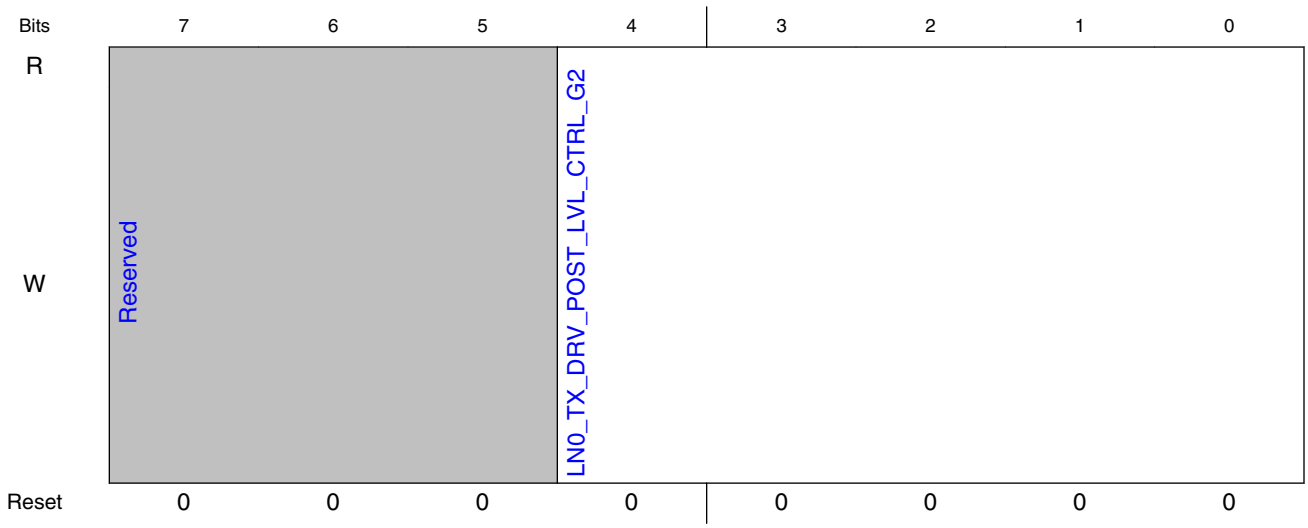
Field	Function
7-6 —	Reserved
5 LN0_OVRD_TX_DRV_POST_LVL_CTRL	Override enable for tx_drv_post_lvl_ctrl_g1
4-0 LN0_TX_DRV_POST_LVL_CTRL_G1	[GEN1] TX driver de-emphasis level 00000: min de-emphasis level … 10100: max de-emphasis level Others: N/A

11.4.3.1.133 (TRSV_REG006)

11.4.3.1.133.1 Offset

Register	Offset
TRSV_REG006	418h

11.4.3.1.133.2 Diagram



11.4.3.1.133.3 Fields

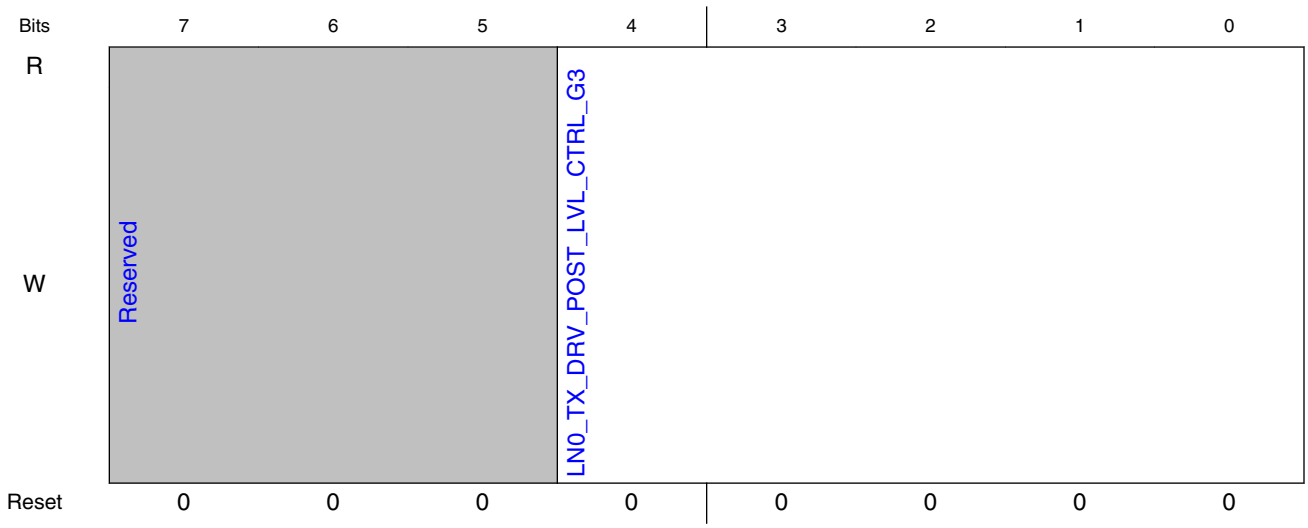
Field	Function
7-5 —	Reserved
4-0 LN0_TX_DRV_POST_LVL_CTRL_G2	[GEN2]

11.4.3.1.134 (TRSV_REG007)

11.4.3.1.134.1 Offset

Register	Offset
TRSV_REG007	41Ch

11.4.3.1.134.2 Diagram



11.4.3.1.134.3 Fields

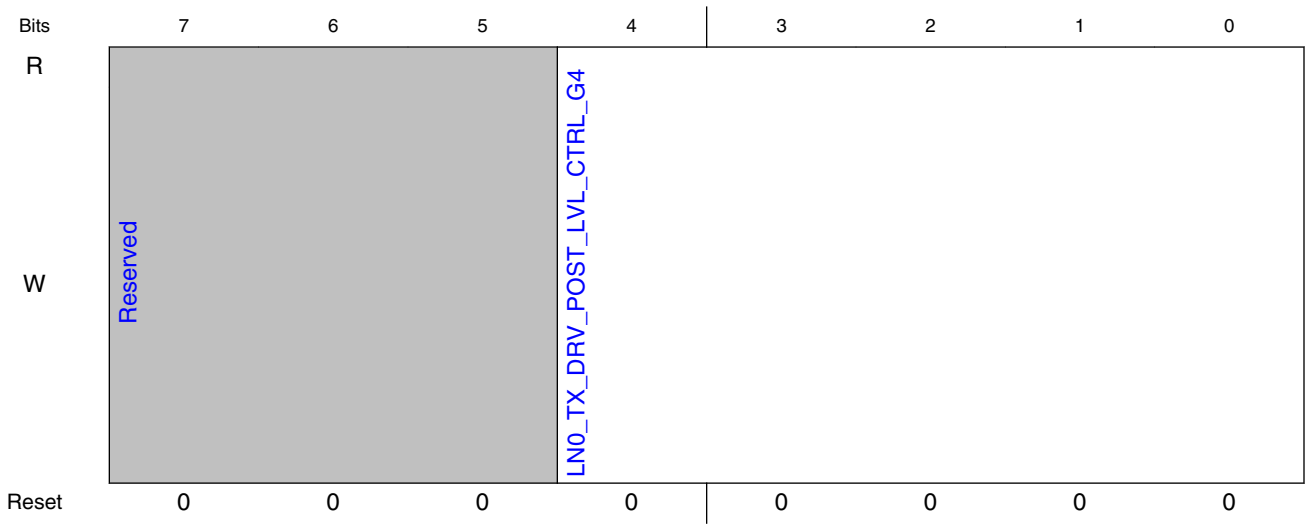
Field	Function
7-5 —	Reserved
4-0 LN0_TX_DRV_POST_LVL_CTRL_G3	[GEN3]

11.4.3.1.135 (TRSV_REG008)

11.4.3.1.135.1 Offset

Register	Offset
TRSV_REG008	420h

11.4.3.1.135.2 Diagram



11.4.3.1.135.3 Fields

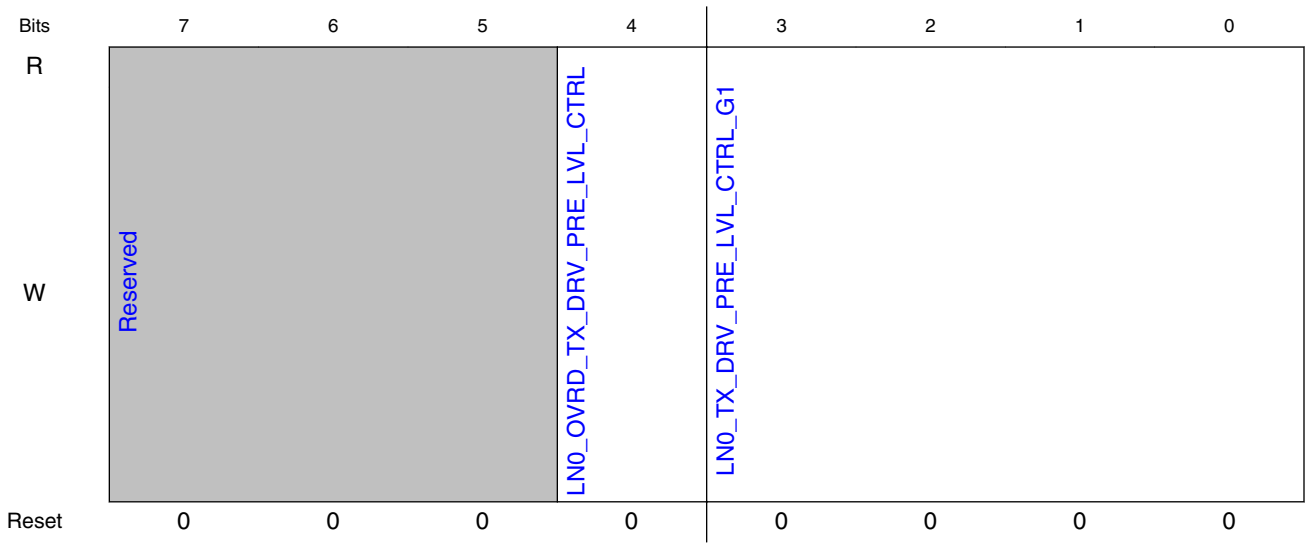
Field	Function
7-5 —	Reserved
4-0 LN0_TX_DRV_POST_LVL_CTRL_G4	[GEN4]

11.4.3.1.136 (TRSV_REG009)

11.4.3.1.136.1 Offset

Register	Offset
TRSV_REG009	424h

11.4.3.1.136.2 Diagram



11.4.3.1.136.3 Fields

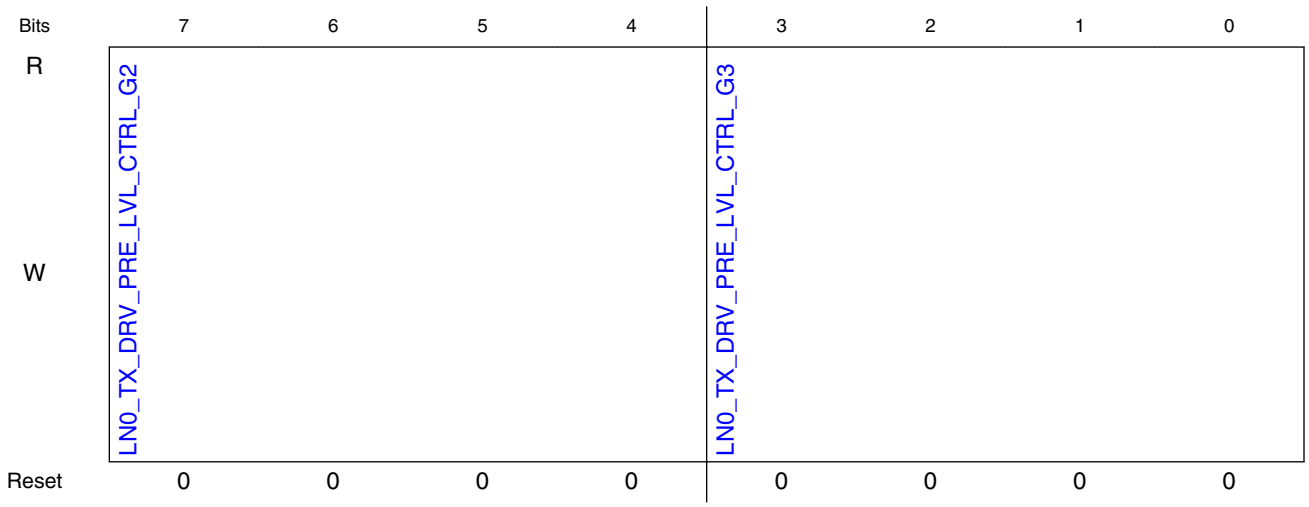
Field	Function
7-5 —	Reserved
4 LN0_OVRD_TX_DRV_PRE_LVL_CTRL	Override enable for tx_drv_pre_lvl_ctrl_g1
3-0 LN0_TX_DRV_PRE_LVL_CTRL_G1	[GEN1] TX driver pre-shoot level 0000: min pre-shoot boosting … 1011: max pre-shoot boosting Others: N/A

11.4.3.1.137 (TRSV_REG00A)

11.4.3.1.137.1 Offset

Register	Offset
TRSV_REG00A	428h

11.4.3.1.137.2 Diagram



11.4.3.1.137.3 Fields

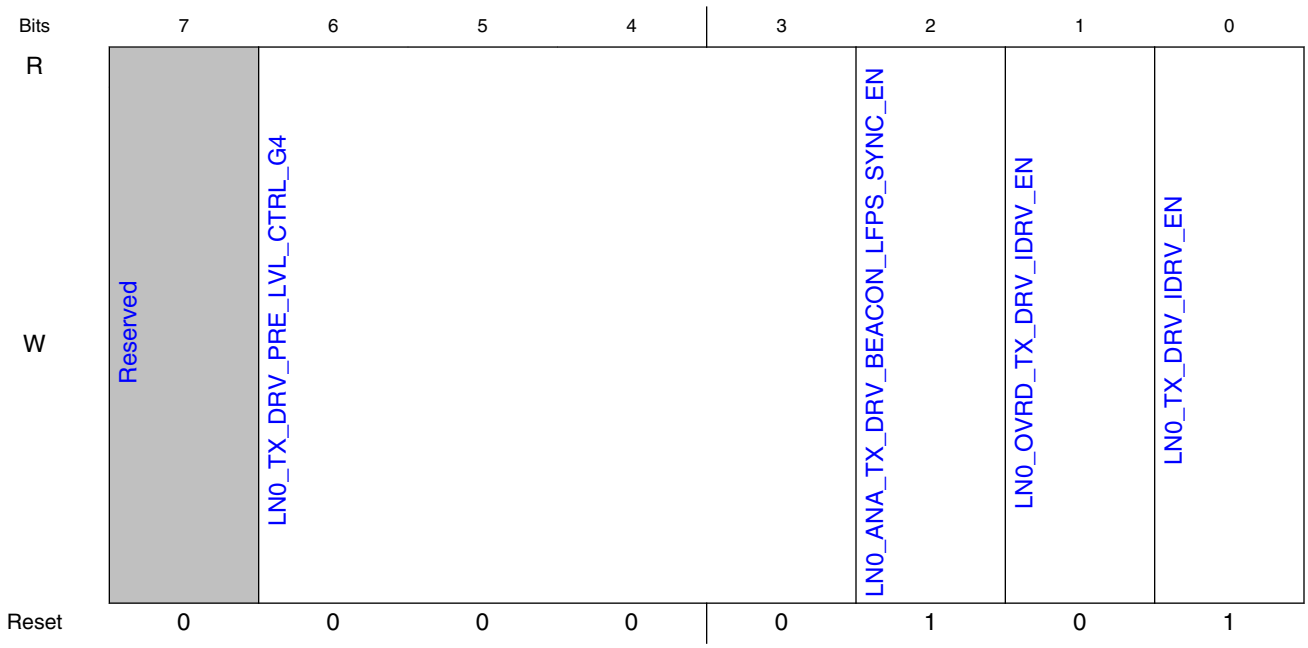
Field	Function
7-4 LN0_TX_DRV_PRE_LVL_CTRL_G2	[GEN2]
3-0 LN0_TX_DRV_PRE_LVL_CTRL_G3	[GEN3]

11.4.3.1.138 (TRSV_REG00B)

11.4.3.1.138.1 Offset

Register	Offset
TRSV_REG00B	42Ch

11.4.3.1.138.2 Diagram



11.4.3.1.138.3 Fields

Field	Function
7 —	Reserved
6-3 LN0_TX_DRV_PRE_LVL_CTRL_G4	[GEN4]
2 LN0_ANA_TX_DRV_BEACON_LFPS_SYNC_EN	TX LFPS/BEACON synchronization enable If enabled, LFPS/BEACON output is self-synchronized so that waveform is not affedted by LFPS/beacon enable timing. 0: Disable (async), 1: Enalbe (sync)
1 LN0_OVRD_TX_DRV_IDRV_EN	Override enable for tx_drv_idrv_en
0 LN0_TX_DRV_IDRV_EN	TX current-driver enable 0: disable, 1: enable

11.4.3.1.139 (TRSV_REG00C)

11.4.3.1.139.1 Offset

Register	Offset
TRSV_REG00C	430h

11.4.3.1.139.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_ANA_TX_DRV_IDRV_IDN_CTRL			LN0_ANA_TX_DRV_IDRV_IUP_CTRL			LN0_ANA_TX_DRV_IDRV_VREF_SEL	LN0_ANA_TX_DRV_ACCDRV_EN
W								
Reset	1	0	0	1	0	0	1	1

11.4.3.1.139.3 Fields

Field	Function
7-5 LN0_ANA_TX_DRV_IDRV_IDN_CTRL	TX current driver pmos current control 000: Min … 111: Max
4-2 LN0_ANA_TX_DRV_IDRV_IUP_CTRL	TX current driver pmos current control 000: Min … 111: Max
1 LN0_ANA_TX_DRV_IDRV_VREF_SEL	TX current driver reference selection 1: Fixed reference, 0: Unity-gain buffer(Default)
0	Enable of Cap. Peaking block.

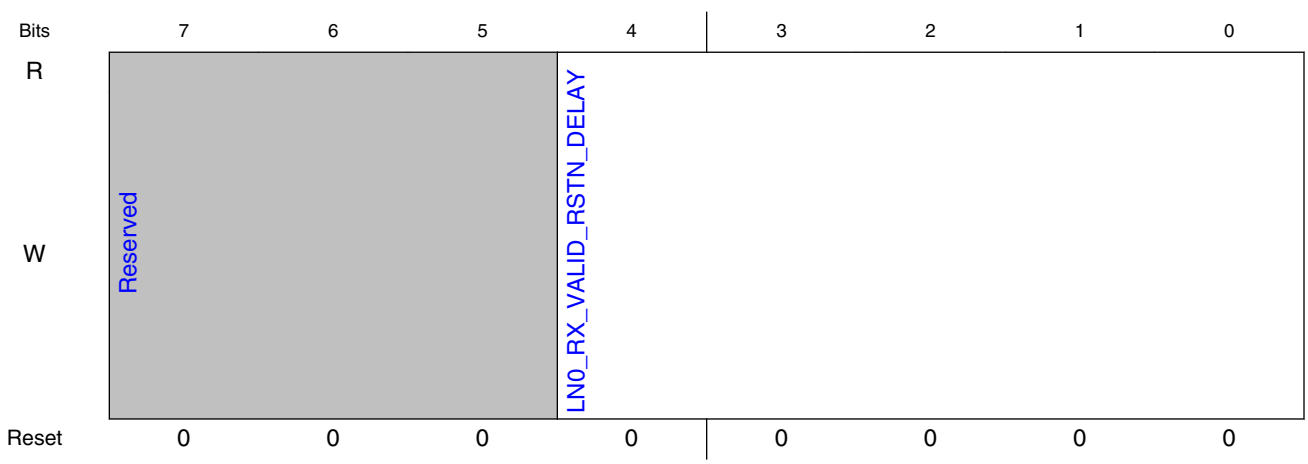
Field	Function
LN0_ANA_TX_DRV_ACCDRV_EN	0: Disable, 1: Enable

11.4.3.1.140 (TRSV_REG00D)

11.4.3.1.140.1 Offset

Register	Offset
TRSV_REG00D	434h

11.4.3.1.140.2 Diagram



11.4.3.1.140.3 Fields

Field	Function
7-5 —	Reserved
4-0 LN0_RX_VALID_RSTN_DELAY	

11.4.3.1.141 (TRSV_REG00E)

11.4.3.1.141.1 Offset

Register	Offset
TRSV_REG00E	438h

11.4.3.1.141.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_TX_DRV_EI_EN_DELAY_SEL_G1		LN0_TX_DRV_EI_EN_DELAY_SEL_G2		LN0_TX_DRV_EI_EN_DELAY_SEL_G3		LN0_TX_DRV_EI_EN_DELAY_SEL_G4	
W								
Reset	0	1	0	1	0	1	0	1

11.4.3.1.141.3 Fields

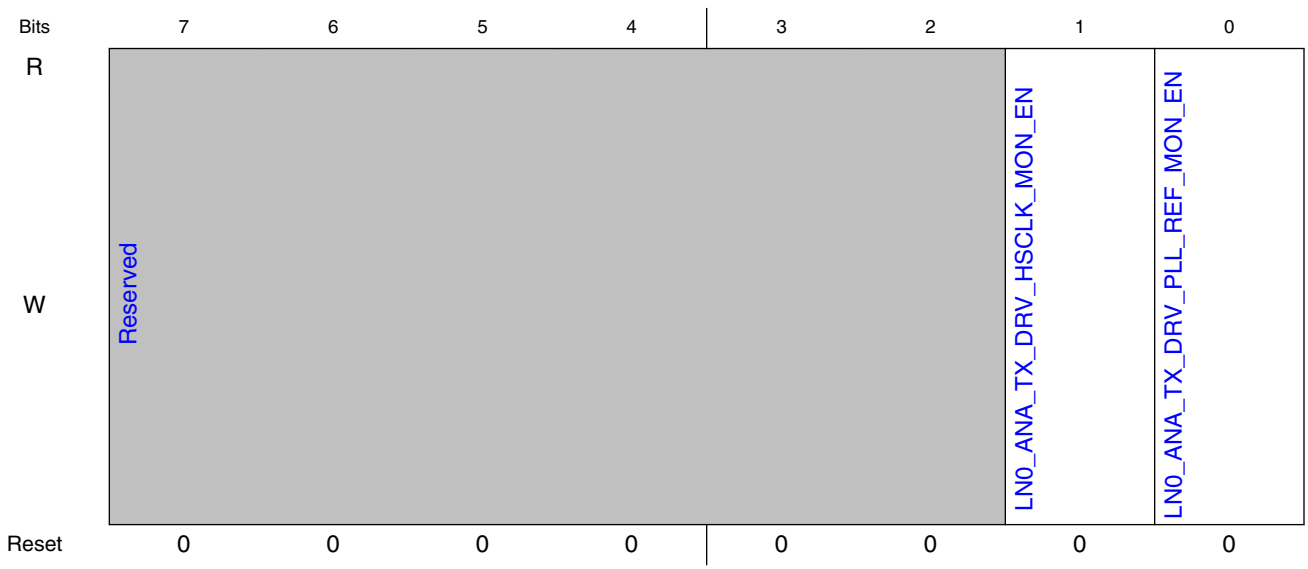
Field	Function
7-6 LN0_TX_DRV_EI_EN_DELAY_SEL_G1	[GEN1] TX EI enable latency control Time delay from EI enable from PCS to real 00: , 01: , 10: , 11:
5-4 LN0_TX_DRV_EI_EN_DELAY_SEL_G2	[GEN2]
3-2 LN0_TX_DRV_EI_EN_DELAY_SEL_G3	[GEN3]
1-0 LN0_TX_DRV_EI_EN_DELAY_SEL_G4	[GEN4]

11.4.3.1.142 (TRSV_REG00F)

11.4.3.1.142.1 Offset

Register	Offset
TRSV_REG00F	43Ch

11.4.3.1.142.2 Diagram



11.4.3.1.142.3 Fields

Field	Function
7-2 —	Reserved
1 LN0_ANA_TX_DRV_HSCLK_MON_EN	Enable of high-speed clock monitor through Tx driver 0: Disable, 1: Enable
0 LN0_ANA_TX_DRV_PLL_REF_MON_EN	Enable of PLL reference clock monitor through Tx driver 0: Disable, 1: Enable

11.4.3.1.143 (TRSV_REG010)

11.4.3.1.143.1 Offset

Register	Offset
TRSV_REG010	440h

11.4.3.1.143.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_TX_JEQ_CAP_CTRL_G1				LN0_TX_JEQ_CAP_CTRL_G2			
W								
Reset	0	0	0	0	0	0	0	0

11.4.3.1.143.3 Fields

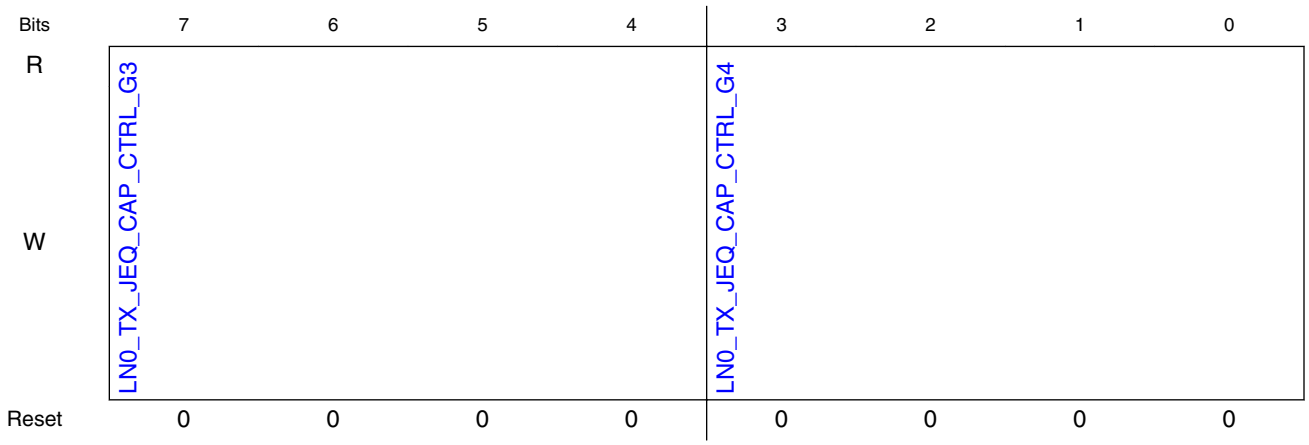
Field	Function
7-4 LN0_TX_JEQ_CAP_CTRL_G1	[GEN1] TX jitter EQ loding capacitance control in thermomether 0000: default 0001: 1 unit cap loading 0011: 2 unit cap loading 0111: 3 unit cap loading 1111: 4 unit cap loading
3-0 LN0_TX_JEQ_CAP_CTRL_G2	[GEN2]

11.4.3.1.144 (TRSV_REG011)

11.4.3.1.144.1 Offset

Register	Offset
TRSV_REG011	444h

11.4.3.1.144.2 Diagram



11.4.3.1.144.3 Fields

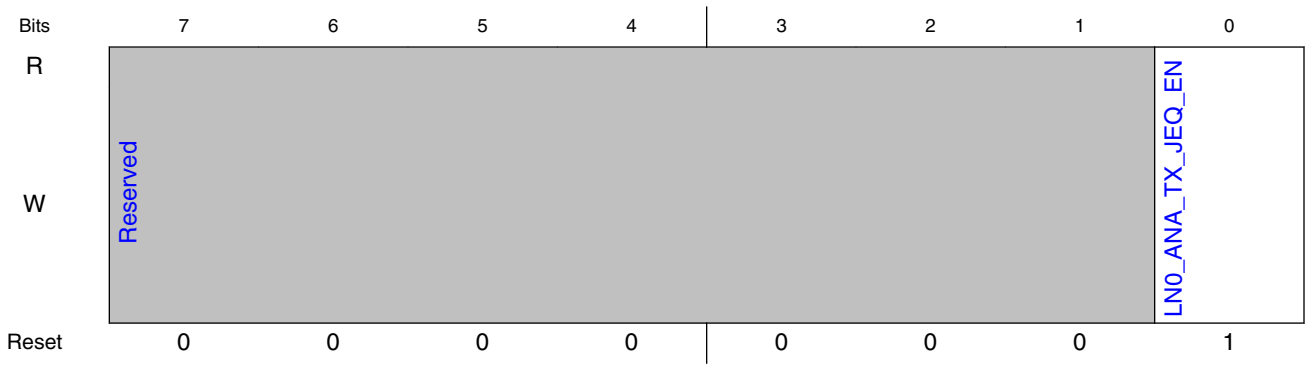
Field	Function
7-4 LN0_TX_JEQ_CAP_CTRL_G3	[GEN3]
3-0 LN0_TX_JEQ_CAP_CTRL_G4	[GEN4]

11.4.3.1.145 (TRSV_REG012)

11.4.3.1.145.1 Offset

Register	Offset
TRSV_REG012	448h

11.4.3.1.145.2 Diagram



11.4.3.1.145.3 Fields

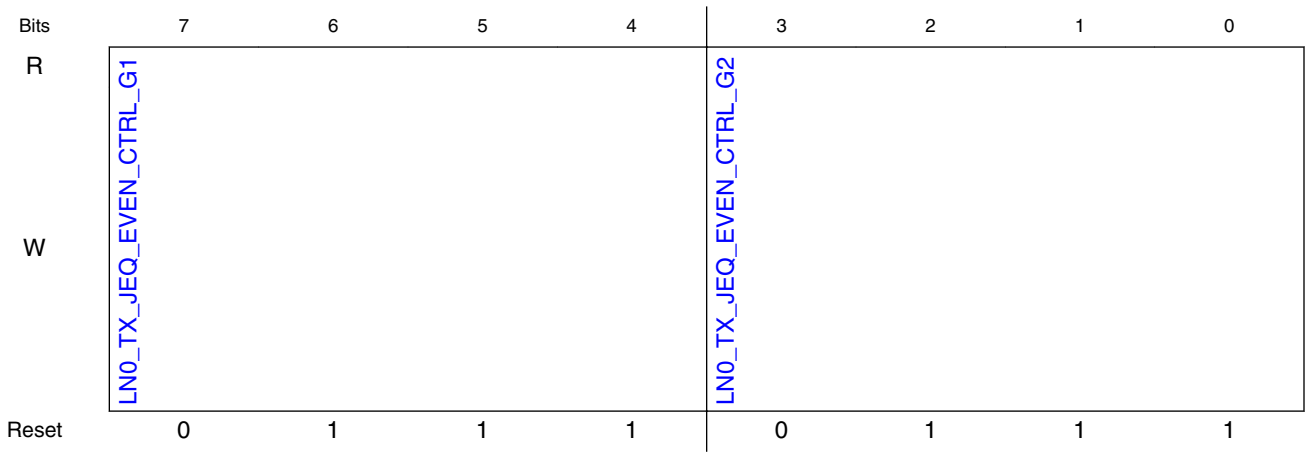
Field	Function
7-1 —	Reserved
0 LN0_ANA_TX_JEQ_EN	TX jitter EQ enable 0: disable 1: enable

11.4.3.1.146 (TRSV_REG013)

11.4.3.1.146.1 Offset

Register	Offset
TRSV_REG013	44Ch

11.4.3.1.146.2 Diagram



11.4.3.1.146.3 Fields

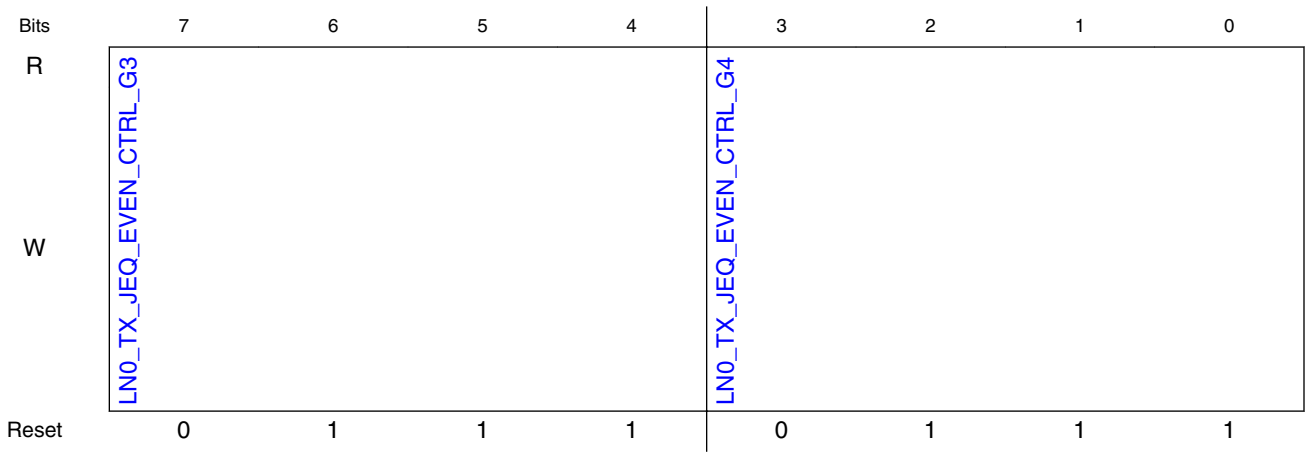
Field	Function
7-4 LN0_TX_JEQ_EVEN_CTRL_G1	[GEN1] TX jitter EQ driver (even) strength control 0000: 0 legs on (jitter EQ driver off) 0100: 2 legs on 1000: 4 legs on 1111: 8 legs on
3-0 LN0_TX_JEQ_EVEN_CTRL_G2	[GEN2]

11.4.3.1.147 (TRSV_REG014)

11.4.3.1.147.1 Offset

Register	Offset
TRSV_REG014	450h

11.4.3.1.147.2 Diagram



11.4.3.1.147.3 Fields

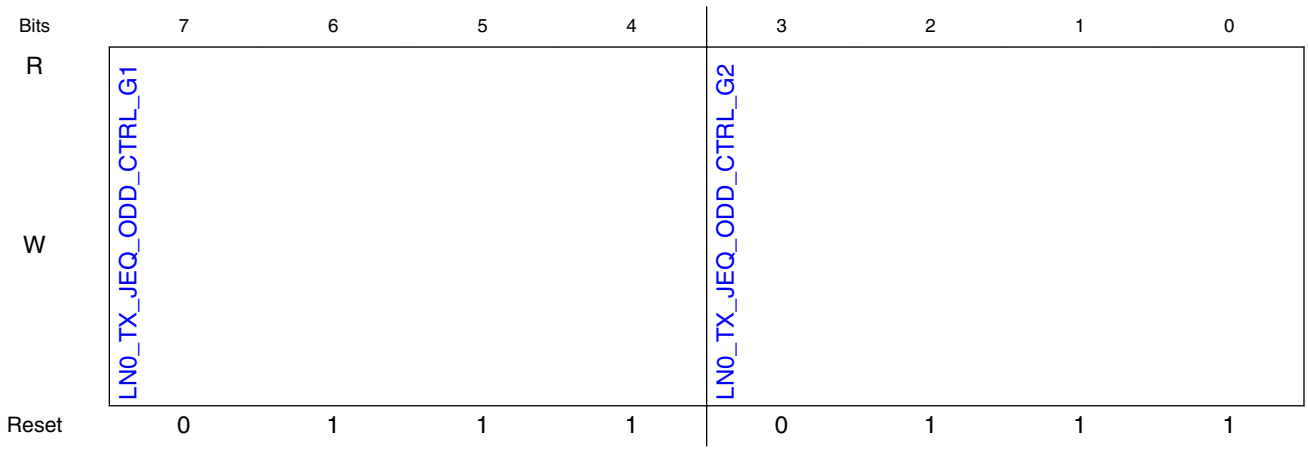
Field	Function
7-4 LN0_TX_JEQ_EVEN_CTRL_G3	[GEN3]
3-0 LN0_TX_JEQ_EVEN_CTRL_G4	[GEN4]

11.4.3.1.148 (TRSV_REG015)

11.4.3.1.148.1 Offset

Register	Offset
TRSV_REG015	454h

11.4.3.1.148.2 Diagram



11.4.3.1.148.3 Fields

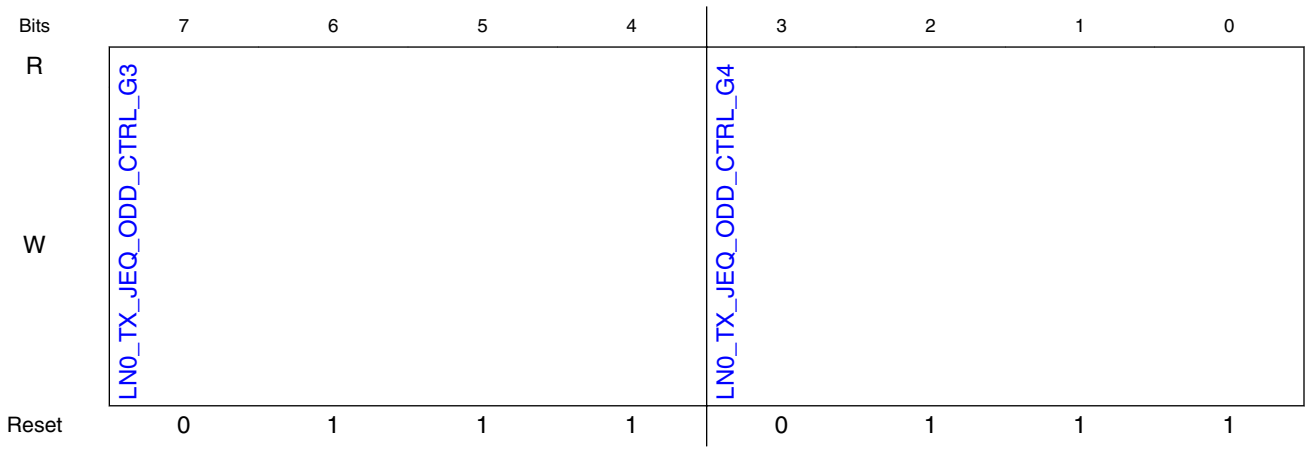
Field	Function
7-4 LN0_TX_JEQ_ODD_CTRL_G1	[GEN1] TX jitter EQ driver (odd) strength control
3-0 LN0_TX_JEQ_ODD_CTRL_G2	[GEN2]

11.4.3.1.149 (TRSV_REG016)

11.4.3.1.149.1 Offset

Register	Offset
TRSV_REG016	458h

11.4.3.1.149.2 Diagram



11.4.3.1.149.3 Fields

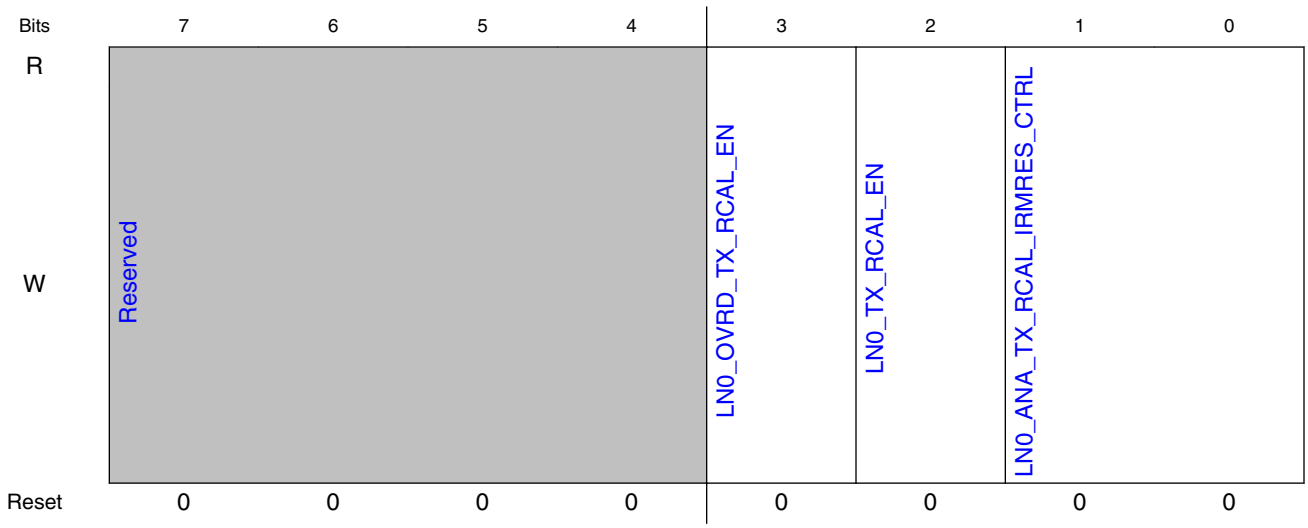
Field	Function
7-4 LN0_TX_JEQ_ODD_CTRL_G3	[GEN3]
3-0 LN0_TX_JEQ_ODD_CTRL_G4	[GEN4]

11.4.3.1.150 (TRSV_REG017)

11.4.3.1.150.1 Offset

Register	Offset
TRSV_REG017	45Ch

11.4.3.1.150.2 Diagram



11.4.3.1.150.3 Fields

Field	Function
7-4 —	Reserved
3 LN0_OVRD_TX_RCAL_EN	Override enable for tx_rcal_en
2 LN0_TX_RCAL_EN	TX RCAL enable 0: Disable, 1: Enable
1-0 LN0_ANA_TX_RCAL_IRMRES_CTRL	TX RCAL rmres bias current control 00: 85uA, 01:100uA, 10:100uA, 11:115uA T-coil + 42.5ohm cal: 2'b00 T-coil + 50ohm cal: 2'b01 w/o T-coil + 42.5ohm cal: 2'b01 w/o T-coil + 50ohm cal: 2'b11

11.4.3.1.151 (TRSV_REG018)

11.4.3.1.151.1 Offset

Register	Offset
TRSV_REG018	460h

11.4.3.1.151.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_TX_RTERM_42P5_EN_G1	LN0_TX_RTERM_42P5_EN_G2	LN0_TX_RTERM_42P5_EN_G3	LN0_TX_RTERM_42P5_EN_G4	LN0_OVRD_TX_RXD_COMP_EN	LN0_TX_RXD_COMP_EN	LN0_OVRD_TX_RXD_EN	LN0_TX_RXD_EN
W								
Reset	0	0	1	1	0	0	0	0

11.4.3.1.151.3 Fields

Field	Function
7 LN0_TX_RTERM_42P5_EN_G1	[GEN1]
6 LN0_TX_RTERM_42P5_EN_G2	[GEN2]
5 LN0_TX_RTERM_42P5_EN_G3	[GEN3]
4 LN0_TX_RTERM_42P5_EN_G4	[GEN4]
3 LN0_OVRD_TX_RXD_COMP_EN	Override enable for tx_rxd_comp_en
2 LN0_TX_RXD_COMP_EN	TX receiver detector comparator enable 0: Disable, 1: Enable
1	Override enable for tx_rxd_en

Table continues on the next page...

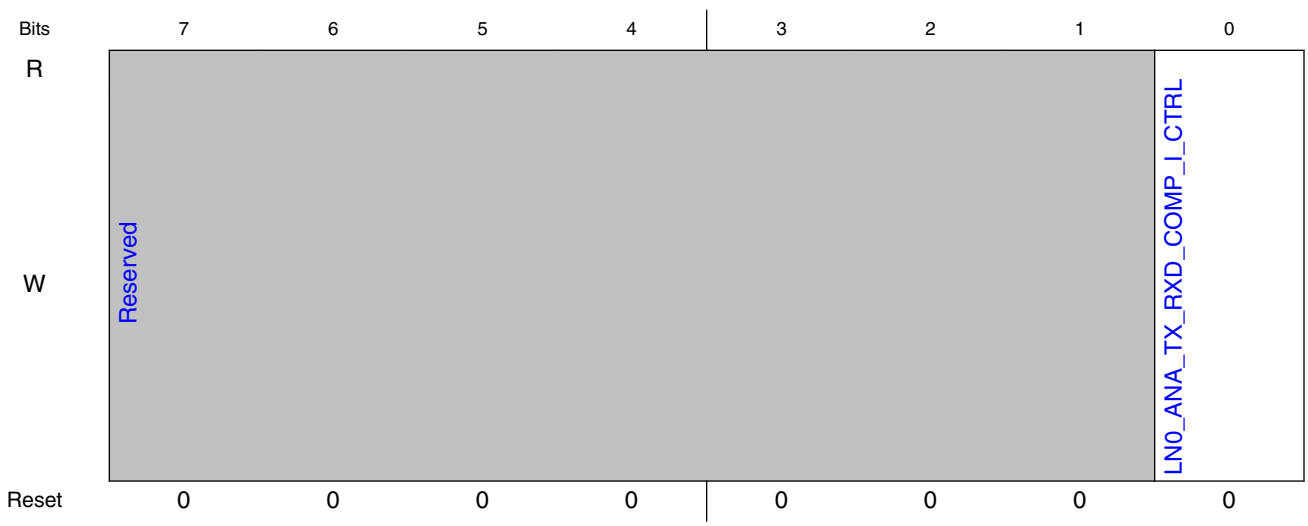
Field	Function
LN0_OVRD_TX_RXD_EN	
0 LN0_TX_RXD_EN	TX receiver detector enable. Drives a transition on the serial data and measures the charge time of the line in order to determine whether a receiver is connected. 0: normal operation, 1: initiate a receiver detect sequence

11.4.3.1.152 (TRSV_REG019)

11.4.3.1.152.1 Offset

Register	Offset
TRSV_REG019	464h

11.4.3.1.152.2 Diagram



11.4.3.1.152.3 Fields

Field	Function
7-1 —	Reserved
0 LN0_ANA_TX_RXD_COMP_I_CTRL	TX receiver detector comparator bias control 0: 1x, 1: 3x

11.4.3.1.153 (TRSV_REG01A)

11.4.3.1.153.1 Offset

Register	Offset
TRSV_REG01A	468h

11.4.3.1.153.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	Reserved		LN0_TX_SER_40BIT_EN_G1	LN0_TX_SER_40BIT_EN_G2	LN0_TX_SER_40BIT_EN_G3	LN0_TX_SER_40BIT_EN_G4	LN0_OVRD_TX_SER_DATA_RSTN	LN0_TX_SER_DATA_RSTN
W								
Reset	0	0	1	1	1	1	0	0

11.4.3.1.153.3 Fields

Field	Function
7-6 —	Reserved
5 LN0_TX_SER_40BIT_EN_G1	[GEN1] TX serializer data width selection 0: 20/16-bit, 1: 40/32-bit
4 LN0_TX_SER_40BIT_EN_G2	[GEN2]
3 LN0_TX_SER_40BIT_EN_G3	[GEN3]
2 LN0_TX_SER_40BIT_EN_G4	[GEN4]

Table continues on the next page...

Field	Function
1 LN0_OVRD_TX_SER_DATA_RSTN	Override enable for tx_ser_data_rstn
0 LN0_TX_SER_DATA_RSTN	TX serializer data-path resetn 0: Reset, 1: Released

11.4.3.1.154 (TRSV_REG01B)

11.4.3.1.154.1 Offset

Register	Offset
TRSV_REG01B	46Ch

11.4.3.1.154.2 Diagram

Bits	7	6	5	4	3	2	1	0
R								
W	LN0_TX_SER_RATE_SEL_G1	LN0_TX_SER_RATE_SEL_G2	LN0_TX_SER_RATE_SEL_G3	LN0_TX_SER_RATE_SEL_G4	LN0_OVRD_TX_SER_CLK_RSTN	LN0_TX_SER_CLK_RSTN	LN0_ANA_TX_CDR_CLK_MON_EN	LN0_ANA_TX_SER_TXCLK_INV
Reset	0	0	1	1	0	0	0	0

11.4.3.1.154.3 Fields

Field	Function
7 LN0_TX_SER_RATE_SEL_G1	[GEN1] TX serializer data rate selection for Gen4 (Need to be controlled with i_tx_en_40bit) 0: 20/40-bit 1: 16/32-bit

Table continues on the next page...

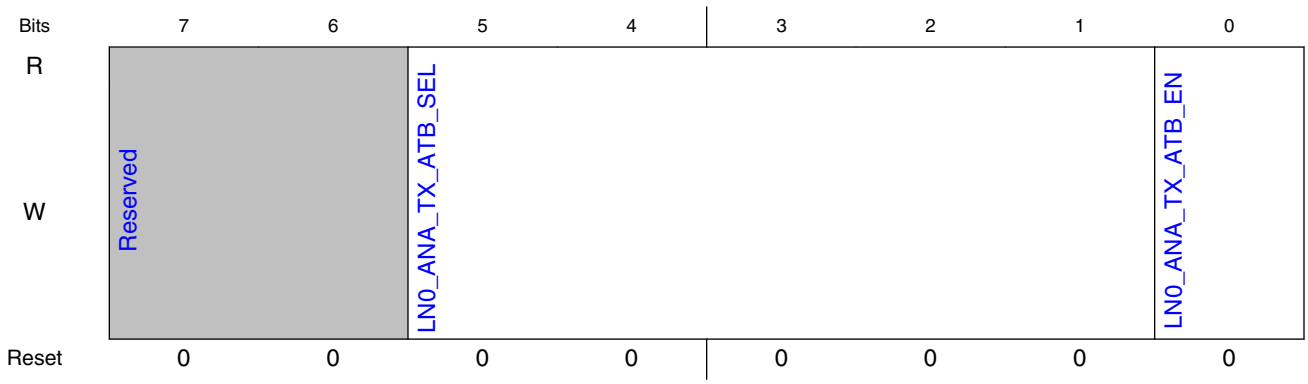
Field	Function
6 LN0_TX_SER_RATE_SEL_G2	[GEN2]
5 LN0_TX_SER_RATE_SEL_G3	[GEN3]
4 LN0_TX_SER_RATE_SEL_G4	[GEN4]
3 LN0_OVRD_TX_SER_CLK_RSTN	Override enable for tx_ser_clk_rstn
2 LN0_TX_SER_CLK_RSTN	TX serializer clock-path resetn 0: Reset, 1: Released
1 LN0_ANA_TX_CDR_CLK_MON_EN	TX serializer clock selection 0: PLL clock, 1: CDR clock
0 LN0_ANA_TX_SER_TXCLK_INV	TX byte clock polarity inversion 0: No inversion, 1: Inversion

11.4.3.1.155 (TRSV_REG01C)

11.4.3.1.155.1 Offset

Register	Offset
TRSV_REG01C	470h

11.4.3.1.155.2 Diagram



11.4.3.1.155.3 Fields

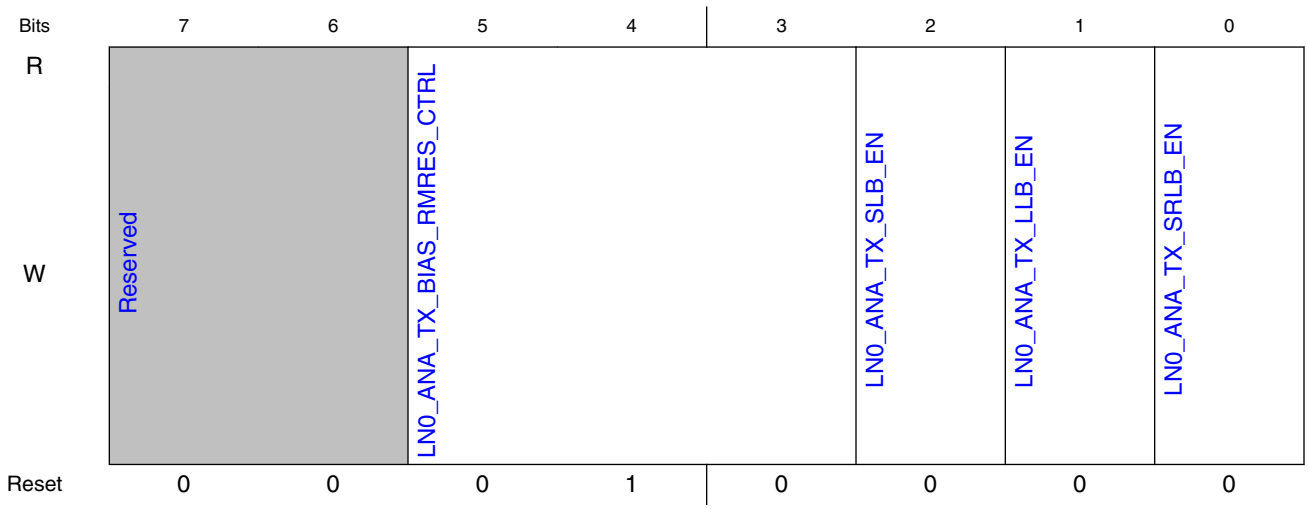
Field	Function
7-6 —	Reserved
5-1 LN0_ANA_TX_ATB_SEL	0000: Serailizer VDD, 0001: Pre Driver VDD, 0010: Driver VDD, 0011: Driver VDDH, 0100: Driver VDD, 0101: Serailizer VSS, 0110: Pre Driver VSS, 0111: Driver VSS, 1000: RCVD detected, 1001: VINM_FB_REPLICA (AUX BIAS), 1010: VBP_BIAS (AUX BIAS), 1011: VBN_BIAS (AUX BIAS), 1XXX: N/A
0 LN0_ANA_TX_ATB_EN	TX ATB enable

11.4.3.1.156 (TRSV_REG01D)

11.4.3.1.156.1 Offset

Register	Offset
TRSV_REG01D	474h

11.4.3.1.156.2 Diagram



11.4.3.1.156.3 Fields

Field	Function
7-6 —	Reserved
5-3 LN0_ANA_TX_BIAS_RMRES_CTRL	RX RMRES bias current control 000: 62%, 001: 87%, 010: 100% (Default), 100: 138%, 111: 156%
2 LN0_ANA_TX_SLB_EN	Serial loopback enable 0: Disable, 1: Enable (PMA digital → TX SER → RX CDR (DES) → PMA digital)
1 LN0_ANA_TX_LLB_EN	Line loopback enalble 0: Disable, 1: Enable (RX pad → RX AFE (CTLE) → TX driver → TX PAD)
0 LN0_ANA_TX_SRLB_EN	Serial retimed loopback enable 0: Disable 1: Enable (RX pad → RX AFE → RX CDR (BBPD) → TX driver → TX pad)

11.4.3.1.157 (TRSV_REG01E)

11.4.3.1.157.1 Offset

Register	Offset
TRSV_REG01E	478h

11.4.3.1.157.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_TX_EQ_2UI_DELAY_EN_G1	LN0_TX_EQ_2UI_DELAY_EN_G2	LN0_TX_EQ_2UI_DELAY_EN_G3	LN0_TX_EQ_2UI_DELAY_EN_G4	LN0_ANA_TX_RESERVED			
W								
Reset	1	0	0	0	0	0	0	0

11.4.3.1.157.3 Fields

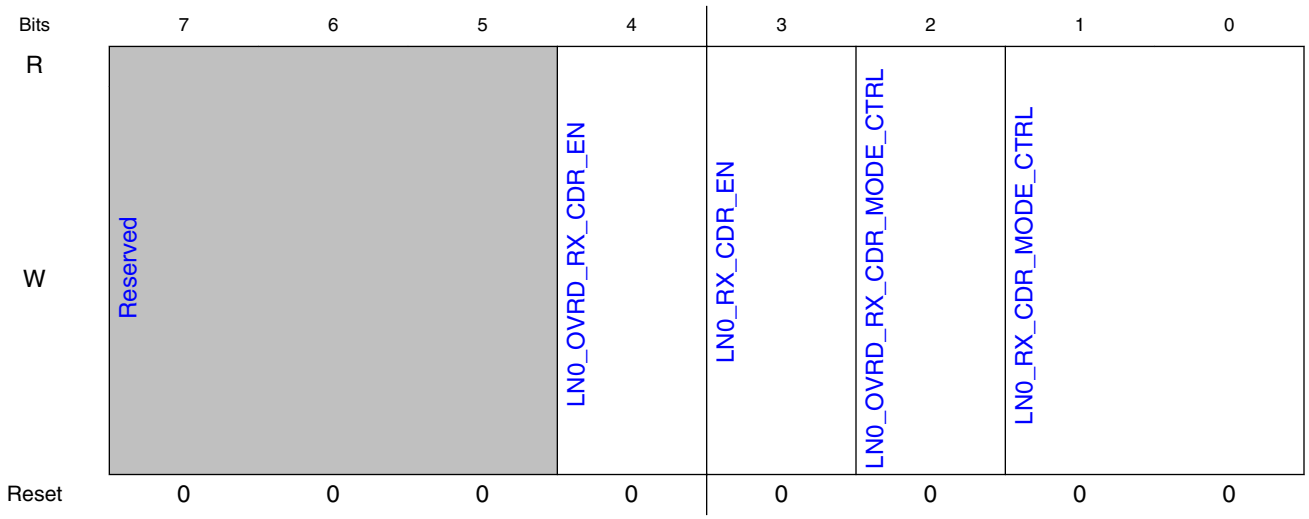
Field	Function
7 LN0_TX_EQ_2UI_DELAY_EN_G1	[GEN1] TX FIR filter delay control when bit-duplication 0: 1UI delay, 1: 2UI delay
6 LN0_TX_EQ_2UI_DELAY_EN_G2	[GEN2]
5 LN0_TX_EQ_2UI_DELAY_EN_G3	[GEN3]
4 LN0_TX_EQ_2UI_DELAY_EN_G4	[GEN4]
3-0 LN0_ANA_TX_RESERVED	Reserved port

11.4.3.1.158 (TRSV_REG01F)

11.4.3.1.158.1 Offset

Register	Offset
TRSV_REG01F	47Ch

11.4.3.1.158.2 Diagram



11.4.3.1.158.3 Fields

Field	Function
7-5 —	Reserved
4 LN0_OVRD_RX_CDR_EN	Override enable for rx_cdr_en
3 LN0_RX_CDR_EN	RX CDR enable 0: Disable, 1: Enable
2 LN0_OVRD_RX_CDR_MODE_CTRL	Override enable for rx_cdr_mode_ctrl
1-0 LN0_RX_CDR_MODE_CTRL	RX CDR mode select 01 : Rx AFC& Calibaration mode, 00 : PLL mode, 10 : CDR clock mode, 11 : CDR data mode

11.4.3.1.159 (TRSV_REG020)

11.4.3.1.159.1 Offset

Register	Offset
TRSV_REG020	480h

11.4.3.1.159.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_RX_CDR_REFDIV_SEL_PLL_G1				LN0_RX_CDR_REFDIV_SEL_PLL_G2			
W								
Reset	0	0	0	0	0	0	0	1

11.4.3.1.159.3 Fields

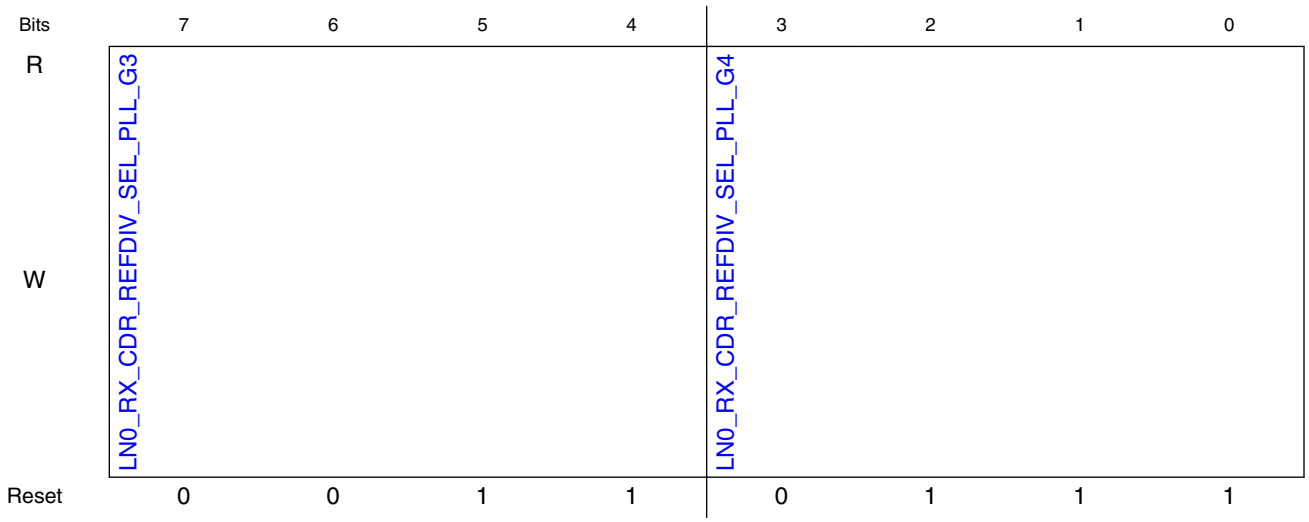
Field	Function
7-4 LN0_RX_CDR_REFDIV_SEL_PLL_G1	[GEN1] [PLL mode] Decision of CDR ref. clock dividing-rate 1111 : div16, 0111 : div8, 0011 : div4, 0001 : div2, 0000: div1
3-0 LN0_RX_CDR_REFDIV_SEL_PLL_G2	[GEN2] [PLL mode]

11.4.3.1.160 (TRSV_REG021)

11.4.3.1.160.1 Offset

Register	Offset
TRSV_REG021	484h

11.4.3.1.160.2 Diagram



11.4.3.1.160.3 Fields

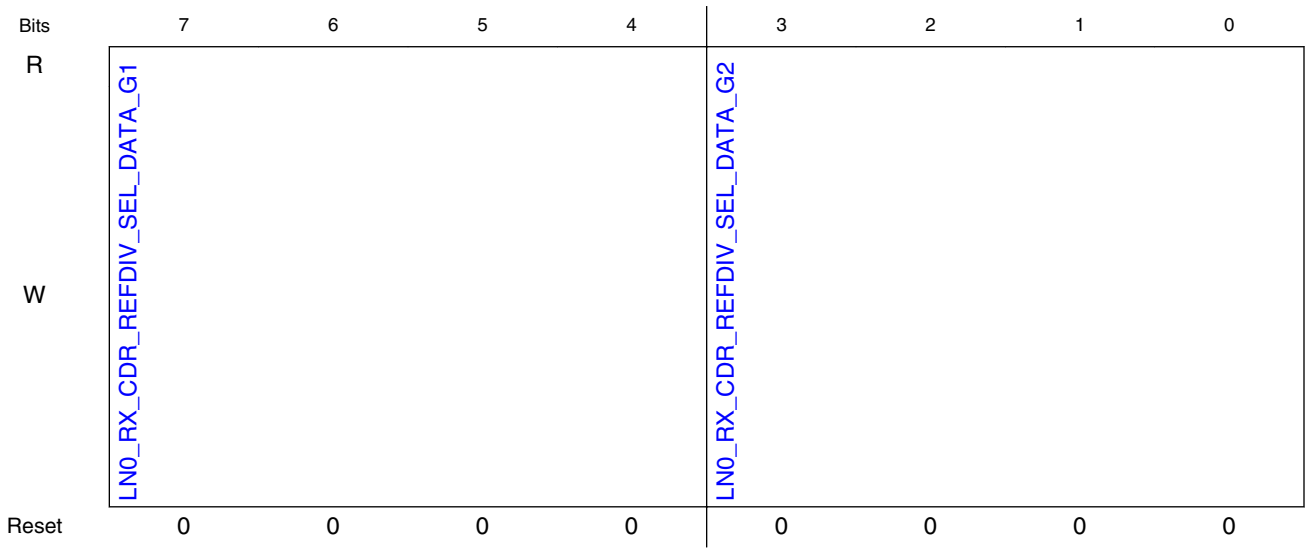
Field	Function
7-4 LN0_RX_CDR_REFDIV_SEL_PLL_G3	[GEN3] [PLL mode]
3-0 LN0_RX_CDR_REFDIV_SEL_PLL_G4	[GEN4] [PLL mode]

11.4.3.1.161 (TRSV_REG022)

11.4.3.1.161.1 Offset

Register	Offset
TRSV_REG022	488h

11.4.3.1.161.2 Diagram



11.4.3.1.161.3 Fields

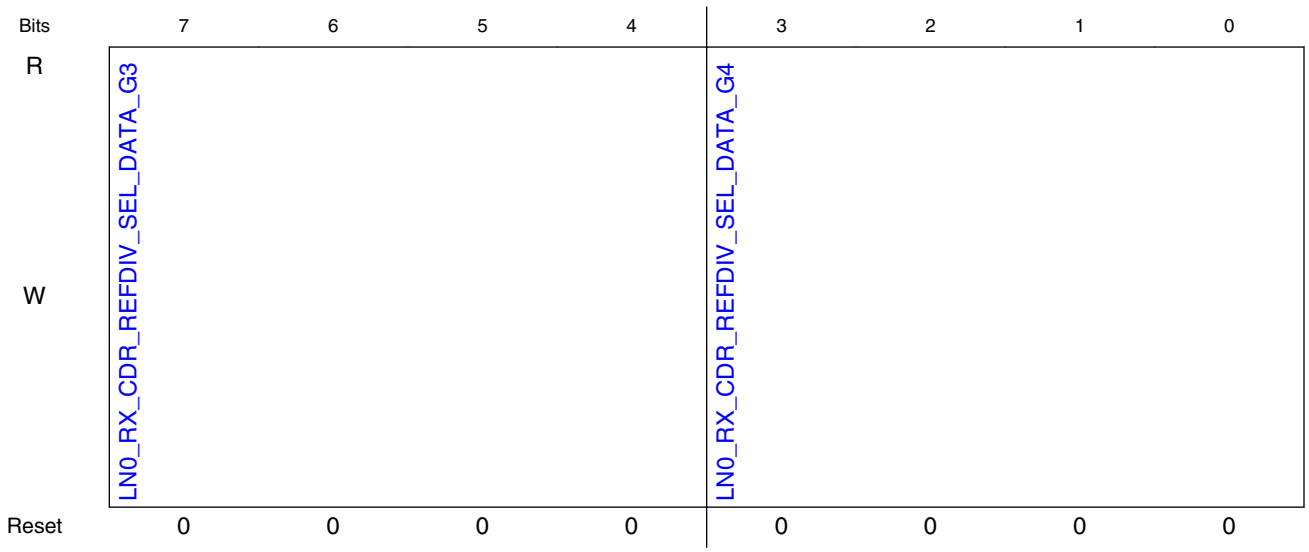
Field	Function
7-4 LN0_RX_CDR_REFDIV_SEL_DATA_G1	[GEN1] [Data mode] Decision of CDR ref. divider ratio 1111 : div16, 0111 : div8, 0011 : div4, 0001 : div2, 0000: div1
3-0 LN0_RX_CDR_REFDIV_SEL_DATA_G2	[GEN2] [Data mode]

11.4.3.1.162 (TRSV_REG023)

11.4.3.1.162.1 Offset

Register	Offset
TRSV_REG023	48Ch

11.4.3.1.162.2 Diagram



11.4.3.1.162.3 Fields

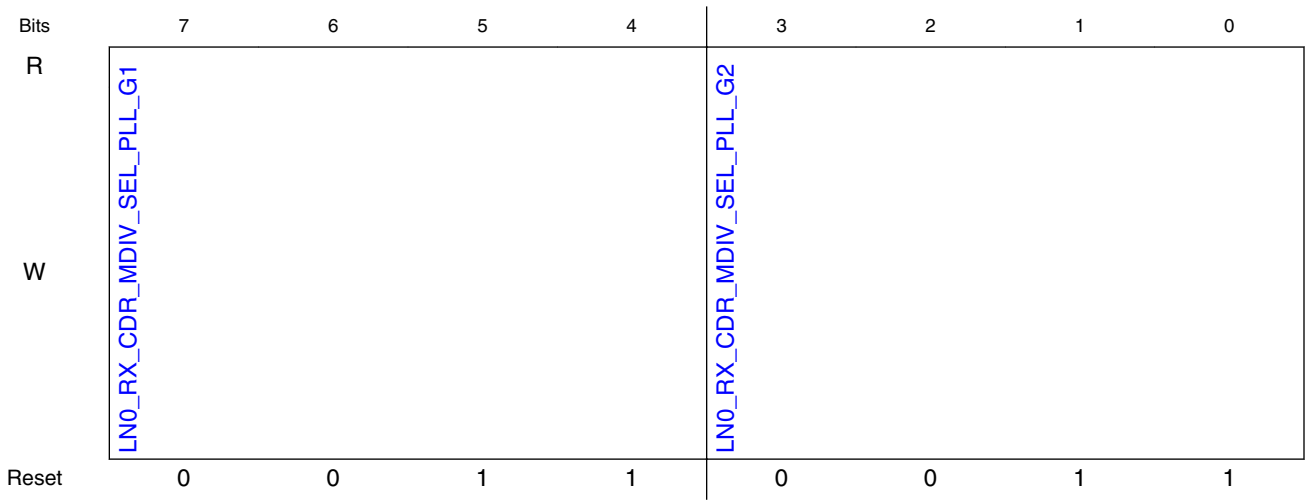
Field	Function
7-4 LN0_RX_CDR_REFDIV_SEL_DATA_G3	[GEN3] [Data mode]
3-0 LN0_RX_CDR_REFDIV_SEL_DATA_G4	[GEN4] [Data mode]

11.4.3.1.163 (TRSV_REG024)

11.4.3.1.163.1 Offset

Register	Offset
TRSV_REG024	490h

11.4.3.1.163.2 Diagram



11.4.3.1.163.3 Fields

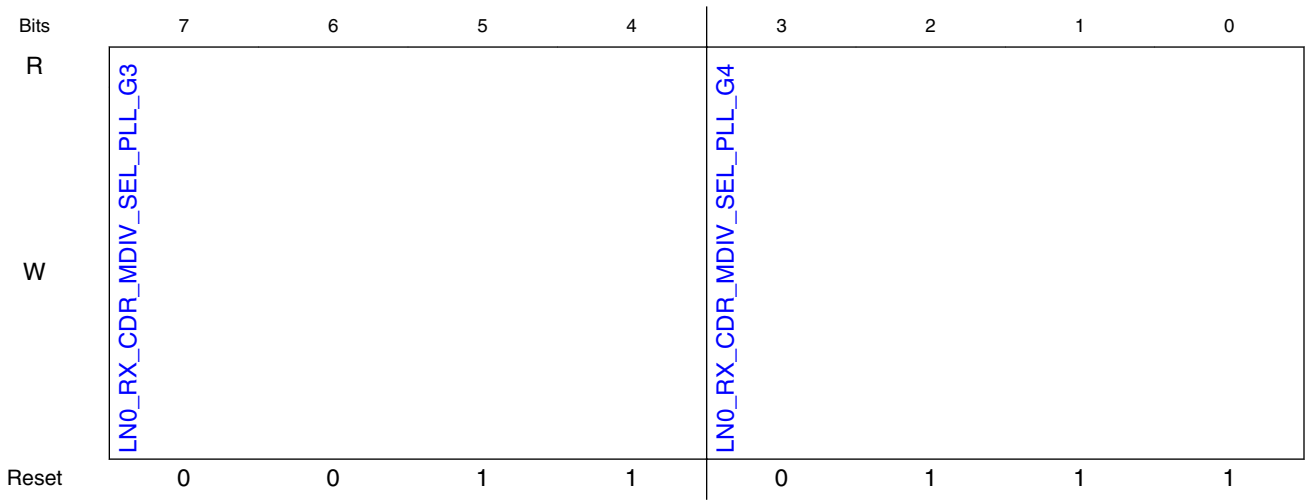
Field	Function
7-4 LN0_RX_CDR_MDIV_SEL_PLL_G1	[GEN1] [PLL mode] Decision of CDR main-divider ratio 1111 : div16, 0111 : div8, 0011 : div4, 0001 : div2, 0000: div1
3-0 LN0_RX_CDR_MDIV_SEL_PLL_G2	[GEN2] [PLL mode]

11.4.3.1.164 (TRSV_REG025)

11.4.3.1.164.1 Offset

Register	Offset
TRSV_REG025	494h

11.4.3.1.164.2 Diagram



11.4.3.1.164.3 Fields

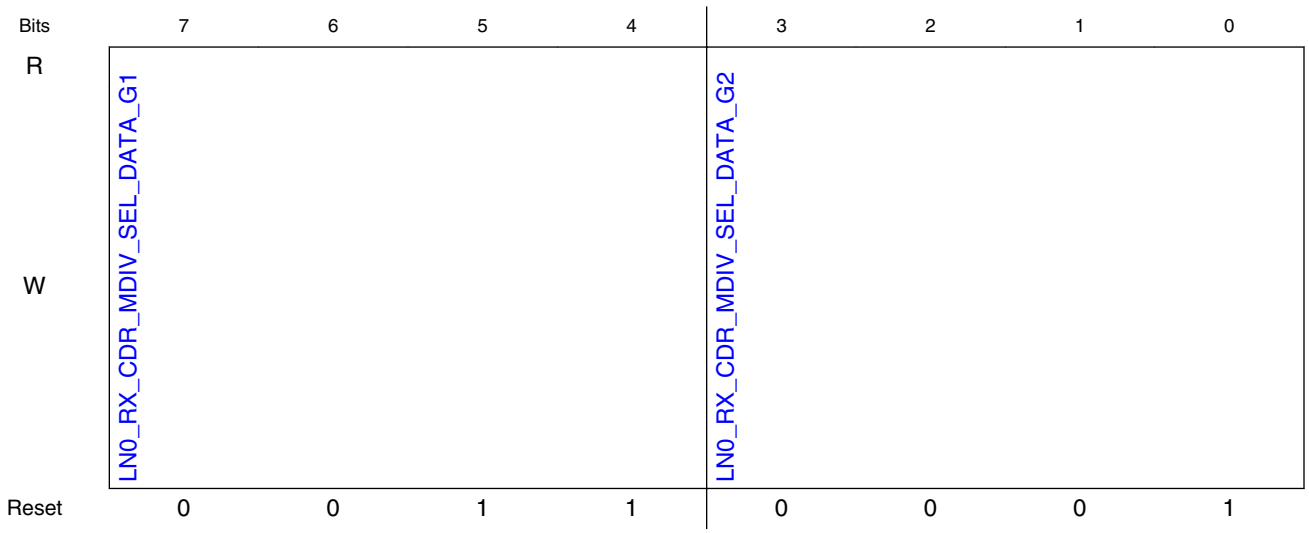
Field	Function
7-4 LN0_RX_CDR_MDIV_SEL_PLL_G3	[GEN3] [PLL mode]
3-0 LN0_RX_CDR_MDIV_SEL_PLL_G4	[GEN4] [PLL mode]

11.4.3.1.165 (TRSV_REG026)

11.4.3.1.165.1 Offset

Register	Offset
TRSV_REG026	498h

11.4.3.1.165.2 Diagram



11.4.3.1.165.3 Fields

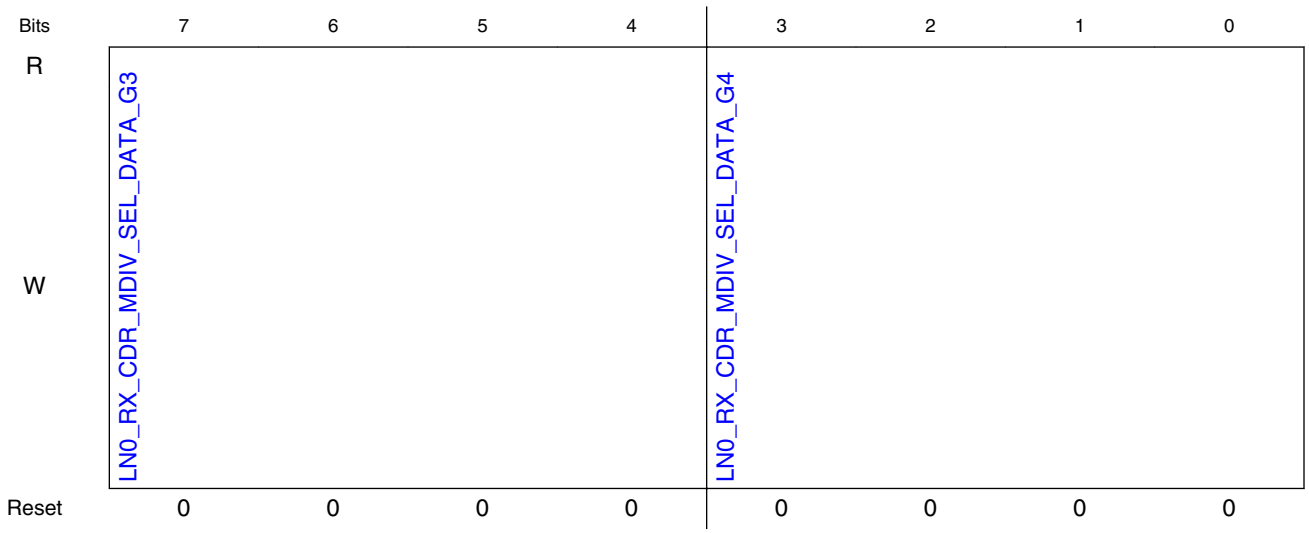
Field	Function
7-4 LN0_RX_CDR_MDIV_SEL_DATA_G1	[GEN1] [Data mode] Decision of CDR main-divider ratio 1111 : div16, 0111 : div8, 0011 : div4, 0001 : div2, 0000: div1
3-0 LN0_RX_CDR_MDIV_SEL_DATA_G2	[GEN2] [Data mode]

11.4.3.1.166 (TRSV_REG027)

11.4.3.1.166.1 Offset

Register	Offset
TRSV_REG027	49Ch

11.4.3.1.166.2 Diagram



11.4.3.1.166.3 Fields

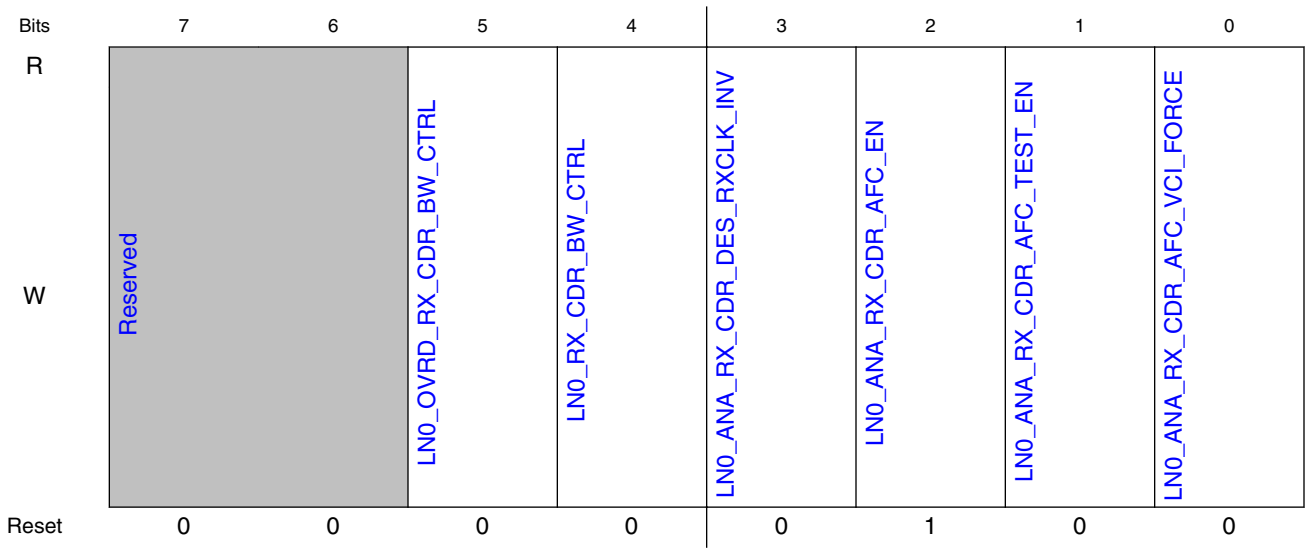
Field	Function
7-4 LN0_RX_CDR_MDIV_SEL_DATA_G3	[GEN3] [Data mode]
3-0 LN0_RX_CDR_MDIV_SEL_DATA_G4	[GEN4] [Data mode]

11.4.3.1.167 (TRSV_REG028)

11.4.3.1.167.1 Offset

Register	Offset
TRSV_REG028	4A0h

11.4.3.1.167.2 Diagram



11.4.3.1.167.3 Fields

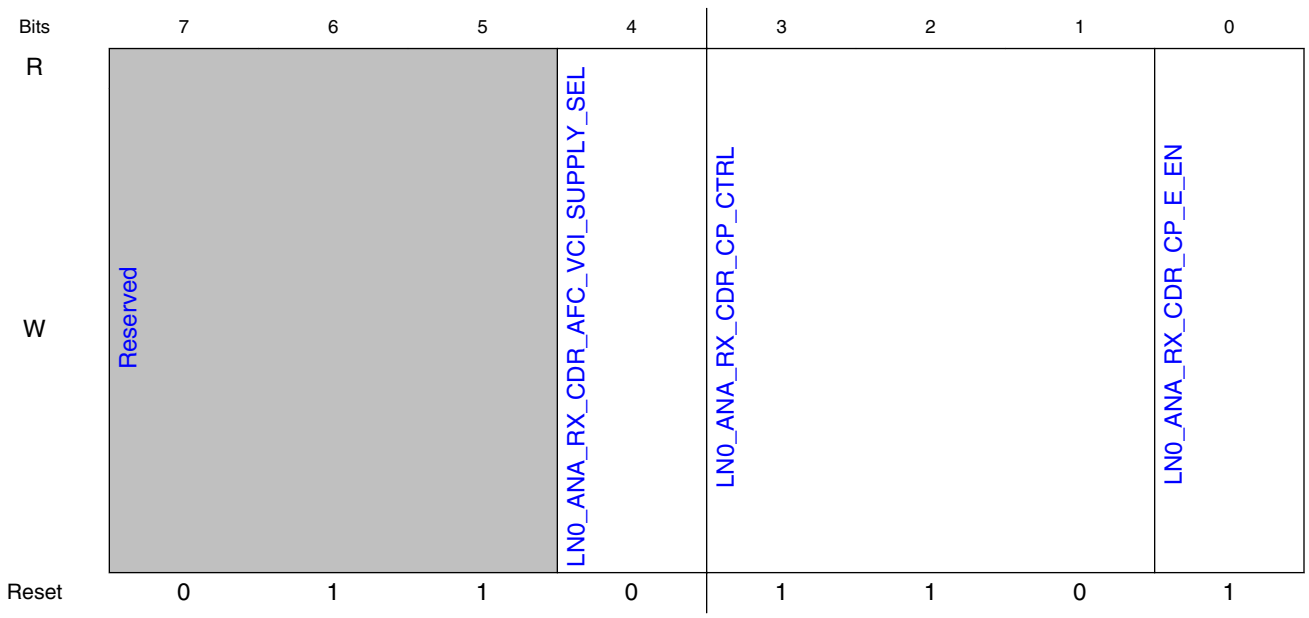
Field	Function
7-6 —	Reserved
5 LN0_OVRD_RX_CDR_BW_CTRL	Override enable for rx_cdr_bw_ctrl
4 LN0_RX_CDR_BW_CTRL	RX CDR bandwidth control 0: wide bandwidth, 1: narror bandwidth
3 LN0_ANA_RX_CDR_DES_RXCLK_INV	RX byte clock polarity inversion 0: No inversion, 1: Inversion
2 LN0_ANA_RX_CDR_AFC_EN	RX CDR AFC enable 0: Disable, 1: Enable
1 LN0_ANA_RX_CDR_AFC_TEST_EN	RX CDR test mode enable 0: Disable, 1: Enable
0 LN0_ANA_RX_CDR_AFC_VCI_FORCE	RX CDR control voltage force

11.4.3.1.168 (TRSV_REG029)

11.4.3.1.168.1 Offset

Register	Offset
TRSV_REG029	4A4h

11.4.3.1.168.2 Diagram



11.4.3.1.168.3 Fields

Field	Function
7-5 —	Reserved
4 LN0_ANA_RX_CDR_AFC_VCI_SUPPLY_SEL	RX CDR VCI reference voltage selection 0: Regulated 820mV or VDD after LPF (depending on i_rx_cdr_cp_reg_in_sel), 1: VDD
3-1 LN0_ANA_RX_CDR_CP_CTRL	RX CDR charge pump current control (Ieven + Iodd) 000: 2.5uA, 001: 3.2uA, 010: 4.2uA, 011: 6.3uA 100: 1.4uA, 101: 1.6uA, 110: 1.8uA, 111: 2.1uA
0	RX CDR even charge-pump enable

Field	Function
LN0_ANA_RX_CDR_CP_E_EN	0: Disable, 1: Enable

11.4.3.1.169 (TRSV_REG02A)

11.4.3.1.169.1 Offset

Register	Offset
TRSV_REG02A	4A8h

11.4.3.1.169.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	Reserved	LN0_ANA_RX_CDR_CP_O_EN	LN0_ANA_RX_CDR_CP_VREG_IN_SEL	LN0_ANA_RX_CDR_CP_VREG_LPF_EN	LN0_OVRD_RX_CDR_FBB_CAL_EN	LN0_RX_CDR_FBB_CAL_EN	LN0_OVRD_RX_CDR_VCO_STARTUP	LN0_RX_CDR_VCO_STARTUP
W								
Reset	0	1	0	0	1	0	0	0

11.4.3.1.169.3 Fields

Field	Function
7 —	Reserved
6 LN0_ANA_RX_CDR_CP_O_EN	RX CDR odd charge-pump enable 0: Disable, 1: Enable
5	RX CDR charge pump regulator reference voltage selection 0: BGR 820mV, 1: VDD

Table continues on the next page...

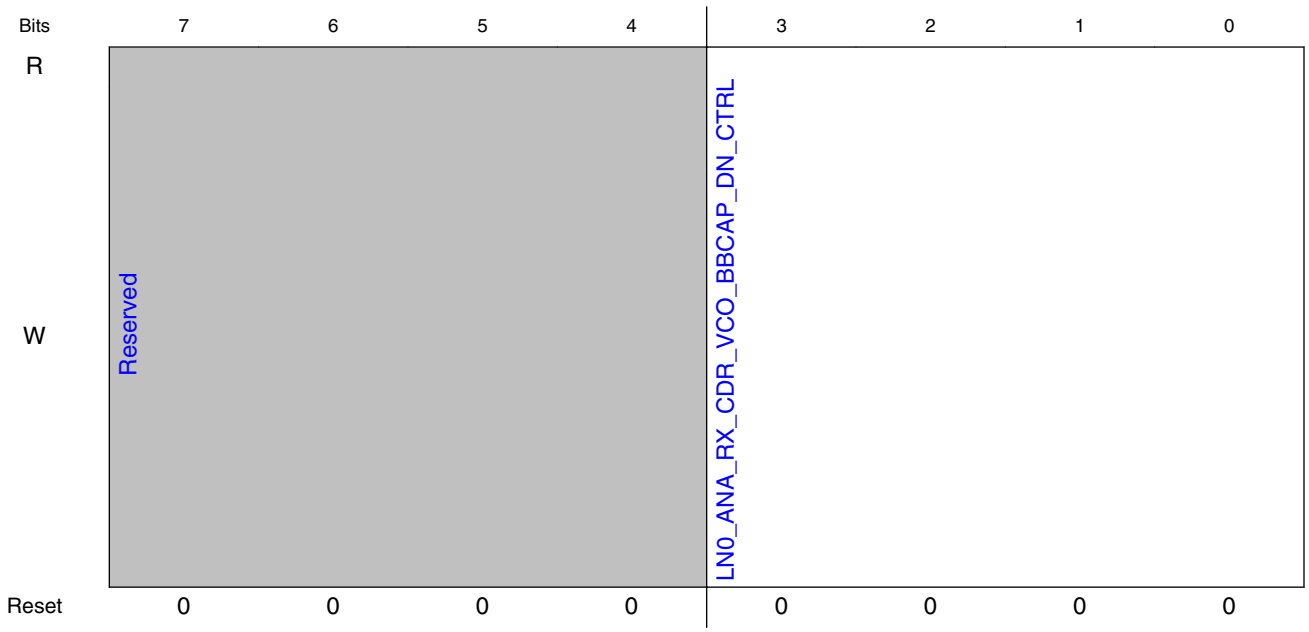
Field	Function
LN0_ANA_RX_CDR_CP_VREG_IN_SEL	
4 LN0_ANA_RX_CDR_CP_VREG_LPF_EN	LPF enable for RX CDR charge pump regualtor 0: Disable, 1: Enable
3 LN0_OVRD_RX_CDR_FBB_CAL_EN	Override enable for rx_cdr_fbb_cal_en
2 LN0_RX_CDR_FBB_CAL_EN	RX CDR FBB calibration enable 0: Disable, 1: Enable
1 LN0_OVRD_RX_CDR_VCO_STARTUP	Override enable for rx_cdr_vco_startup
0 LN0_RX_CDR_VCO_STARTUP	RX CDR VCO startup signal, low to high transition

11.4.3.1.170 (TRSV_REG02B)

11.4.3.1.170.1 Offset

Register	Offset
TRSV_REG02B	4ACh

11.4.3.1.170.2 Diagram



11.4.3.1.170.3 Fields

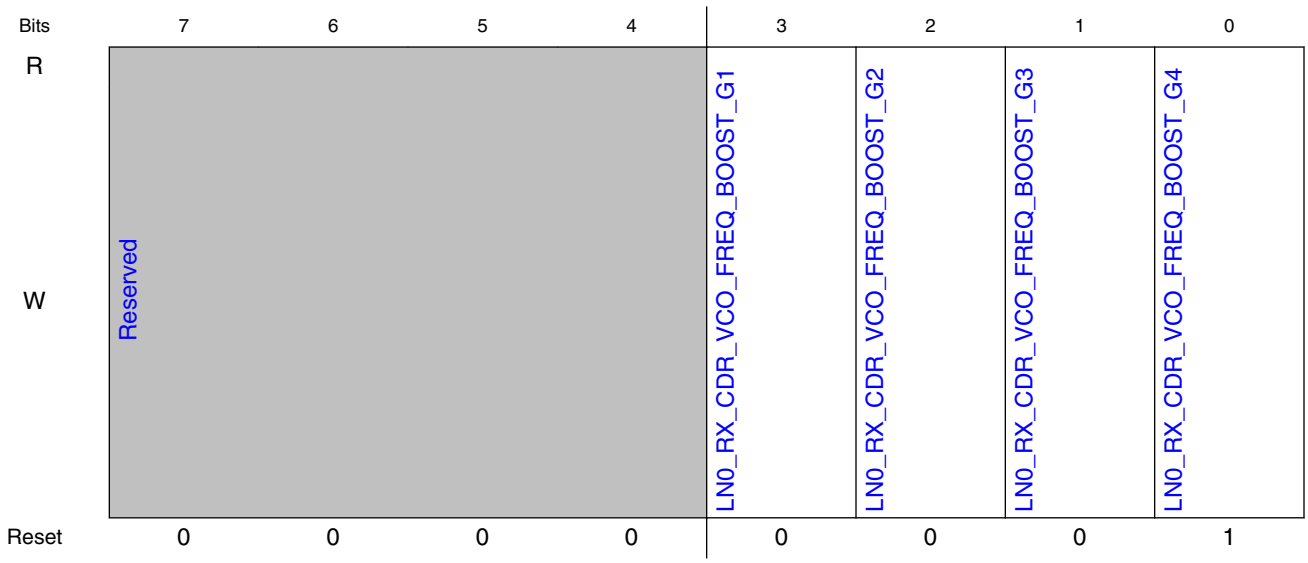
Field	Function
7-4 —	Reserved
3-0 LN0_ANA_RX_CDR_VCO_BBCAP_DN_CTRL	RX CDR BBVCO dummy cap control to decrease frequency 0000: Min freq, 1111: Max freq.

11.4.3.1.171 (TRSV_REG02C)

11.4.3.1.171.1 Offset

Register	Offset
TRSV_REG02C	4B0h

11.4.3.1.171.2 Diagram



11.4.3.1.171.3 Fields

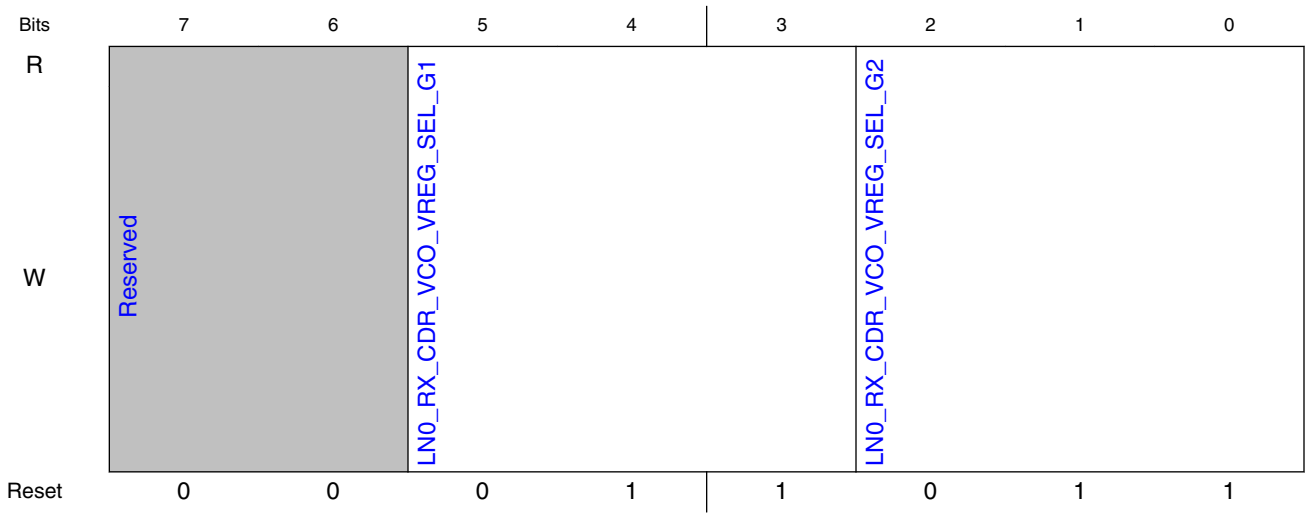
Field	Function
7-4 —	Reserved
3 LN0_RX_CDR_VCO_FREQ_BOOST_G1	[GEN1] RX CDR VCO frequency boost enable 0: Disable, 1: Enable (High freqency)
2 LN0_RX_CDR_VCO_FREQ_BOOST_G2	[GEN2]
1 LN0_RX_CDR_VCO_FREQ_BOOST_G3	[GEN3]
0 LN0_RX_CDR_VCO_FREQ_BOOST_G4	[GEN4]

11.4.3.1.172 (TRSV_REG02D)

11.4.3.1.172.1 Offset

Register	Offset
TRSV_REG02D	4B4h

11.4.3.1.172.2 Diagram



11.4.3.1.172.3 Fields

Field	Function
7-6 —	Reserved
5-3 LN0_RX_CDR_VCO_VREG_SEL_G1	[GEN1] RX CDR voltage regualtor selection 000: 1.0 X Vctrl, 001: 1.2 X Vctrl, 010: 1.4 X Vctrl, 011: 1.6 X Vctrl, 100: 1.8 X Vctrl, 101: 2.0 X Vctrl, 110: 2.2 X Vctrl, 111: 2.4 X Vctrl
2-0 LN0_RX_CDR_VCO_VREG_SEL_G2	[GEN2]

11.4.3.1.173 (TRSV_REG02E)

11.4.3.1.173.1 Offset

Register	Offset
TRSV_REG02E	4B8h

11.4.3.1.173.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_RX_CDR_VCO_VREG_SEL_G3			LN0_RX_CDR_VCO_VREG_SEL_G4			LN0_OVRD_RX_CTLE_EN	LN0_RX_CTLE_EN
W								
Reset	0	1	1	1	0	0	0	0

11.4.3.1.173.3 Fields

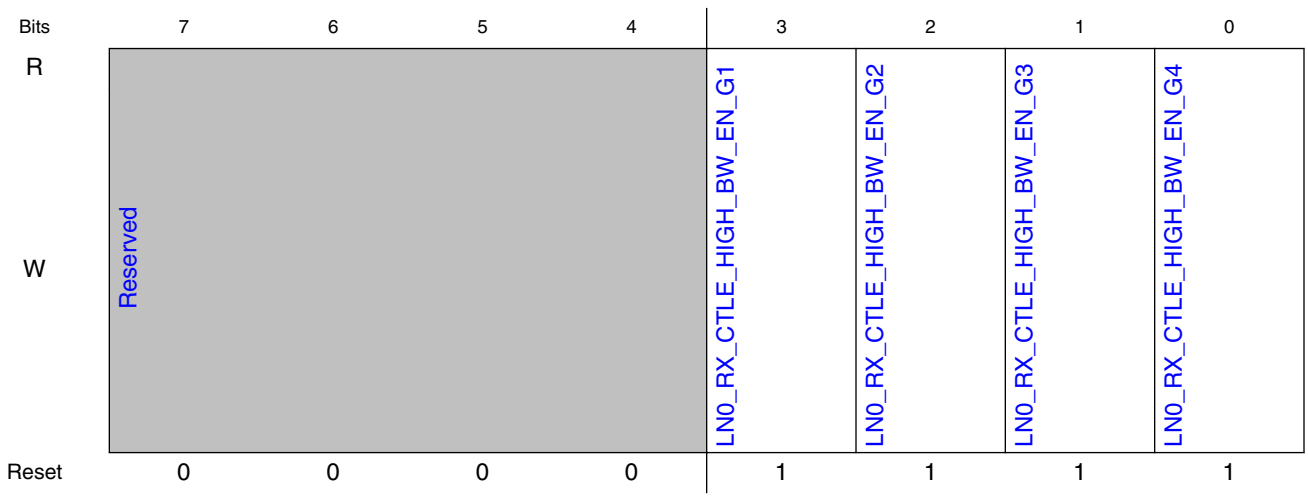
Field	Function
7-5 LN0_RX_CDR_VCO_VREG_SEL_G3	[GEN3]
4-2 LN0_RX_CDR_VCO_VREG_SEL_G4	[GEN4]
1 LN0_OVRD_RX_CTLE_EN	Override enable for rx_ctle_en
0 LN0_RX_CTLE_EN	RX CTLE enable 0: Disable, 1: Enable

11.4.3.1.174 (TRSV_REG02F)

11.4.3.1.174.1 Offset

Register	Offset
TRSV_REG02F	4BCh

11.4.3.1.174.2 Diagram



11.4.3.1.174.3 Fields

Field	Function
7-4 —	Reserved
3 LN0_RX_CTLE_HIGH_BW_EN_G1	[GEN1] RX CTLE bandwidth enhancement by boosting up current 0: Nominal BW, 1: Higher BW
2 LN0_RX_CTLE_HIGH_BW_EN_G2	[GEN2]
1 LN0_RX_CTLE_HIGH_BW_EN_G3	[GEN3]
0	[GEN4]

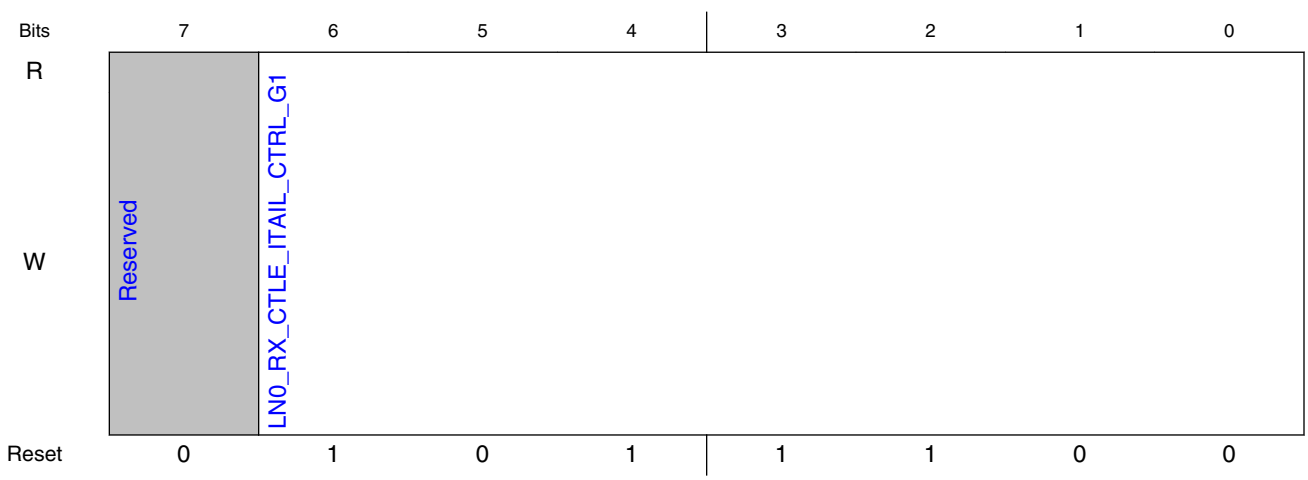
Field	Function
LN0_RX_CTLE_HIGH_BW_EN_G4	

11.4.3.1.175 (TRSV_REG030)

11.4.3.1.175.1 Offset

Register	Offset
TRSV_REG030	4C0h

11.4.3.1.175.2 Diagram



11.4.3.1.175.3 Fields

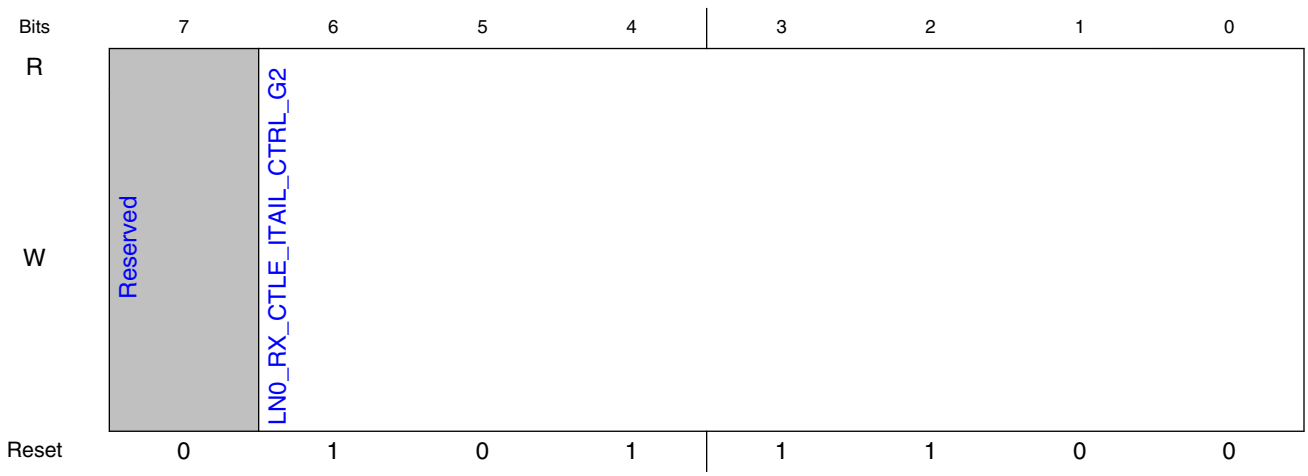
Field	Function
7 —	Reserved
6-0 LN0_RX_CTLE_ITAIL_CTRL_G1	[GEN1] RX CTLE main tail current

11.4.3.1.176 (TRSV_REG031)

11.4.3.1.176.1 Offset

Register	Offset
TRSV_REG031	4C4h

11.4.3.1.176.2 Diagram



11.4.3.1.176.3 Fields

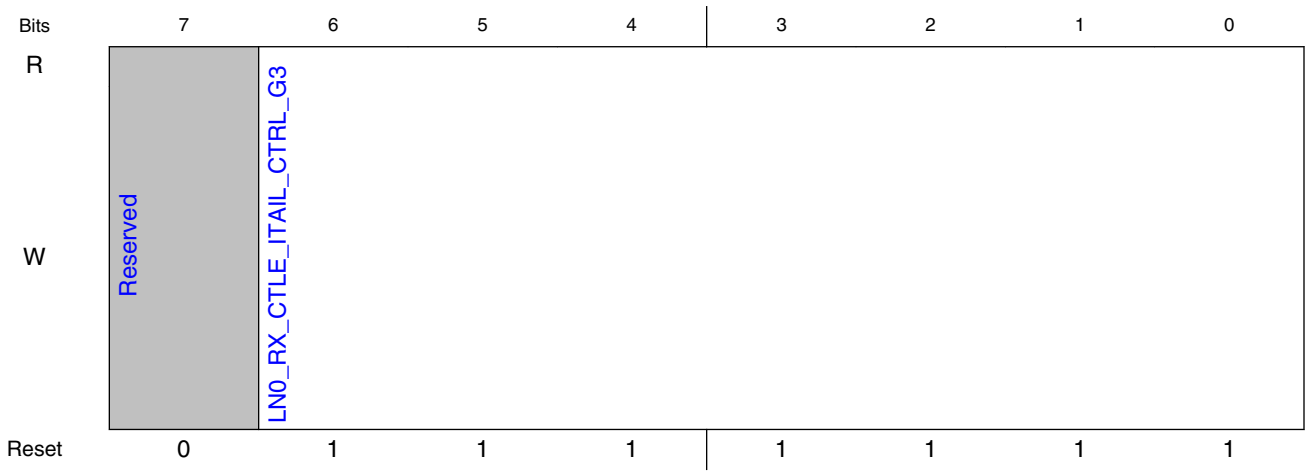
Field	Function
7 —	Reserved
6-0 LN0_RX_CTLE_ITAIL_CTRL_G2	[GEN2]

11.4.3.1.177 (TRSV_REG032)

11.4.3.1.177.1 Offset

Register	Offset
TRSV_REG032	4C8h

11.4.3.1.177.2 Diagram



11.4.3.1.177.3 Fields

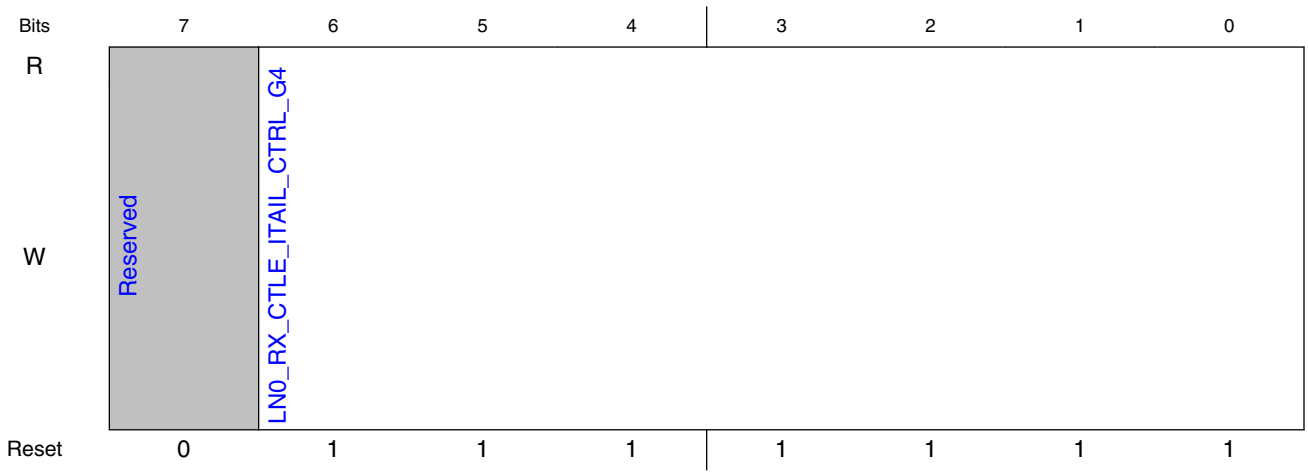
Field	Function
7 —	Reserved
6-0 LN0_RX_CTLE_ITAIL_CTRL_G3	[GEN3]

11.4.3.1.178 (TRSV_REG033)

11.4.3.1.178.1 Offset

Register	Offset
TRSV_REG033	4CCh

11.4.3.1.178.2 Diagram



11.4.3.1.178.3 Fields

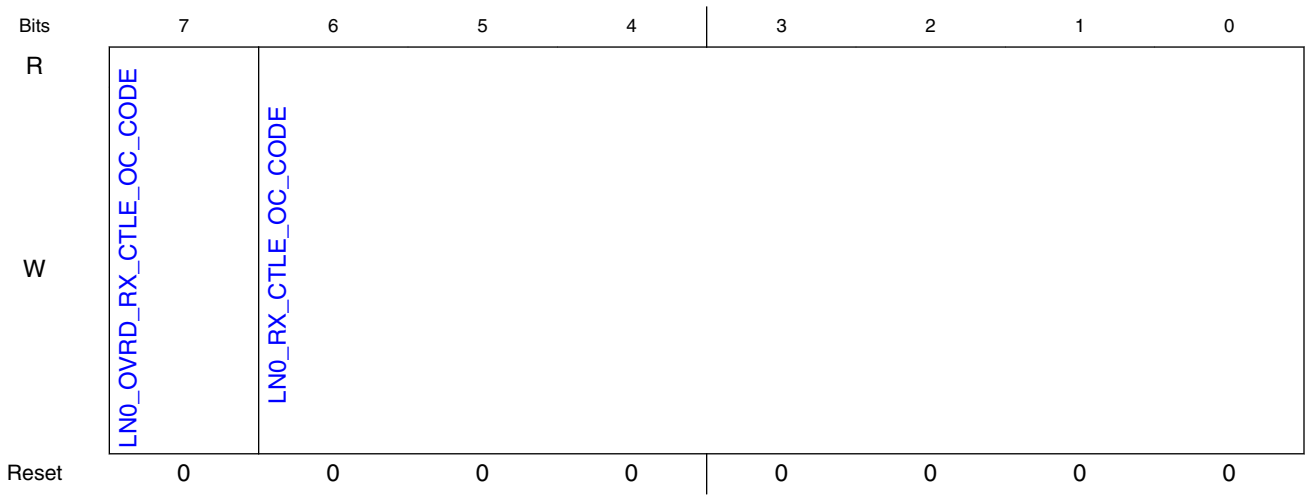
Field	Function
7 —	Reserved
6-0 LN0_RX_CTLE_ITAIL_CTRL_G4	[GEN4]

11.4.3.1.179 (TRSV_REG034)

11.4.3.1.179.1 Offset

Register	Offset
TRSV_REG034	4D0h

11.4.3.1.179.2 Diagram



11.4.3.1.179.3 Fields

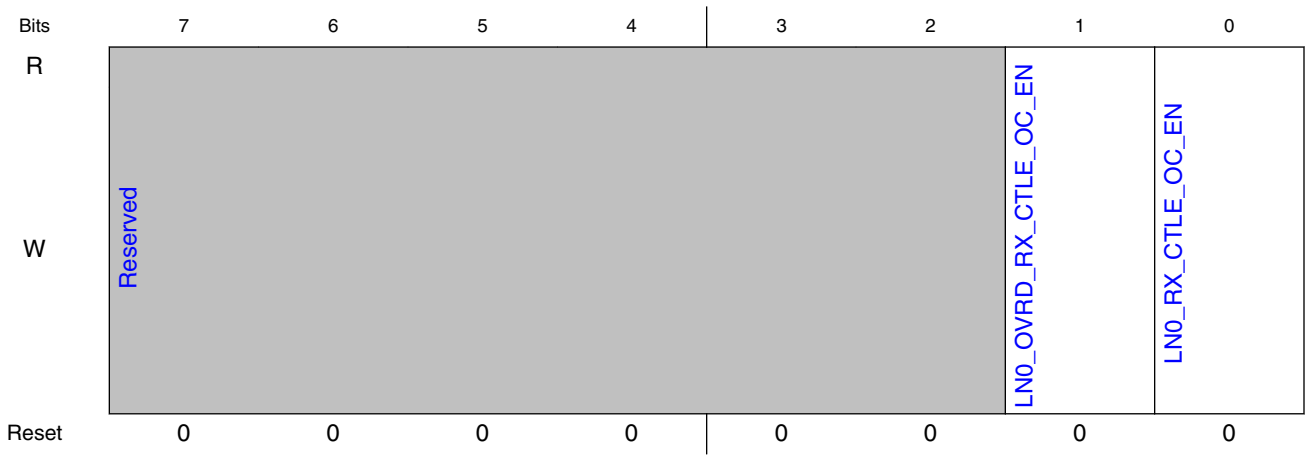
Field	Function
7 LN0_OVRD_RX_CTLE_OC_CODE	Override enable for rx_ctle_oc_code
6-0 LN0_RX_CTLE_OC_CODE	RX CTLE manual offset code 2 MSB: Range control - 00: Disable, 01: Min, 10: Medium, 11: Max range 5 LSB: Offset magnitude - binary weighted and 5'b10000 is center code.

11.4.3.1.180 (TRSV_REG035)

11.4.3.1.180.1 Offset

Register	Offset
TRSV_REG035	4D4h

11.4.3.1.180.2 Diagram



11.4.3.1.180.3 Fields

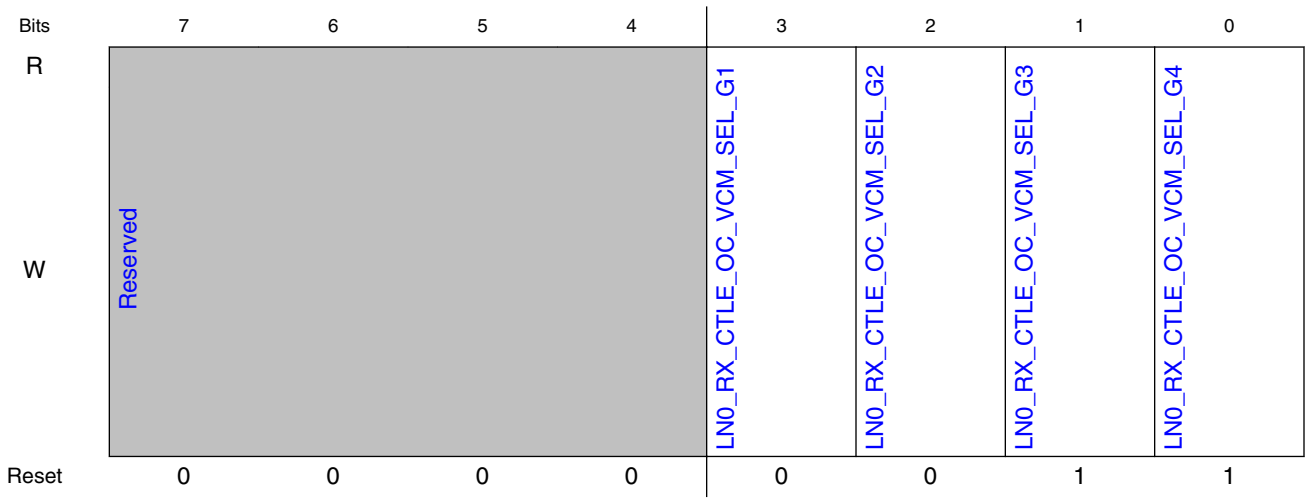
Field	Function
7-2 —	Reserved
1 LN0_OVRD_RX_CTLE_OC_EN	Override enable for rx_ctle_oc_en
0 LN0_RX_CTLE_OC_EN	RX CTLE offset calibration enable 0: Disable, 1: Enable

11.4.3.1.181 (TRSV_REG036)

11.4.3.1.181.1 Offset

Register	Offset
TRSV_REG036	4D8h

11.4.3.1.181.2 Diagram



11.4.3.1.181.3 Fields

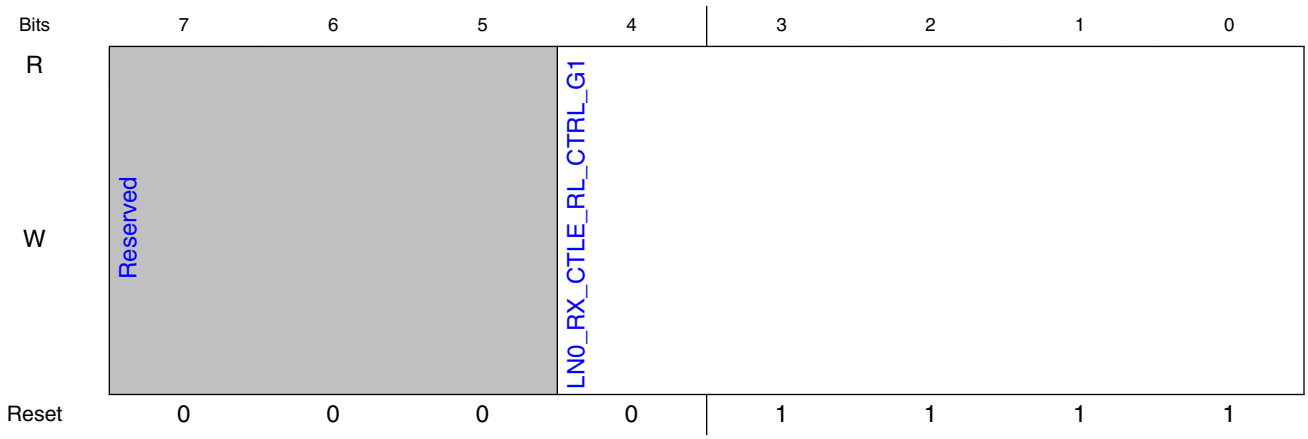
Field	Function
7-4 —	Reserved
3 LN0_RX_CTLE_OC_VCM_SEL_G1	[GEN1] RX CTLE input common-mode selection in offset calibration 0: GND, 1: VCM (~200mV)
2 LN0_RX_CTLE_OC_VCM_SEL_G2	[GEN2]
1 LN0_RX_CTLE_OC_VCM_SEL_G3	[GEN3]
0 LN0_RX_CTLE_OC_VCM_SEL_G4	[GEN4]

11.4.3.1.182 (TRSV_REG037)

11.4.3.1.182.1 Offset

Register	Offset
TRSV_REG037	4DCh

11.4.3.1.182.2 Diagram



11.4.3.1.182.3 Fields

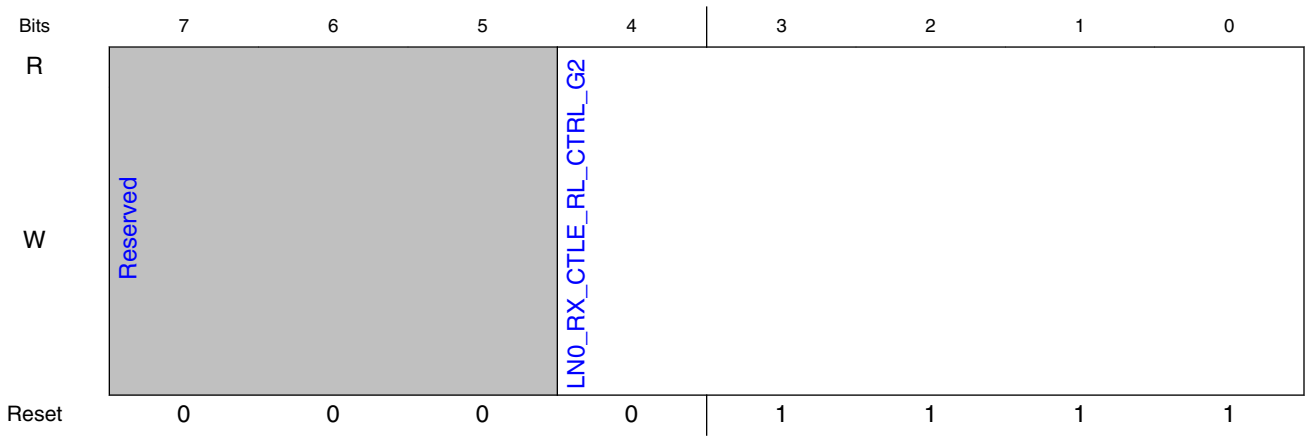
Field	Function
7-5 —	Reserved
4-0 LN0_RX_CTLE_RL_CTRL_G1	[GEN1] RX CTLE load resistance control for Gen1 0000: Max resistance, 1111: Min resistance 3'b001 for Gen1 and 2, 3'b011 for Gen3, and 3'b111 for Gen4 is recommended

11.4.3.1.183 (TRSV_REG038)

11.4.3.1.183.1 Offset

Register	Offset
TRSV_REG038	4E0h

11.4.3.1.183.2 Diagram



11.4.3.1.183.3 Fields

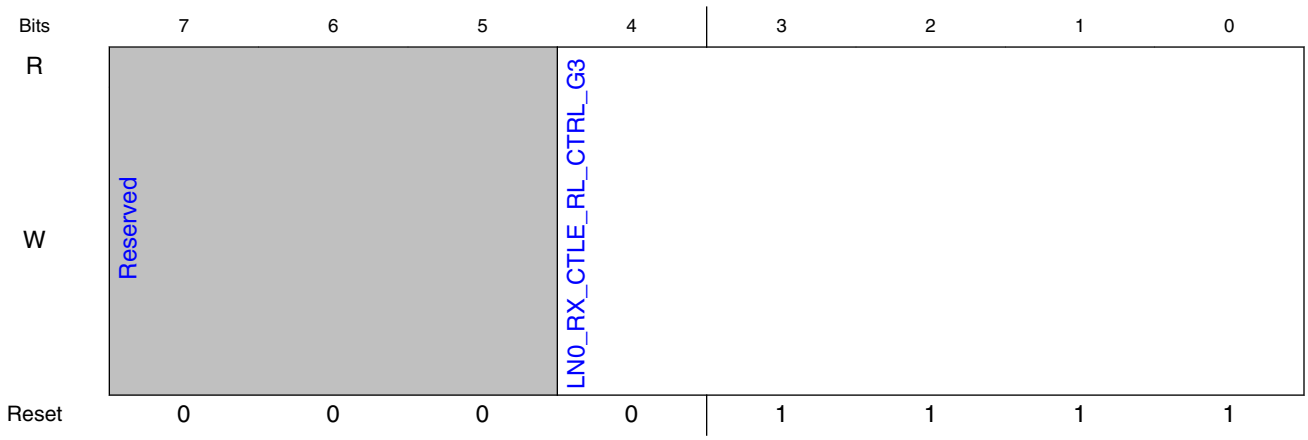
Field	Function
7-5 —	Reserved
4-0 LN0_RX_CTLE_RL_CTRL_G2	[GEN2]

11.4.3.1.184 (TRSV_REG039)

11.4.3.1.184.1 Offset

Register	Offset
TRSV_REG039	4E4h

11.4.3.1.184.2 Diagram



11.4.3.1.184.3 Fields

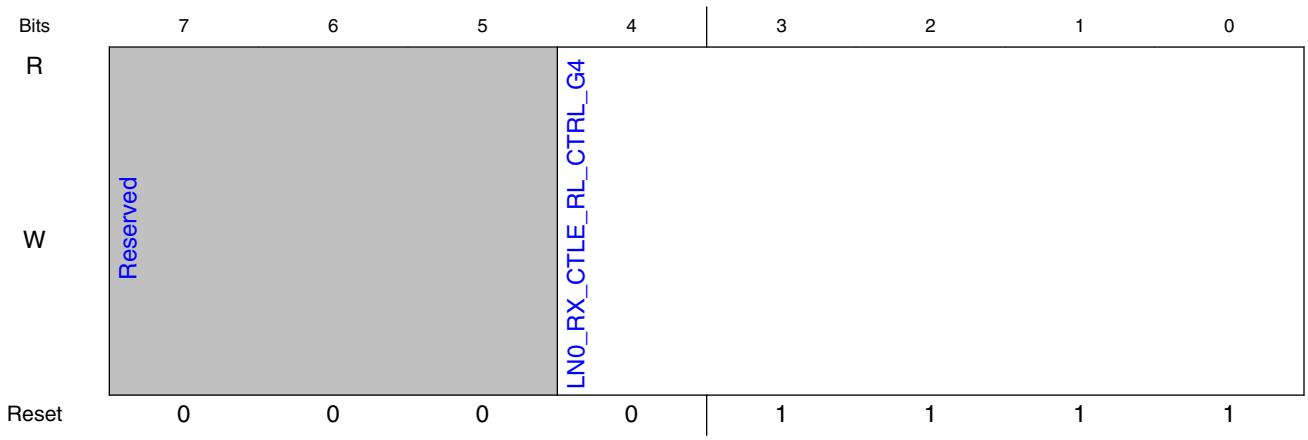
Field	Function
7-5 —	Reserved
4-0 LN0_RX_CTLE_RL_CTRL_G3	[GEN3]

11.4.3.1.185 (TRSV_REG03A)

11.4.3.1.185.1 Offset

Register	Offset
TRSV_REG03A	4E8h

11.4.3.1.185.2 Diagram



11.4.3.1.185.3 Fields

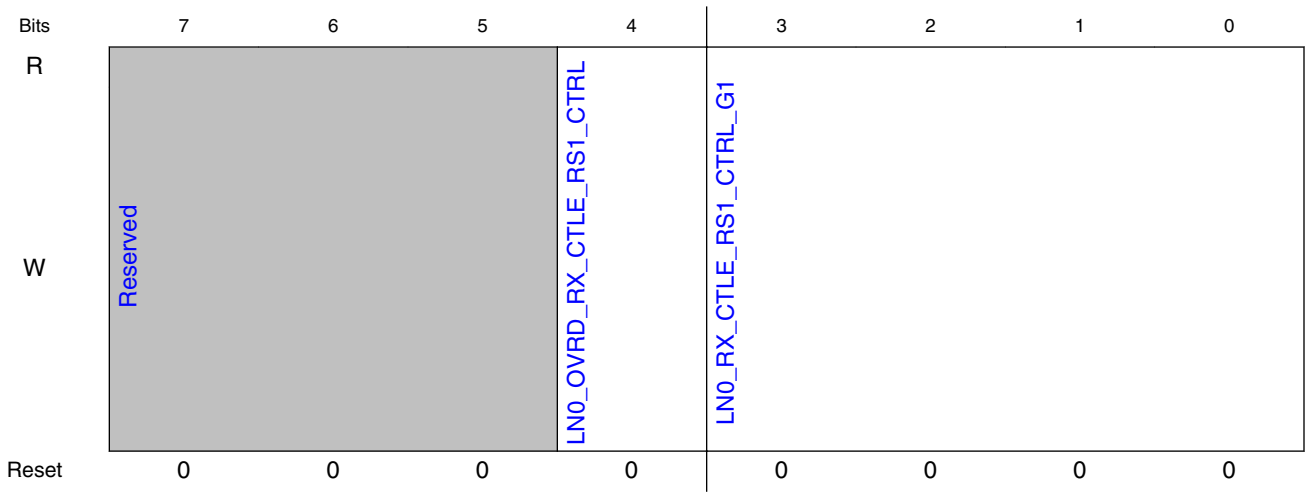
Field	Function
7-5 —	Reserved
4-0 LN0_RX_CTLE_RL_CTRL_G4	[GEN4]

11.4.3.1.186 (TRSV_REG03B)

11.4.3.1.186.1 Offset

Register	Offset
TRSV_REG03B	4ECh

11.4.3.1.186.2 Diagram



11.4.3.1.186.3 Fields

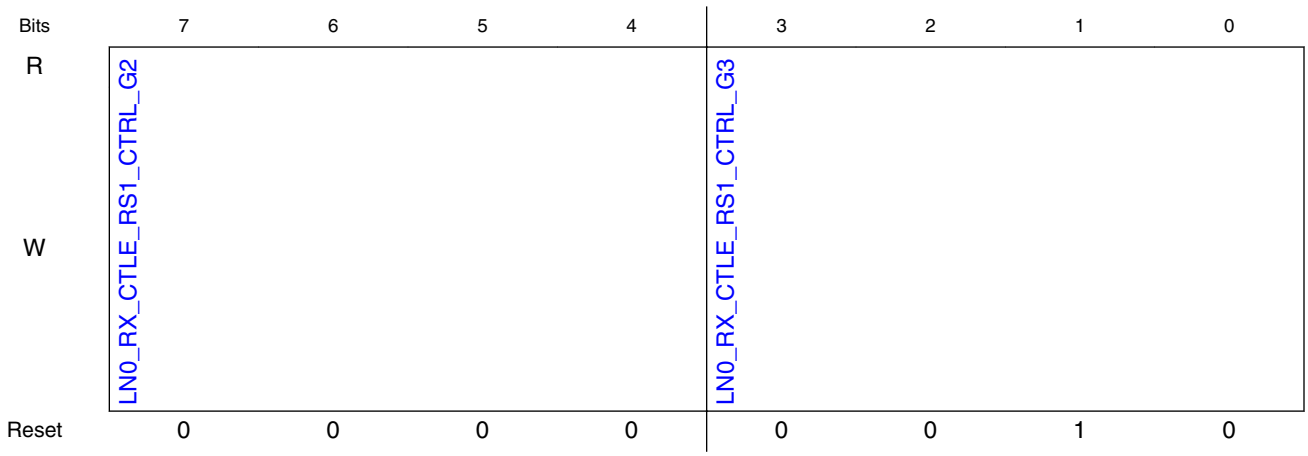
Field	Function
7-5 —	Reserved
4 LN0_OVRD_RX_CTLE_RS1_CTRL	Override enable for rx_ctle_rs1_ctrl_g1
3-0 LN0_RX_CTLE_RS1_CTRL_G1	[GEN1] RX CTLE 1st stage source series resistance control 000: Min resistance, … , 111: Max resistance

11.4.3.1.187 (TRSV_REG03C)

11.4.3.1.187.1 Offset

Register	Offset
TRSV_REG03C	4F0h

11.4.3.1.187.2 Diagram



11.4.3.1.187.3 Fields

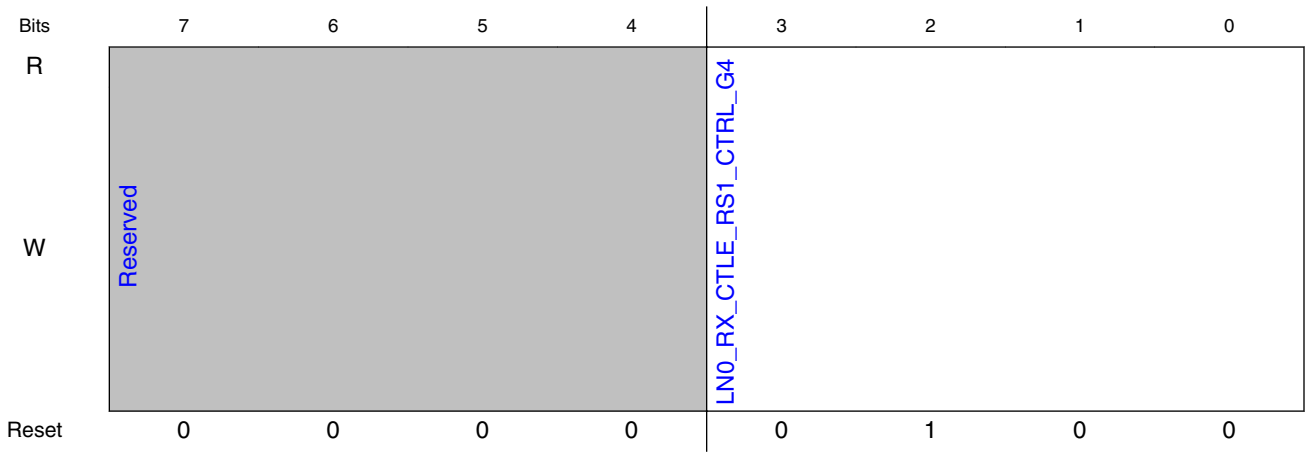
Field	Function
7-4 LN0_RX_CTLE_RS1_CTRL_G2	[GEN2]
3-0 LN0_RX_CTLE_RS1_CTRL_G3	[GEN3]

11.4.3.1.188 (TRSV_REG03D)

11.4.3.1.188.1 Offset

Register	Offset
TRSV_REG03D	4F4h

11.4.3.1.188.2 Diagram



11.4.3.1.188.3 Fields

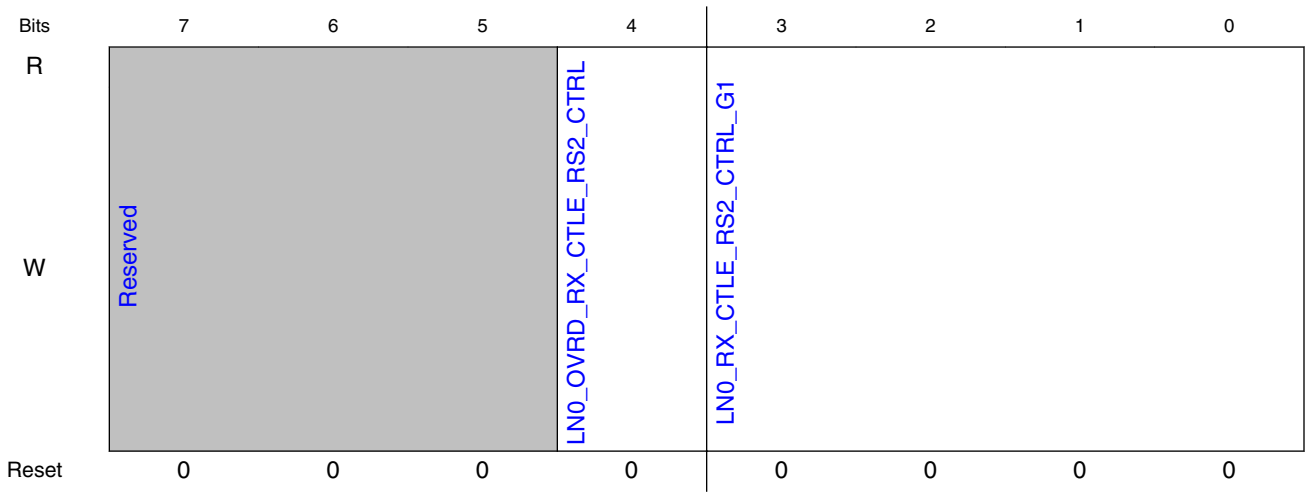
Field	Function
7-4 —	Reserved
3-0 LN0_RX_CTLE_RS1_CTRL_G4	[GEN4]

11.4.3.1.189 (TRSV_REG03E)

11.4.3.1.189.1 Offset

Register	Offset
TRSV_REG03E	4F8h

11.4.3.1.189.2 Diagram



11.4.3.1.189.3 Fields

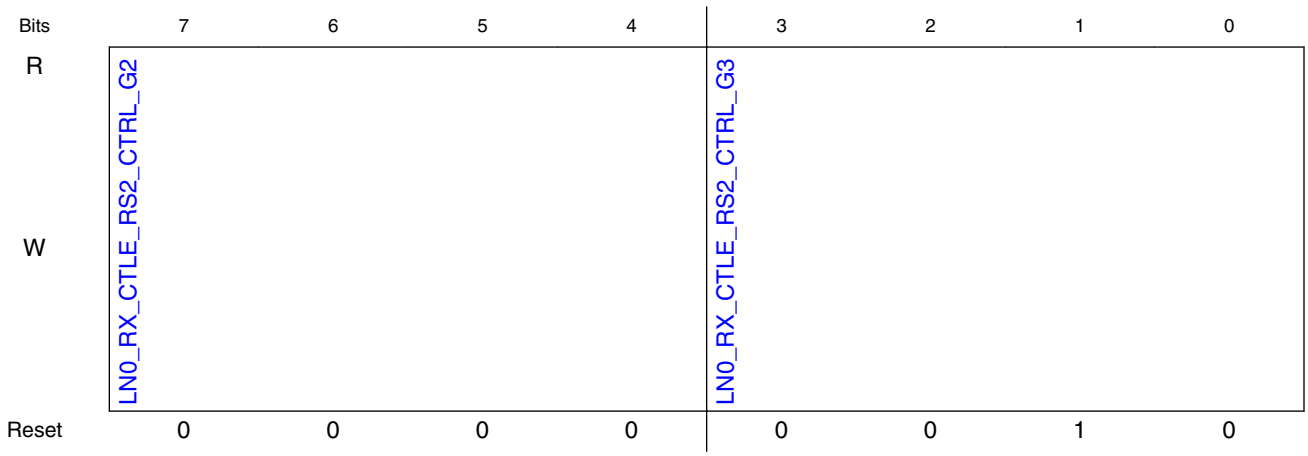
Field	Function
7-5 —	Reserved
4 LN0_OVRD_RX_CTLE_RS2_CTRL	Override enable for rx_ctle_rs2_ctrl_g1
3-0 LN0_RX_CTLE_RS2_CTRL_G1	[GEN1] RX CTLE 2nd stage source series resistance control

11.4.3.1.190 (TRSV_REG03F)

11.4.3.1.190.1 Offset

Register	Offset
TRSV_REG03F	4FCh

11.4.3.1.190.2 Diagram



11.4.3.1.190.3 Fields

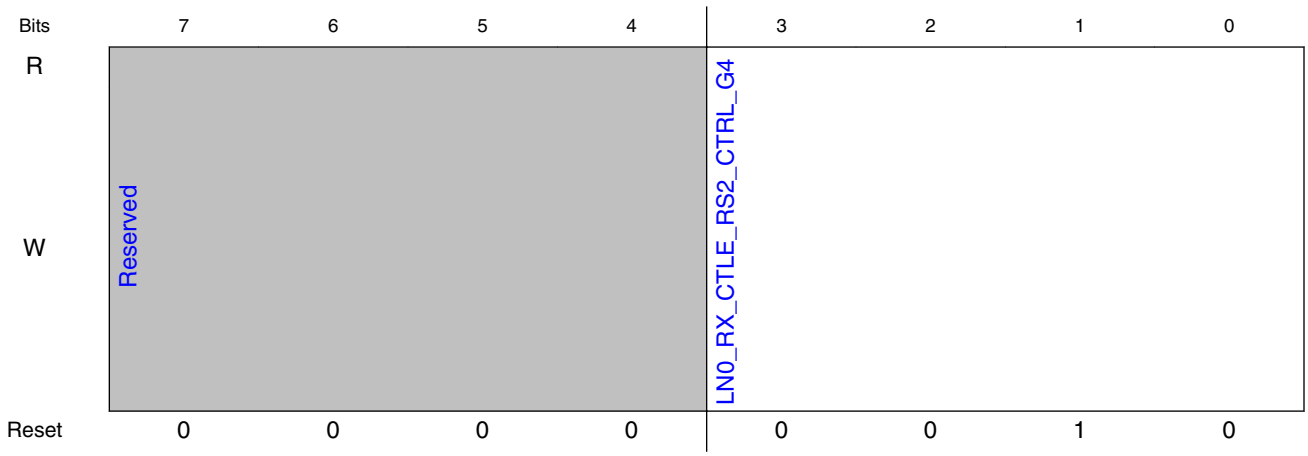
Field	Function
7-4 LN0_RX_CTLE_RS2_CTRL_G2	[GEN2]
3-0 LN0_RX_CTLE_RS2_CTRL_G3	[GEN3]

11.4.3.1.191 (TRSV_REG040)

11.4.3.1.191.1 Offset

Register	Offset
TRSV_REG040	500h

11.4.3.1.191.2 Diagram



11.4.3.1.191.3 Fields

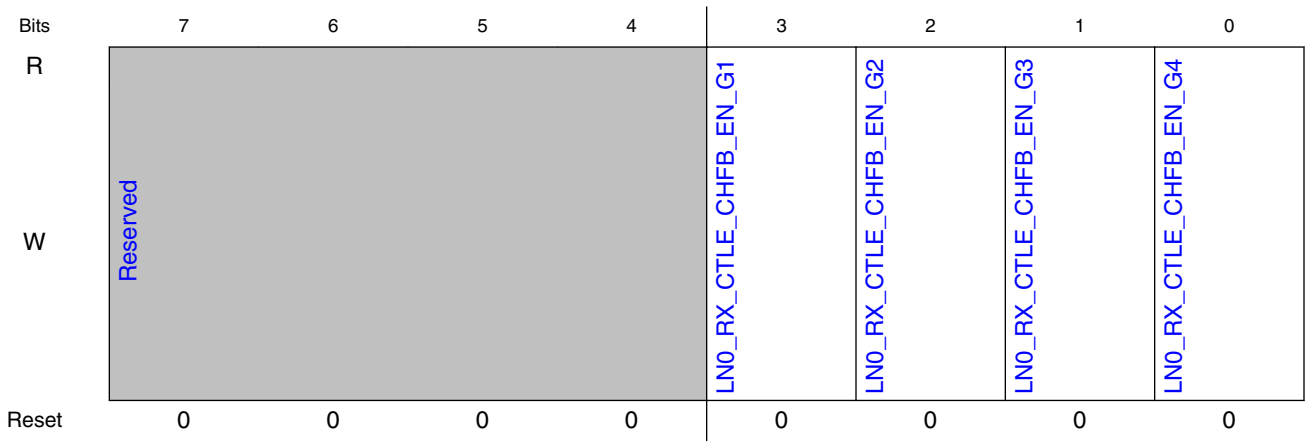
Field	Function
7-4 —	Reserved
3-0 LN0_RX_CTLE_RS2_CTRL_G4	[GEN4]

11.4.3.1.192 (TRSV_REG041)

11.4.3.1.192.1 Offset

Register	Offset
TRSV_REG041	504h

11.4.3.1.192.2 Diagram



11.4.3.1.192.3 Fields

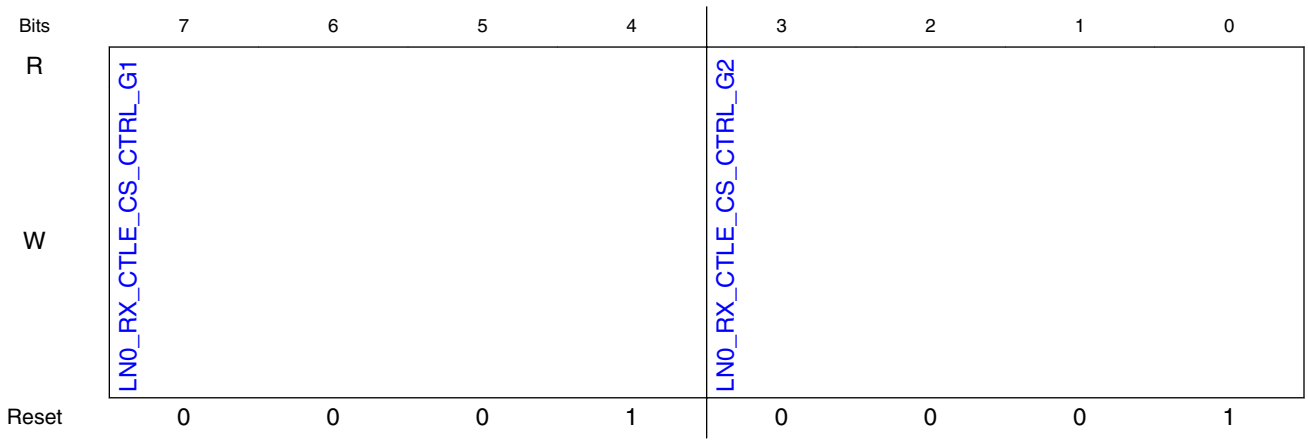
Field	Function
7-4 —	Reserved
3 LN0_RX_CTLE_CHFB_EN_G1	[GEN1] RX CTLE Cherry-Hooper feedback amplifier enable 0: Disable, 1: Enable
2 LN0_RX_CTLE_CHFB_EN_G2	[GEN2]
1 LN0_RX_CTLE_CHFB_EN_G3	[GEN3]
0 LN0_RX_CTLE_CHFB_EN_G4	[GEN4]

11.4.3.1.193 (TRSV_REG042)

11.4.3.1.193.1 Offset

Register	Offset
TRSV_REG042	508h

11.4.3.1.193.2 Diagram



11.4.3.1.193.3 Fields

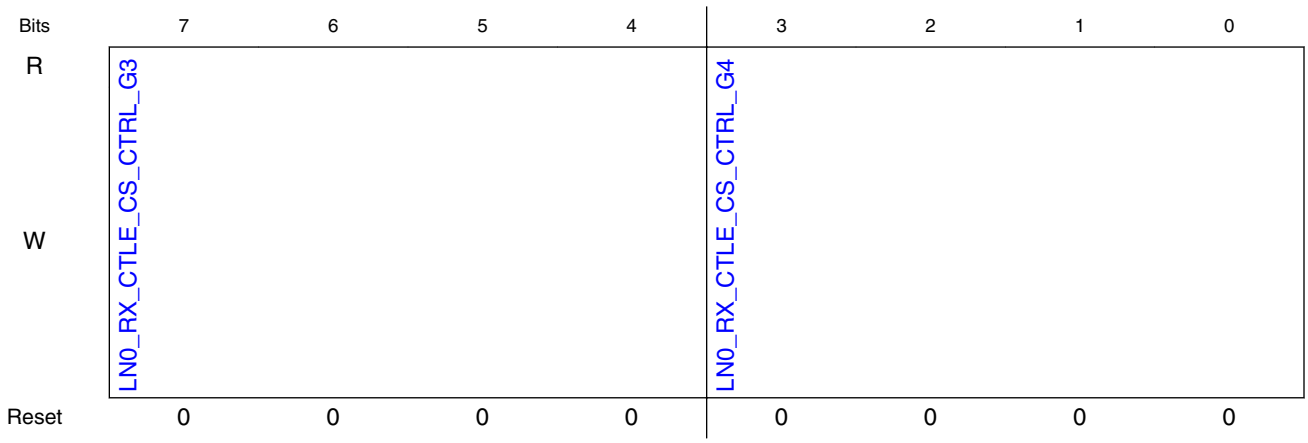
Field	Function
7-4 LN0_RX_CTLE_CS_CTRL_G1	[GEN1] CTLE capacitance control. 4'h0=Gen4,3 4'h3=Gen2 4'h7=Gen1
3-0 LN0_RX_CTLE_CS_CTRL_G2	[GEN2]

11.4.3.1.194 (TRSV_REG043)

11.4.3.1.194.1 Offset

Register	Offset
TRSV_REG043	50Ch

11.4.3.1.194.2 Diagram



11.4.3.1.194.3 Fields

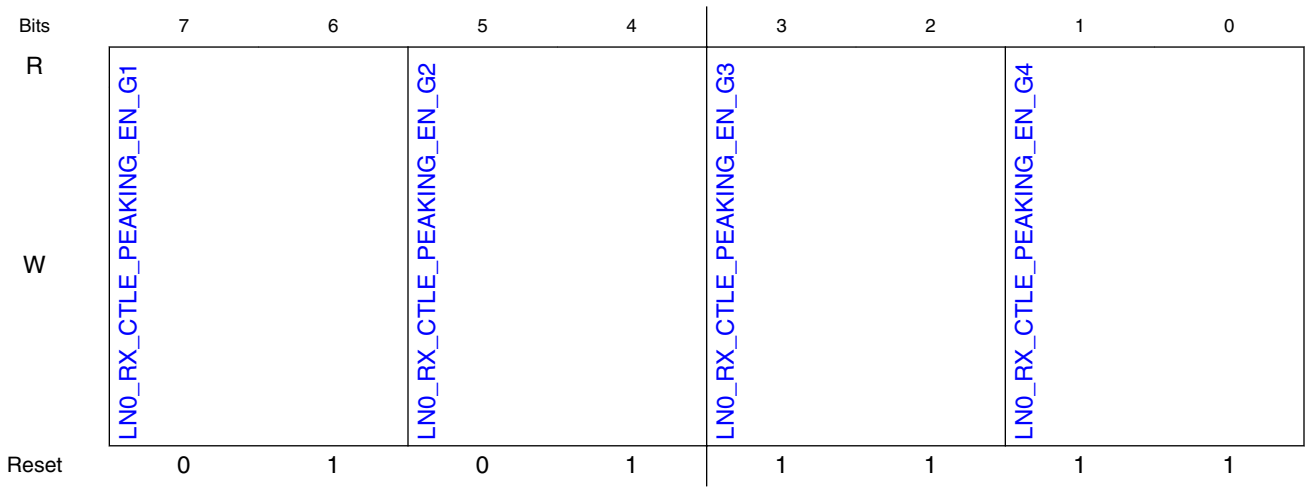
Field	Function
7-4 LN0_RX_CTLE_CS_CTRL_G3	[GEN3]
3-0 LN0_RX_CTLE_CS_CTRL_G4	[GEN4]

11.4.3.1.195 (TRSV_REG044)

11.4.3.1.195.1 Offset

Register	Offset
TRSV_REG044	510h

11.4.3.1.195.2 Diagram



11.4.3.1.195.3 Fields

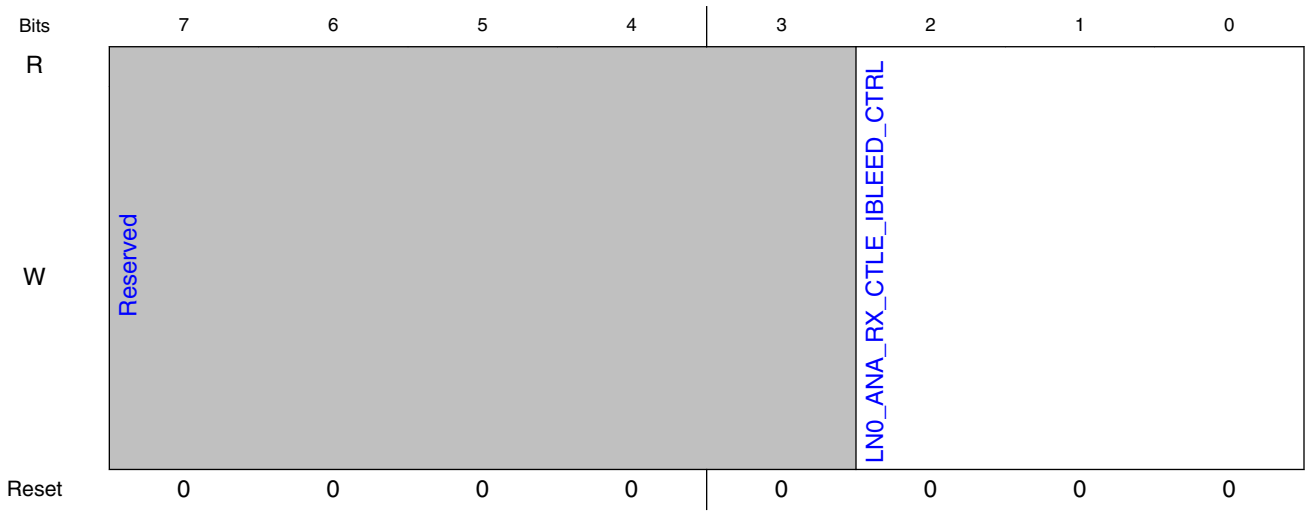
Field	Function
7-6 LN0_RX_CTLE_PEAKING_EN_G1	[GEN1] RX CTLE stage enable for Gen1 [0]: 1st stage, [1]: 2nd stage, [3]=3rd stage 0: Disable, 1: Enable
5-4 LN0_RX_CTLE_PEAKING_EN_G2	[GEN2]
3-2 LN0_RX_CTLE_PEAKING_EN_G3	[GEN3]
1-0 LN0_RX_CTLE_PEAKING_EN_G4	[GEN4]

11.4.3.1.196 (TRSV_REG045)

11.4.3.1.196.1 Offset

Register	Offset
TRSV_REG045	514h

11.4.3.1.196.2 Diagram



11.4.3.1.196.3 Fields

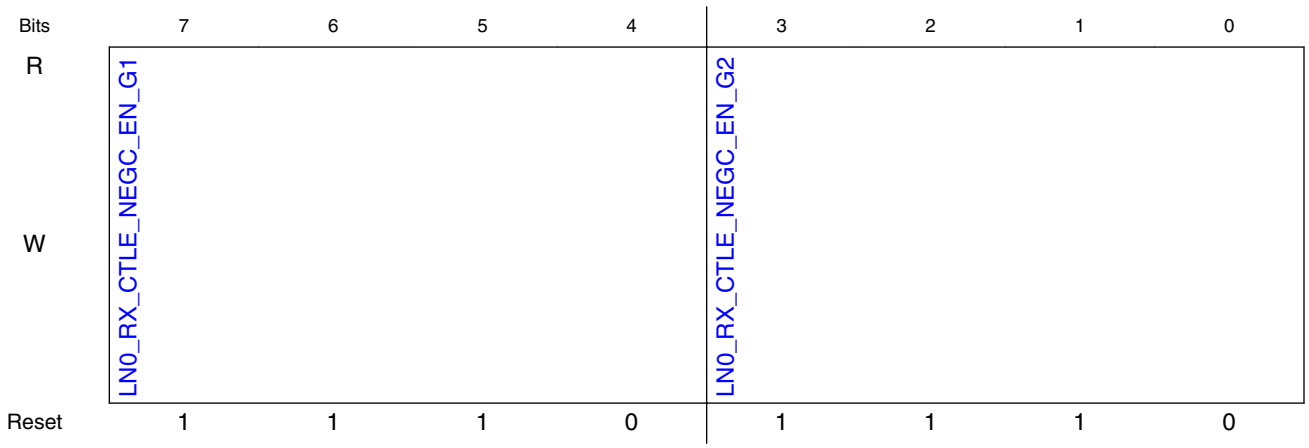
Field	Function
7-3 —	Reserved
2-0 LN0_ANA_RX_CTLE_IBLEED_CTRL	RX CTLE bleeder current control 000: No bleeder current, … , 111: Max bleeder current

11.4.3.1.197 (TRSV_REG046)

11.4.3.1.197.1 Offset

Register	Offset
TRSV_REG046	518h

11.4.3.1.197.2 Diagram



11.4.3.1.197.3 Fields

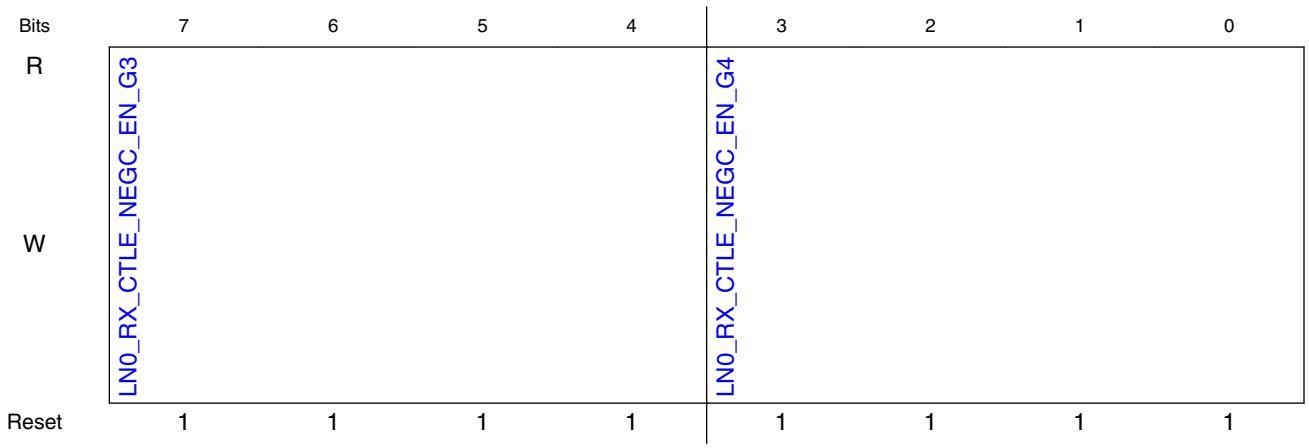
Field	Function
7-4 LN0_RX_CTLE_NEGC_EN_G1	[GEN1] RX CTLE negative-C enable 0: Disable, 1: Enable
3-0 LN0_RX_CTLE_NEGC_EN_G2	[GEN2]

11.4.3.1.198 (TRSV_REG047)

11.4.3.1.198.1 Offset

Register	Offset
TRSV_REG047	51Ch

11.4.3.1.198.2 Diagram



11.4.3.1.198.3 Fields

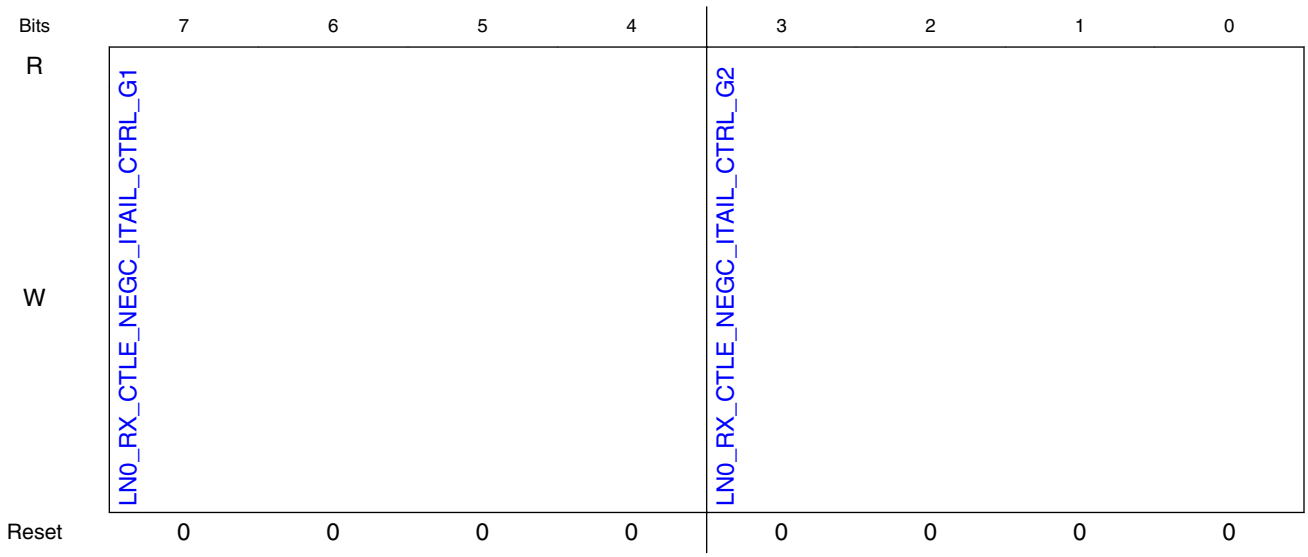
Field	Function
7-4 LN0_RX_CTLE_NEGC_EN_G3	[GEN3]
3-0 LN0_RX_CTLE_NEGC_EN_G4	[GEN4]

11.4.3.1.199 (TRSV_REG048)

11.4.3.1.199.1 Offset

Register	Offset
TRSV_REG048	520h

11.4.3.1.199.2 Diagram



11.4.3.1.199.3 Fields

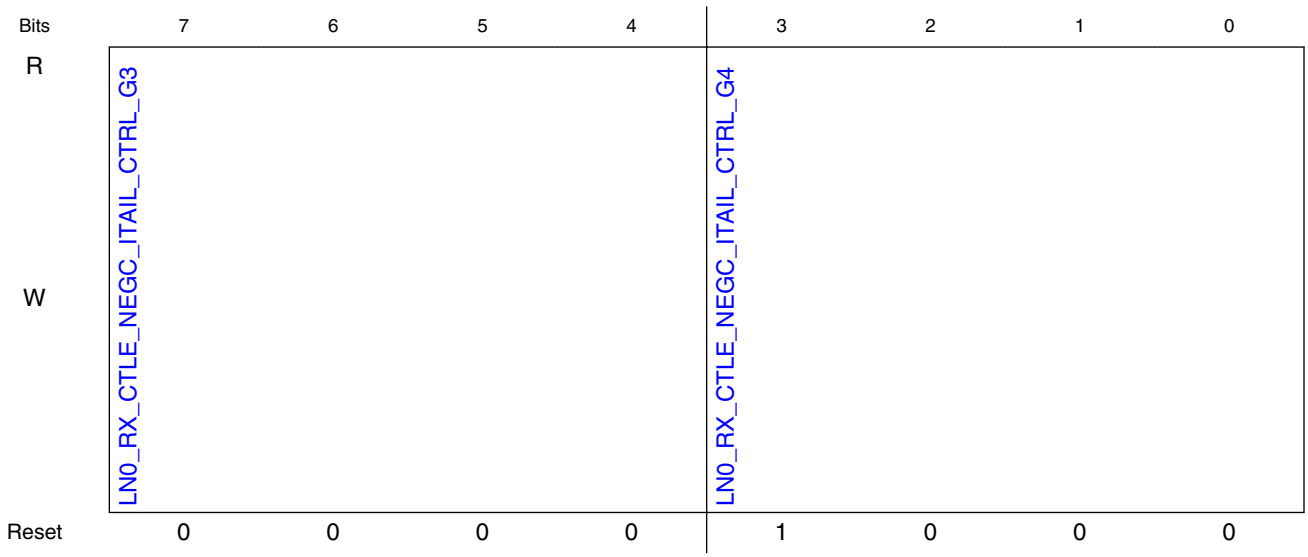
Field	Function
7-4 LN0_RX_CTLE_NEGC_ITAIL_CTRL_G1	[GEN1] RX CTLE negative-C tail current control 0000:, 1111:
3-0 LN0_RX_CTLE_NEGC_ITAIL_CTRL_G2	[GEN2]

11.4.3.1.200 (TRSV_REG049)

11.4.3.1.200.1 Offset

Register	Offset
TRSV_REG049	524h

11.4.3.1.200.2 Diagram



11.4.3.1.200.3 Fields

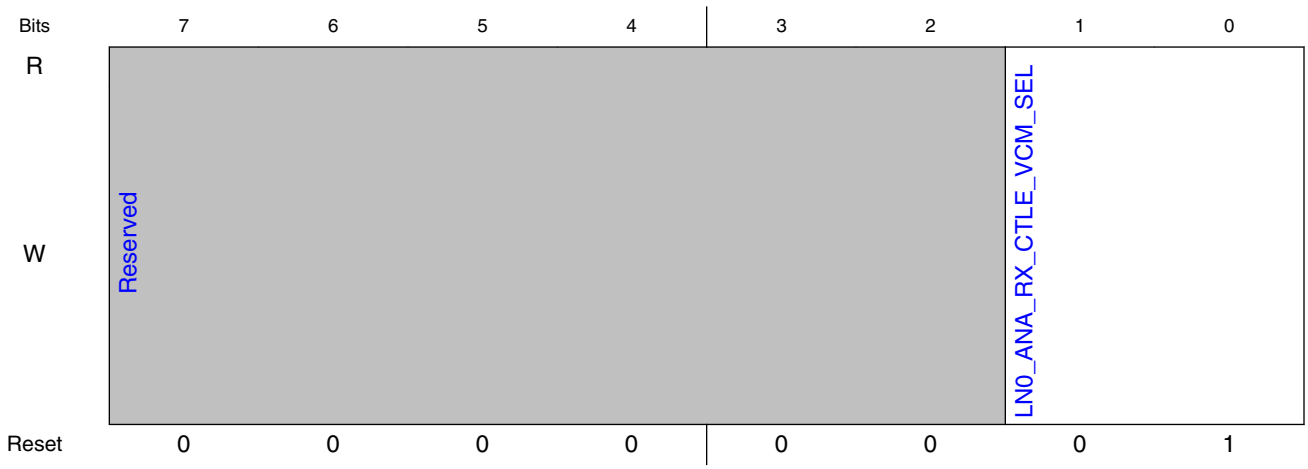
Field	Function
7-4 LN0_RX_CTLE_NEGC_ITAIL_CTRL_G3	[GEN3]
3-0 LN0_RX_CTLE_NEGC_ITAIL_CTRL_G4	[GEN4]

11.4.3.1.201 (TRSV_REG04A)

11.4.3.1.201.1 Offset

Register	Offset
TRSV_REG04A	528h

11.4.3.1.201.2 Diagram



11.4.3.1.201.3 Fields

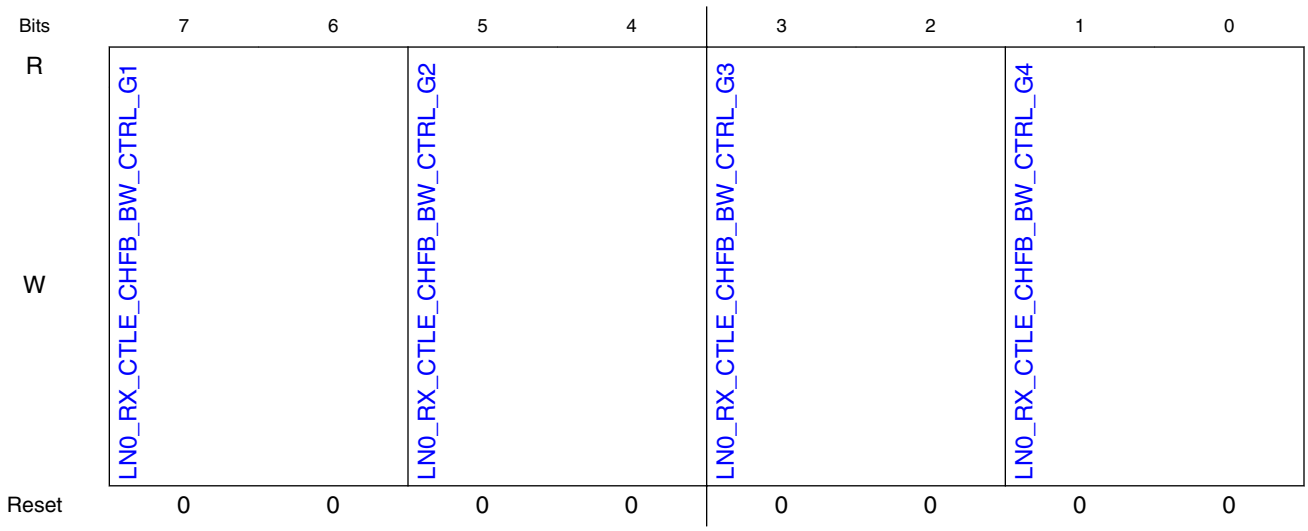
Field	Function
7-2 —	Reserved
1-0 LN0_ANA_RX_CTLE_VCM_SEL	RX AFE (CTLE output) common-mode voltage selection 00=150mV, 01: 180mV, 10: 210mV, 11: 230mV

11.4.3.1.202 (TRSV_REG04B)

11.4.3.1.202.1 Offset

Register	Offset
TRSV_REG04B	52Ch

11.4.3.1.202.2 Diagram



11.4.3.1.202.3 Fields

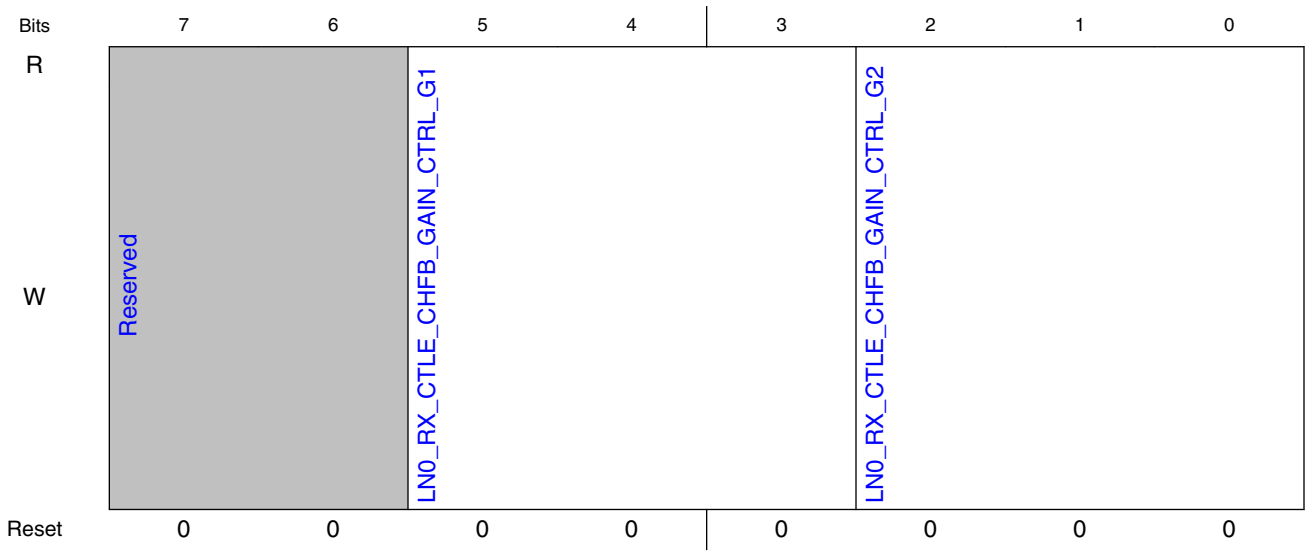
Field	Function
7-6 LN0_RX_CTLE_CHFB_BW_CTRL_G1	[GEN1] RX CTLE Cherry-Hooper feedback amplifier bandwidth control 00: , 01: , 10: , 11
5-4 LN0_RX_CTLE_CHFB_BW_CTRL_G2	[GEN2]
3-2 LN0_RX_CTLE_CHFB_BW_CTRL_G3	[GEN3]
1-0 LN0_RX_CTLE_CHFB_BW_CTRL_G4	[GEN4]

11.4.3.1.203 (TRSV_REG04C)

11.4.3.1.203.1 Offset

Register	Offset
TRSV_REG04C	530h

11.4.3.1.203.2 Diagram



11.4.3.1.203.3 Fields

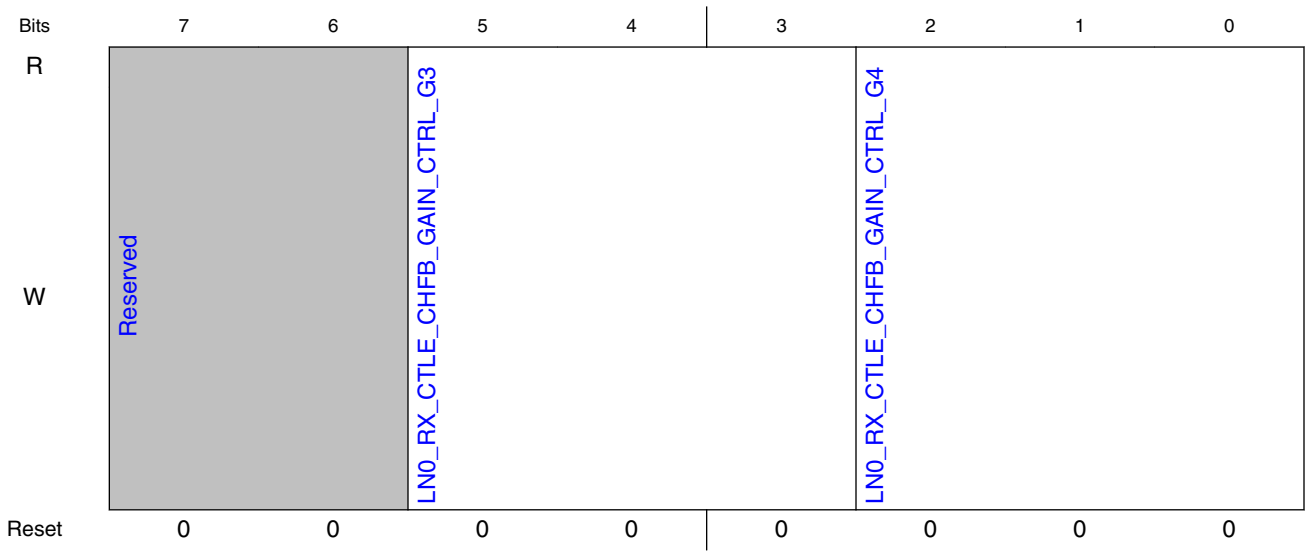
Field	Function
7-6 —	Reserved
5-3 LN0_RX_CTLE_CHFB_GAIN_CTRL_G1	[GEN1] RX CTLE Cherry-Hooper feedback amplifier gain control 000: , 001: , 010: , 011, 100: , 101: , 110: , 111,
2-0 LN0_RX_CTLE_CHFB_GAIN_CTRL_G2	[GEN2]

11.4.3.1.204 (TRSV_REG04D)

11.4.3.1.204.1 Offset

Register	Offset
TRSV_REG04D	534h

11.4.3.1.204.2 Diagram



11.4.3.1.204.3 Fields

Field	Function
7-6 —	Reserved
5-3 LN0_RX_CTLE_CHFB_GAIN_CTRL_G3	[GEN3]
2-0 LN0_RX_CTLE_CHFB_GAIN_CTRL_G4	[GEN4]

11.4.3.1.205 (TRSV_REG04E)

11.4.3.1.205.1 Offset

Register	Offset
TRSV_REG04E	538h

11.4.3.1.205.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_RX_CTLE_ACTIVE_LOAD_CTRL_G1		LN0_RX_CTLE_ACTIVE_LOAD_CTRL_G2		LN0_RX_CTLE_ACTIVE_LOAD_CTRL_G3		LN0_RX_CTLE_ACTIVE_LOAD_CTRL_G4	
W								
Reset	0	0	0	0	0	0	0	0

11.4.3.1.205.3 Fields

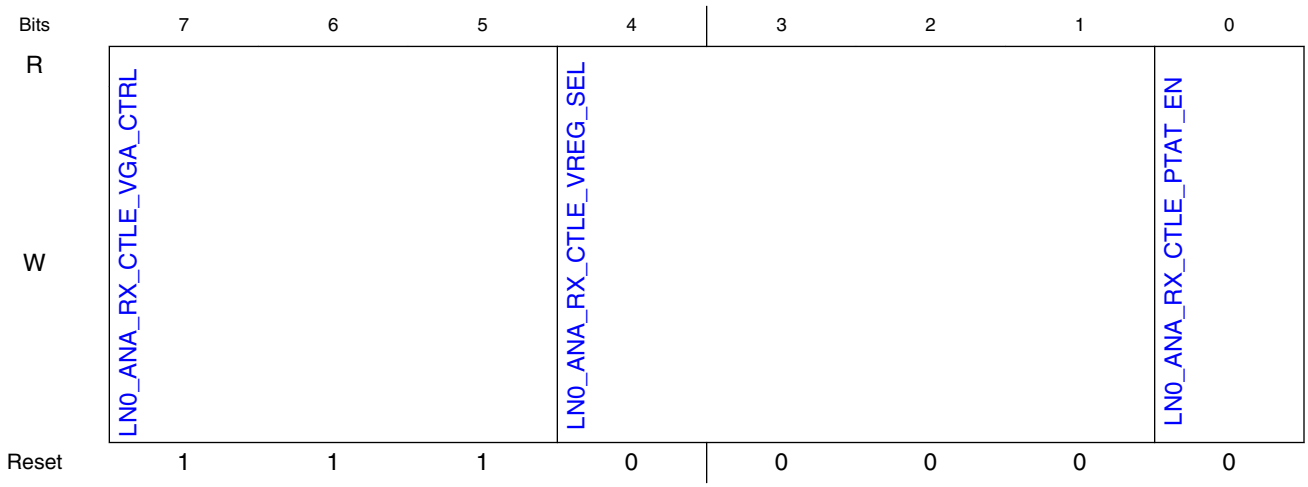
Field	Function
7-6 LN0_RX_CTLE_ACTIVE_LOAD_CTRL_G1	[GEN1] RX CTLE active load control 00: , 01: , 10: , 11
5-4 LN0_RX_CTLE_ACTIVE_LOAD_CTRL_G2	[GEN2]
3-2 LN0_RX_CTLE_ACTIVE_LOAD_CTRL_G3	[GEN3]
1-0 LN0_RX_CTLE_ACTIVE_LOAD_CTRL_G4	[GEN4]

11.4.3.1.206 (TRSV_REG04F)

11.4.3.1.206.1 Offset

Register	Offset
TRSV_REG04F	53Ch

11.4.3.1.206.2 Diagram



11.4.3.1.206.3 Fields

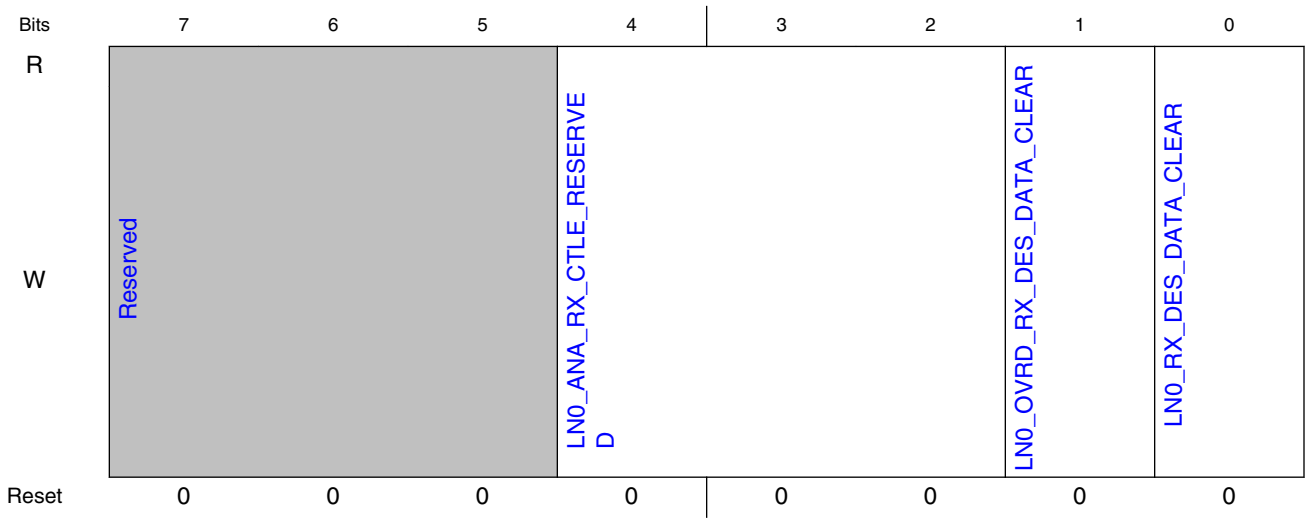
Field	Function
7-5 LN0_ANA_RX_CTLE_VGA_CTRL	RX CTLE stage3 gain control 000:Max. gain , 111: Max. gain
4-1 LN0_ANA_RX_CTLE_VREG_SEL	RX CTLE voltage regulator output voltage 0000: 820mV, 1111: Max voltage
0 LN0_ANA_RX_CTLE_PTAT_EN	RX CTLE PTAT current enable 0: Disable, 1: Enable

11.4.3.1.207 (TRSV_REG050)

11.4.3.1.207.1 Offset

Register	Offset
TRSV_REG050	540h

11.4.3.1.207.2 Diagram



11.4.3.1.207.3 Fields

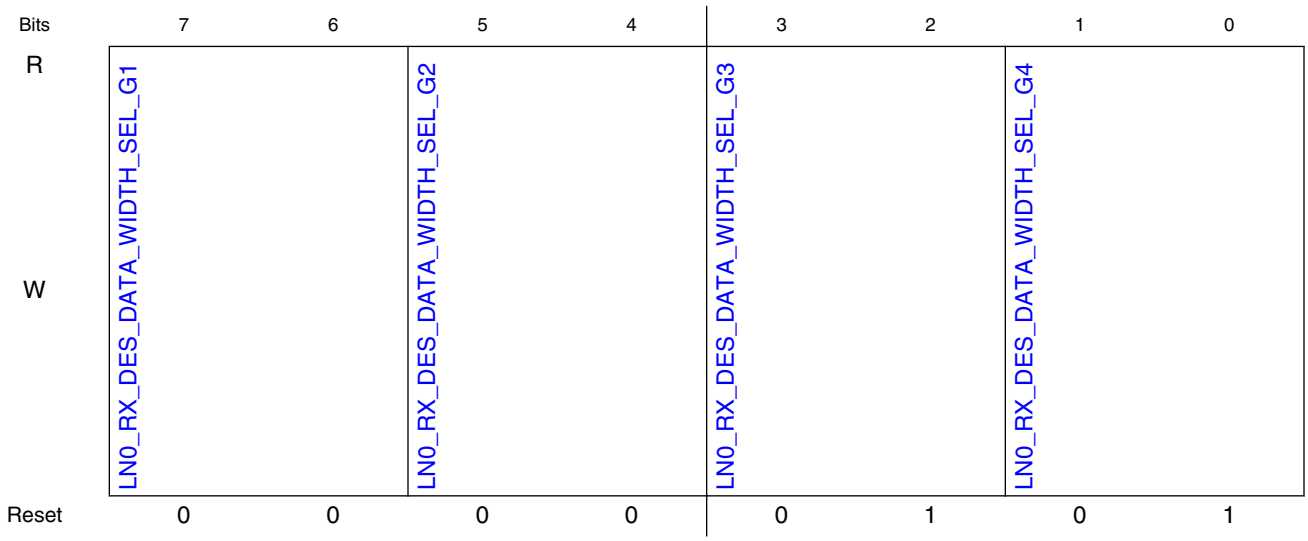
Field	Function
7-5 —	Reserved
4-2 LN0_ANA_RX_CTLE_RESERVED	Reserved port
1 LN0_OVRD_RX_DES_DATA_CLEAR	Override enable for rx_des_data_clear
0 LN0_RX_DES_DATA_CLEAR	RX deserializer data clear to prevent garbage data 0: Normal operation, 1: Clear (all 0)

11.4.3.1.208 (TRSV_REG051)

11.4.3.1.208.1 Offset

Register	Offset
TRSV_REG051	544h

11.4.3.1.208.2 Diagram



11.4.3.1.208.3 Fields

Field	Function
7-6 LN0_RX_DES_DATA_WIDTH_SEL_G1	[GEN1] RX deserializer data width selection 00: 2:40 mode, 01: 2:32 mode, 10: 2:20 mode, Others: N/A
5-4 LN0_RX_DES_DATA_WIDTH_SEL_G2	[GEN2]
3-2 LN0_RX_DES_DATA_WIDTH_SEL_G3	[GEN3]
1-0 LN0_RX_DES_DATA_WIDTH_SEL_G4	[GEN4]

11.4.3.1.209 (TRSV_REG052)

11.4.3.1.209.1 Offset

Register	Offset
TRSV_REG052	548h

11.4.3.1.209.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_OVRD_RX_DES_EN	LN0_RX_DES_EN	LN0_OVRD_RX_DES_NON_DATA_SEL	LN0_RX_DES_NON_DATA_SEL	LN0_OVRD_RX_DES_RSTN	LN0_RX_DES_RSTN	LN0_ANA_RX_DES_DATA_CLEAR_DELAY_SEL	
W								
Reset	0	0	0	0	0	0	0	0

11.4.3.1.209.3 Fields

Field	Function
7 LN0_OVRD_RX_DES_EN	Override enable for rx_des_en
6 LN0_RX_DES_EN	RX deserializer enable 0: Disable, 1: Enable
5 LN0_OVRD_RX_DES_NON_DATA_SEL	Override enable for rx_des_non_data_sel
4	RX deserializer non-data selection for edge/error sampler calibration

Table continues on the next page...

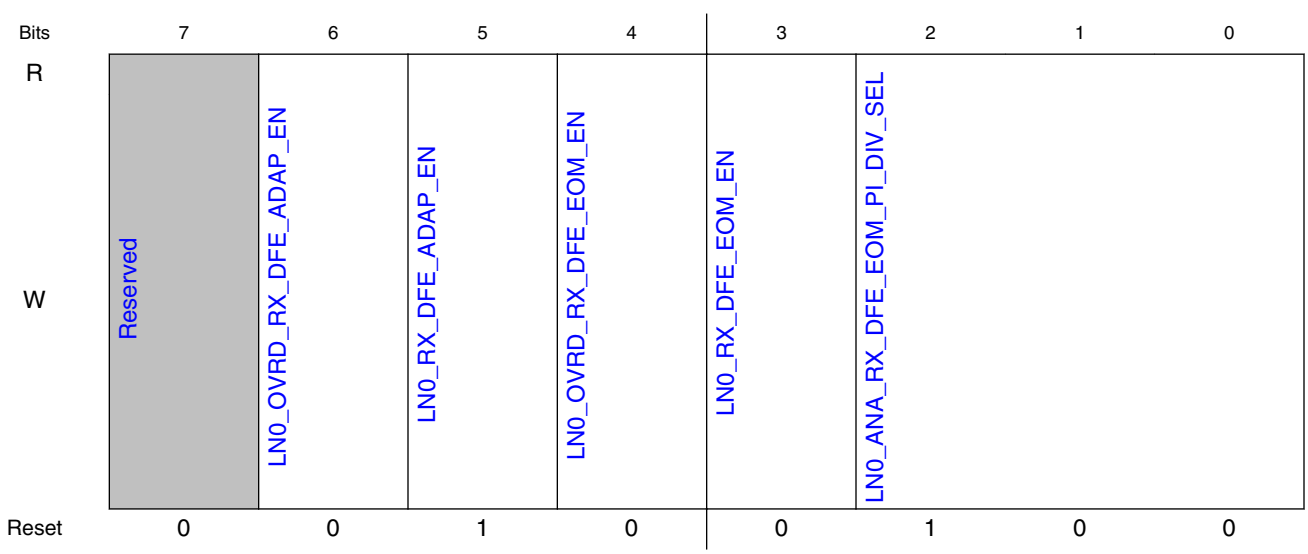
Field	Function
LN0_RX_DES_NON_DATA_SEL	0: Error, 1: Edge
3 LN0_OVRD_RX_DES_RSTN	Override enable for rx_des_rstn
2 LN0_RX_DES_RSTN	RX deserializer reset 0: Reset, 1: Released
1-0 LN0_ANA_RX_DES_DATA_CLEAR_DELAY_SEL	

11.4.3.1.210 (TRSV_REG053)

11.4.3.1.210.1 Offset

Register	Offset
TRSV_REG053	54Ch

11.4.3.1.210.2 Diagram



11.4.3.1.210.3 Fields

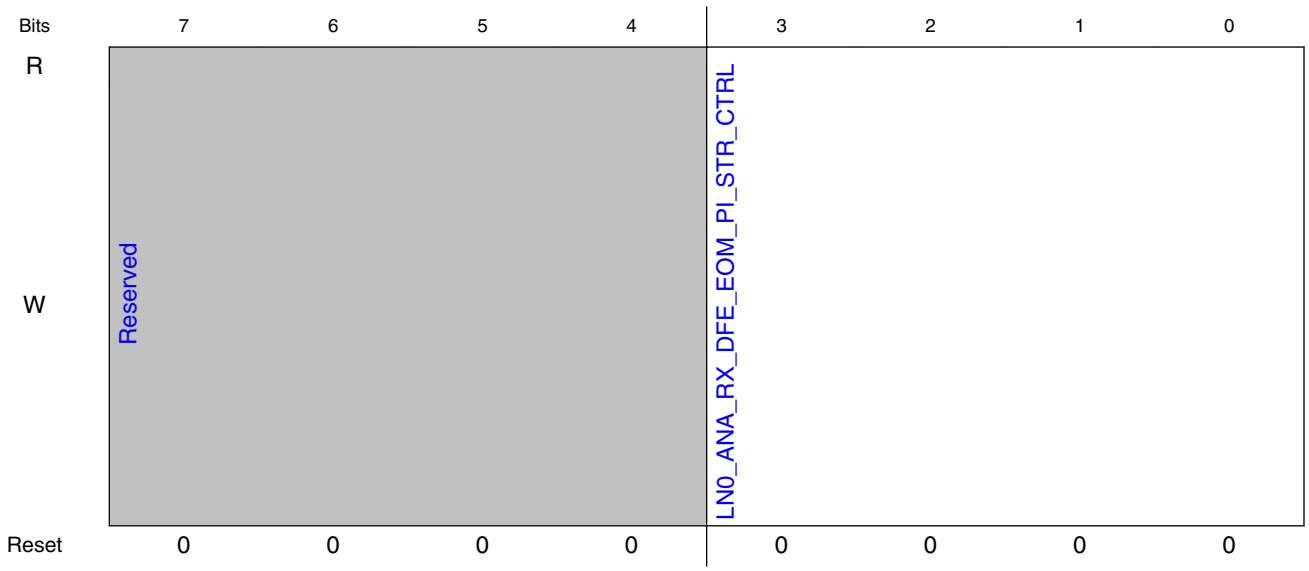
Field	Function
7 —	Reserved
6 LN0_OVRD_RX_DFE_ADAP_EN	Override enable for rx_dfe_adap_en
5 LN0_RX_DFE_ADAP_EN	RX DFE adaptation path enable. Only one of i_rx_dfe_adap_en and i_rx_eom_en should be "1". 0: Disable, 1: Enable
4 LN0_OVRD_RX_DFE_EOM_EN	Override enable for rx_dfe_eom_en
3 LN0_RX_DFE_EOM_EN	RX EOM enable 0: Disable, 1: Enable
2-0 LN0_ANA_RX_DFE_EOM_PI_DIV_SEL	Clock divider control before RX EOM phase interpolator 000 : div8, 001 : div4, 01x : div2, 1xx: div1

11.4.3.1.211 (TRSV_REG054)

11.4.3.1.211.1 Offset

Register	Offset
TRSV_REG054	550h

11.4.3.1.211.2 Diagram



11.4.3.1.211.3 Fields

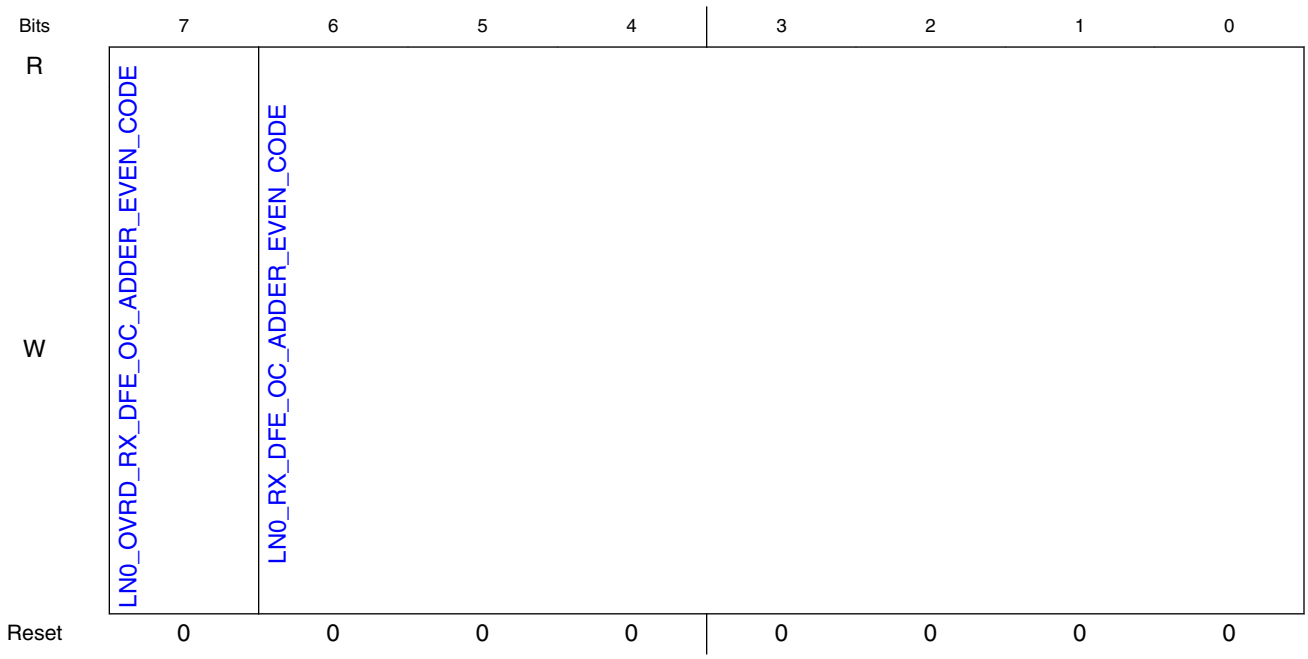
Field	Function
7-4 —	Reserved
3-0 LN0_ANA_RX_DFE_EOM_PI_STR_CTRL	RX EOM PI drive strengh in pre-buffer stage 0000: slower slope … 1111: steeper slope

11.4.3.1.212 (TRSV_REG055)

11.4.3.1.212.1 Offset

Register	Offset
TRSV_REG055	554h

11.4.3.1.212.2 Diagram



11.4.3.1.212.3 Fields

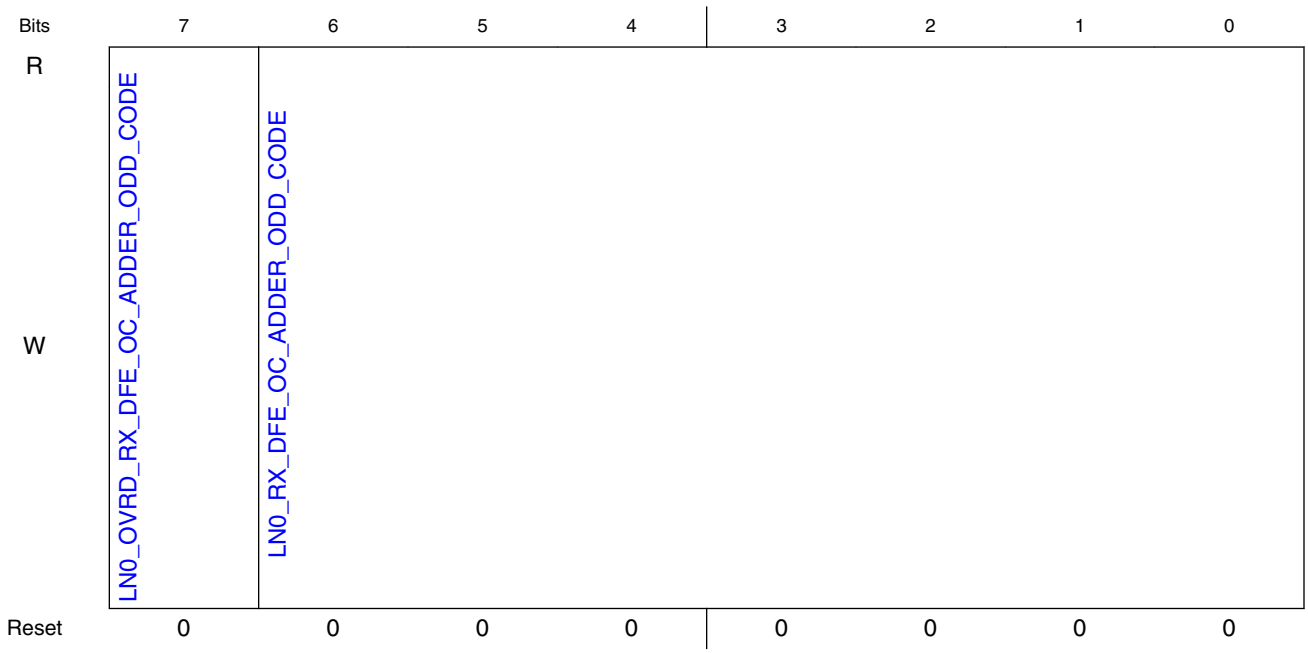
Field	Function
7 LN0_OVRD_RX_DFE_OC_ADDER_EVEN_CODE	Override enable for rx_dfe_oc_adder_even_code
6-0 LN0_RX_DFE_OC_ADDER_EVEN_CODE	RX DFE even data path offset calibration code

11.4.3.1.213 (TRSV_REG056)

11.4.3.1.213.1 Offset

Register	Offset
TRSV_REG056	558h

11.4.3.1.213.2 Diagram



11.4.3.1.213.3 Fields

Field	Function
7 LN0_OVRD_RX_DFE_OC_ADDER_ODD_CODE	Override enable for rx_dfe_oc_adder_odd_code
6-0 LN0_RX_DFE_OC_ADDER_ODD_CODE	RX DFE odd data path offset calibration code

11.4.3.1.214 (TRSV_REG057)

11.4.3.1.214.1 Offset

Register	Offset
TRSV_REG057	55Ch

11.4.3.1.214.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_OVRD_RX_DFE_OC_DAC_ADDER_EVEN_CODE	LN0_RX_DFE_OC_DAC_ADDER_EVEN_CODE	LN0_OVRD_RX_DFE_OC_DAC_ADDER_ODD_CODE	LN0_RX_DFE_OC_DAC_ADDER_ODD_CODE	LN0_OVRD_RX_DFE_OC_DAC_EDGE_EVEN_CODE	LN0_RX_DFE_OC_DAC_EDGE_EVEN_CODE		
W								
Reset	0	0	0	0	0	0	0	0

11.4.3.1.214.3 Fields

Field	Function
7 LN0_OVRD_RX_DFE_OC_DAC_ADDER_EVEN_CODE	Override enable for rx_dfe_oc_dac_adder_even_code
6 LN0_RX_DFE_OC_DAC_ADDER_EVEN_CODE	Fine control of zero-crossing in RX DFE DAC for even adder offset calibration
5 LN0_OVRD_RX_DFE_OC_DAC_ADDER_ODD_CODE	Override enable for rx_dfe_oc_dac_adder_odd_code
4 LN0_RX_DFE_OC_DAC_ADDER_ODD_CODE	Fine control of zero-crossing in RX DFE DAC for odd adder offset calibration
3	Override enable for rx_dfe_oc_dac_edge_even_code

Table continues on the next page...

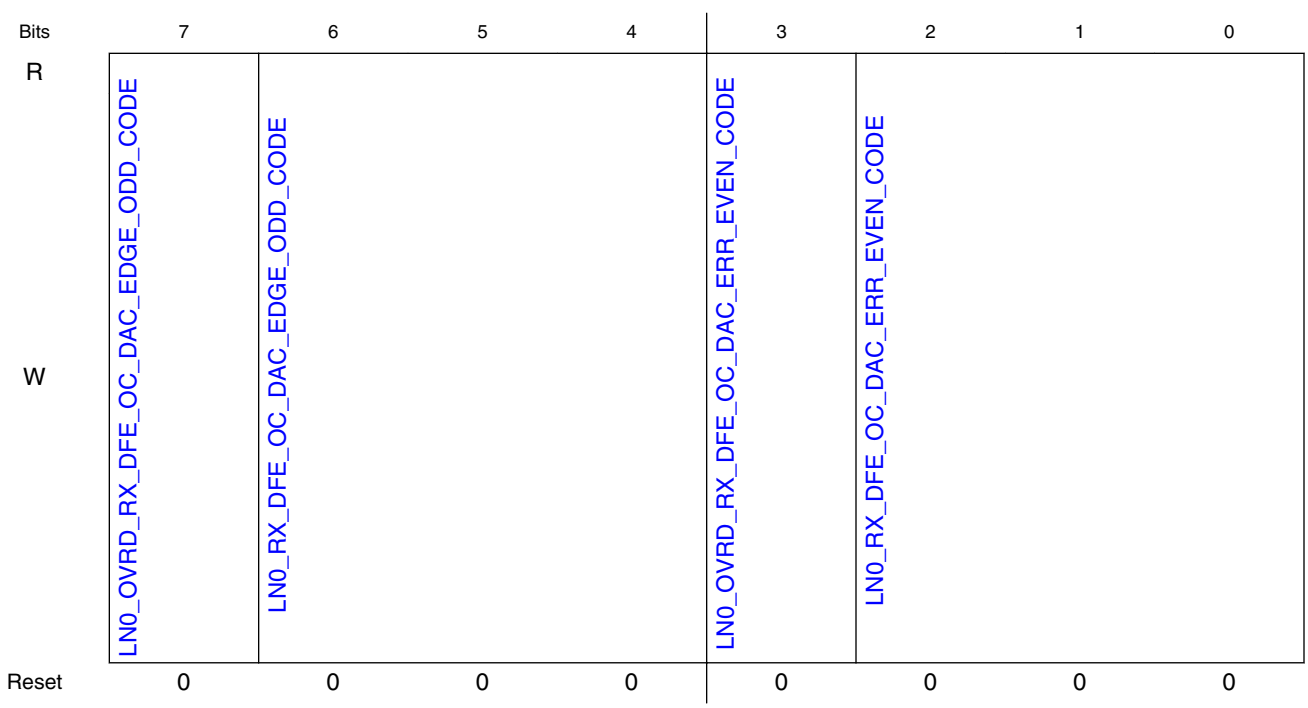
Field	Function
LN0_OVRD_RX_DFE_OC_DAC_EDGE_EVEN_CODE	
2-0 LN0_RX_DFE_OC_DAC_EDGE_EVEN_CODE	Fine control of zero-crossing in RX DFE DAC for even edge path offset calibration

11.4.3.1.215 (TRSV_REG058)

11.4.3.1.215.1 Offset

Register	Offset
TRSV_REG058	560h

11.4.3.1.215.2 Diagram



11.4.3.1.215.3 Fields

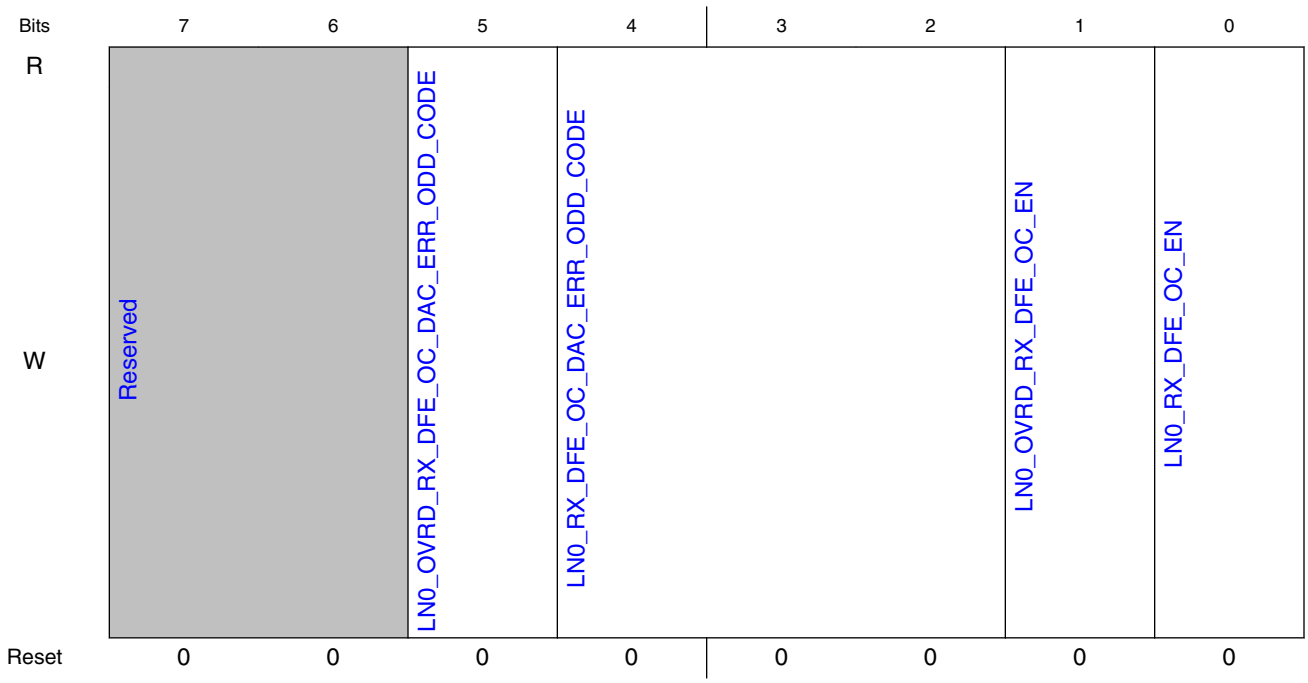
Field	Function
7 LN0_OVRD_RX_DFE_OC_DAC_EDGE_ODD_CODE	Override enable for rx_dfe_oc_dac_edge_odd_code
6-4 LN0_RX_DFE_OC_DAC_EDGE_ODD_CODE	Fine control of zero-crossing in RX DFE DAC for odd edge path offset calibration
3 LN0_OVRD_RX_DFE_OC_DAC_ERR_EVEN_CODE	Override enable for rx_dfe_oc_dac_err_even_code
2-0 LN0_RX_DFE_OC_DAC_ERR_EVEN_CODE	Fine control of zero-crossing in RX DFE DAC for even error path offset calibration

11.4.3.1.216 (TRSV_REG059)

11.4.3.1.216.1 Offset

Register	Offset
TRSV_REG059	564h

11.4.3.1.216.2 Diagram



11.4.3.1.216.3 Fields

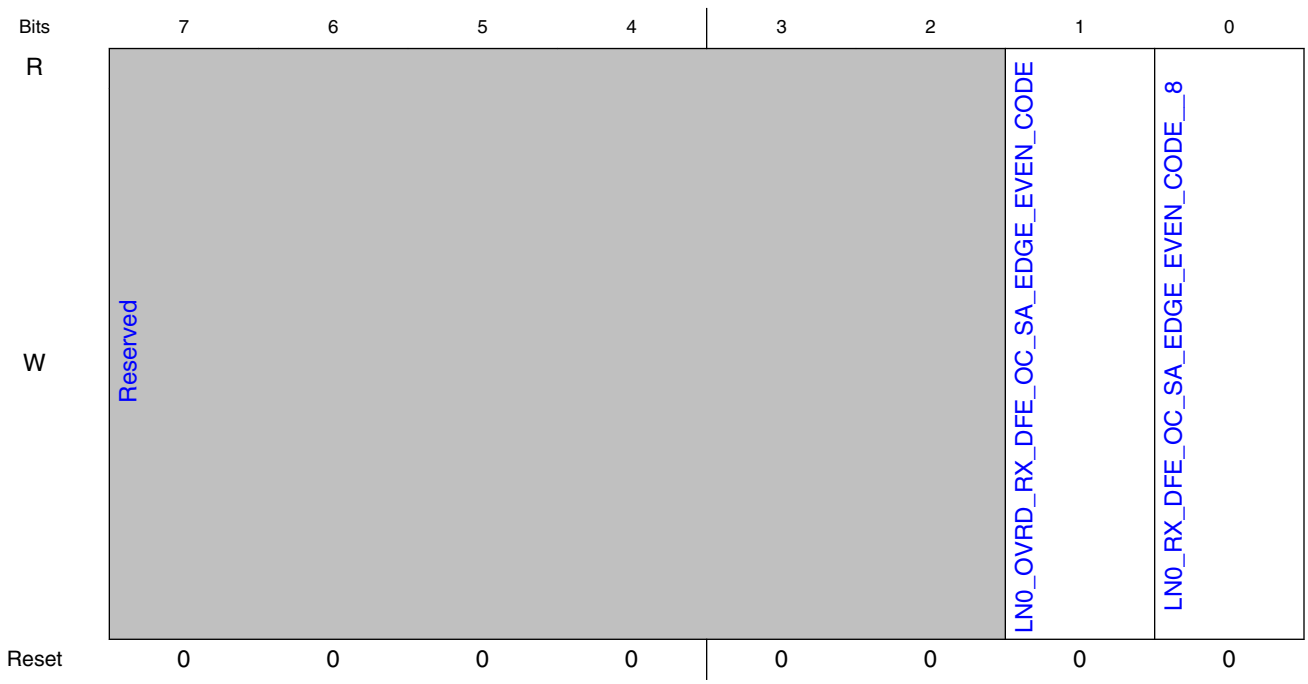
Field	Function
7-6 —	Reserved
5 LN0_OVRD_RX_DFE_OC_DAC_ERR_ODD_CODE	Override enable for rx_dfe_oc_dac_err_odd_code
4-2 LN0_RX_DFE_OC_DAC_ERR_ODD_CODE	Fine control of zero-crossing in RX DFE DAC for odd error path offset calibration
1 LN0_OVRD_RX_DFE_OC_EN	Override enable for rx_dfe_oc_en
0 LN0_RX_DFE_OC_EN	RX DFE offset calibration progress enable 0: Disable (Normal operation of CTLE), 1: Enable (CTLE output forced to VCM)

11.4.3.1.217 (TRSV_REG05A)

11.4.3.1.217.1 Offset

Register	Offset
TRSV_REG05A	568h

11.4.3.1.217.2 Diagram



11.4.3.1.217.3 Fields

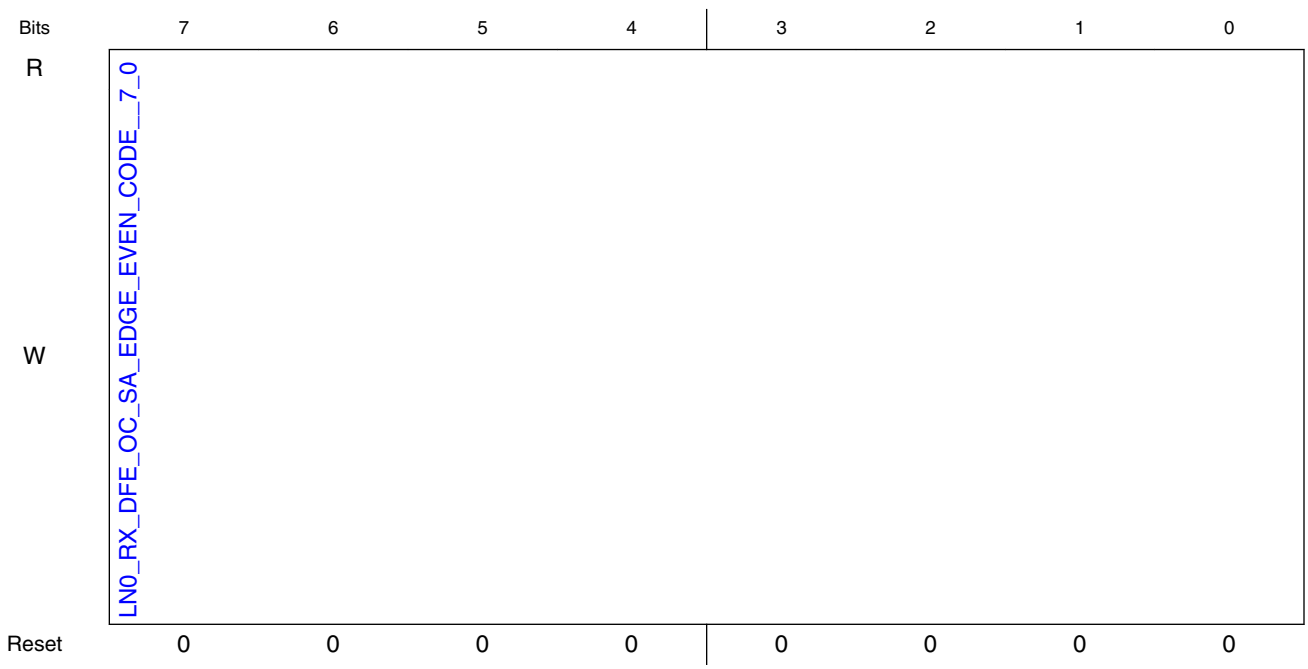
Field	Function
7-2 —	Reserved
1 LN0_OVRD_RX_DFE_OC_SA_EDGE_EVEN_CODE	Override enable for rx_dfe_oc_sa_edge_even_code
0 LN0_RX_DFE_OC_SA_EDGE_EVEN_CODE__8	RX DFE even edge path offset calibration code

11.4.3.1.218 (TRSV_REG05B)

11.4.3.1.218.1 Offset

Register	Offset
TRSV_REG05B	56Ch

11.4.3.1.218.2 Diagram



11.4.3.1.218.3 Fields

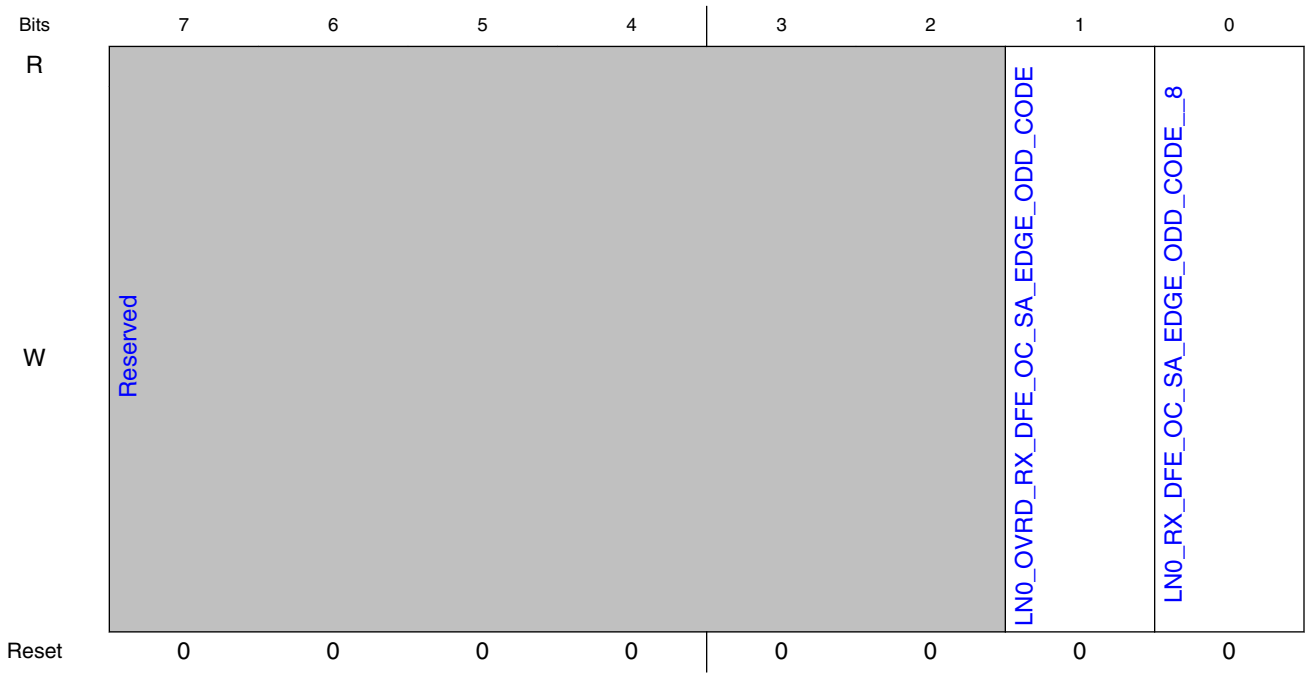
Field	Function
7-0 LN0_RX_DFE_OC_SA_EDGE_EVEN_CODE__7_0	RX DFE even edge path offset calibration code

11.4.3.1.219 (TRSV_REG05C)

11.4.3.1.219.1 Offset

Register	Offset
TRSV_REG05C	570h

11.4.3.1.219.2 Diagram



11.4.3.1.219.3 Fields

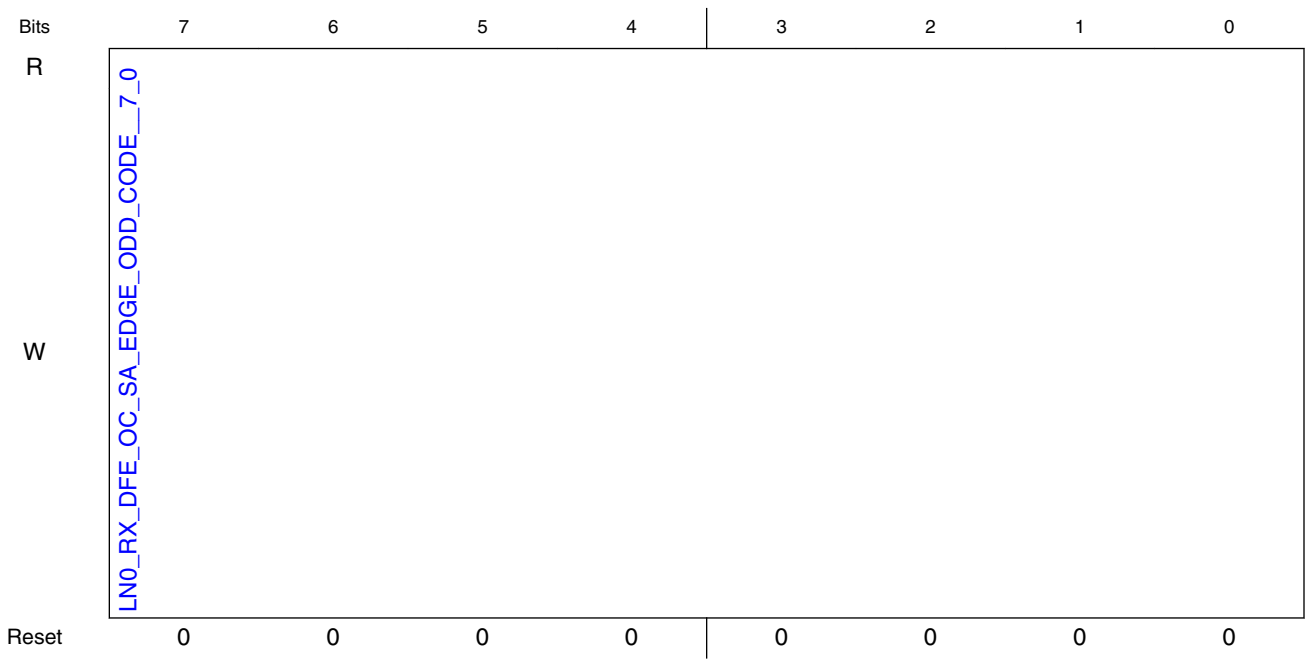
Field	Function
7-2 —	Reserved
1 LN0_OVRD_RX_DFE_OC_SA_EDGE_ODD_CODE	Override enable for rx_dfe_oc_sa_edge_odd_code
0 LN0_RX_DFE_OC_SA_EDGE_ODD_CODE__8	RX DFE odd edge path offset calibration code

11.4.3.1.220 (TRSV_REG05D)

11.4.3.1.220.1 Offset

Register	Offset
TRSV_REG05D	574h

11.4.3.1.220.2 Diagram



11.4.3.1.220.3 Fields

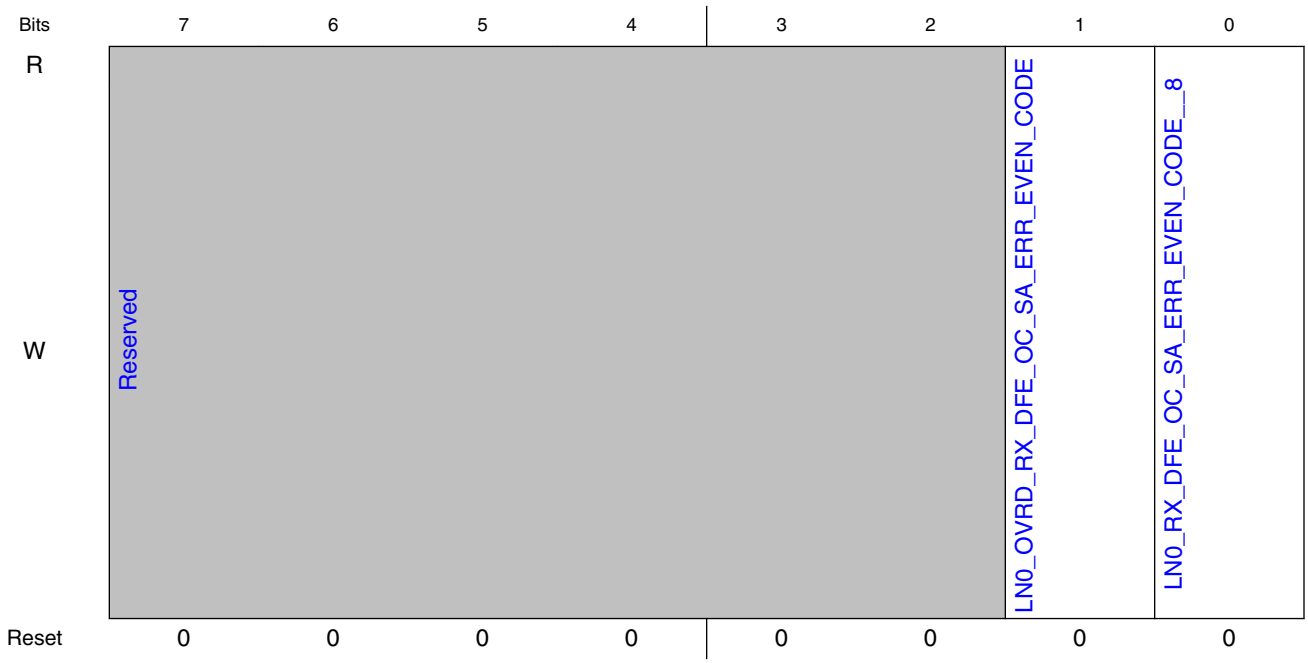
Field	Function
7-0 LN0_RX_DFE_OC_SA_EDGE_ODD_CODE__7_0	RX DFE odd edge path offset calibration code

11.4.3.1.221 (TRSV_REG05E)

11.4.3.1.221.1 Offset

Register	Offset
TRSV_REG05E	578h

11.4.3.1.221.2 Diagram



11.4.3.1.221.3 Fields

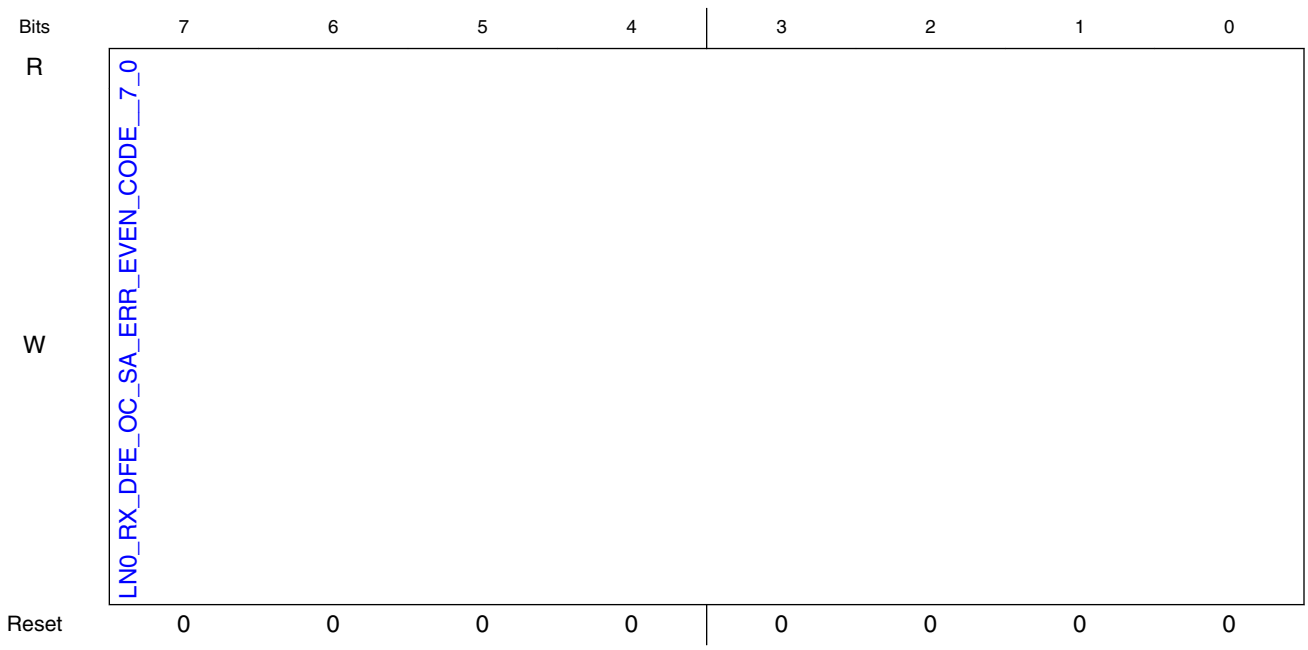
Field	Function
7-2 —	Reserved
1 LN0_OVRD_RX_DFE_OC_SA_ERR_EVEN_CODE	Override enable for rx_dfe_oc_sa_err_even_code
0 LN0_RX_DFE_OC_SA_ERR_EVEN_CODE__8	RX DFE even error path offset calibration code

11.4.3.1.222 (TRSV_REG05F)

11.4.3.1.222.1 Offset

Register	Offset
TRSV_REG05F	57Ch

11.4.3.1.222.2 Diagram



11.4.3.1.222.3 Fields

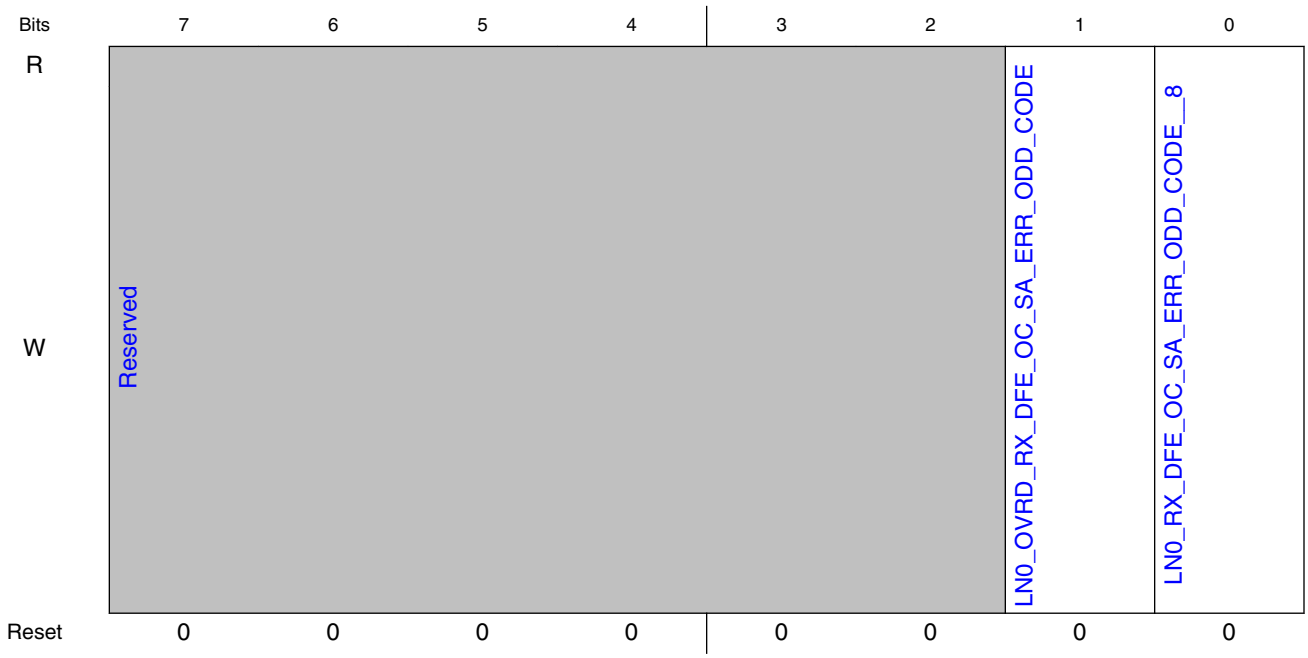
Field	Function
7-0 LN0_RX_DFE_OC_SA_ERR_EVEN_CODE__7_0	RX DFE even error path offset calibration code

11.4.3.1.223 (TRSV_REG060)

11.4.3.1.223.1 Offset

Register	Offset
TRSV_REG060	580h

11.4.3.1.223.2 Diagram



11.4.3.1.223.3 Fields

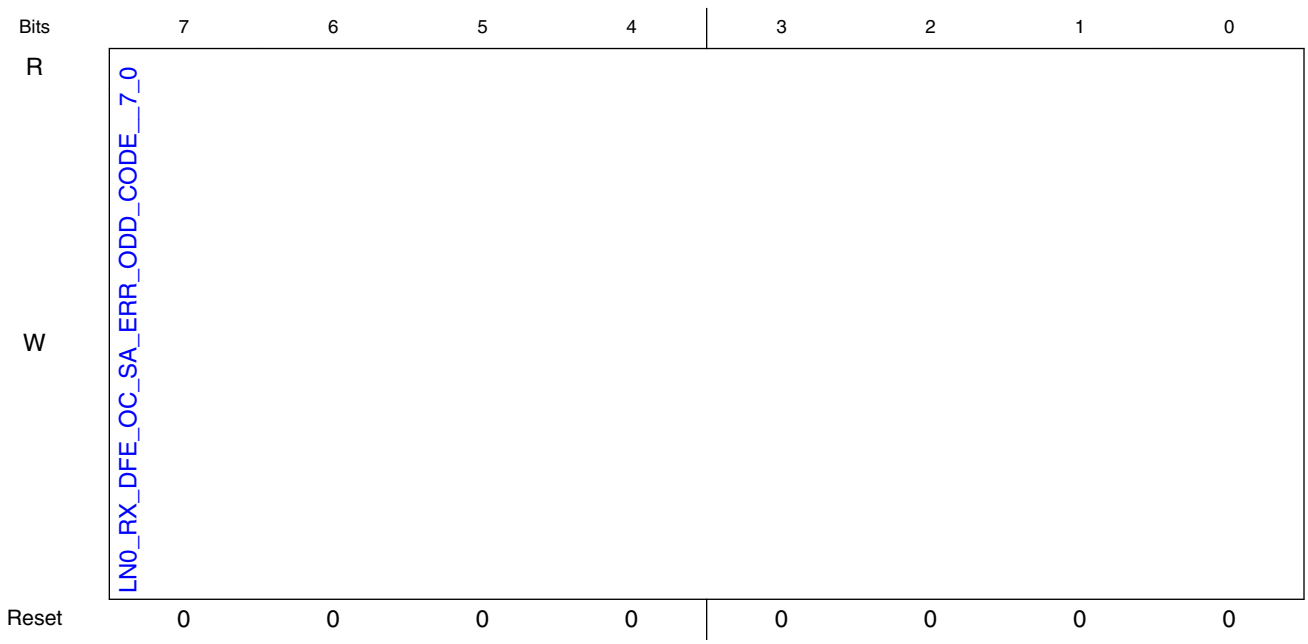
Field	Function
7-2 —	Reserved
1 LN0_OVRD_RX_DFE_OC_SA_ERR_ODD_CODE	Override enable for rx_dfe_oc_sa_err_odd_code
0 LN0_RX_DFE_OC_SA_ERR_ODD_CODE__8	RX DFE odd error path offset calibration code

11.4.3.1.224 (TRSV_REG061)

11.4.3.1.224.1 Offset

Register	Offset
TRSV_REG061	584h

11.4.3.1.224.2 Diagram



11.4.3.1.224.3 Fields

Field	Function
7-0 LN0_RX_DFE_OC_SA_ERR_ODD_CODE__7_0	RX DFE odd error path offset calibration code

11.4.3.1.225 (TRSV_REG062)

11.4.3.1.225.1 Offset

Register	Offset
TRSV_REG062	588h

11.4.3.1.225.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_OVRD_RX_DFE_SA_DATA_EVEN_OC_EN	LN0_RX_DFE_SA_DATA_EVEN_OC_EN	LN0_OVRD_RX_DFE_SA_DATA_ODD_OC_EN	LN0_RX_DFE_SA_DATA_ODD_OC_EN	LN0_OVRD_RX_DFE_SA_EDGE_OC_EN	LN0_RX_DFE_SA_EDGE_OC_EN	LN0_OVRD_RX_DFE_SA_ERR_OC_EN	LN0_RX_DFE_SA_ERR_OC_EN
W								
Reset	0	0	0	0	0	0	0	0

11.4.3.1.225.3 Fields

Field	Function
7 LN0_OVRD_RX_DFE_SA_DATA_EVEN_OC_EN	Override enable for rx_dfe_sa_data_even_oc_en
6 LN0_RX_DFE_SA_DATA_EVEN_OC_EN	RX DFE data odd path enable in offset calibration (If all of rx_dfe_sa_*_oc_en are 0, all path are autimatically activated as normal mode) 0: Disable, 1: only data odd path alive
5 LN0_OVRD_RX_DFE_SA_DATA_ODD_OC_EN	Override enable for rx_dfe_sa_data_odd_oc_en
4	RX DFE data even path enable in offset calibration (If all of rx_dfe_sa_*_oc_en are 0, all path are autimatically activated as normal mode)

Table continues on the next page...

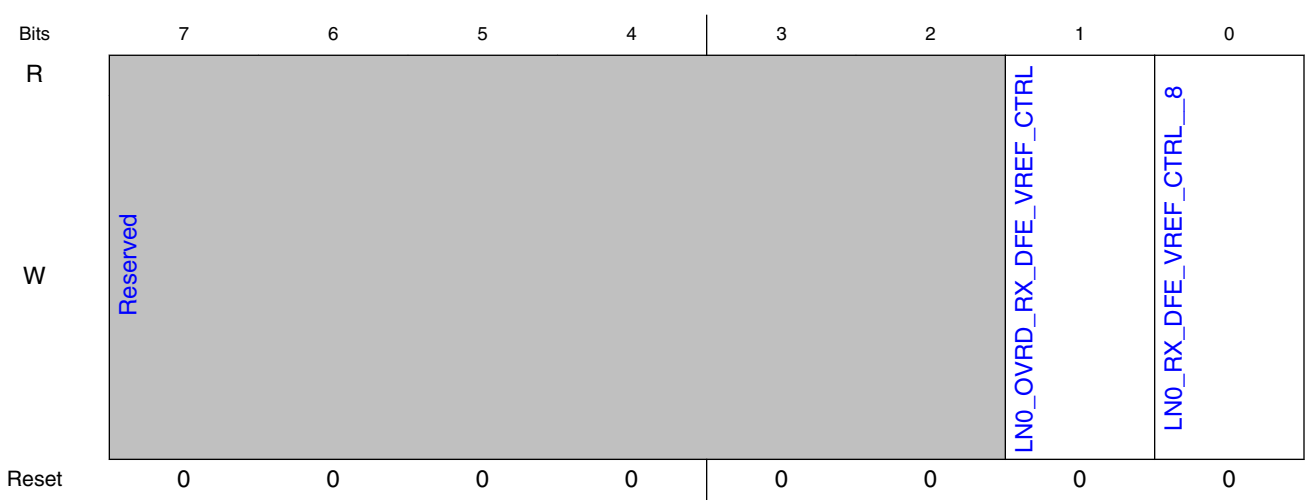
Field	Function
LN0_RX_DFE_SA_DATA_ODD_OC_EN	0: Disable, 1: only data even path alive
3 LN0_OVRD_RX_DFE_SA_EDGE_OC_EN	Override enable for rx_dfe_sa_edge_oc_en
2 LN0_RX_DFE_SA_EDGE_OC_EN	RX DFE edge path enable in offset calibration (If all of rx_dfe_sa_*_oc_en are 0, all path are autimatically activated as normal mode) 0: Disable, 1: only edge path alive
1 LN0_OVRD_RX_DFE_SA_ERR_OC_EN	Override enable for rx_dfe_sa_err_oc_en
0 LN0_RX_DFE_SA_ERR_OC_EN	RX DFE error path enable in offset calibration (If all of rx_dfe_sa_*_oc_en are 0, all path are autimatically activated as normal mode) 0: Disable, 1: only error path alive

11.4.3.1.226 (TRSV_REG063)

11.4.3.1.226.1 Offset

Register	Offset
TRSV_REG063	58Ch

11.4.3.1.226.2 Diagram



11.4.3.1.226.3 Fields

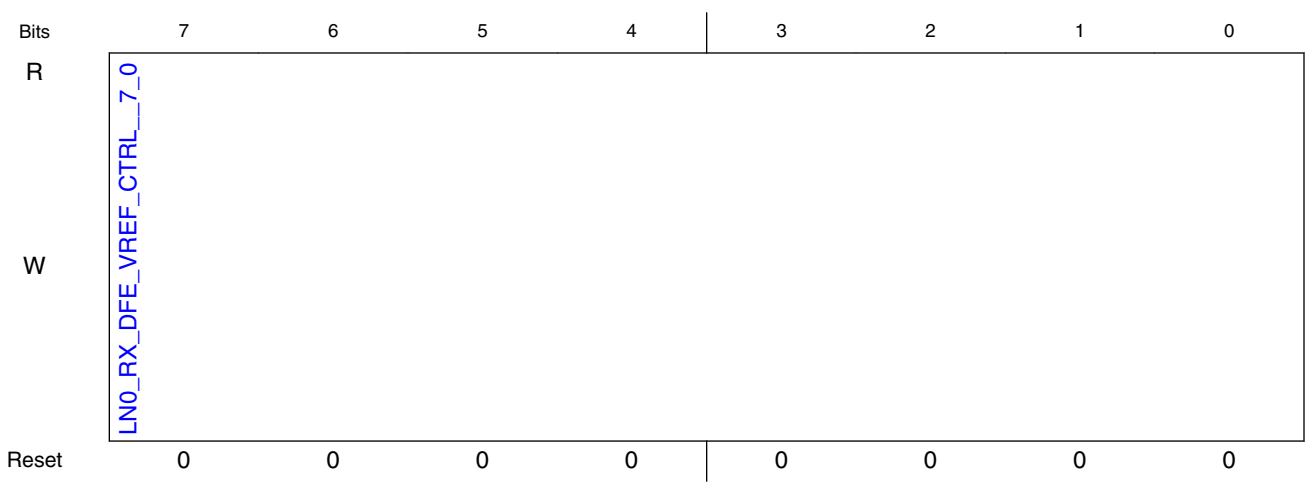
Field	Function
7-2 —	Reserved
1 LN0_OVRD_RX_DFE_VREF_CTRL	Override enable for rx_dfe_vref_ctrl
0 LN0_RX_DFE_VREF_CTRL__8	RX DFE Vref control 0000_0000: -300mVdiff ... 1111_1111: +300mVdiff

11.4.3.1.227 (TRSV_REG064)

11.4.3.1.227.1 Offset

Register	Offset
TRSV_REG064	590h

11.4.3.1.227.2 Diagram



11.4.3.1.227.3 Fields

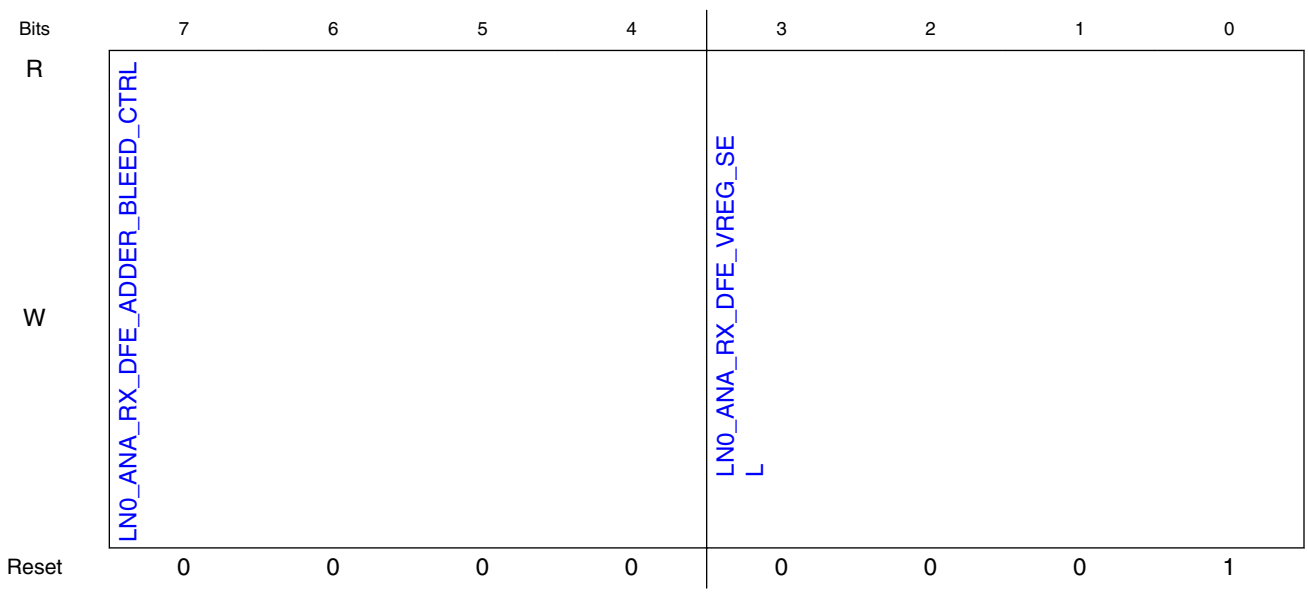
Field	Function
7-0 LN0_RX_DFE_VREF_CTRL__7_0	RX DFE Vref control 0000_0000: -300mVdiff ... 1111_1111: +300mVdiff

11.4.3.1.228 (TRSV_REG065)

11.4.3.1.228.1 Offset

Register	Offset
TRSV_REG065	594h

11.4.3.1.228.2 Diagram



11.4.3.1.228.3 Fields

Field	Function
7-4	

Table continues on the next page...

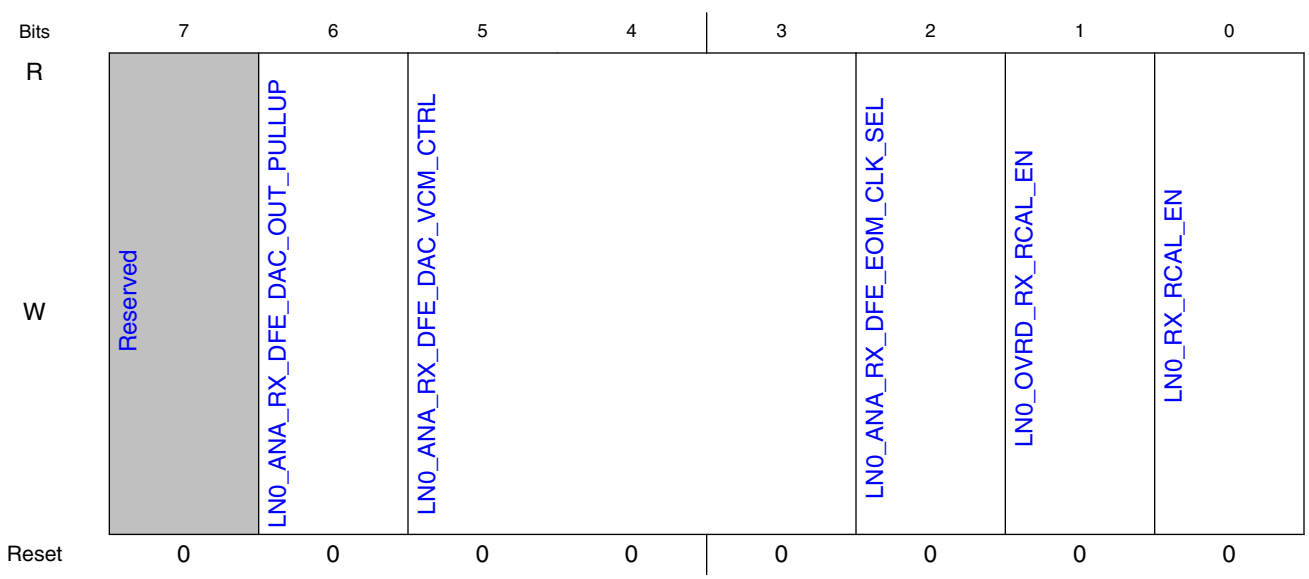
Field	Function
LN0_ANA_RX_DFE_ADDER_BLEED_CTRL	
3-0 LN0_ANA_RX_DFE_VREG_SEL	N/A, DFE bias current control [3:2]: DFE adder bias current, [1:0]: OC DAC bias current. 00: 37.5uA, 01: 50uA, 10: 62.5uA, 11: 75uA

11.4.3.1.229 (TRSV_REG066)

11.4.3.1.229.1 Offset

Register	Offset
TRSV_REG066	598h

11.4.3.1.229.2 Diagram



11.4.3.1.229.3 Fields

Field	Function
7 —	Reserved
6	Pull-up all DAC output in RX DFE to disable all offset code effect

Table continues on the next page...

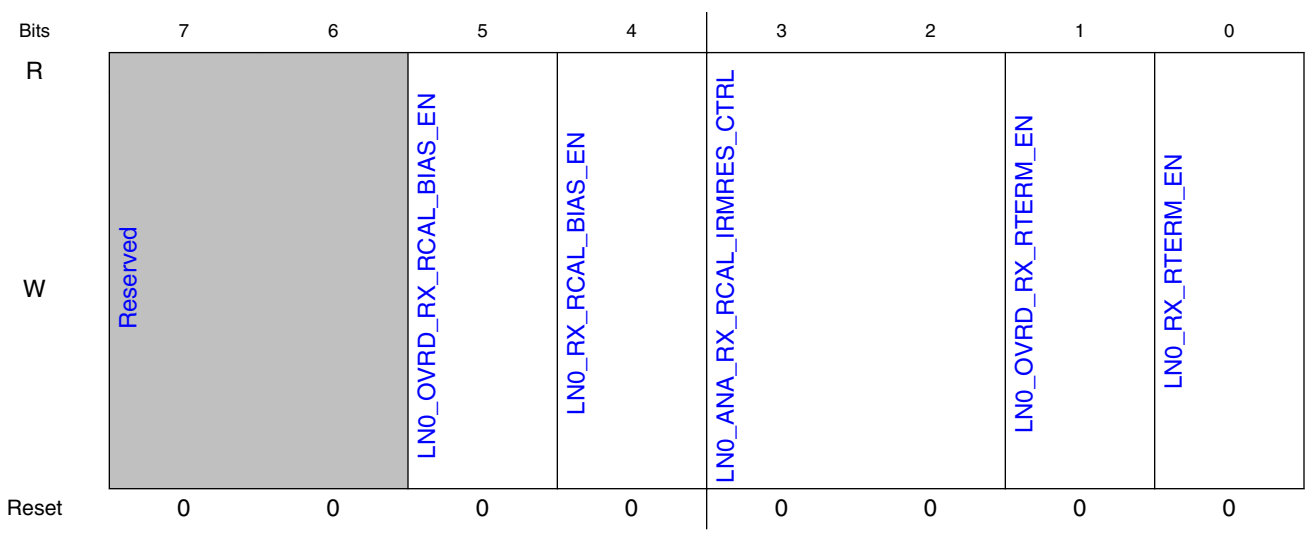
Field	Function
LN0_ANA_RX_DFE_DAC_OUT_PULLUP	0: DAC output enable (offset code effect on), 1: DAC output pull-up (offset code effect off)
5-3 LN0_ANA_RX_DFE_DAC_VCM_CTRL	
2 LN0_ANA_RX_DFE_EOM_CLK_SEL	RX EOM clock selection 0: CDR clock, 1: PLL clock
1 LN0_OVRD_RX_RCAL_EN	Override enable for rx_rcal_en
0 LN0_RX_RCAL_EN	RX RCAL enable 0: Disable, 1: Enable

11.4.3.1.230 (TRSV_REG067)

11.4.3.1.230.1 Offset

Register	Offset
TRSV_REG067	59Ch

11.4.3.1.230.2 Diagram



11.4.3.1.230.3 Fields

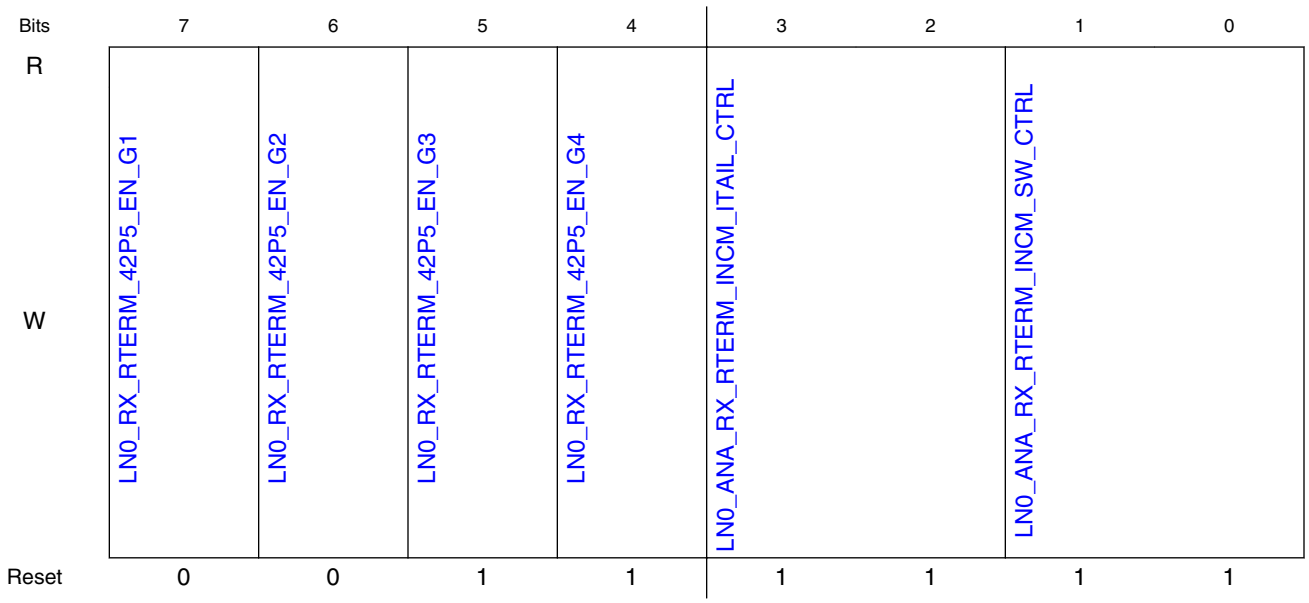
Field	Function
7-6 —	Reserved
5 LN0_OVRD_RX_RCAL_BIAS_EN	Override enable for rx_rcal_bias_en
4 LN0_RX_RCAL_BIAS_EN	RX RCAL bias current enable 0: Disable, 1: Enable
3-2 LN0_ANA_RX_RCAL_IRMRES_CTRL	
1 LN0_OVRD_RX_RTERM_EN	Override enable for rx_rterm_en
0 LN0_RX_RTERM_EN	RX RTERM enable 0: Disable, 1: Enable

11.4.3.1.231 (TRSV_REG068)

11.4.3.1.231.1 Offset

Register	Offset
TRSV_REG068	5A0h

11.4.3.1.231.2 Diagram



11.4.3.1.231.3 Fields

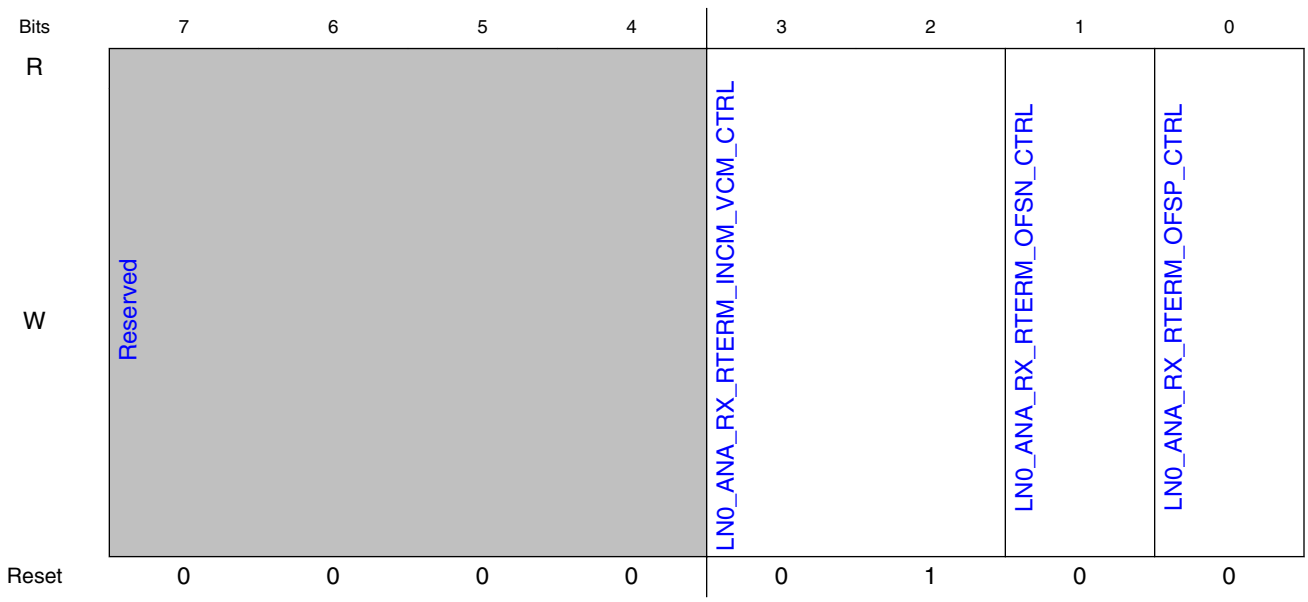
Field	Function
7 LN0_RX_RTERM_42P5_EN_G1	[GEN1] RX RTERM resistance shift 0: 50ohm, 1: 42.5ohm
6 LN0_RX_RTERM_42P5_EN_G2	[GEN2]
5 LN0_RX_RTERM_42P5_EN_G3	[GEN3]
4 LN0_RX_RTERM_42P5_EN_G4	[GEN4]
3-2 LN0_ANA_RX_RTERM_INCM_ITAIL_CTRL	RX RTERM single-ended impedance control by current control 00: Max, .. , 11: Min
1-0 LN0_ANA_RX_RTERM_INCM_SW_CTRL	RX RTERM single-ended impedance control by switch control 00: Max, .. , 11: Min

11.4.3.1.232 (TRSV_REG069)

11.4.3.1.232.1 Offset

Register	Offset
TRSV_REG069	5A4h

11.4.3.1.232.2 Diagram



11.4.3.1.232.3 Fields

Field	Function
7-4 —	Reserved
3-2 LN0_ANA_RX_RTERM_INCM_VCM_CTRL	RX RTERM output common-mode voltage control 00: 180mV, 01: 225mV, 10: 270mV, 11: 300mV
1 LN0_ANA_RX_RTERM_OFSN_CTRL	Offset code for RX RTERM N node
0	Offset code for RX RTERM P node

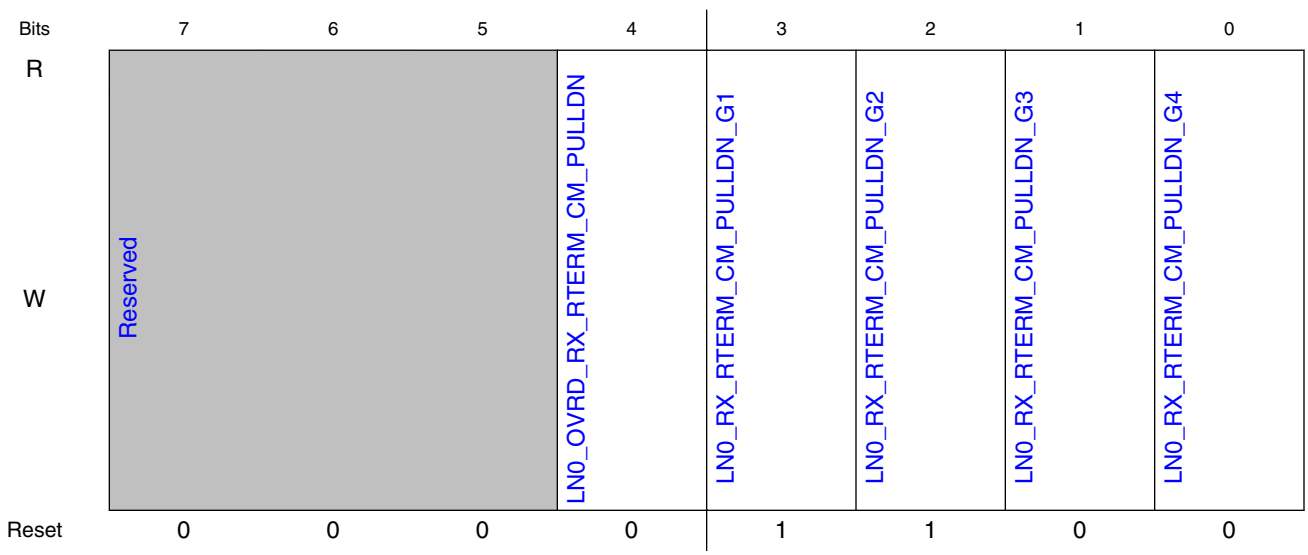
Field	Function
LN0_ANA_RX_RTERM_OFSP_CTRL	0: No offset added, 1: +1 offset code added

11.4.3.1.233 (TRSV_REG06A)

11.4.3.1.233.1 Offset

Register	Offset
TRSV_REG06A	5A8h

11.4.3.1.233.2 Diagram



11.4.3.1.233.3 Fields

Field	Function
7-5 —	Reserved
4 LN0_OVRD_RX_RTERM_CM_PULLDN	Override enable for rx_rterm_cm_pulldn_g1
3	[GEN1] RX RTERM termination voltage pull-down

Table continues on the next page...

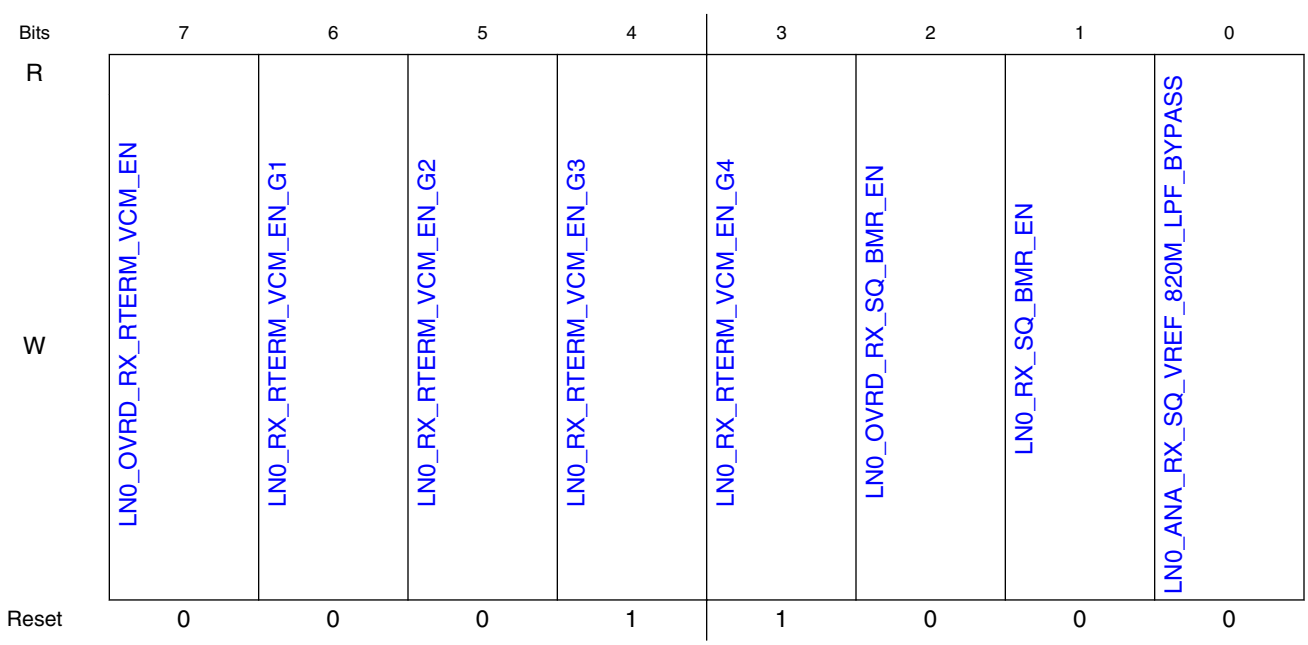
Field	Function
LN0_RX_RTERM_CM_PULLDN_G1	0: No pull-down, 1: Pull-down
2 LN0_RX_RTERM_CM_PULLDN_G2	[GEN2]
1 LN0_RX_RTERM_CM_PULLDN_G3	[GEN3]
0 LN0_RX_RTERM_CM_PULLDN_G4	[GEN4]

11.4.3.1.234 (TRSV_REG06B)

11.4.3.1.234.1 Offset

Register	Offset
TRSV_REG06B	5ACh

11.4.3.1.234.2 Diagram



11.4.3.1.234.3 Fields

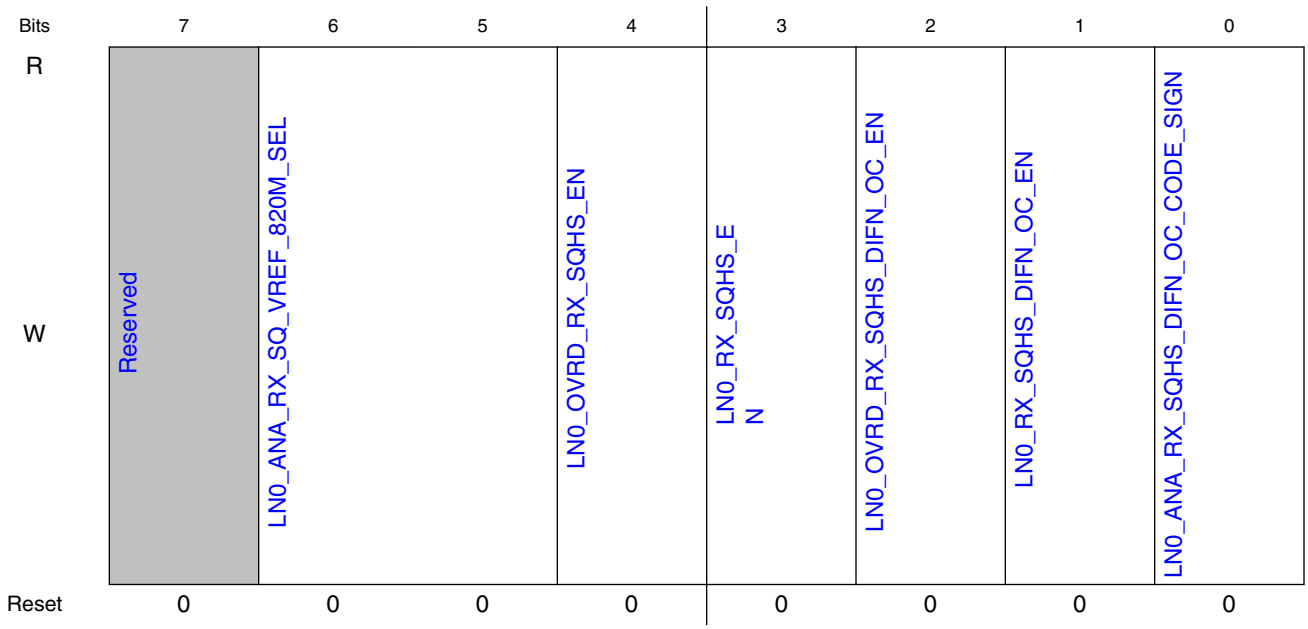
Field	Function
7 LN0_OVRD_RX_RTERM_VCM_EN	Override enable for rx_rterm_vcm_en_g1
6 LN0_RX_RTERM_VCM_EN_G1	[GEN1]
5 LN0_RX_RTERM_VCM_EN_G2	[GEN2]
4 LN0_RX_RTERM_VCM_EN_G3	[GEN3]
3 LN0_RX_RTERM_VCM_EN_G4	[GEN4]
2 LN0_OVRD_RX_SQ_BMR_EN	Override enable for rx_sq_bmr_en
1 LN0_RX_SQ_BMR_EN	
0 LN0_ANA_RX_SQ_VREF_820M_LPF_BYPASS	

11.4.3.1.235 (TRSV_REG06C)

11.4.3.1.235.1 Offset

Register	Offset
TRSV_REG06C	5B0h

11.4.3.1.235.2 Diagram



11.4.3.1.235.3 Fields

Field	Function
7 —	Reserved
6-5 LN0_ANA_RX_SQ_VREF_820M_SEL	
4 LN0_OVRD_RX_SQHS_EN	Override enable for rx_sqhs_en
3 LN0_RX_SQHS_EN	RX high-speed squelch enable 0: Disable, 1: Enable
2 LN0_OVRD_RX_SQHS_DIFN_OC_EN	Override enable for rx_sqhs_difn_oc_en
1 LN0_RX_SQHS_DIFN_OC_EN	RX high-squelch diff-N offset calibration enable
0 LN0_ANA_RX_SQHS_DIFN_OC_CODE_SIGN	RX high-squelch diff-N offset sign 0: Positive, 1: Negative

11.4.3.1.236 (TRSV_REG06D)

11.4.3.1.236.1 Offset

Register	Offset
TRSV_REG06D	5B4h

11.4.3.1.236.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_OVRD_RX_SQHS_DIFN_OC_CODE	LN0_RX_SQHS_DIFN_OC_CODE				LN0_ANA_RX_SQHS_DIFN_SKEWBUF_EN	LN0_OVRD_RX_SQHS_DIFP_OC_EN	LN0_RX_SQHS_DIFP_OC_EN
W								
Reset	0	0	0	0	0	1	0	0

11.4.3.1.236.3 Fields

Field	Function
7 LN0_OVRD_RX_SQHS_DIFN_OC_CODE	Override enable for rx_sqhs_difn_oc_code
6-3 LN0_RX_SQHS_DIFN_OC_CODE	RX high-squelch diff-N manual offset code
2	

Table continues on the next page...

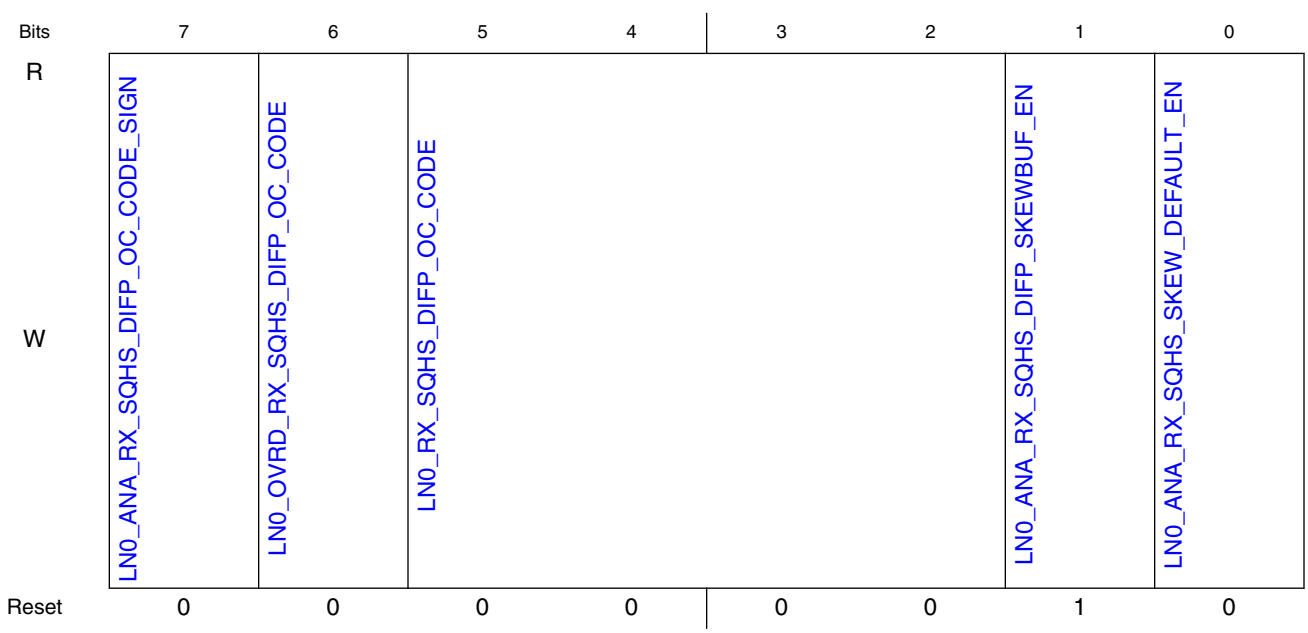
Field	Function
LN0_ANA_RX_SQHS_DIFN_SKEWBUF_EN	
1 LN0_OVRD_RX_SQHS_DIFP_OC_EN	Override enable for rx_sqhs_difp_oc_en
0 LN0_RX_SQHS_DIFP_OC_EN	RX high-squelch diff-P offset calibration enable

11.4.3.1.237 (TRSV_REG06E)

11.4.3.1.237.1 Offset

Register	Offset
TRSV_REG06E	5B8h

11.4.3.1.237.2 Diagram



11.4.3.1.237.3 Fields

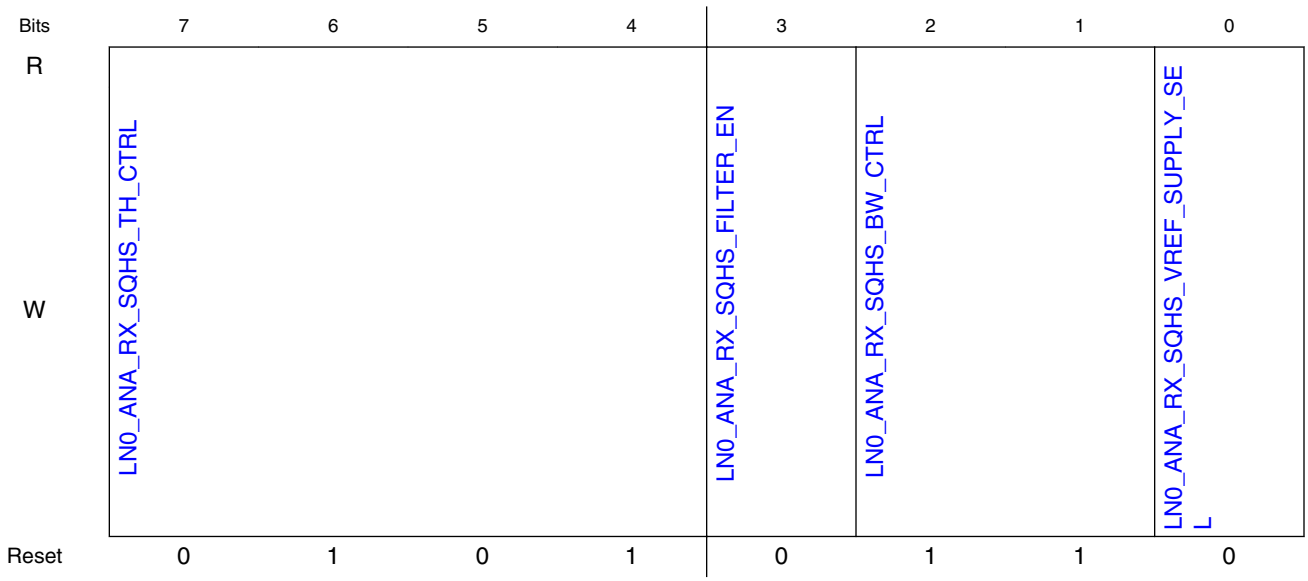
Field	Function
7 LN0_ANA_RX_SQHS_DIFP_OC_CODE_SIGN	RX high-squelch diff-P offset sign 0: Positive, 1: Negative
6 LN0_OVRD_RX_SQHS_DIFP_OC_CODE	Override enable for rx_sqhs_difp_oc_code
5-2 LN0_RX_SQHS_DIFP_OC_CODE	RX squelch diff-P manual offset code 0000: (-) Max, … 1111: (+) Max
1 LN0_ANA_RX_SQHS_DIFP_SKEWBUF_EN	Enable the high speed Squelch DIFP SKEW BUFFER
0 LN0_ANA_RX_SQHS_SKEW_DEFAULT_EN	Fixed skew for PCIe/SATA Squelch

11.4.3.1.238 (TRSV_REG06F)

11.4.3.1.238.1 Offset

Register	Offset
TRSV_REG06F	5BCh

11.4.3.1.238.2 Diagram



11.4.3.1.238.3 Fields

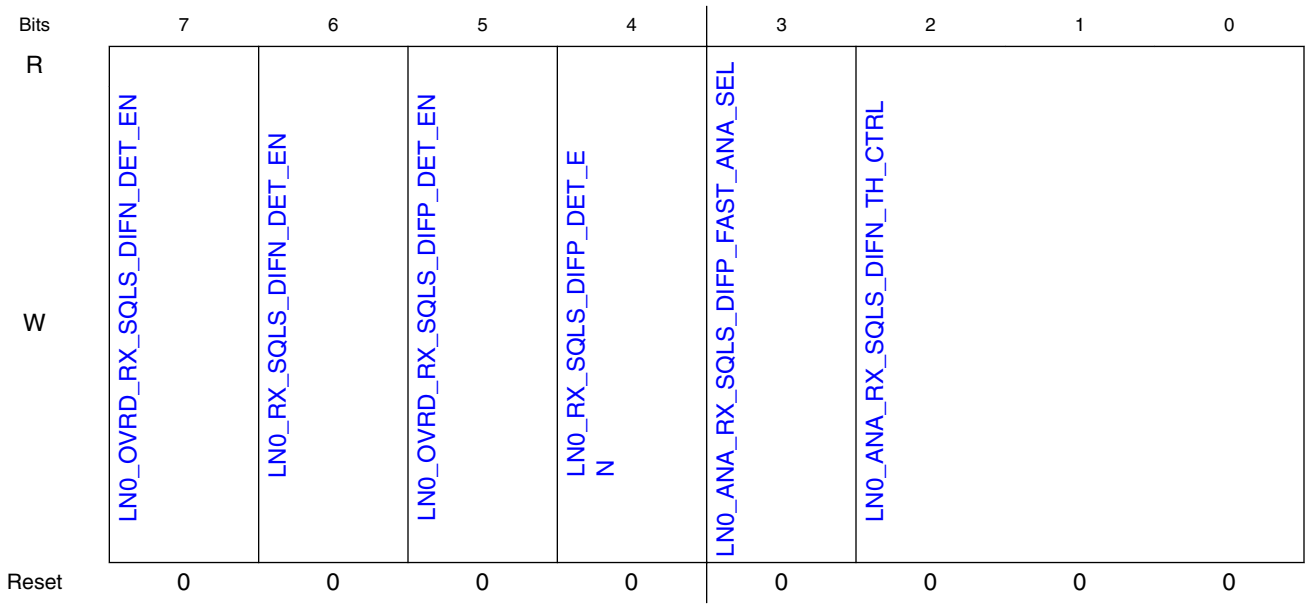
Field	Function
7-4 LN0_ANA_RX_SQHS_TH_CTRL	RX squelch threshold voltage selection 0000: 0mV, 0001: 45mV, 0010: 60mV, 0011: 75mV 0100: 90mV, 0101: 105mV, 0110: 120mV, 0111: 135mV 1000: 150mV, 1001: 165mV, 1010: 180mV, 1011: 195mV 1100: 210mV, 1101: 225mV, 1110: 240mV, 1111: 255mV
3 LN0_ANA_RX_SQHS_FILTER_EN	SQHS loss detector enable
2-1 LN0_ANA_RX_SQHS_BW_CTRL	
0 LN0_ANA_RX_SQHS_VREF_SUPPLY_SEL	Selection of supply voltage of reference voltage for threshold calibration of HS SQ 0: VDD, 1: VREG

11.4.3.1.239 (TRSV_REG070)

11.4.3.1.239.1 Offset

Register	Offset
TRSV_REG070	5C0h

11.4.3.1.239.2 Diagram



11.4.3.1.239.3 Fields

Field	Function
7 LN0_OVRD_RX_SQLS_DIFN_DET_EN	Override enable for rx_sqls_difn_det_en
6 LN0_RX_SQLS_DIFN_DET_EN	RX low-speed DIFN squelch enable 0: Disable, 1: Enable
5 LN0_OVRD_RX_SQLS_DIFP_DET_EN	Override enable for rx_sqls_difp_det_en
4 LN0_RX_SQLS_DIFP_DET_EN	RX low-speed DIFP squelch enable 0: Disable, 1: Enable
3	RX DIFP detect signal selection for PRE_DATA_VALID and DATA_VALID signal

Table continues on the next page...

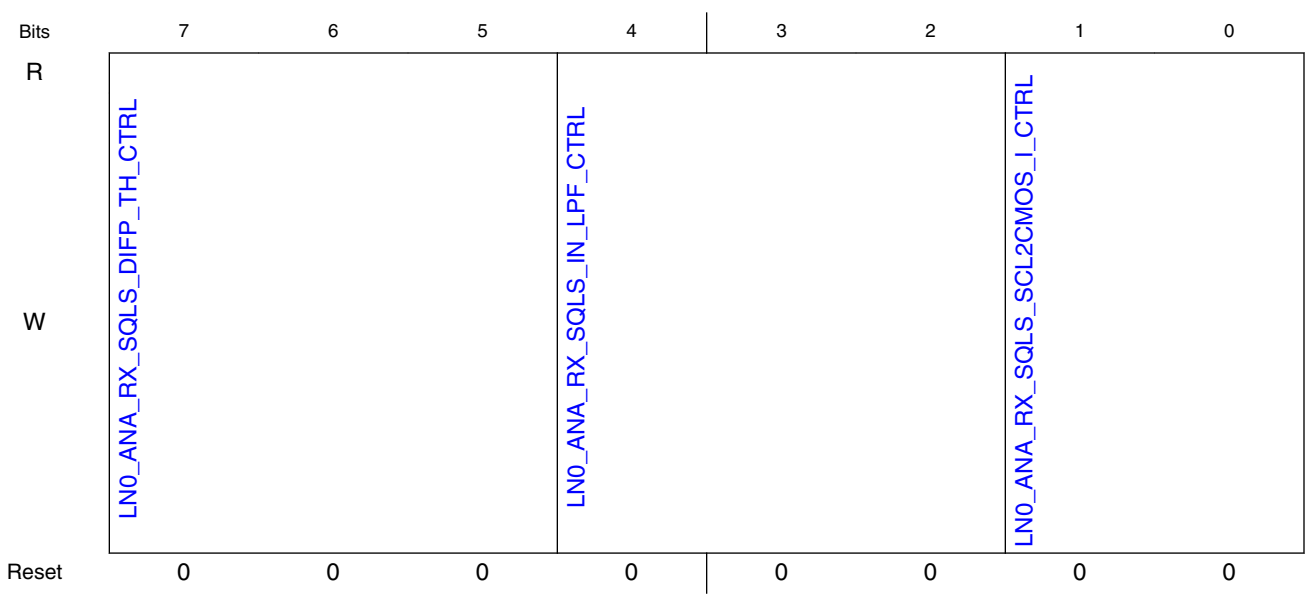
Field	Function
LN0_ANA_RX_SQLS_DIFP_FAST_ANA_SEL	0: low-speed squelch detect signal, 1: low-speed squelch detect signal detect signal
2-0 LN0_ANA_RX_SQLS_DIFN_TH_CTRL	DIFN in MPHY, LFPS in USB: 0:40m, 1:60m, 2:80m, 3:96m, 4:116m, 5:135m, 6: 154m, 7:174m

11.4.3.1.240 (TRSV_REG071)

11.4.3.1.240.1 Offset

Register	Offset
TRSV_REG071	5C4h

11.4.3.1.240.2 Diagram



11.4.3.1.240.3 Fields

Field	Function
7-5	DIFP in MPHY 0:40m, 1:60m, 2:80m, 3:96m, 4:116m, 5:135m, 6: 154m, 7:174m

Table continues on the next page...

Field	Function
LN0_ANA_RX_SQLS_DIFP_TH_CTRL	
4-2 LN0_ANA_RX_SQLS_IN_LPF_CTRL	Low pass filter resistor control for Squelch input : 00:30MHz,01:60MHz,, 10:100MHz, 11: 150MHz
1-0 LN0_ANA_RX_SQLS_SCL2CMOS_I_CTRL	Current controls for low-speed Squelch comparator current source 00 (Min BW) - 11:Max BW

11.4.3.1.241 (TRSV_REG072)

11.4.3.1.241.1 Offset

Register	Offset
TRSV_REG072	5C8h

11.4.3.1.241.2 Diagram

Bits	7	6	5	4	3	2	1	0
R								
W	LN0_OVRD_RX_PWM_OSC_EN	LN0_RX_PWM_OSC_EN	LN0_OVRD_RX_PWM_CNT_EN	LN0_RX_PWM_CNT_EN	LN0_OVRD_RX_PWM_RSTN	LN0_RX_PWM_RSTN	LN0_OVRD_RX_PWM_AFC_RSTN	LN0_RX_PWM_AFC_RSTN
Reset	0	0	0	0	0	0	0	0

11.4.3.1.241.3 Fields

Field	Function
7	Override enable for rx_pwm_osc_en

Table continues on the next page...

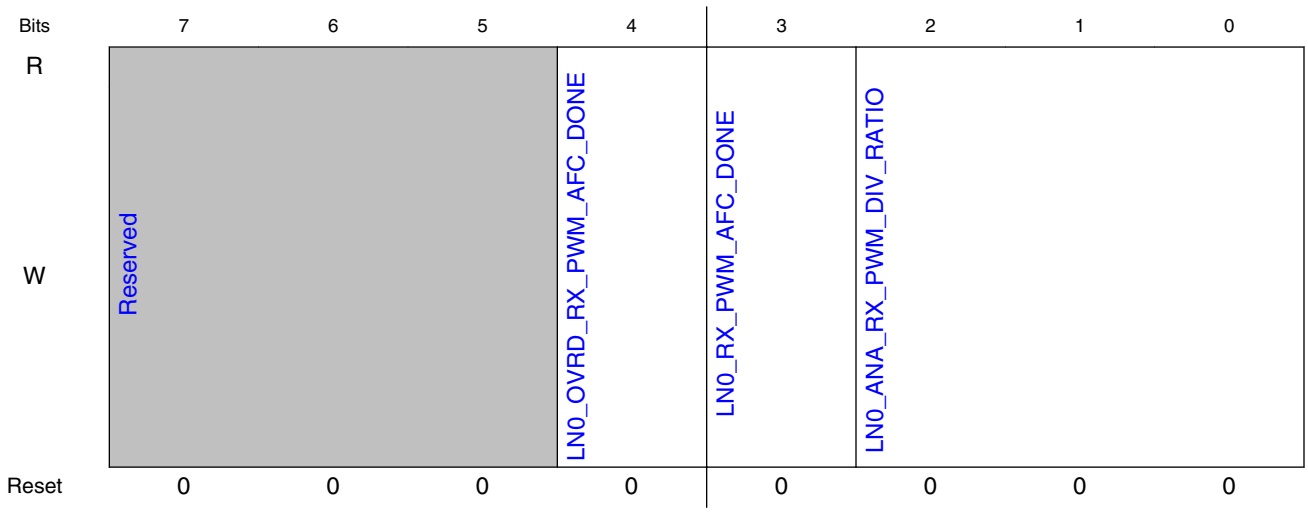
Field	Function
LN0_OVRD_RX_PWM_OSC_EN	
6 LN0_RX_PWM_OSC_EN	RX MPHY PWM oscillator enable which is used in analog RX block in order to oversample 0: Disable, 1: Enable
5 LN0_OVRD_RX_PWM_CNT_EN	Override enable for rx_pwm_cnt_en
4 LN0_RX_PWM_CNT_EN	Enalbe counter clock for PWM over sampling 0: Disable, 1: Enable
3 LN0_OVRD_RX_PWM_RSTN	Override enable for rx_pwm_rstn
2 LN0_RX_PWM_RSTN	RX MPHY PWM reset
1 LN0_OVRD_RX_PWM_AFC_RSTN	Override enable for rx_pwm_afc_rstn
0 LN0_RX_PWM_AFC_RSTN	RX MPHY PWM AFC reset

11.4.3.1.242 (TRSV_REG073)

11.4.3.1.242.1 Offset

Register	Offset
TRSV_REG073	5CCh

11.4.3.1.242.2 Diagram



11.4.3.1.242.3 Fields

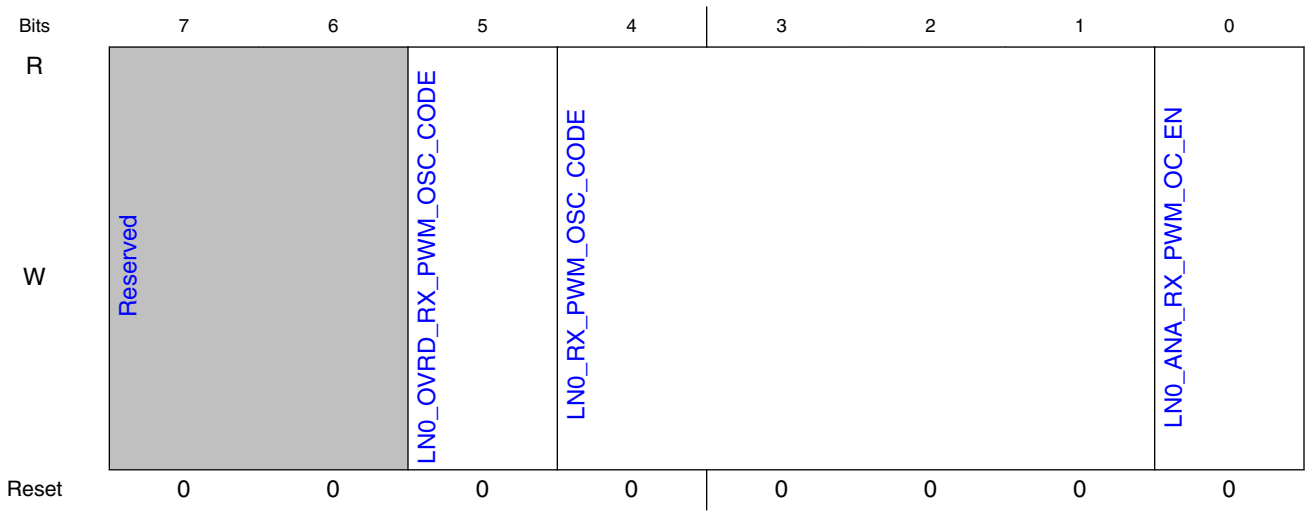
Field	Function
7-5 —	Reserved
4 LN0_OVRD_RX_PWM_AFC_DONE	Override enable for rx_pwm_afc_done
3 LN0_RX_PWM_AFC_DONE	RX MPHY PWM AFC done signal
2-0 LN0_ANA_RX_PWM_DIV_RATIO	RX MPHY PWM oversampling clock divide ratio from PWM oscillator 000: 001:

11.4.3.1.243 (TRSV_REG074)

11.4.3.1.243.1 Offset

Register	Offset
TRSV_REG074	5D0h

11.4.3.1.243.2 Diagram



11.4.3.1.243.3 Fields

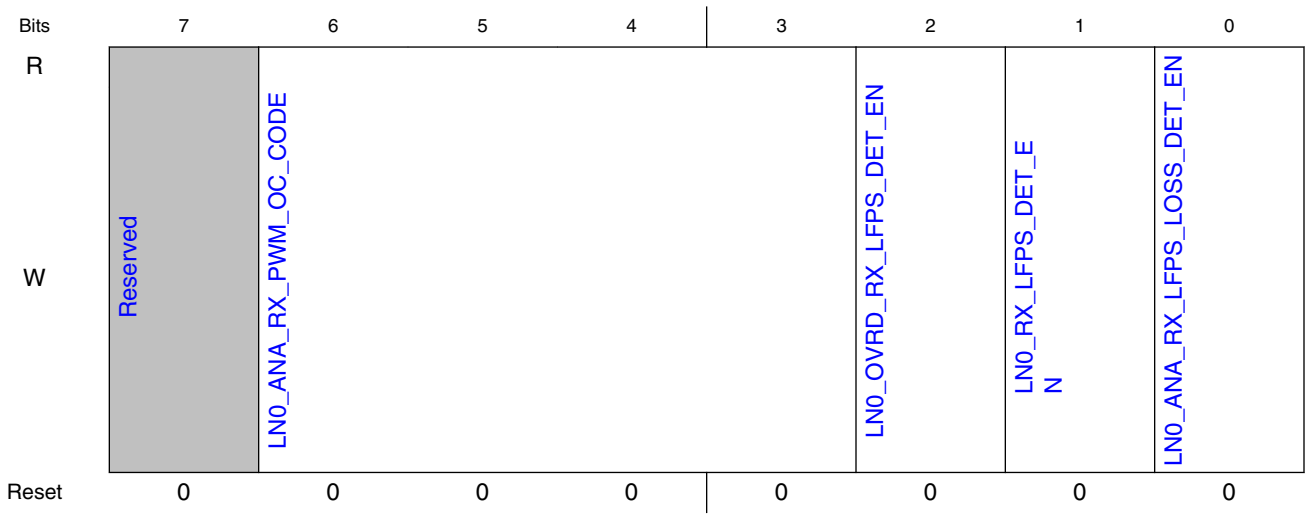
Field	Function
7-6 —	Reserved
5 LN0_OVRD_RX_PWM_OSC_CODE	Override enable for rx_pwm_osc_code
4-1 LN0_RX_PWM_OSC_CODE	RX MPHY PWM AFC code for oscillator
0 LN0_ANA_RX_PWM_OC_EN	

11.4.3.1.244 (TRSV_REG075)

11.4.3.1.244.1 Offset

Register	Offset
TRSV_REG075	5D4h

11.4.3.1.244.2 Diagram



11.4.3.1.244.3 Fields

Field	Function
7 —	Reserved
6-3 LN0_ANA_RX_PWM_OC_CODE	min(-7 or 0_111){maximum negative offset} - max(+7 or 1_111) {maximum positive offset} <3>: Sign bit <2:0> Offset magnitude bits{000 no offset, 111=max offset} 1_000(+0) or 0000(-0) are center code
2 LN0_OVRD_RX_LFPS_DET_EN	Override enable for rx_lfps_det_en
1 LN0_RX_LFPS_DET_EN	LFPS detector enable 0 : Disable, 1:Enable
0 LN0_ANA_RX_LFPS_LOSS_DET_EN	LFPS loss detector enable 0 : Disable, 1:Enable

11.4.3.1.245 (TRSV_REG076)

11.4.3.1.245.1 Offset

Register	Offset
TRSV_REG076	5D8h

11.4.3.1.245.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_OVRD_RX_BIAS_EN	LN0_RX_BIAS_EN	LN0_ANA_RX_BIAS_RMRES_CTRL			LN0_ANA_RX_SLB_EN	LN0_ANA_RX_LLB_EN	LN0_ANA_RX_SRLB_EN
W								
Reset	0	0	0	1	0	0	0	0

11.4.3.1.245.3 Fields

Field	Function
7 LN0_OVRD_RX_BIAS_EN	Override enable for rx_bias_en
6 LN0_RX_BIAS_EN	RX bias current enable 0: Disable all RX bias current, 1: Enable
5-3 LN0_ANA_RX_BIAS_RMRES_CTRL	RX RMRES bias current control 000: 62%, 001: 87%, 010: 100% (Default), 100: 138%, 111: 156%
2 LN0_ANA_RX_SLB_EN	Serial loopback enable 0: Disable, 1: Enable (PMA digital → TX SER → RX CDR (DES) → PMA digital)
1 LN0_ANA_RX_LLB_EN	Line loopback enalble 0: Disable, 1: Enable (RX pad → RX AFE (CTLE) → TX driver → TX PAD)
0 LN0_ANA_RX_SRLB_EN	Serial retimed loopback enable 0: Disable 1: Enable (RX pad → RX AFE → RX CDR (BBPD) → TX driver → TX pad)

11.4.3.1.246 (TRSV_REG077)

11.4.3.1.246.1 Offset

Register	Offset
TRSV_REG077	5DCh

11.4.3.1.246.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_ANA_RX_SRLB_EVEN_ODD_SEL	LN0_ANA_RX_SRLB_DATA_EDGE_SEL	LN0_ANA_RX_LLB_ACCAP_EN	LN0_ANA_RX_LLB_DUTY_CTRL			LN0_ANA_RX_LLB_ITAIL_CTRL	
W								
Reset	0	0	0	0	0	0	0	0

11.4.3.1.246.3 Fields

Field	Function
7 LN0_ANA_RX_SRLB_EVEN_ODD_SEL	Serial retimed loopback path selection 0: Even path, 1: Odd path
6 LN0_ANA_RX_SRLB_DATA_EDGE_SEL	Serial retimed loopback path selection 0: Data path, 1: Edge path
5 LN0_ANA_RX_LLB_ACCAP_EN	Line loopback path selection 0: SCL-to-CMOS, 1: AC-coupled

Table continues on the next page...

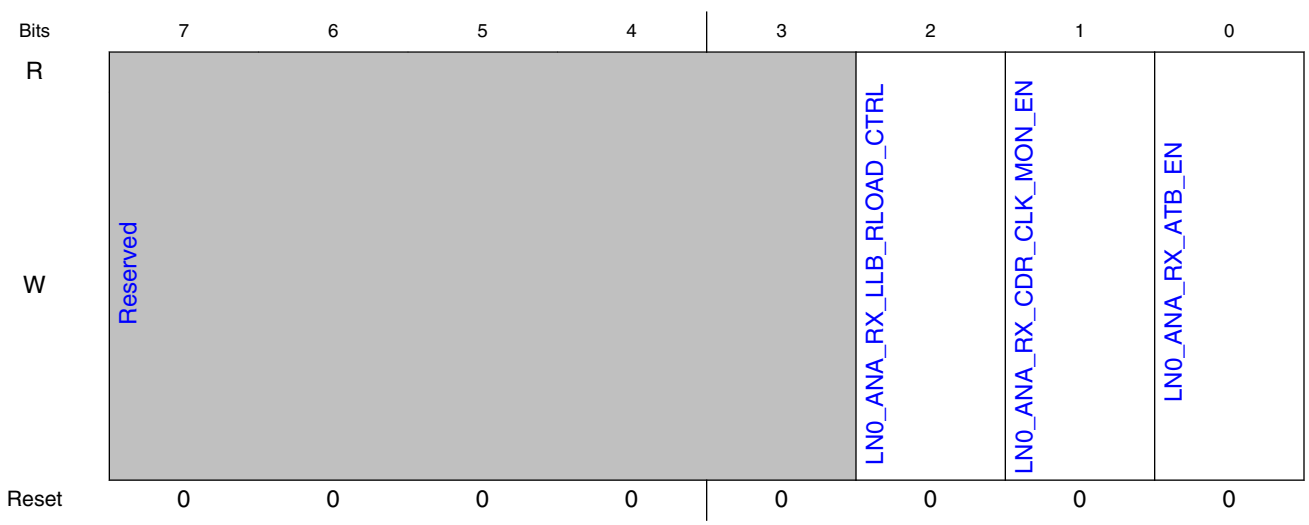
Field	Function
4-2 LN0_ANA_RX_LLB_DUTY_CTRL	Line loopback duty-ratio control 000:, 111:
1-0 LN0_ANA_RX_LLB_ITAIL_CTRL	Line loopback tail-current control 00:, 11:

11.4.3.1.247 (TRSV_REG078)

11.4.3.1.247.1 Offset

Register	Offset
TRSV_REG078	5E0h

11.4.3.1.247.2 Diagram



11.4.3.1.247.3 Fields

Field	Function
7-3 —	Reserved
2	Line loopback load resistance control

Table continues on the next page...

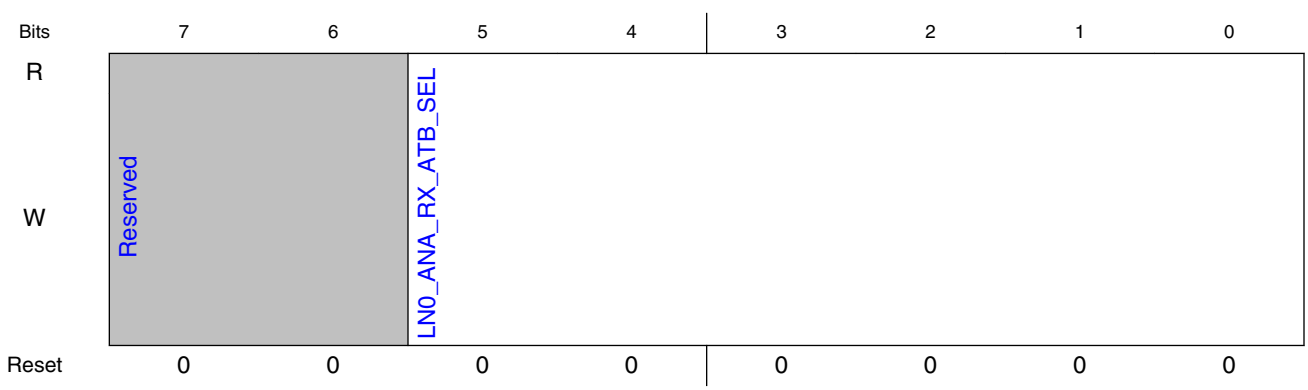
Field	Function
LN0_ANA_RX_LLB_RLOAD_CTRL	0:, 1:
1 LN0_ANA_RX_CDR_CLK_MON_EN	
0 LN0_ANA_RX_ATB_EN	RX ATB enable

11.4.3.1.248 (TRSV_REG079)

11.4.3.1.248.1 Offset

Register	Offset
TRSV_REG079	5E4h

11.4.3.1.248.2 Diagram



11.4.3.1.248.3 Fields

Field	Function
7-6 —	Reserved
5-0 LN0_ANA_RX_ATB_SEL	When i_sfr_rx_atb_en=1 and i_sfr_rx_atb_sel<5>=0, RX AFE nodes are under monitoring. According to i_sfr_rx_atb_sel<4:0>,

Field	Function
	00000: LLB_SCL_INN, 00001: LLB_SCL_INP, 00010: OUTN, 00011: OUTP, 00100: VSS, 00101: VCM_INCMGENOUT, 00110: CTLE_VCM, 00111: CTLE_NBIAS, 01000: CTLE_PCAS, 01001: CTLE_PBIAS, 01010: SQOFSN, 01011: SQOFSP, 01100: SQ_PBIAS, 01101: VSS, 01110: VSS, 01111: VDD When i_sfr_rx_atb_en=1 and i_sfr_rx_atb_sel<5>=0, RX CDR nodes are under monitoring. According to i_sfr_rx_atb_sel<4:0>, 00000: VCI SET, 00001: VCM, 00010: VREG 820mV LPF, 00011: CP VDDH, 00100: CP FB, 00101: DCC VDD, 00110:DCC VSS, 00111: IQ-DIV VDD, 01000: IQ-DIV VSS, 01001: BBPD/DFE VDD, 01010: BBPD/DFE VSS, 01011: DFE VREFP, 01100: DFE VREFN, 01101: DFE VOFSP DE, 01110: DFE VOFSN DE, 01111: N/A

11.4.3.1.249 (TRSV_REG07A)

11.4.3.1.249.1 Offset

Register	Offset
TRSV_REG07A	5E8h

11.4.3.1.249.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_ANA_RX_RESERVED							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.249.3 Fields

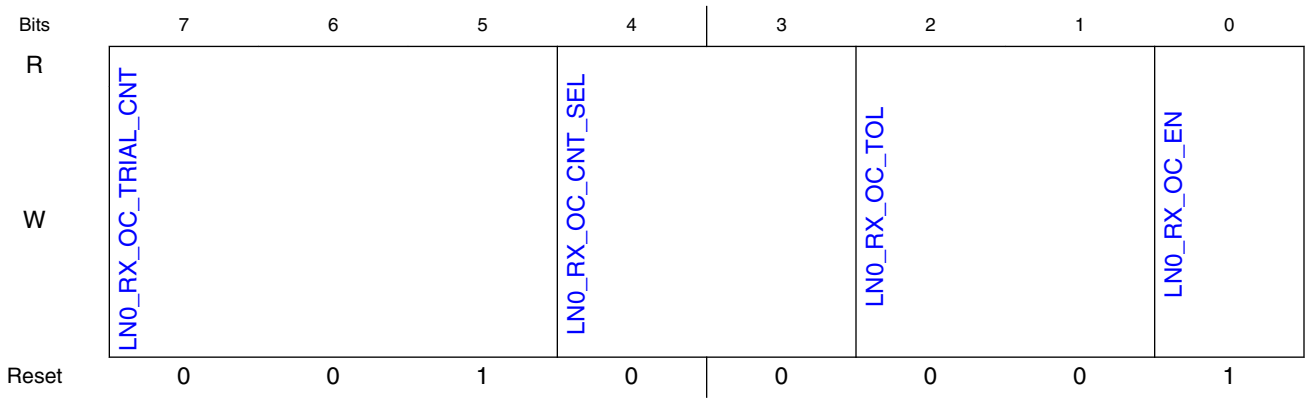
Field	Function
7-0 LN0_ANA_RX_RESERVED	Reserved port

11.4.3.1.250 (TRSV_REG07B)

11.4.3.1.250.1 Offset

Register	Offset
TRSV_REG07B	5ECh

11.4.3.1.250.2 Diagram



11.4.3.1.250.3 Fields

Field	Function
7-5 LN0_RX_OC_TRIAL_CNT	RX offset calibration trial number selection 00: 2, 01: 4, 10: 6, 11: 8
4-3 LN0_RX_OC_CNT_SEL	RX SQ offset calibraiton counter selection for waiting time 00: 10, 01: 15, 20: 4, 11: 25
2-1 LN0_RX_OC_TOL	RX offset calibration enable 0: Disable, 1: Enable
0 LN0_RX_OC_EN	RX offset calibration tolerance for average value 00: 2, 01: 3, 10: 4, 11: 5

11.4.3.1.251 (TRSV_REG07C)

11.4.3.1.251.1 Offset

Register	Offset
TRSV_REG07C	5F0h

11.4.3.1.251.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_RX_OC_UPD_CNT_SEL		LN0_RX_OC_BYPASS_DFE_SA_ERR_ODD	LN0_RX_OC_BYPASS_DFE_SA_ERR_EVEN	LN0_RX_OC_BYPASS_DFE_SA_EDGE_ODD	LN0_RX_OC_BYPASS_DFE_SA_EDGE_EVEN	LN0_RX_OC_BYPASS_DFE_ADDER_ODD_FINAL	LN0_RX_OC_BYPASS_DFE_ADDER_EVEN_FINAL
W								
Reset	0	0	0	0	0	0	0	0

11.4.3.1.251.3 Fields

Field	Function
7-6 LN0_RX_OC_UPD_CNT_SEL	RX offset calibration code wating time selection for SA & CTLE only 00: 4, 01: 8, 10: 12, 11:16
5 LN0_RX_OC_BYPASS_DFE_SA_ERR_ODD	Bypass
4 LN0_RX_OC_BYPASS_DFE_SA_ERR_EVEN	
3	

Table continues on the next page...

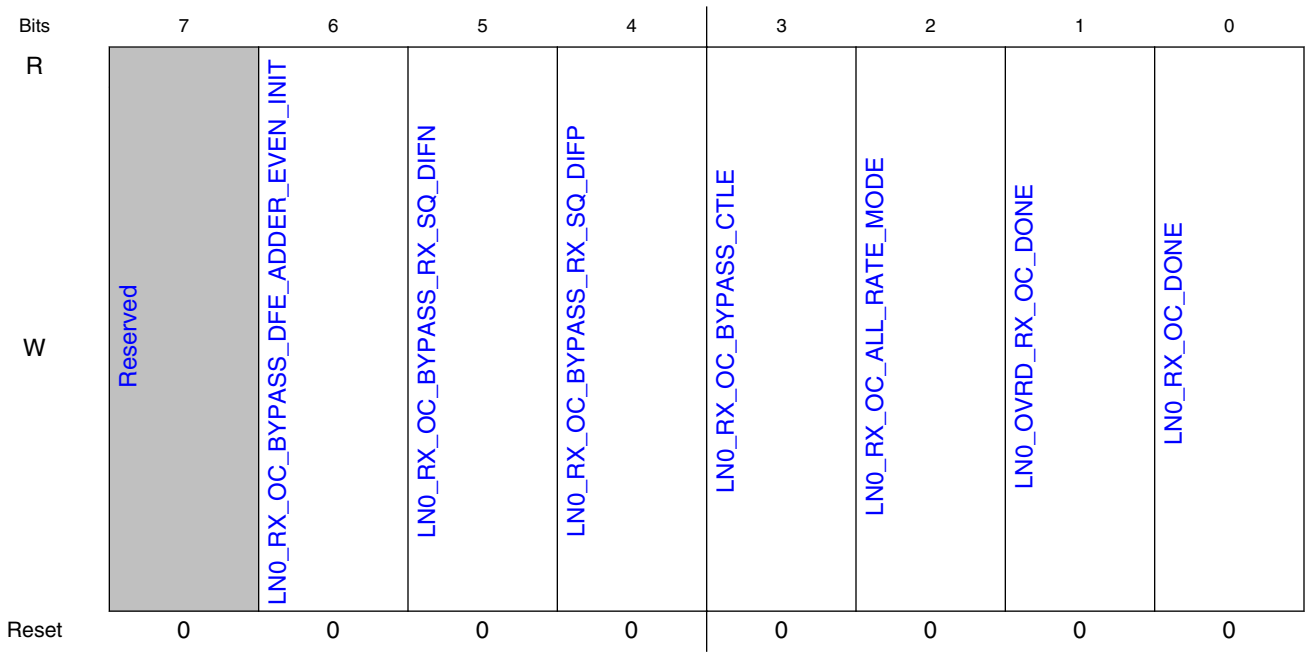
Field	Function
LN0_RX_OC_BYPASS_DFE_SA_EDGE_ODD	
2 LN0_RX_OC_BYPASS_DFE_SA_EDGE_EVEN	
1 LN0_RX_OC_BYPASS_DFE_ADDER_ODD_FINAL	
0 LN0_RX_OC_BYPASS_DFE_ADDER_EVEN_FINAL	

11.4.3.1.252 (TRSV_REG07D)

11.4.3.1.252.1 Offset

Register	Offset
TRSV_REG07D	5F4h

11.4.3.1.252.2 Diagram



11.4.3.1.252.3 Fields

Field	Function
7 — Reserved	
6 LN0_RX_OC_BYPASS_DFE_ADDER_EVEN_INIT	
5 LN0_RX_OC_BYPASS_RX_SQ_DIFN	
4 LN0_RX_OC_BYPASS_RX_SQ_DIFP	
3 LN0_RX_OC_BYPASS_CTLE	Bypass offset calibration for CTLE
2 LN0_RX_OC_ALL_RATE_MODE	

Table continues on the next page...

Field	Function
1 LN0_OVRD_RX_OC_DONE	Override enable for rx_oc_done
0 LN0_RX_OC_DONE	RX offset calibration overide value

11.4.3.1.253 (TRSV_REG07E)

11.4.3.1.253.1 Offset

Register	Offset
TRSV_REG07E	5F8h

11.4.3.1.253.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_RX_SSLMS_C0_INIT							
Reset	1	1	0	0	0	0	0	0

11.4.3.1.253.3 Fields

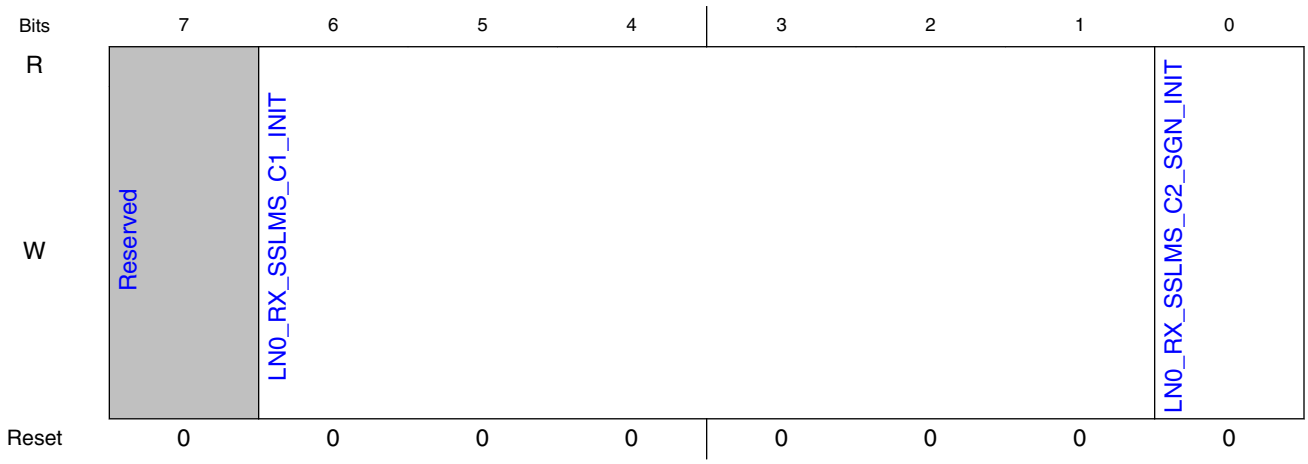
Field	Function
7-0 LN0_RX_SSLMS_C0_INIT	

11.4.3.1.254 (TRSV_REG07F)

11.4.3.1.254.1 Offset

Register	Offset
TRSV_REG07F	5FCh

11.4.3.1.254.2 Diagram



11.4.3.1.254.3 Fields

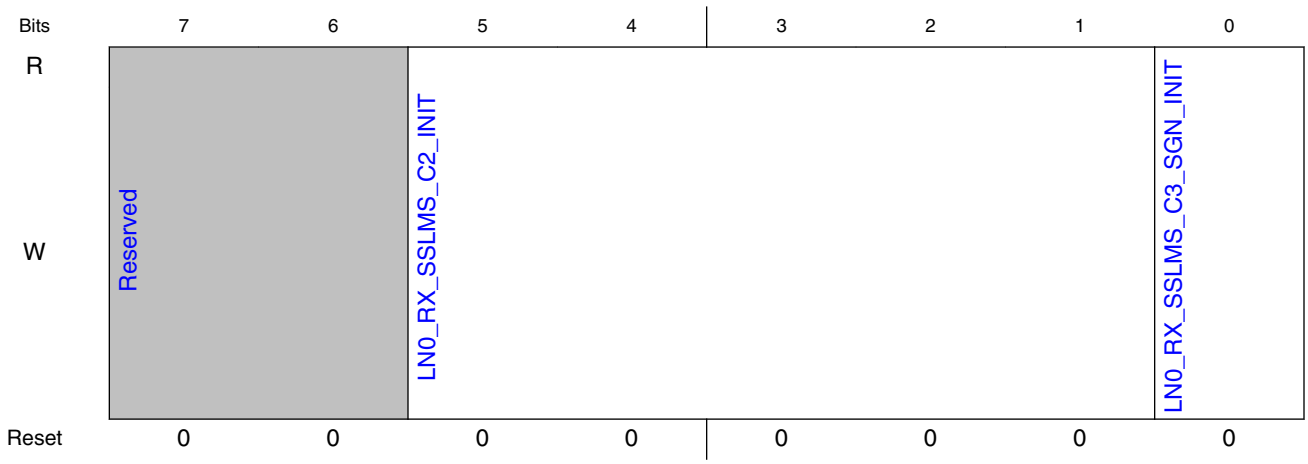
Field	Function
7 —	Reserved
6-1 LN0_RX_SSLMS_C1_INIT	
0 LN0_RX_SSLMS_C2_SGN_INIT	

11.4.3.1.255 (TRSV_REG080)

11.4.3.1.255.1 Offset

Register	Offset
TRSV_REG080	600h

11.4.3.1.255.2 Diagram



11.4.3.1.255.3 Fields

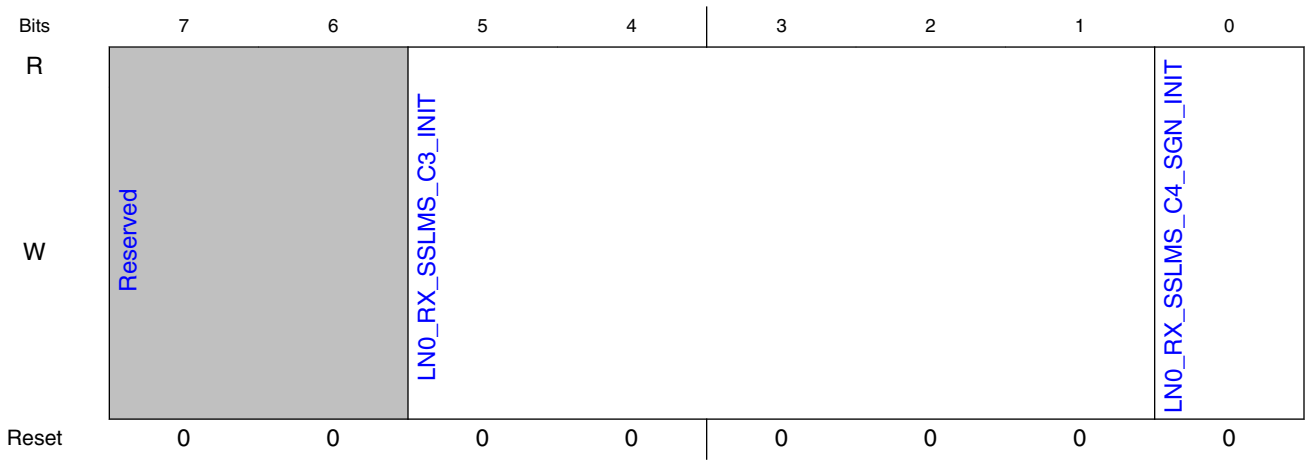
Field	Function
7-6 —	Reserved
5-1 LN0_RX_SSLMS_C2_INIT	
0 LN0_RX_SSLMS_C3_SGN_INIT	

11.4.3.1.256 (TRSV_REG081)

11.4.3.1.256.1 Offset

Register	Offset
TRSV_REG081	604h

11.4.3.1.256.2 Diagram



11.4.3.1.256.3 Fields

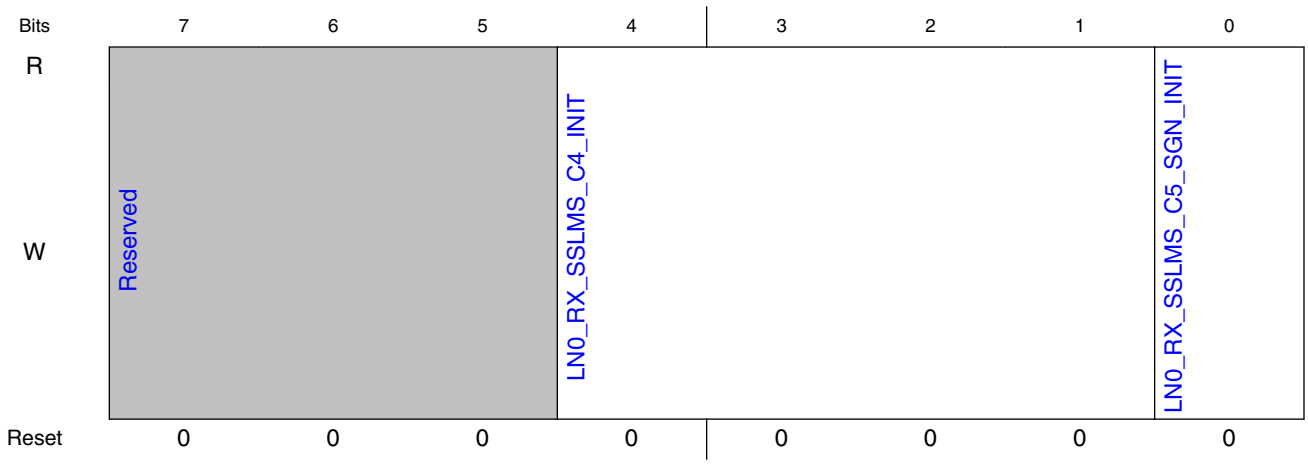
Field	Function
7-6 —	Reserved
5-1 LN0_RX_SSLMS_C3_INIT	
0 LN0_RX_SSLMS_C4_SGN_INIT	

11.4.3.1.257 (TRSV_REG082)

11.4.3.1.257.1 Offset

Register	Offset
TRSV_REG082	608h

11.4.3.1.257.2 Diagram



11.4.3.1.257.3 Fields

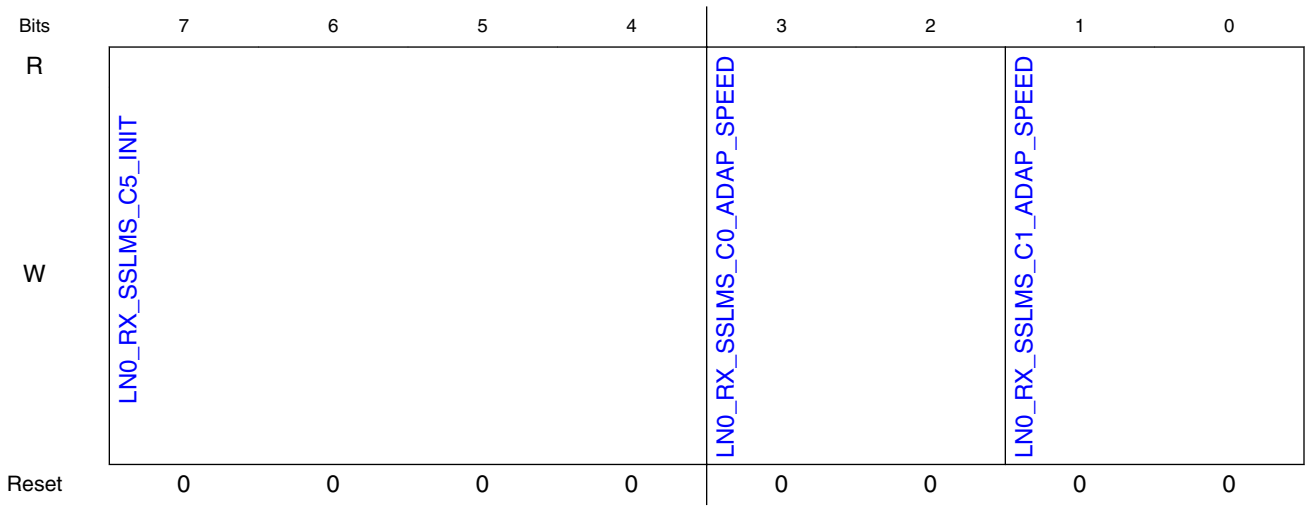
Field	Function
7-5 —	Reserved
4-1 LN0_RX_SSLMS_C4_INIT	
0 LN0_RX_SSLMS_C5_SGN_INIT	

11.4.3.1.258 (TRSV_REG083)

11.4.3.1.258.1 Offset

Register	Offset
TRSV_REG083	60Ch

11.4.3.1.258.2 Diagram



11.4.3.1.258.3 Fields

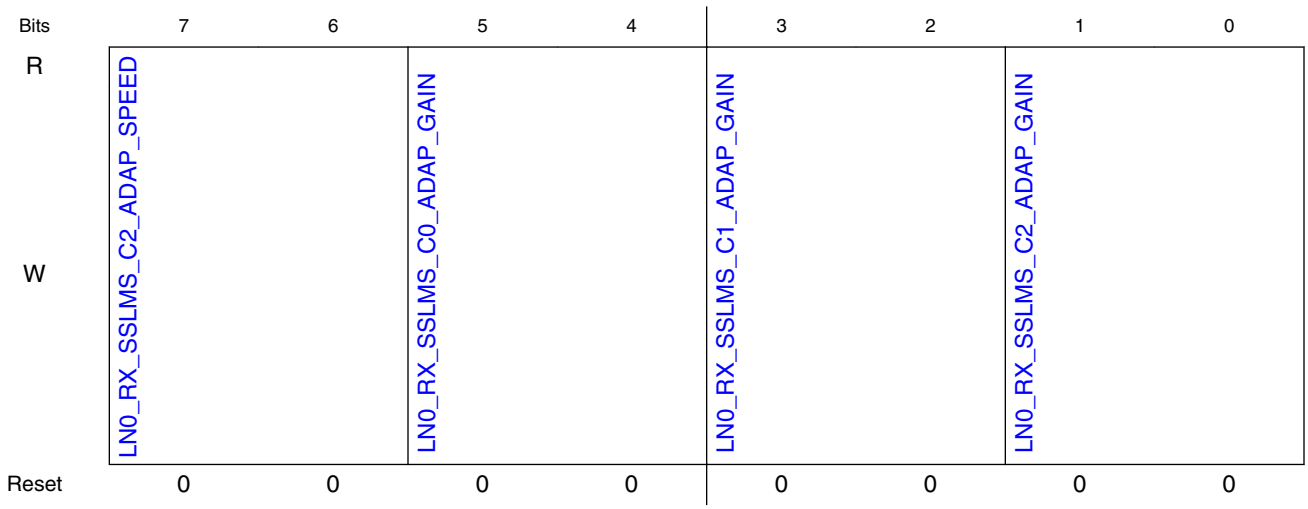
Field	Function
7-4 LN0_RX_SSLMS_C5_INIT	
3-2 LN0_RX_SSLMS_C0_ADAP_SPEED	RX DFE SSLMS c0 adaptation speed selection 00: 1/1024, 01: 1/2048, 10: 1/4096, 11: 1/8192
1-0 LN0_RX_SSLMS_C1_ADAP_SPEED	RX DFE SSLMS c1 adaptation speed selection 00: 1/1024, 01: 1/2048, 10: 1/4096, 11: 1/8192

11.4.3.1.259 (TRSV_REG084)

11.4.3.1.259.1 Offset

Register	Offset
TRSV_REG084	610h

11.4.3.1.259.2 Diagram



11.4.3.1.259.3 Fields

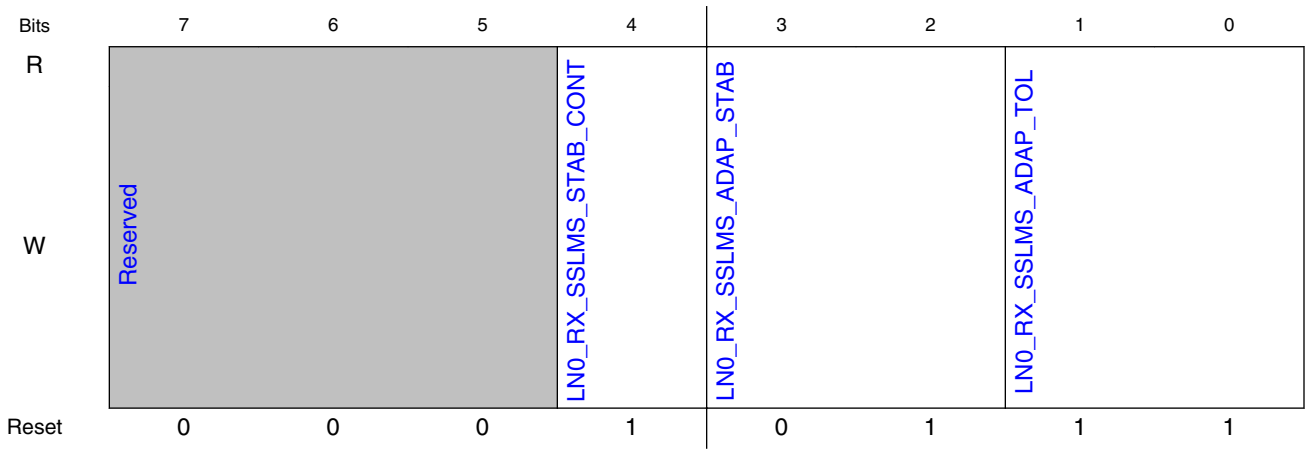
Field	Function
7-6 LN0_RX_SSLMS_C2_ADAP_SPEED	RX DFE SSLMS c2 adaptation speed selection 00: 1/1024, 01: 1/2048, 10: 1/4096, 11: 1/8192
5-4 LN0_RX_SSLMS_C0_ADAP_GAIN	
3-2 LN0_RX_SSLMS_C1_ADAP_GAIN	
1-0 LN0_RX_SSLMS_C2_ADAP_GAIN	

11.4.3.1.260 (TRSV_REG085)

11.4.3.1.260.1 Offset

Register	Offset
TRSV_REG085	614h

11.4.3.1.260.2 Diagram



11.4.3.1.260.3 Fields

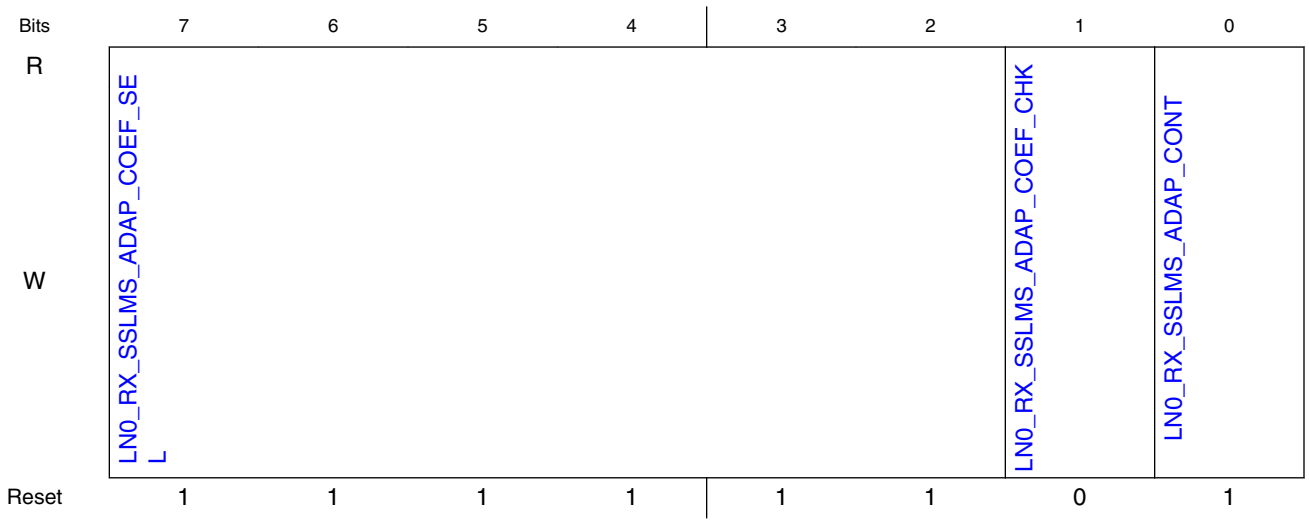
Field	Function
7-5 —	Reserved
4 LN0_RX_SSLMS_STAB_CONT	
3-2 LN0_RX_SSLMS_ADAP_STAB	
1-0 LN0_RX_SSLMS_ADAP_TOL	

11.4.3.1.261 (TRSV_REG086)

11.4.3.1.261.1 Offset

Register	Offset
TRSV_REG086	618h

11.4.3.1.261.2 Diagram



11.4.3.1.261.3 Fields

Field	Function
7-2 LN0_RX_SSLMS_ADAP_COEF_SEL	
1 LN0_RX_SSLMS_ADAP_COEF_CHK	
0 LN0_RX_SSLMS_ADAP_CONT	

11.4.3.1.262 (TRSV_REG087)

11.4.3.1.262.1 Offset

Register	Offset
TRSV_REG087	61Ch

11.4.3.1.262.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_RX_SSLMS_ADAP_TIMEOUT_SEL		LN0_RX_SSLMS_ADAP_TIMEOUT_EN	LN0_RX_SSLMS_RSTN	LN0_OVRD_RX_SSLMS_ADAP_EN	LN0_RX_SSLMS_ADAP_EN	LN0_OVRD_RX_SSLMS_ADAP_HOLD	LN0_RX_SSLMS_ADAP_HOLD
W								
Reset	0	0	1	1	0	0	0	0

11.4.3.1.262.3 Fields

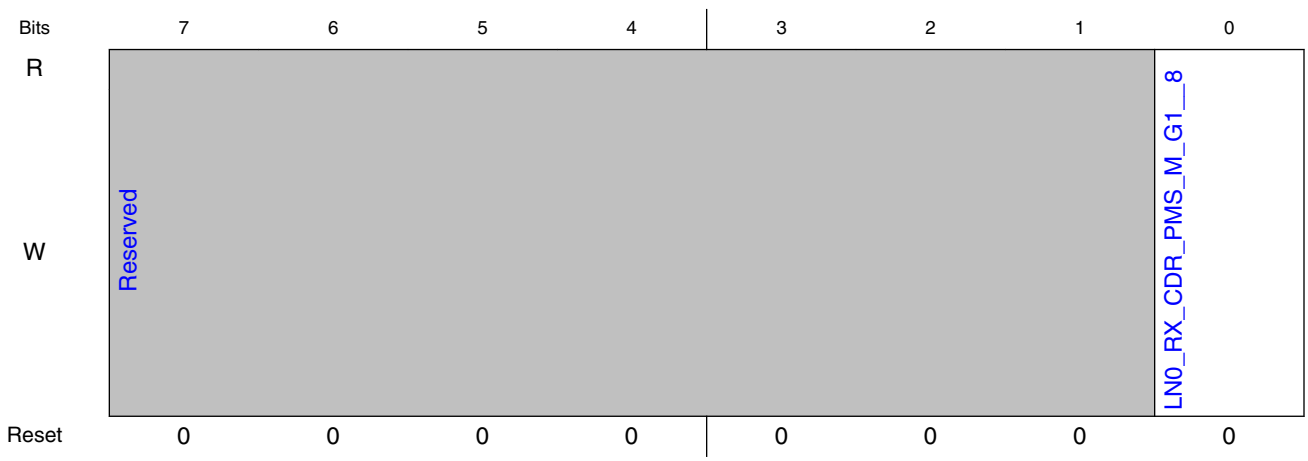
Field	Function
7-6 LN0_RX_SSLMS_ADAP_TIMEOUT_SEL	
5 LN0_RX_SSLMS_ADAP_TIMEOUT_EN	
4 LN0_RX_SSLMS_RSTN	
3 LN0_OVRD_RX_SSLMS_ADAP_EN	Override enable for rx_sslms_adap_en
2 LN0_RX_SSLMS_ADAP_EN	
1 LN0_OVRD_RX_SSLMS_ADAP_HOLD	Override enable for rx_sslms_adap_hold
0 LN0_RX_SSLMS_ADAP_HOLD	

11.4.3.1.263 (TRSV_REG088)

11.4.3.1.263.1 Offset

Register	Offset
TRSV_REG088	620h

11.4.3.1.263.2 Diagram



11.4.3.1.263.3 Fields

Field	Function
7-1 —	Reserved
0 LN0_RX_CDR_PMS_M_G1__8	

11.4.3.1.264 (TRSV_REG089)

11.4.3.1.264.1 Offset

Register	Offset
TRSV_REG089	624h

11.4.3.1.264.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_RX_CDR_PMS_M_G1__7_0							
Reset	1	1	0	0	1	0	0	0

11.4.3.1.264.3 Fields

Field	Function
7-0 LN0_RX_CDR_PMS_M_G1__7_0	

11.4.3.1.265 (TRSV_REG08A)

11.4.3.1.265.1 Offset

Register	Offset
TRSV_REG08A	628h

11.4.3.1.265.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	Reserved							LN0_RX_CDR_PMS_M_G2__8
Reset	0	0	0	0	0	0	0	0

11.4.3.1.265.3 Fields

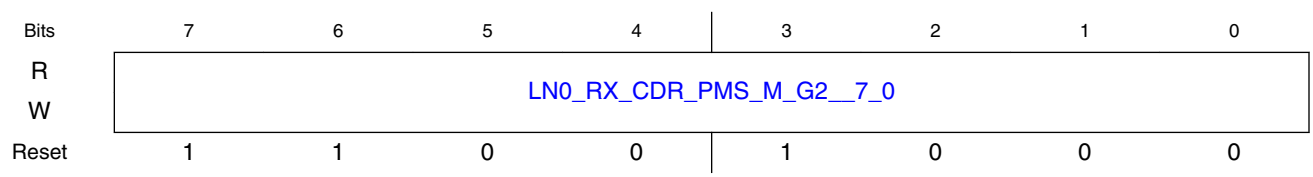
Field	Function
7-1 —	Reserved
0 LN0_RX_CDR_PMS_M_G2__8	

11.4.3.1.266 (TRSV_REG08B)

11.4.3.1.266.1 Offset

Register	Offset
TRSV_REG08B	62Ch

11.4.3.1.266.2 Diagram



11.4.3.1.266.3 Fields

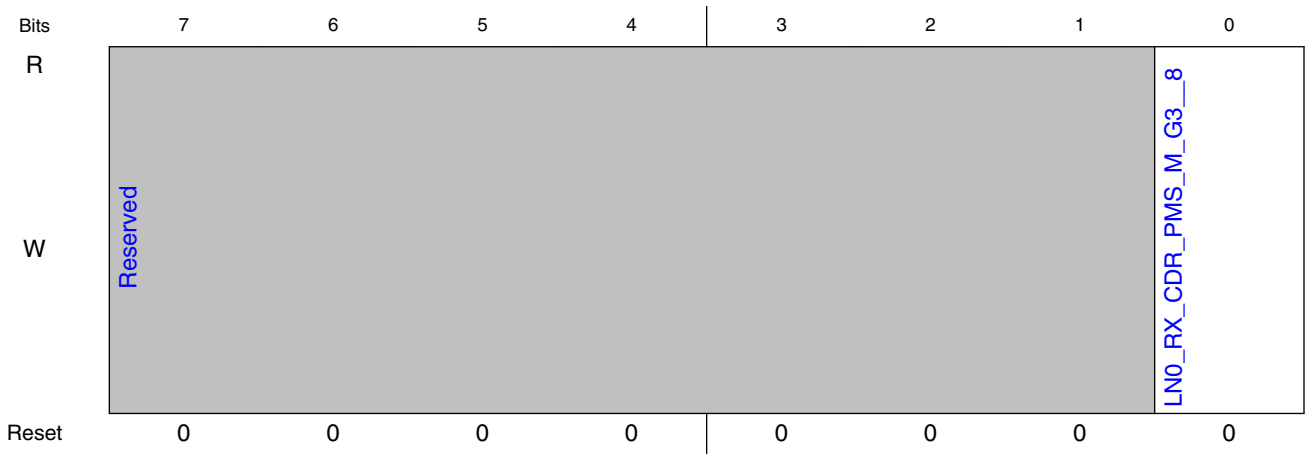
Field	Function
7-0 LN0_RX_CDR_PMS_M_G2__7_0	

11.4.3.1.267 (TRSV_REG08C)

11.4.3.1.267.1 Offset

Register	Offset
TRSV_REG08C	630h

11.4.3.1.267.2 Diagram



11.4.3.1.267.3 Fields

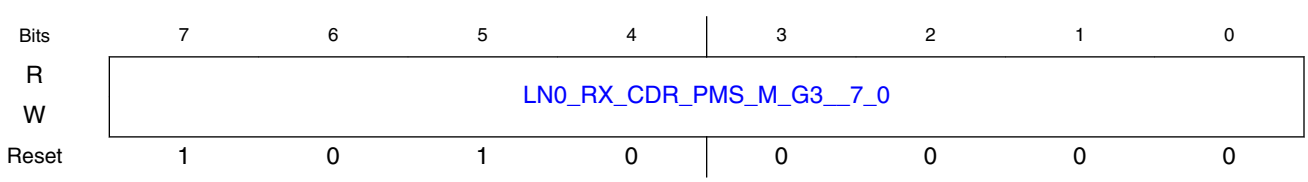
Field	Function
7-1 —	Reserved
0 LN0_RX_CDR_PMS_M_G3__8	

11.4.3.1.268 (TRSV_REG08D)

11.4.3.1.268.1 Offset

Register	Offset
TRSV_REG08D	634h

11.4.3.1.268.2 Diagram



11.4.3.1.268.3 Fields

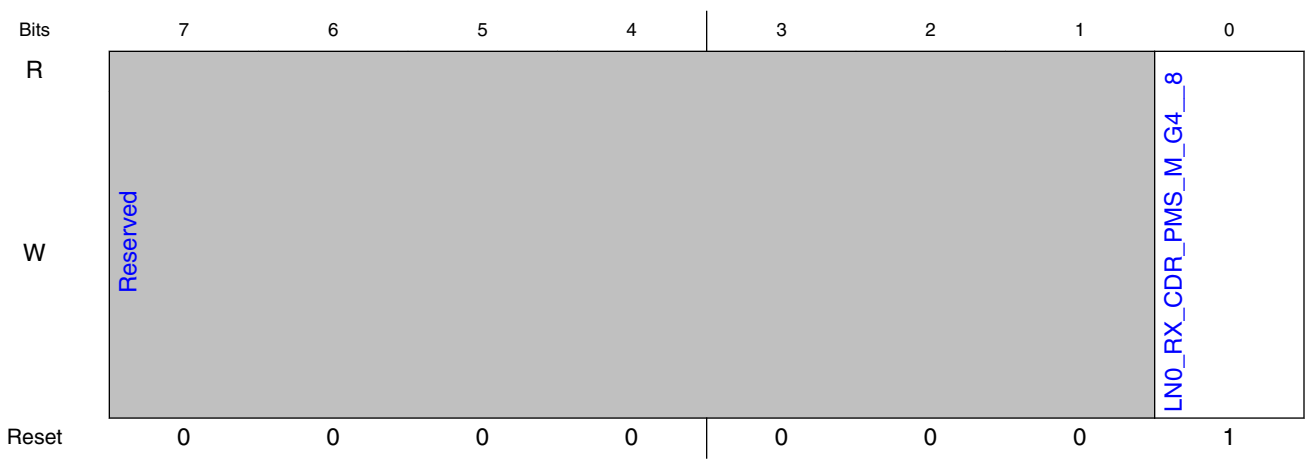
Field	Function
7-0 LN0_RX_CDR_PMS_M_G3__7_0	

11.4.3.1.269 (TRSV_REG08E)

11.4.3.1.269.1 Offset

Register	Offset
TRSV_REG08E	638h

11.4.3.1.269.2 Diagram



11.4.3.1.269.3 Fields

Field	Function
7-1 —	Reserved
0 LN0_RX_CDR_PMS_M_G4__8	

11.4.3.1.270 (TRSV_REG08F)

11.4.3.1.270.1 Offset

Register	Offset
TRSV_REG08F	63Ch

11.4.3.1.270.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_RX_CDR_PMS_M_G4__7_0							
Reset	0	1	0	0	0	0	0	0

11.4.3.1.270.3 Fields

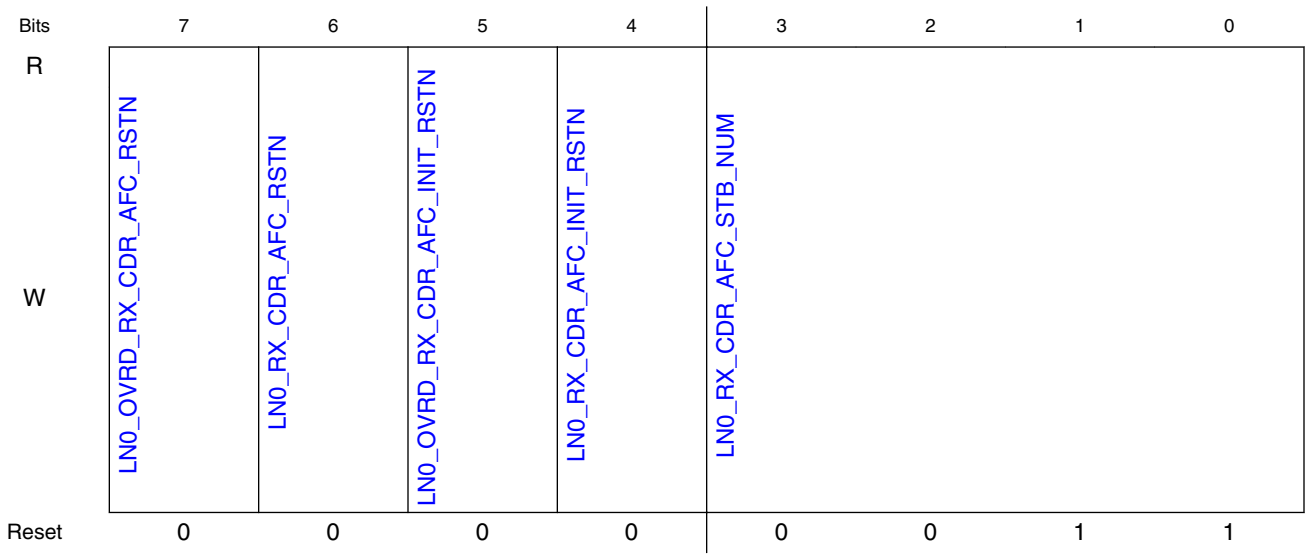
Field	Function
7-0 LN0_RX_CDR_PMS_M_G4__7_0	

11.4.3.1.271 (TRSV_REG090)

11.4.3.1.271.1 Offset

Register	Offset
TRSV_REG090	640h

11.4.3.1.271.2 Diagram



11.4.3.1.271.3 Fields

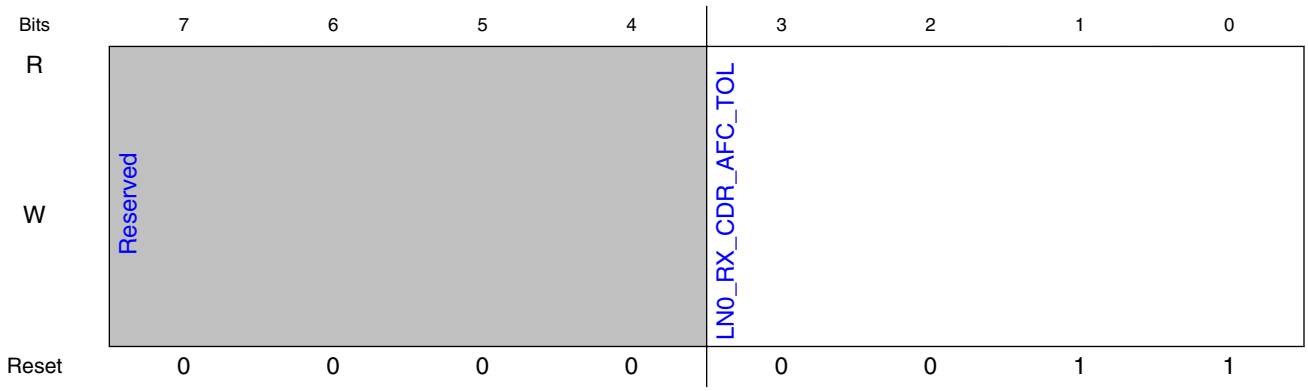
Field	Function
7 LN0_OVRD_RX_CDR_AFC_RSTN	Override enable for rx_cdr_afc_rstn
6 LN0_RX_CDR_AFC_RSTN	
5 LN0_OVRD_RX_CDR_AFC_INIT_RSTN	Override enable for rx_cdr_afc_init_rstn
4 LN0_RX_CDR_AFC_INIT_RSTN	
3-0 LN0_RX_CDR_AFC_STB_NUM	

11.4.3.1.272 (TRSV_REG091)

11.4.3.1.272.1 Offset

Register	Offset
TRSV_REG091	644h

11.4.3.1.272.2 Diagram



11.4.3.1.272.3 Fields

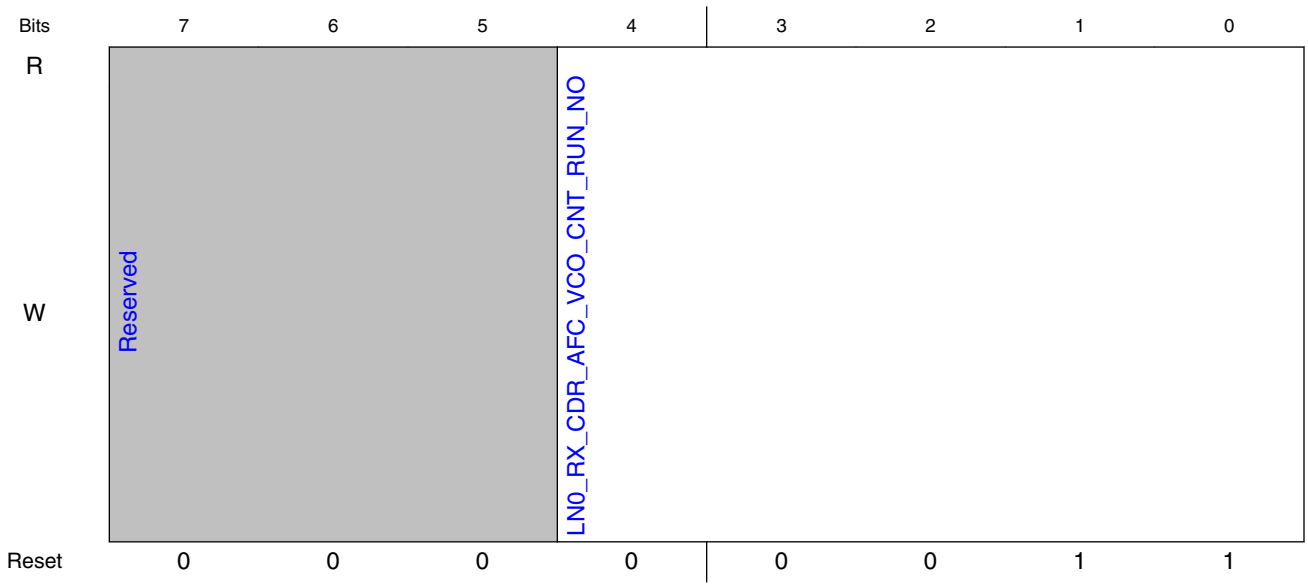
Field	Function
7-4 —	Reserved
3-0 LN0_RX_CDR_AFC_TOL	

11.4.3.1.273 (TRSV_REG092)

11.4.3.1.273.1 Offset

Register	Offset
TRSV_REG092	648h

11.4.3.1.273.2 Diagram



11.4.3.1.273.3 Fields

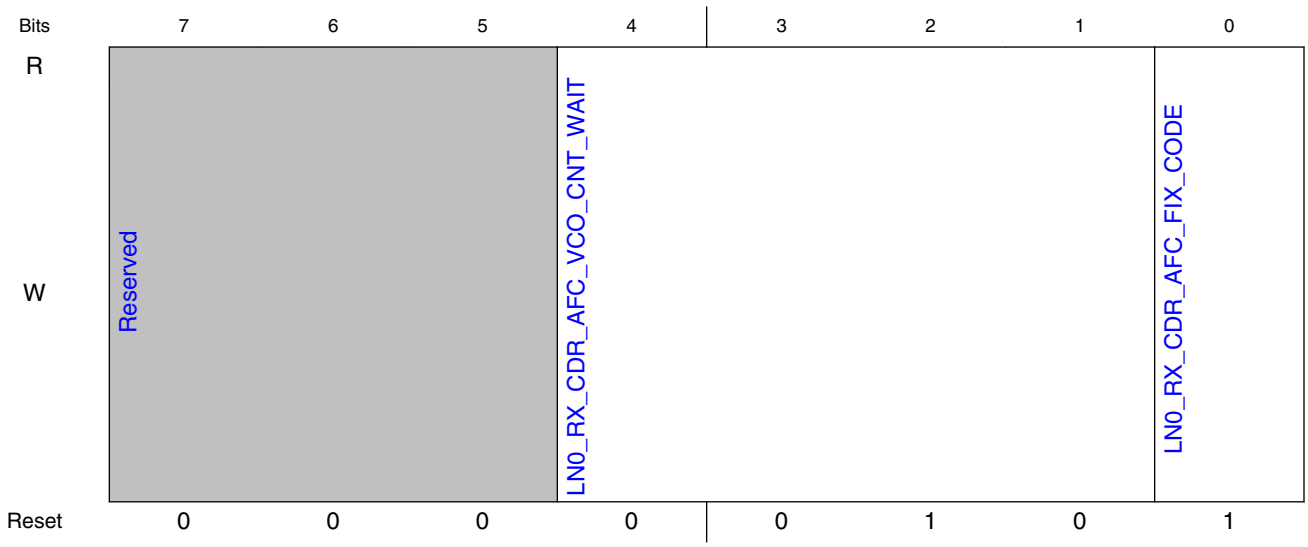
Field	Function
7-5 —	Reserved
4-0 LN0_RX_CDR_AFC_VCO_CNT_RUN_NO	

11.4.3.1.274 (TRSV_REG093)

11.4.3.1.274.1 Offset

Register	Offset
TRSV_REG093	64Ch

11.4.3.1.274.2 Diagram



11.4.3.1.274.3 Fields

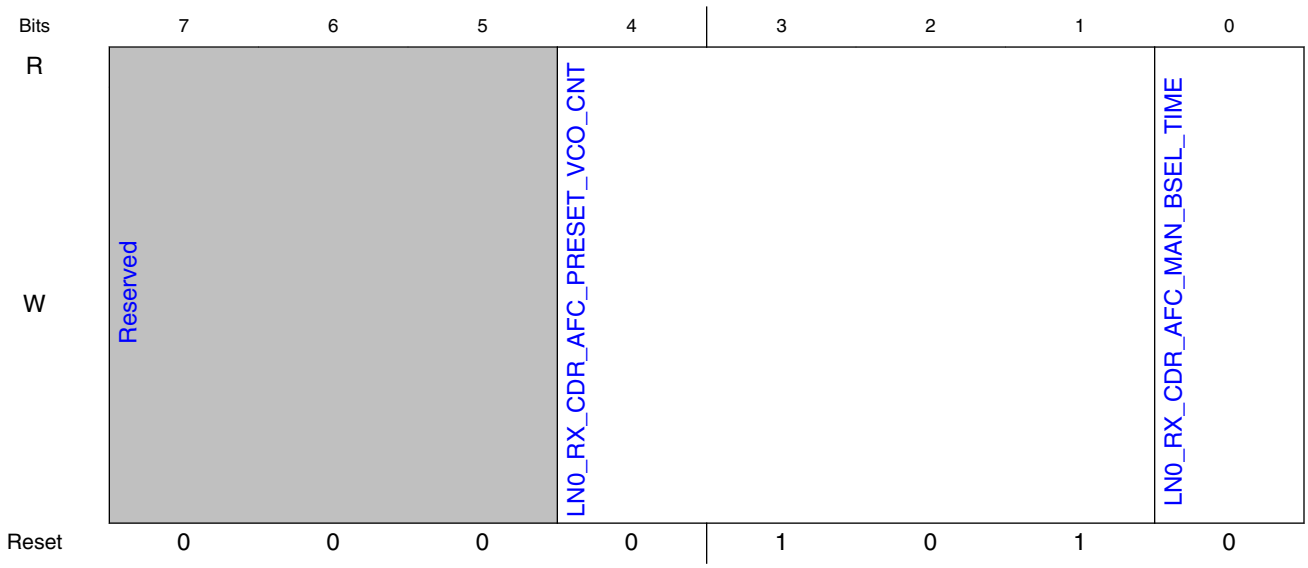
Field	Function
7-5 —	Reserved
4-1 LN0_RX_CDR_AFC_VCO_CNT_WAIT	
0 LN0_RX_CDR_AFC_FIX_CODE	

11.4.3.1.275 (TRSV_REG094)

11.4.3.1.275.1 Offset

Register	Offset
TRSV_REG094	650h

11.4.3.1.275.2 Diagram



11.4.3.1.275.3 Fields

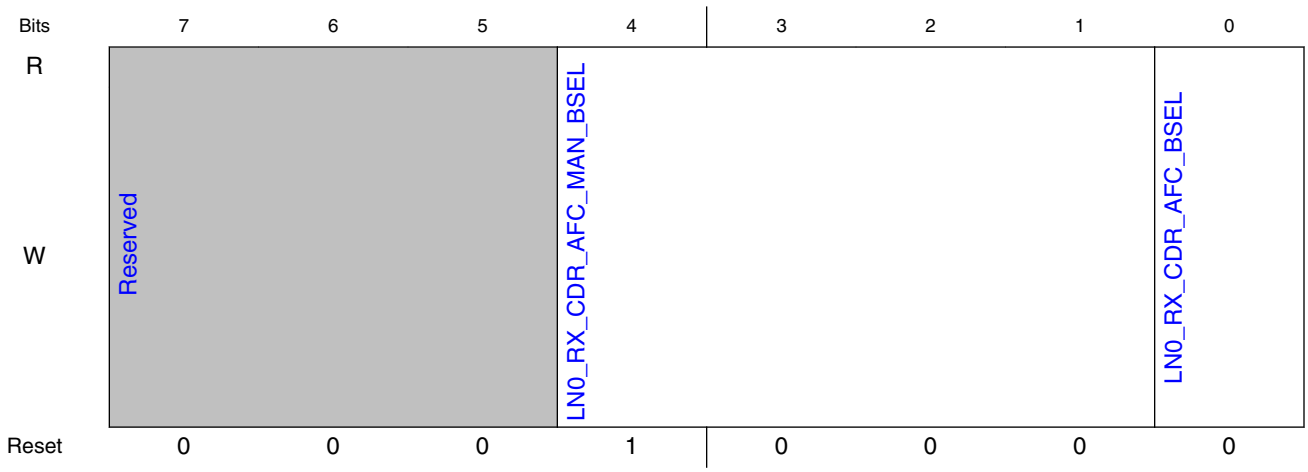
Field	Function
7-5 —	Reserved
4-1 LN0_RX_CDR_AFC_PRESET_VCO_CNT	
0 LN0_RX_CDR_AFC_MAN_BSEL_TIME	

11.4.3.1.276 (TRSV_REG095)

11.4.3.1.276.1 Offset

Register	Offset
TRSV_REG095	654h

11.4.3.1.276.2 Diagram



11.4.3.1.276.3 Fields

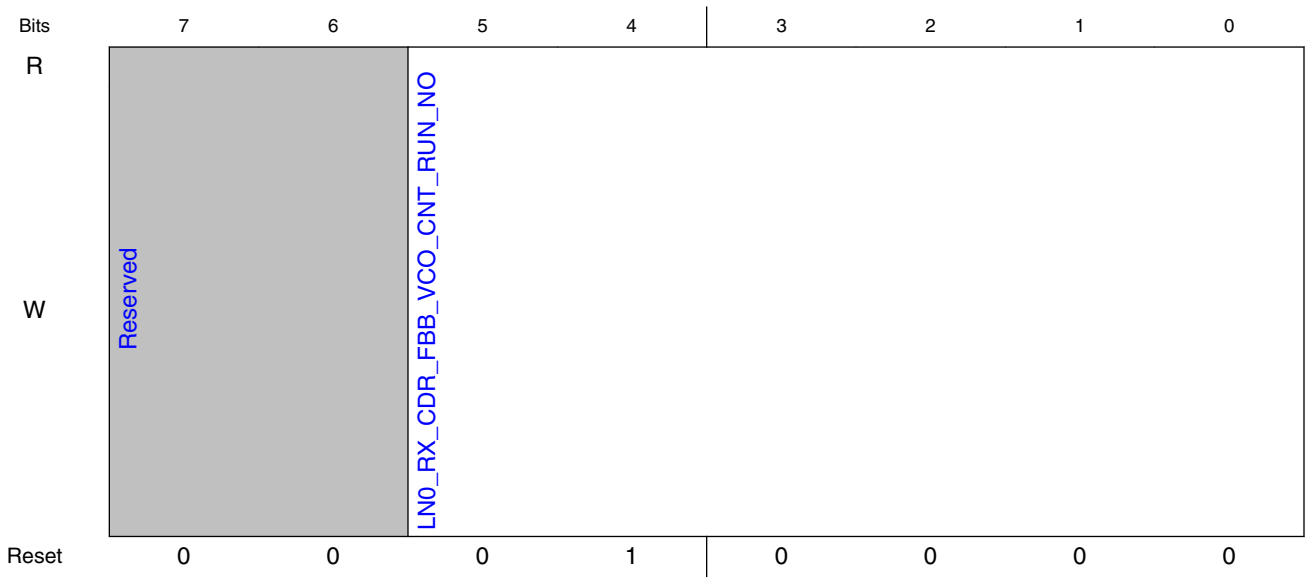
Field	Function
7-5 —	Reserved
4-1 LN0_RX_CDR_AFC_MAN_BSEL	
0 LN0_RX_CDR_AFC_BSEL	

11.4.3.1.277 (TRSV_REG096)

11.4.3.1.277.1 Offset

Register	Offset
TRSV_REG096	658h

11.4.3.1.277.2 Diagram



11.4.3.1.277.3 Fields

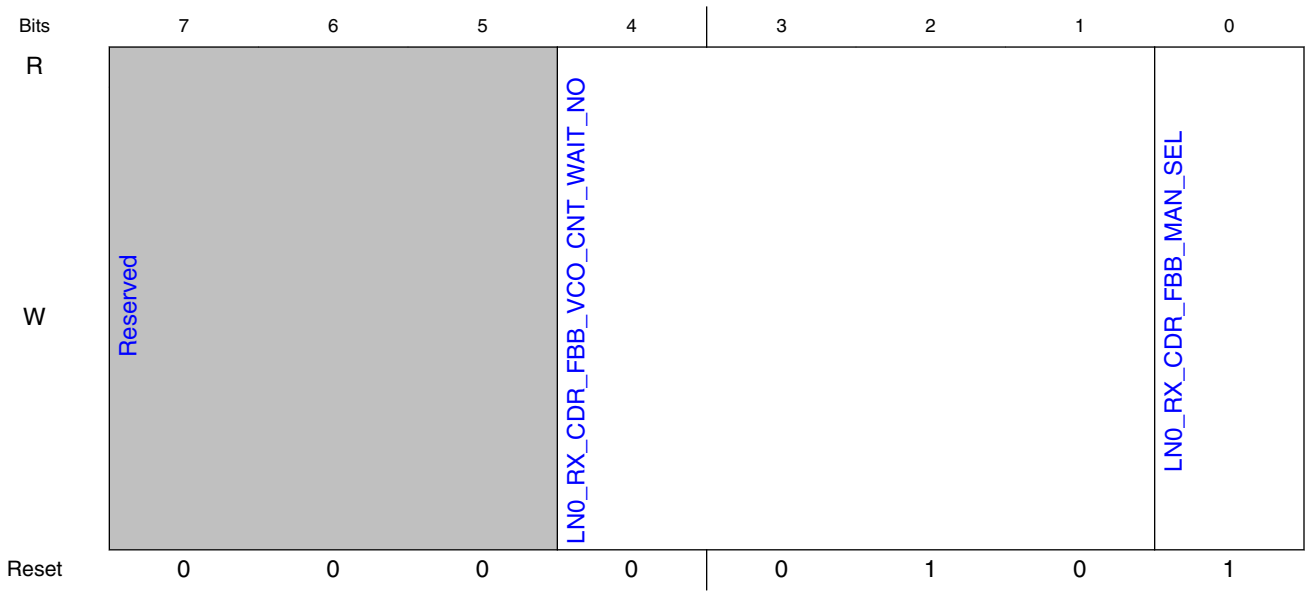
Field	Function
7-6 —	Reserved
5-0 LN0_RX_CDR_FBB_VCO_CNT_RUN_NO	

11.4.3.1.278 (TRSV_REG097)

11.4.3.1.278.1 Offset

Register	Offset
TRSV_REG097	65Ch

11.4.3.1.278.2 Diagram



11.4.3.1.278.3 Fields

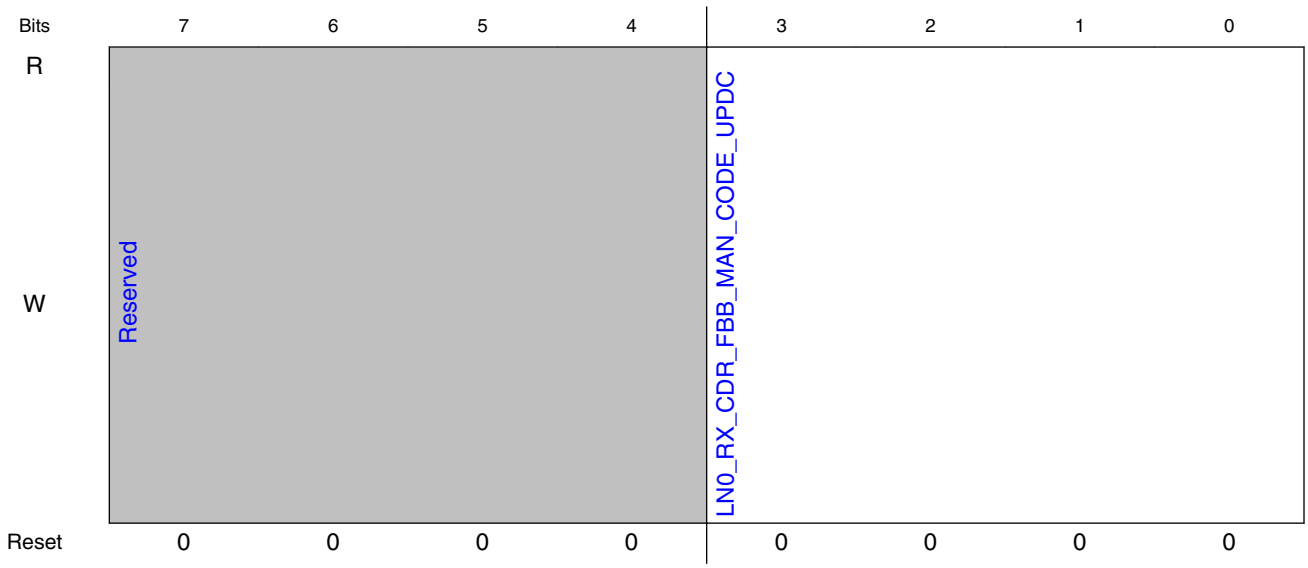
Field	Function
7-5 —	Reserved
4-1 LN0_RX_CDR_FBB_VCO_CNT_WAIT_NO	
0 LN0_RX_CDR_FBB_MAN_SEL	

11.4.3.1.279 (TRSV_REG098)

11.4.3.1.279.1 Offset

Register	Offset
TRSV_REG098	660h

11.4.3.1.279.2 Diagram



11.4.3.1.279.3 Fields

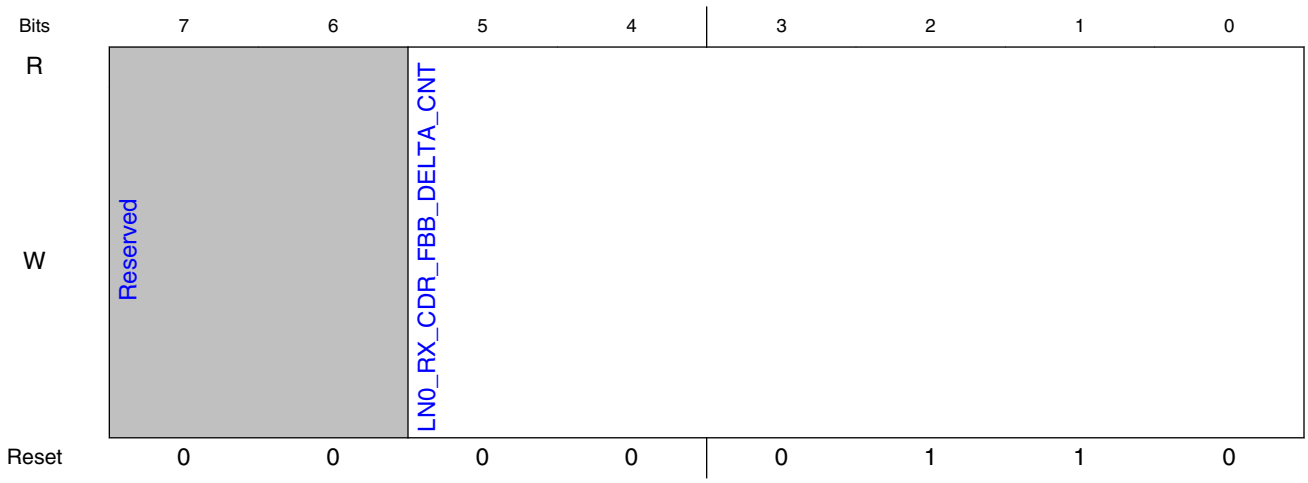
Field	Function
7-4 —	Reserved
3-0 LN0_RX_CDR_FBB_MAN_CODE_UPDC	

11.4.3.1.280 (TRSV_REG099)

11.4.3.1.280.1 Offset

Register	Offset
TRSV_REG099	664h

11.4.3.1.280.2 Diagram



11.4.3.1.280.3 Fields

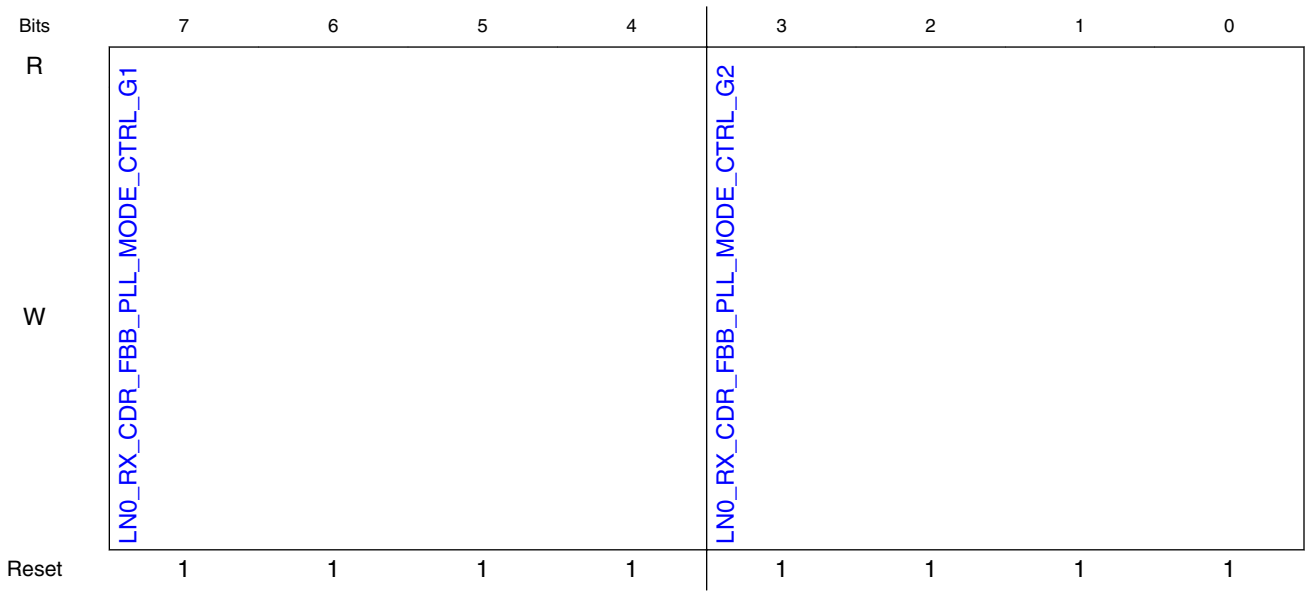
Field	Function
7-6 —	Reserved
5-0 LN0_RX_CDR_FBB_DELTA_CNT	Target delta number in VCO counter in CDR FBB cal mode Delta = fFBB(target) / fref * NRUN If Fvco=5GHz, fref=25MHz, target fFBB=9MHz, and NRUN=16, then delta = 9/25*17~=6

11.4.3.1.281 (TRSV_REG09A)

11.4.3.1.281.1 Offset

Register	Offset
TRSV_REG09A	668h

11.4.3.1.281.2 Diagram



11.4.3.1.281.3 Fields

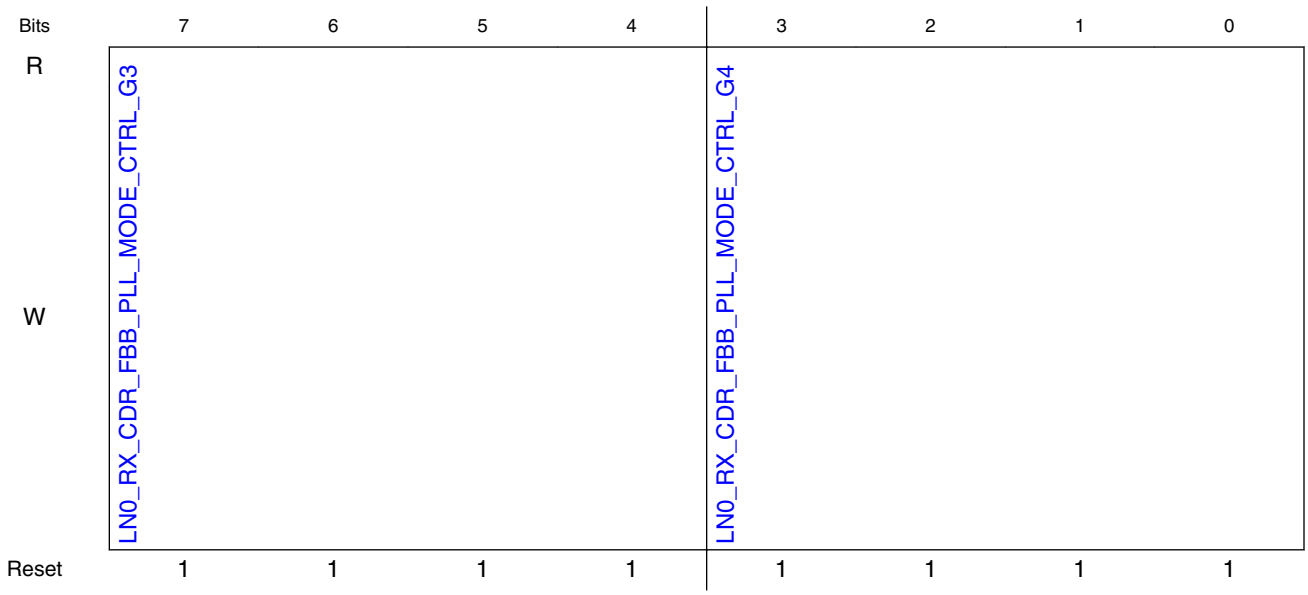
Field	Function
7-4 LN0_RX_CDR_FBB_PLL_MODE_CTRL_G1	[GEN1] RX CDR BBVCO FBB gain control in PLL mode 0000: Lowest gain ... 1111: Highest gain
3-0 LN0_RX_CDR_FBB_PLL_MODE_CTRL_G2	[GEN2]

11.4.3.1.282 (TRSV_REG09B)

11.4.3.1.282.1 Offset

Register	Offset
TRSV_REG09B	66Ch

11.4.3.1.282.2 Diagram



11.4.3.1.282.3 Fields

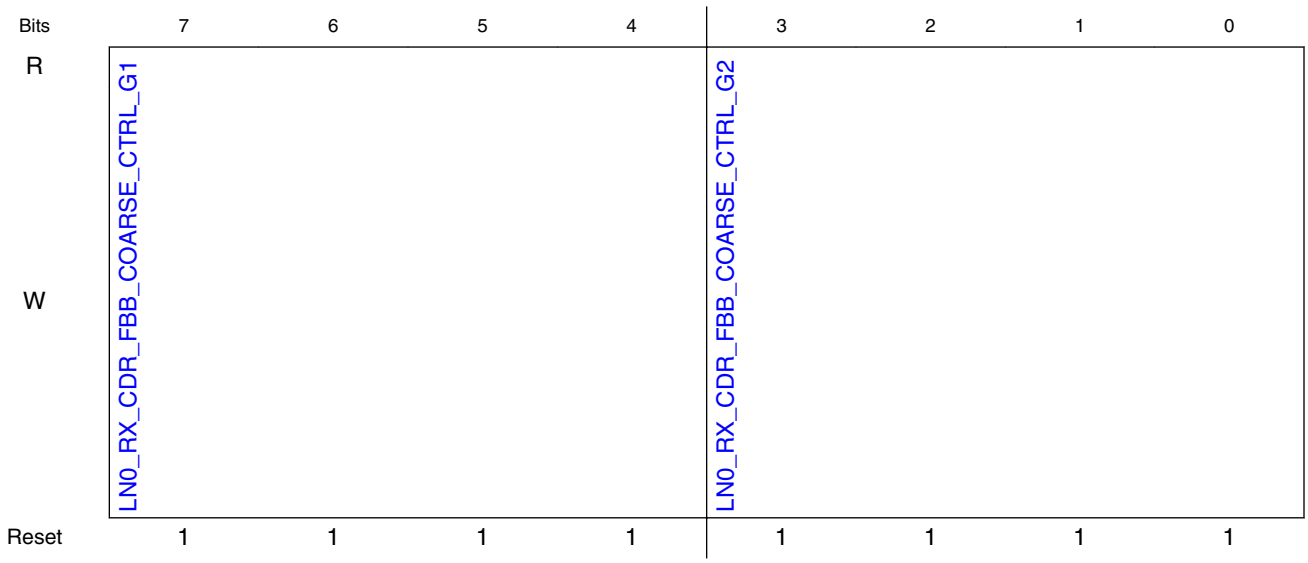
Field	Function
7-4 LN0_RX_CDR_FBB_PLL_MODE_CTRL_G3	[GEN3]
3-0 LN0_RX_CDR_FBB_PLL_MODE_CTRL_G4	[GEN4]

11.4.3.1.283 (TRSV_REG09C)

11.4.3.1.283.1 Offset

Register	Offset
TRSV_REG09C	670h

11.4.3.1.283.2 Diagram



11.4.3.1.283.3 Fields

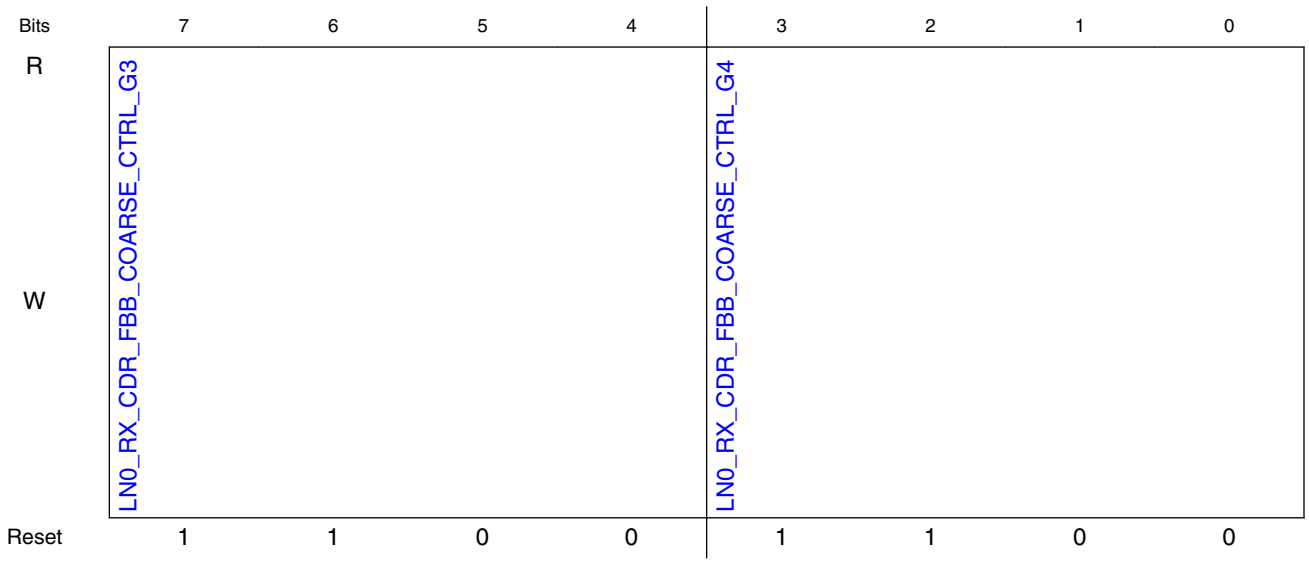
Field	Function
7-4 LN0_RX_CDR_FBB_COARSE_CTRL_G1	[GEN1] RX CDR BBVCO FBB gain control in coarse mode (high bandwidth) 0000: Lowest gain ... 1111: Highest gain
3-0 LN0_RX_CDR_FBB_COARSE_CTRL_G2	[GEN2]

11.4.3.1.284 (TRSV_REG09D)

11.4.3.1.284.1 Offset

Register	Offset
TRSV_REG09D	674h

11.4.3.1.284.2 Diagram



11.4.3.1.284.3 Fields

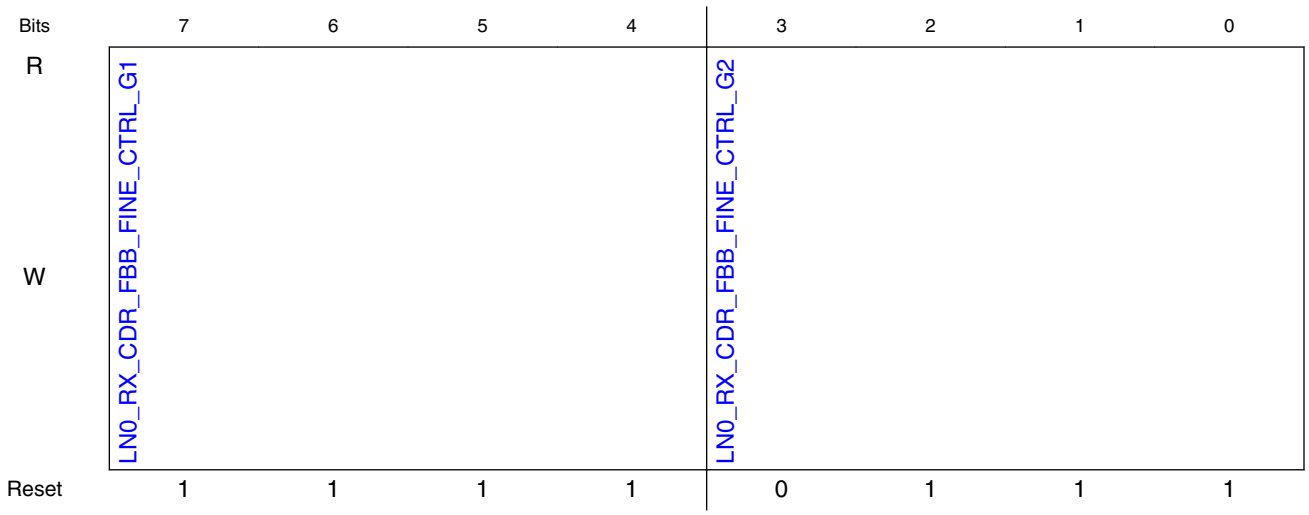
Field	Function
7-4 LN0_RX_CDR_FBB_COARSE_CTRL_G3	[GEN3]
3-0 LN0_RX_CDR_FBB_COARSE_CTRL_G4	[GEN4]

11.4.3.1.285 (TRSV_REG09E)

11.4.3.1.285.1 Offset

Register	Offset
TRSV_REG09E	678h

11.4.3.1.285.2 Diagram



11.4.3.1.285.3 Fields

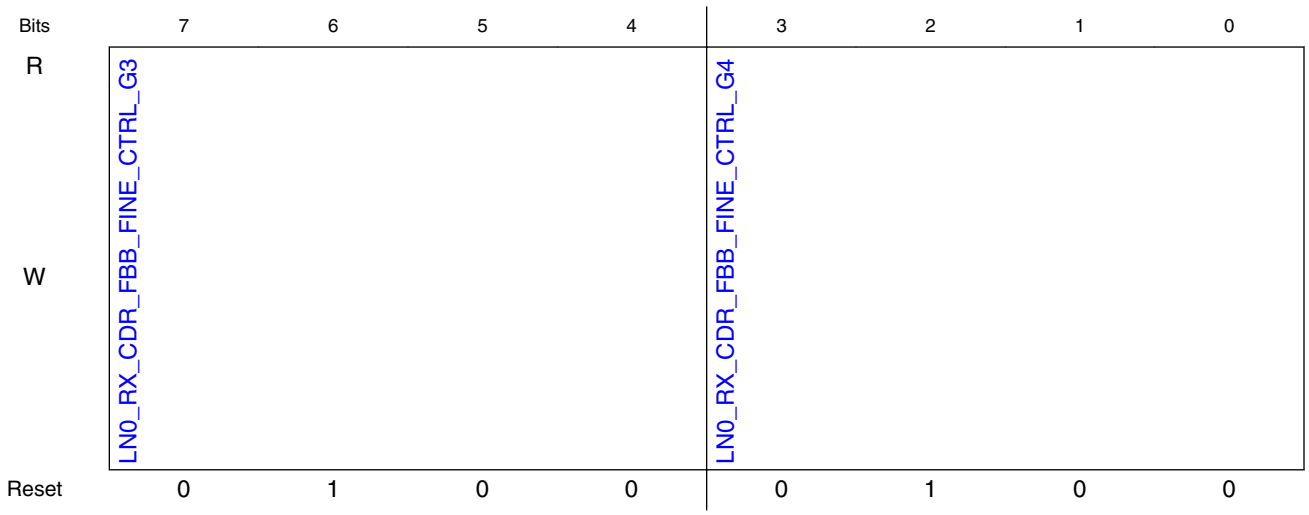
Field	Function
7-4 LN0_RX_CDR_FBB_FINE_CTRL_G1	[GEN1] RX CDR BBVCO FBB gain control in fine mode (low bandwidth) 0000: Lowest gain ... 1111: Highest gain
3-0 LN0_RX_CDR_FBB_FINE_CTRL_G2	[GEN2]

11.4.3.1.286 (TRSV_REG09F)

11.4.3.1.286.1 Offset

Register	Offset
TRSV_REG09F	67Ch

11.4.3.1.286.2 Diagram



11.4.3.1.286.3 Fields

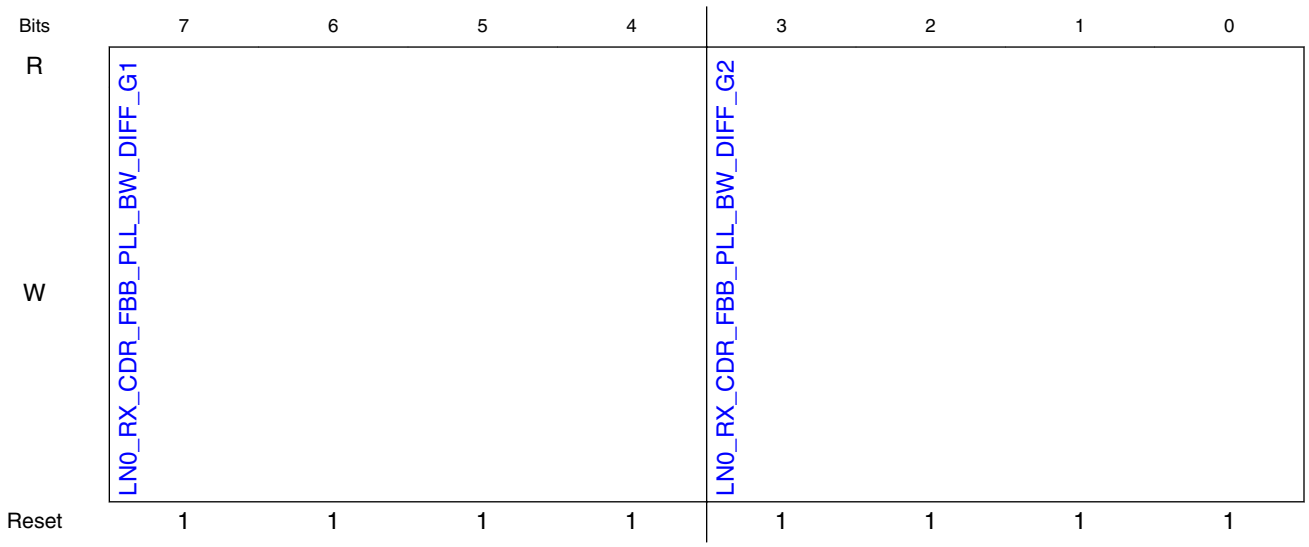
Field	Function
7-4 LN0_RX_CDR_FBB_FINE_CTRL_G3	[GEN3]
3-0 LN0_RX_CDR_FBB_FINE_CTRL_G4	[GEN4]

11.4.3.1.287 (TRSV_REG0A0)

11.4.3.1.287.1 Offset

Register	Offset
TRSV_REG0A0	680h

11.4.3.1.287.2 Diagram



11.4.3.1.287.3 Fields

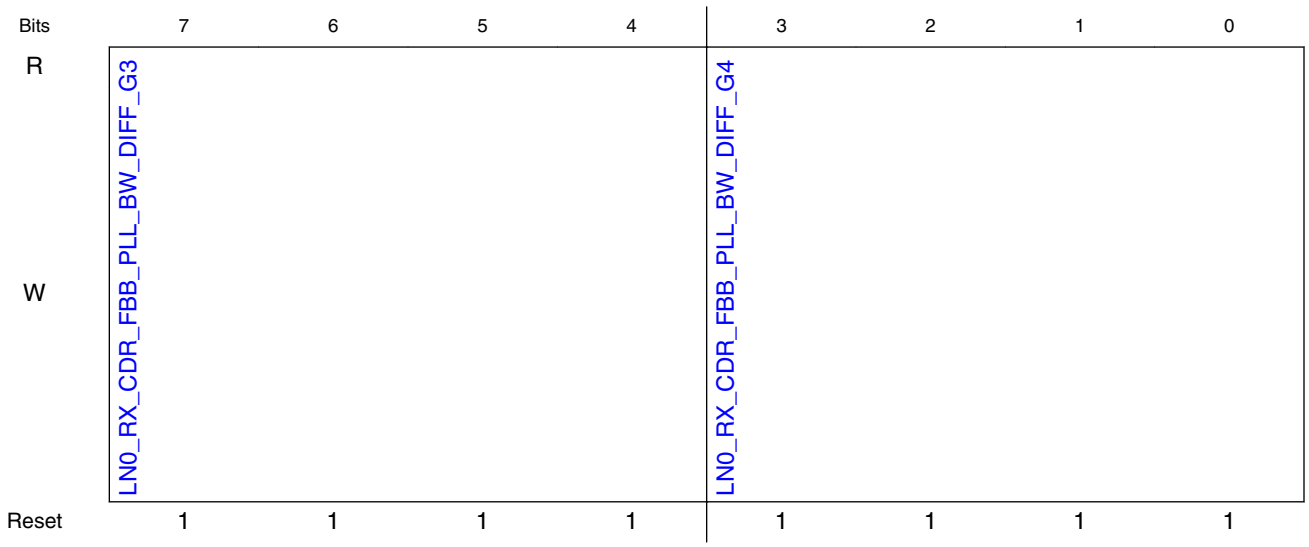
Field	Function
7-4 LN0_RX_CDR_FBB_PLL_BW_DIFF_G1	[GEN4]
3-0 LN0_RX_CDR_FBB_PLL_BW_DIFF_G2	[GEN4]

11.4.3.1.288 (TRSV_REG0A1)

11.4.3.1.288.1 Offset

Register	Offset
TRSV_REG0A1	684h

11.4.3.1.288.2 Diagram



11.4.3.1.288.3 Fields

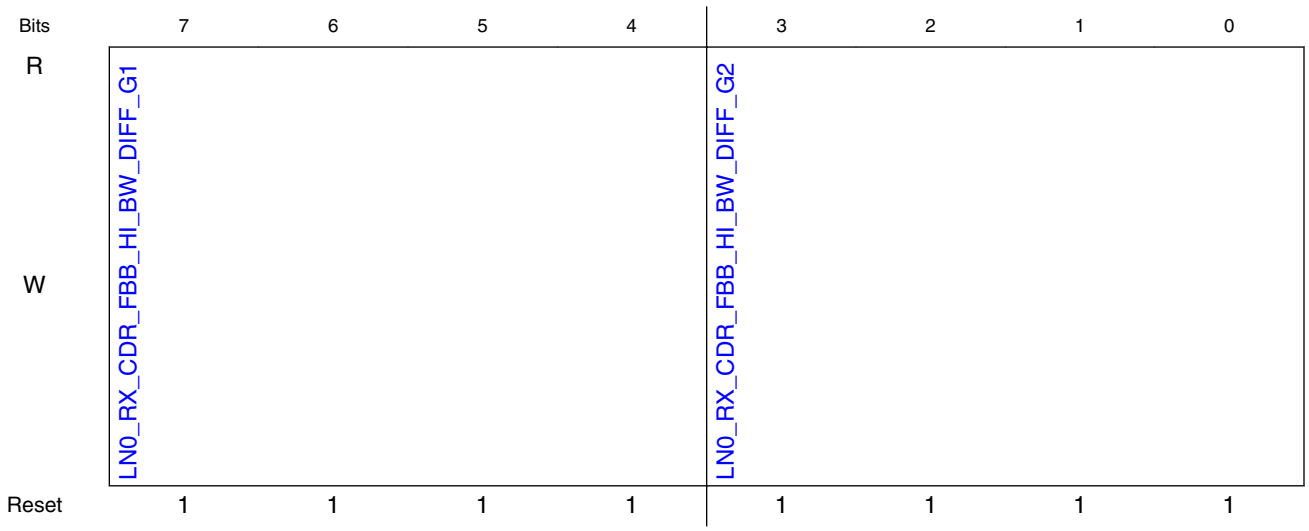
Field	Function
7-4 LN0_RX_CDR_FBB_PLL_BW_DIFF_G3	[GEN4]
3-0 LN0_RX_CDR_FBB_PLL_BW_DIFF_G4	[GEN4]

11.4.3.1.289 (TRSV_REG0A2)

11.4.3.1.289.1 Offset

Register	Offset
TRSV_REG0A2	688h

11.4.3.1.289.2 Diagram



11.4.3.1.289.3 Fields

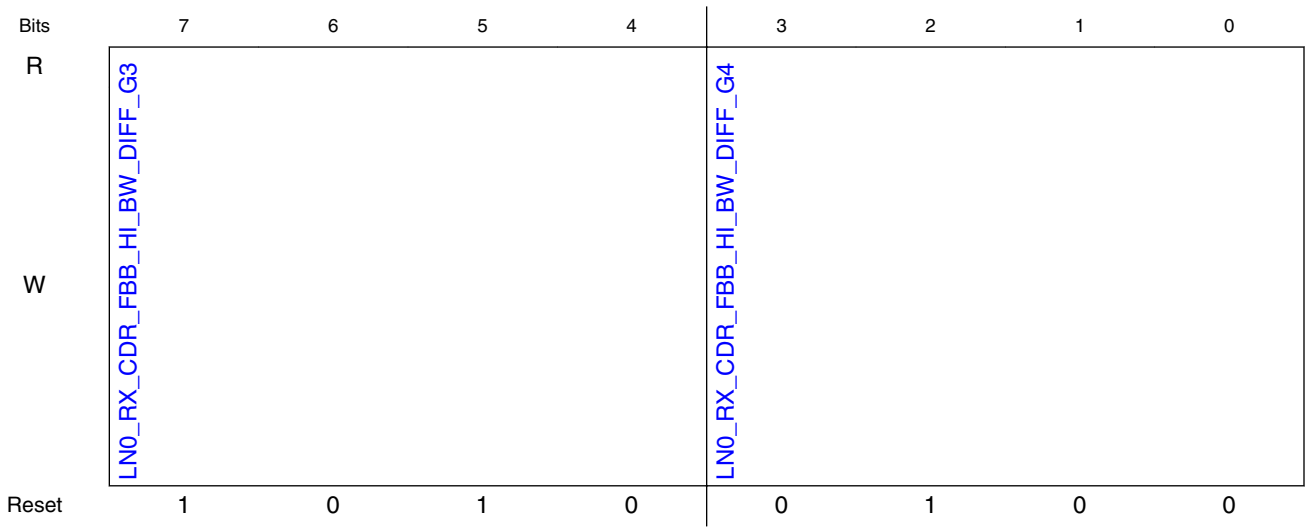
Field	Function
7-4 LN0_RX_CDR_FBB_HI_BW_DIFF_G1	[GEN4]
3-0 LN0_RX_CDR_FBB_HI_BW_DIFF_G2	[GEN4]

11.4.3.1.290 (TRSV_REG0A3)

11.4.3.1.290.1 Offset

Register	Offset
TRSV_REG0A3	68Ch

11.4.3.1.290.2 Diagram



11.4.3.1.290.3 Fields

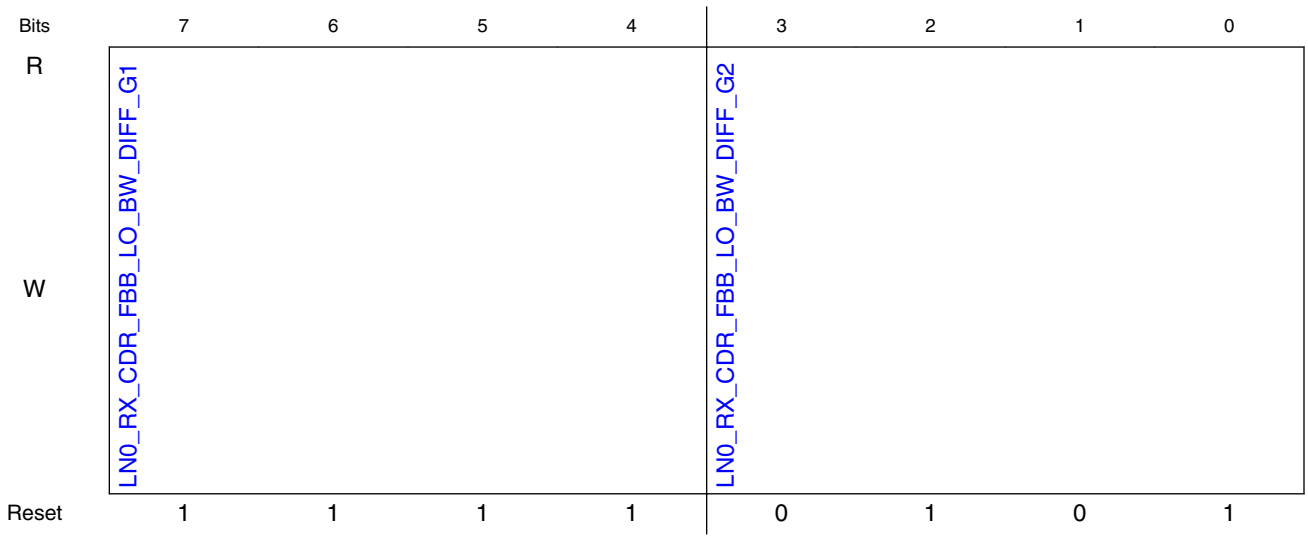
Field	Function
7-4 LN0_RX_CDR_FBB_HI_BW_DIFF_G3	[GEN4]
3-0 LN0_RX_CDR_FBB_HI_BW_DIFF_G4	[GEN4]

11.4.3.1.291 (TRSV_REG0A4)

11.4.3.1.291.1 Offset

Register	Offset
TRSV_REG0A4	690h

11.4.3.1.291.2 Diagram



11.4.3.1.291.3 Fields

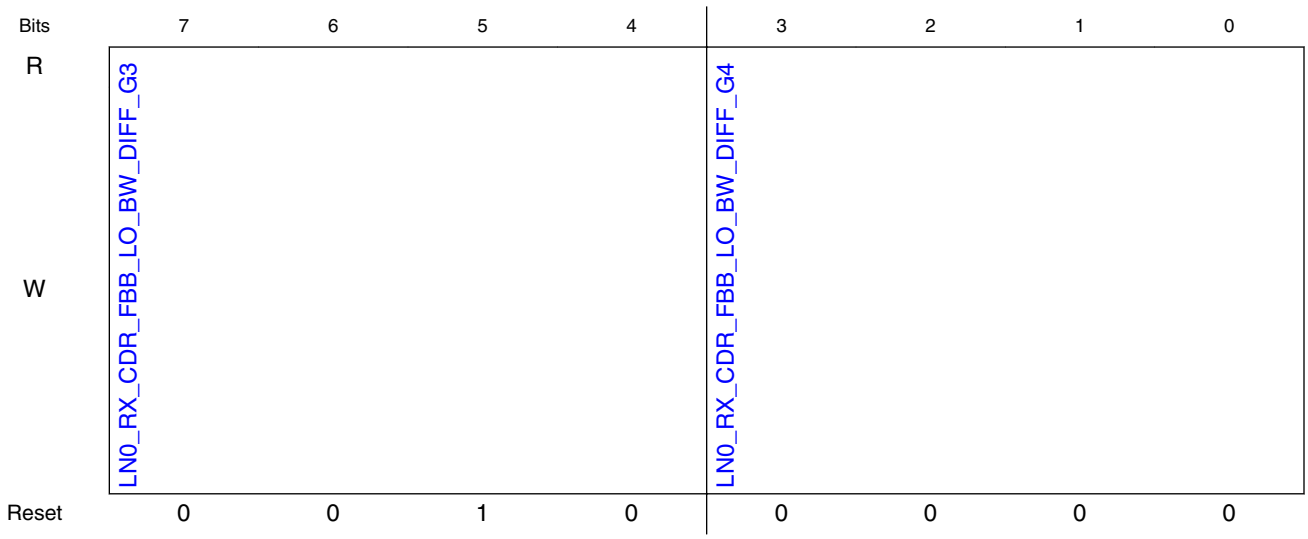
Field	Function
7-4 LN0_RX_CDR_FBB_LO_BW_DIFF_G1	[GEN4]
3-0 LN0_RX_CDR_FBB_LO_BW_DIFF_G2	[GEN4]

11.4.3.1.292 (TRSV_REG0A5)

11.4.3.1.292.1 Offset

Register	Offset
TRSV_REG0A5	694h

11.4.3.1.292.2 Diagram



11.4.3.1.292.3 Fields

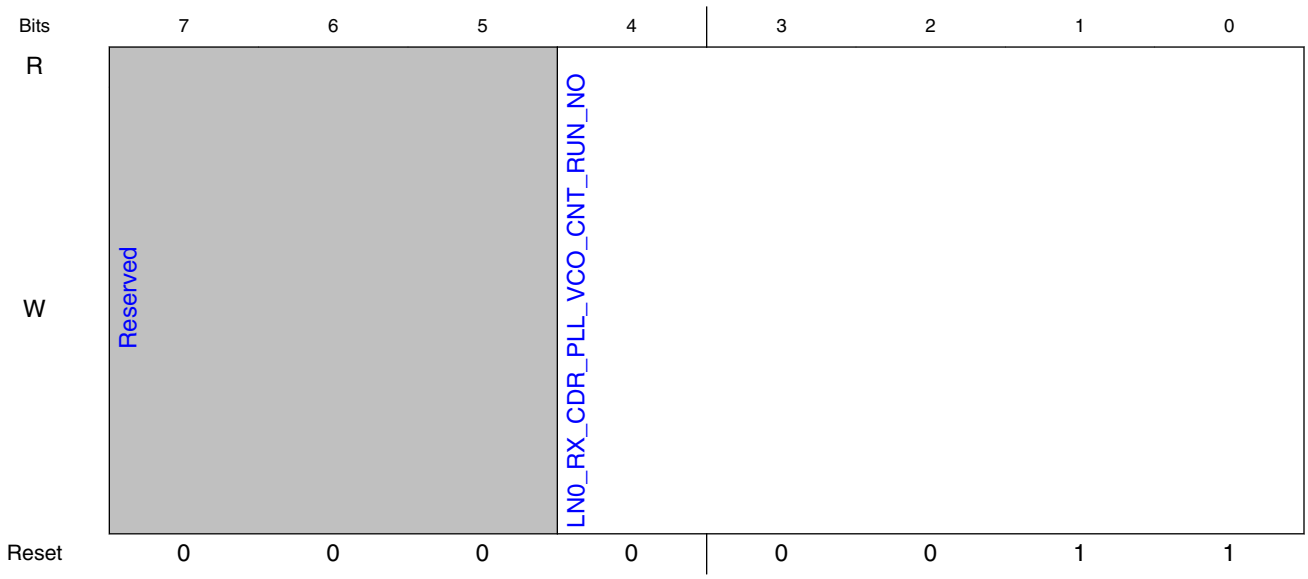
Field	Function
7-4 LN0_RX_CDR_FBB_LO_BW_DIFF_G3	[GEN4]
3-0 LN0_RX_CDR_FBB_LO_BW_DIFF_G4	[GEN4]

11.4.3.1.293 (TRSV_REG0A6)

11.4.3.1.293.1 Offset

Register	Offset
TRSV_REG0A6	698h

11.4.3.1.293.2 Diagram



11.4.3.1.293.3 Fields

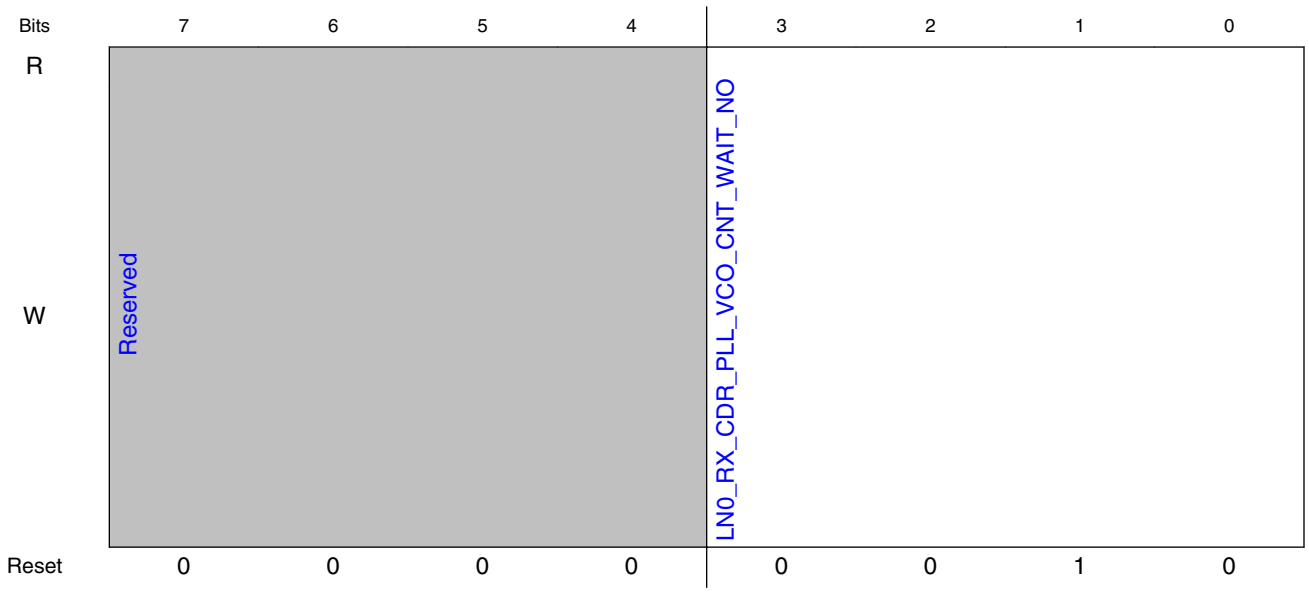
Field	Function
7-5 —	Reserved
4-0 LN0_RX_CDR_PLL_VCO_CNT_RUN_NO	

11.4.3.1.294 (TRSV_REG0A7)

11.4.3.1.294.1 Offset

Register	Offset
TRSV_REG0A7	69Ch

11.4.3.1.294.2 Diagram



11.4.3.1.294.3 Fields

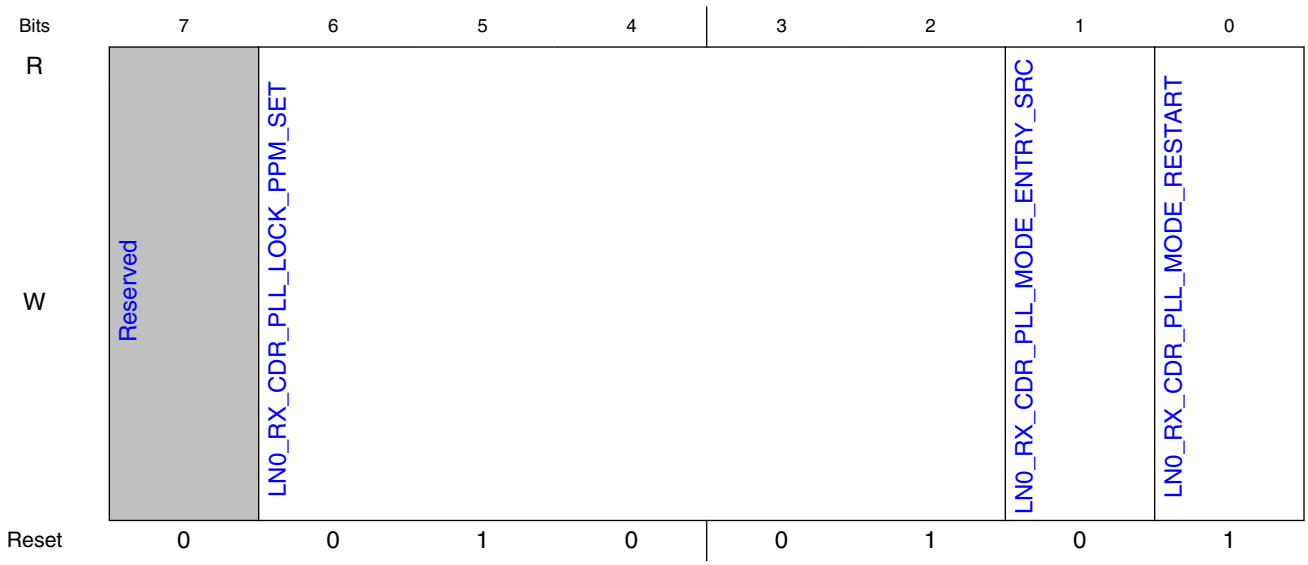
Field	Function
7-4 —	Reserved
3-0 LN0_RX_CDR_PLL_VCO_CNT_WAIT_NO	

11.4.3.1.295 (TRSV_REG0A8)

11.4.3.1.295.1 Offset

Register	Offset
TRSV_REG0A8	6A0h

11.4.3.1.295.2 Diagram



11.4.3.1.295.3 Fields

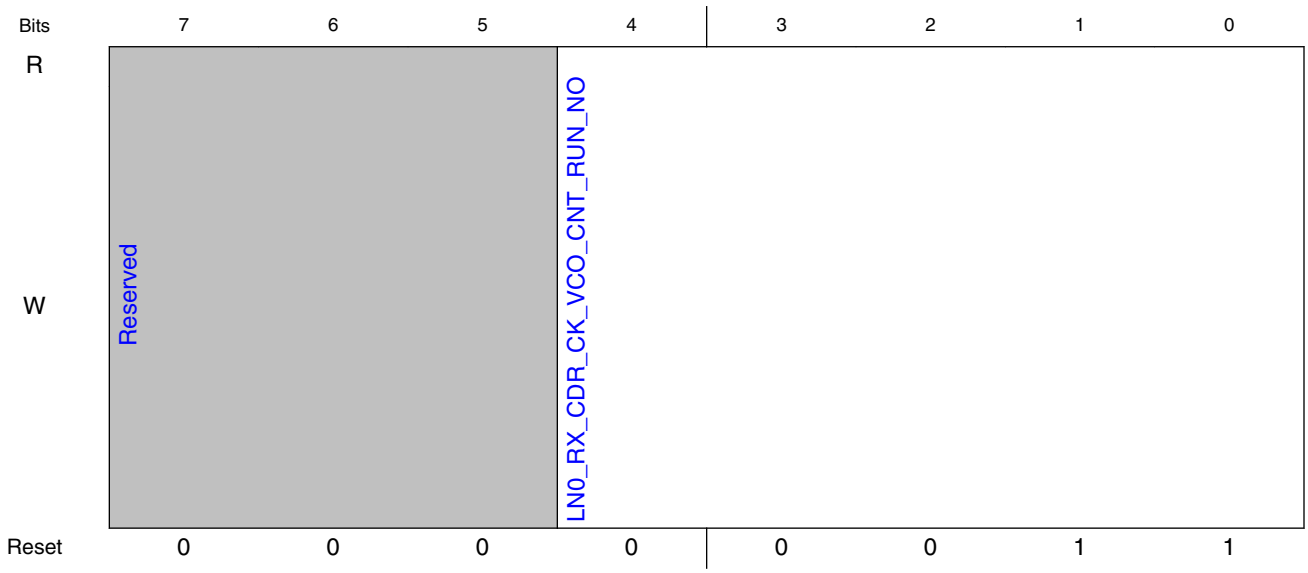
Field	Function
7 —	Reserved
6-2 LN0_RX_CDR_PLL_LOCK_PPM_SET	
1 LN0_RX_CDR_PLL_MODE_ENTRY_SRC	
0 LN0_RX_CDR_PLL_MODE_RESTART	

11.4.3.1.296 (TRSV_REG0A9)

11.4.3.1.296.1 Offset

Register	Offset
TRSV_REG0A9	6A4h

11.4.3.1.296.2 Diagram



11.4.3.1.296.3 Fields

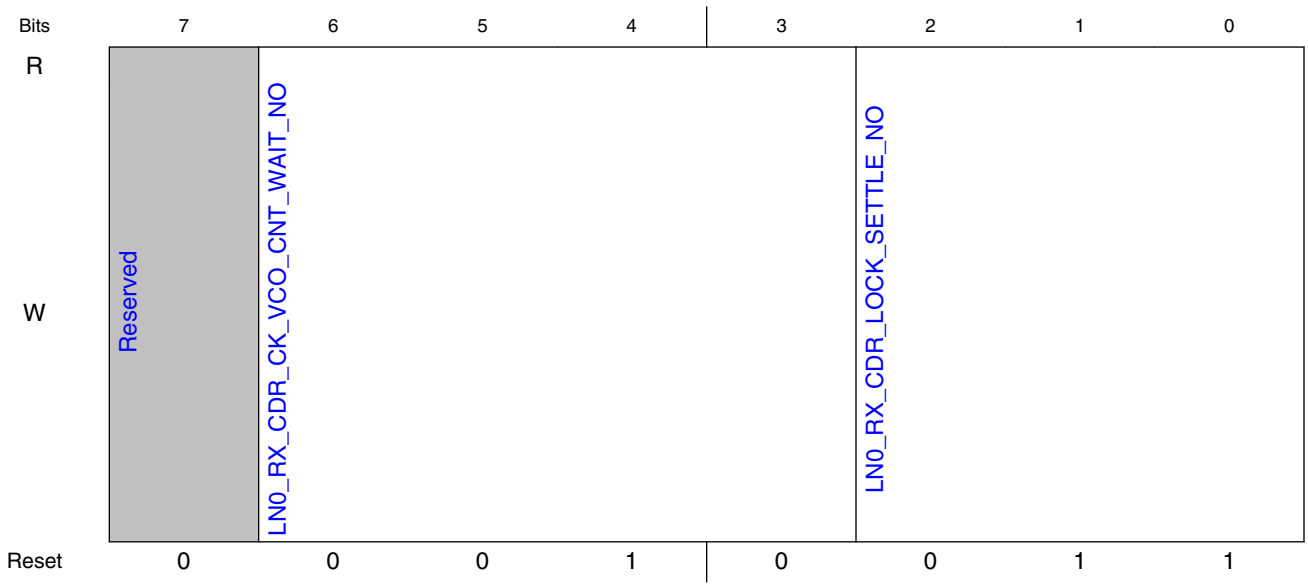
Field	Function
7-5 —	Reserved
4-0 LN0_RX_CDR_CK_VCO_CNT_RUN_NO	

11.4.3.1.297 (TRSV_REG0AA)

11.4.3.1.297.1 Offset

Register	Offset
TRSV_REG0AA	6A8h

11.4.3.1.297.2 Diagram



11.4.3.1.297.3 Fields

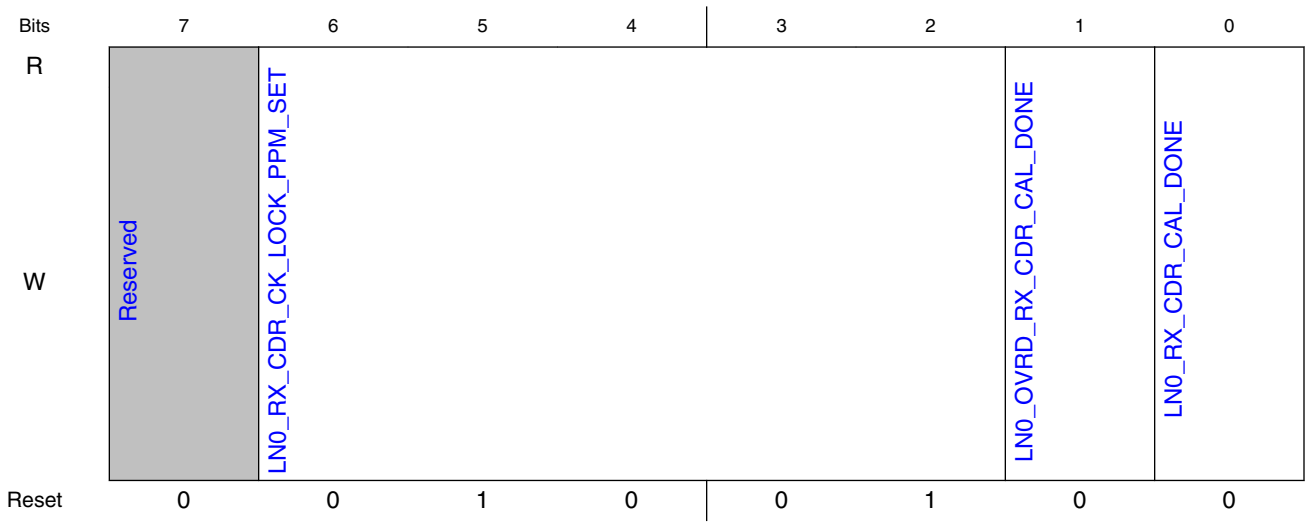
Field	Function
7 —	Reserved
6-3 LN0_RX_CDR_CK_VCO_CNT_WAIT_NO	
2-0 LN0_RX_CDR_LOCK_SETTLE_NO	

11.4.3.1.298 (TRSV_REG0AB)

11.4.3.1.298.1 Offset

Register	Offset
TRSV_REG0AB	6ACh

11.4.3.1.298.2 Diagram



11.4.3.1.298.3 Fields

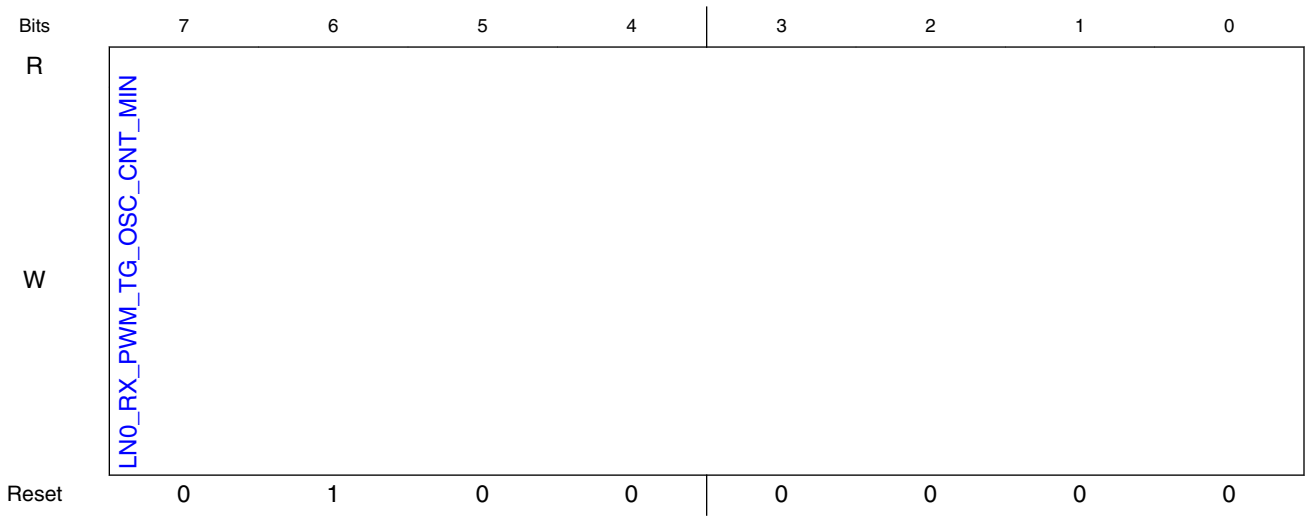
Field	Function
7 —	Reserved
6-2 LN0_RX_CDR_CK_LOCK_PPM_SET	
1 LN0_OVRD_RX_CDR_CAL_DONE	Override enable for rx_cdr_cal_done
0 LN0_RX_CDR_CAL_DONE	

11.4.3.1.299 (TRSV_REG0AC)

11.4.3.1.299.1 Offset

Register	Offset
TRSV_REG0AC	6B0h

11.4.3.1.299.2 Diagram



11.4.3.1.299.3 Fields

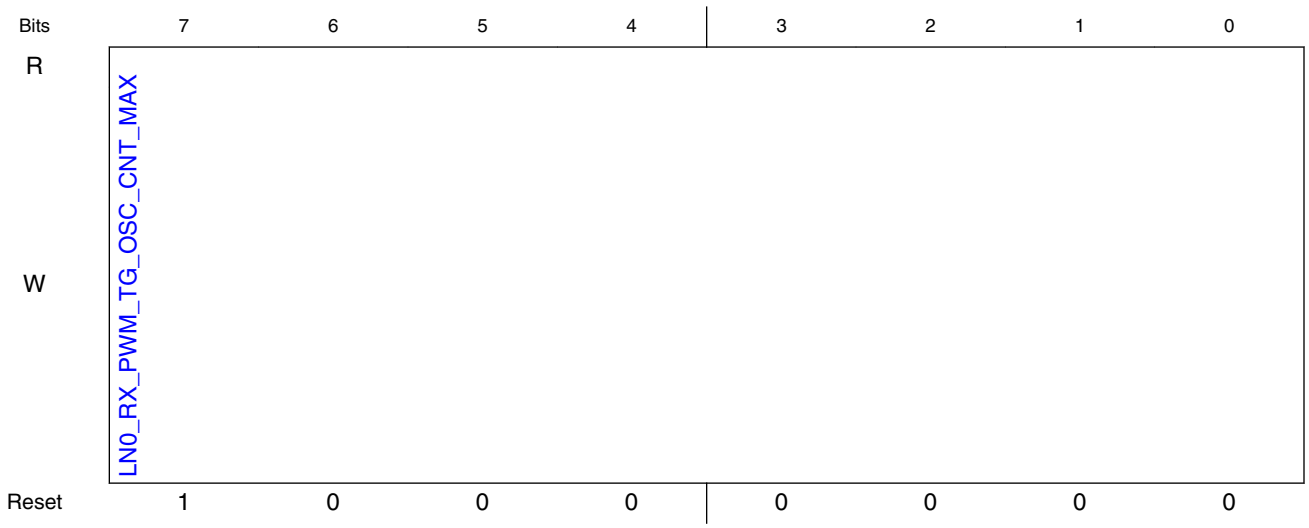
Field	Function
7-0 LN0_RX_PWM_TG_OSC_CNT_MIN	

11.4.3.1.300 (TRSV_REG0AD)

11.4.3.1.300.1 Offset

Register	Offset
TRSV_REG0AD	6B4h

11.4.3.1.300.2 Diagram



11.4.3.1.300.3 Fields

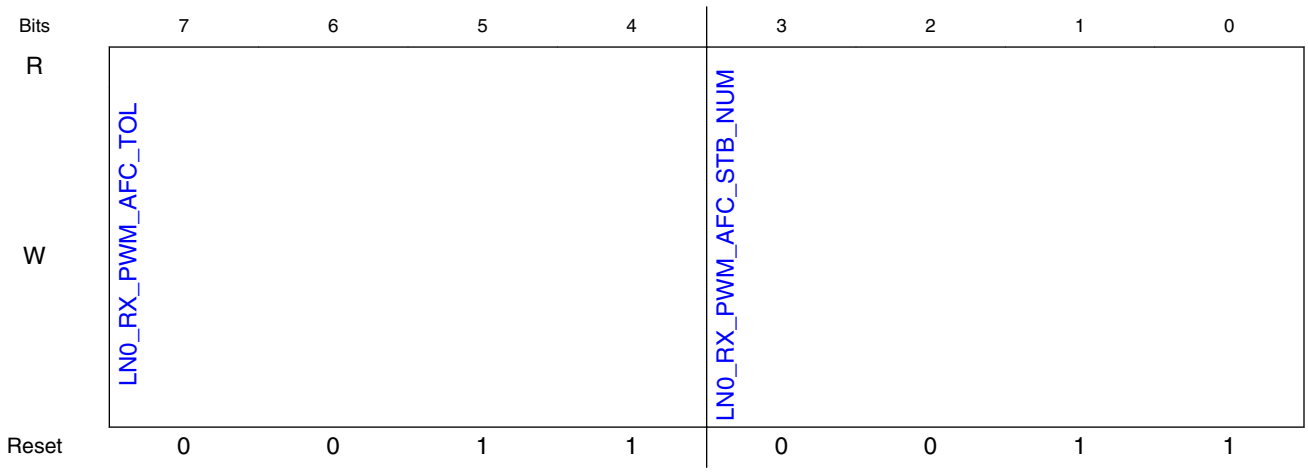
Field	Function
7-0 LN0_RX_PWM_TG_OSC_CNT_MAX	

11.4.3.1.301 (TRSV_REG0AE)

11.4.3.1.301.1 Offset

Register	Offset
TRSV_REG0AE	6B8h

11.4.3.1.301.2 Diagram



11.4.3.1.301.3 Fields

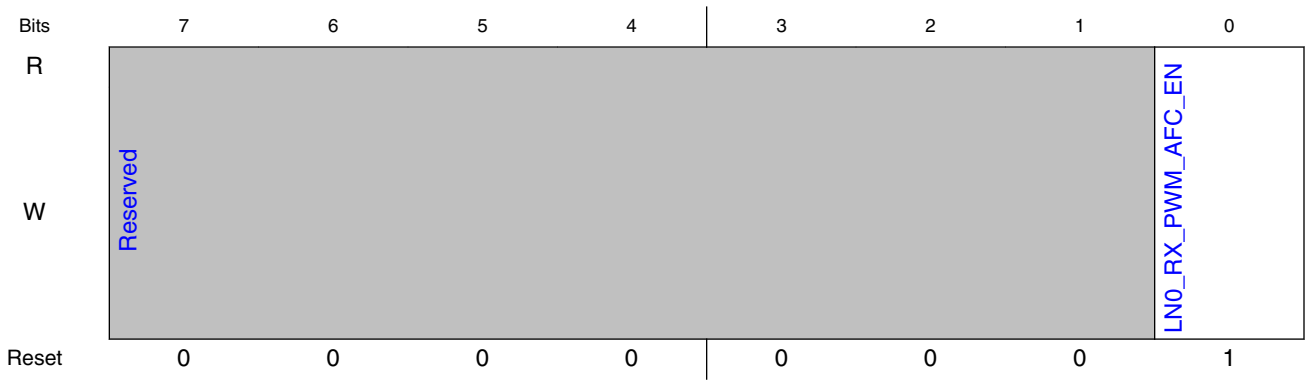
Field	Function
7-4 LN0_RX_PWM_AFC_TOL	
3-0 LN0_RX_PWM_AFC_STB_NUM	

11.4.3.1.302 (TRSV_REG0AF)

11.4.3.1.302.1 Offset

Register	Offset
TRSV_REG0AF	6BCh

11.4.3.1.302.2 Diagram



11.4.3.1.302.3 Fields

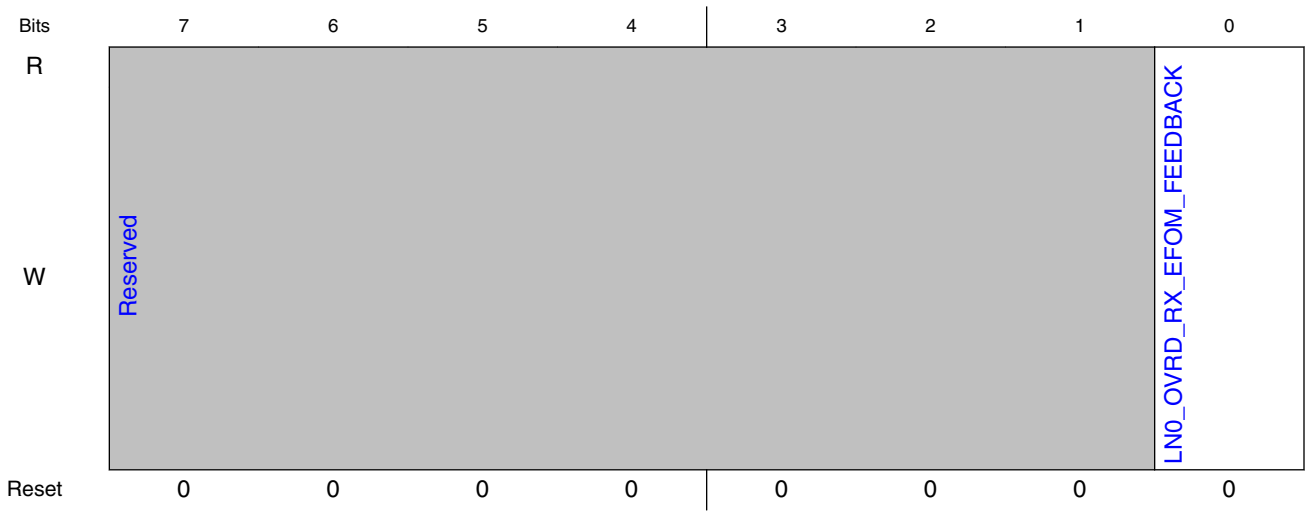
Field	Function
7-1 —	Reserved
0 LN0_RX_PWM_AFC_EN	

11.4.3.1.303 (TRSV_REG0B0)

11.4.3.1.303.1 Offset

Register	Offset
TRSV_REG0B0	6C0h

11.4.3.1.303.2 Diagram



11.4.3.1.303.3 Fields

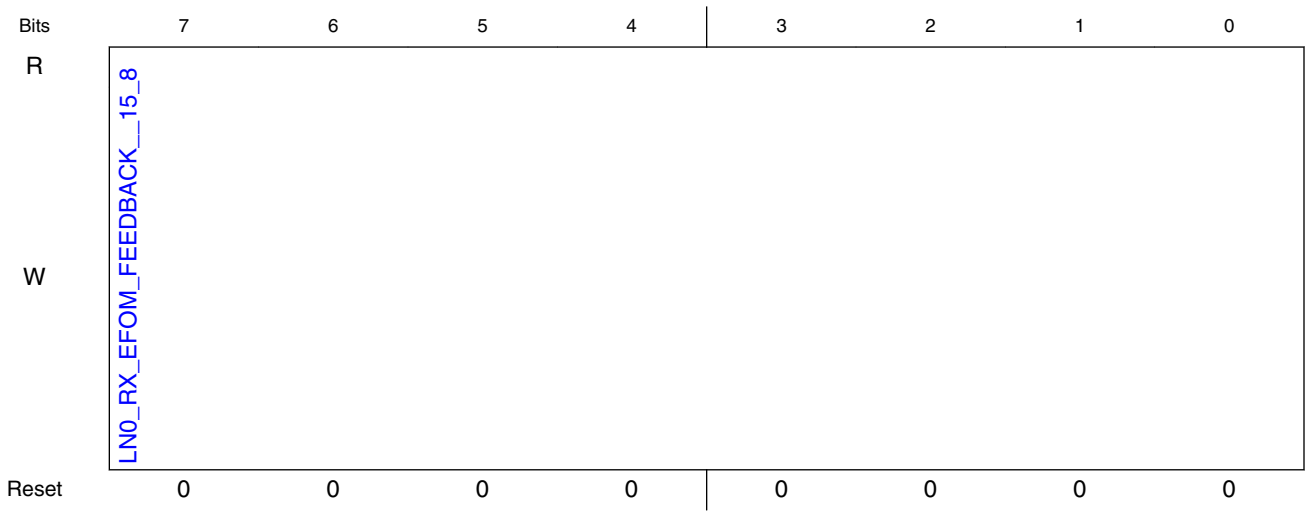
Field	Function
7-1 —	Reserved
0 LN0_OVRD_RX_EFOM_FEEDBACK	Override enable for

11.4.3.1.304 (TRSV_REG0B1)

11.4.3.1.304.1 Offset

Register	Offset
TRSV_REG0B1	6C4h

11.4.3.1.304.2 Diagram



11.4.3.1.304.3 Fields

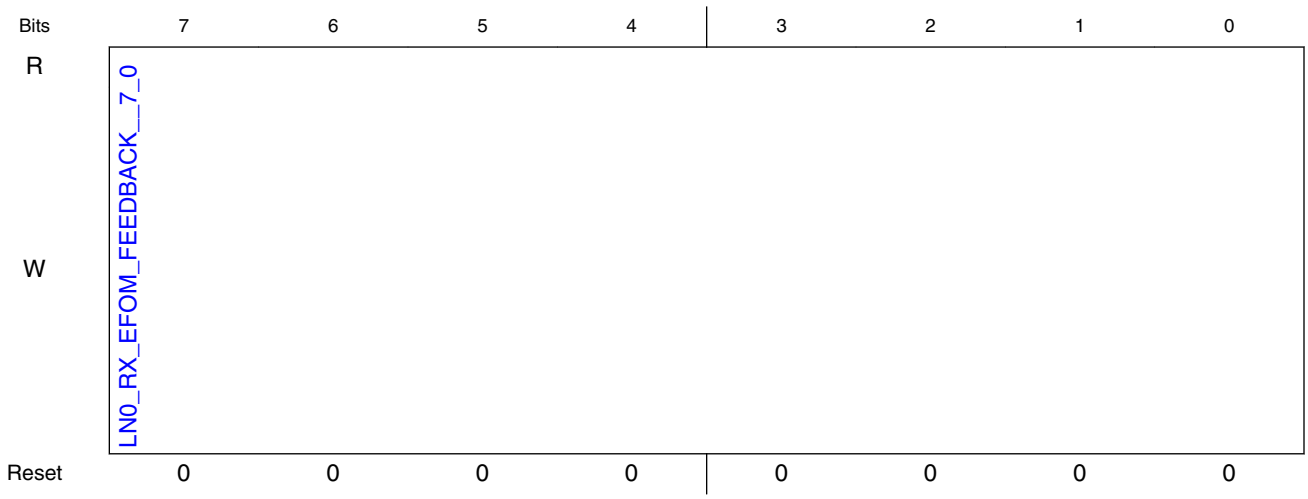
Field	Function
7-0 LN0_RX_EFOM_FEEDBACK__15_8	

11.4.3.1.305 (TRSV_REG0B2)

11.4.3.1.305.1 Offset

Register	Offset
TRSV_REG0B2	6C8h

11.4.3.1.305.2 Diagram



11.4.3.1.305.3 Fields

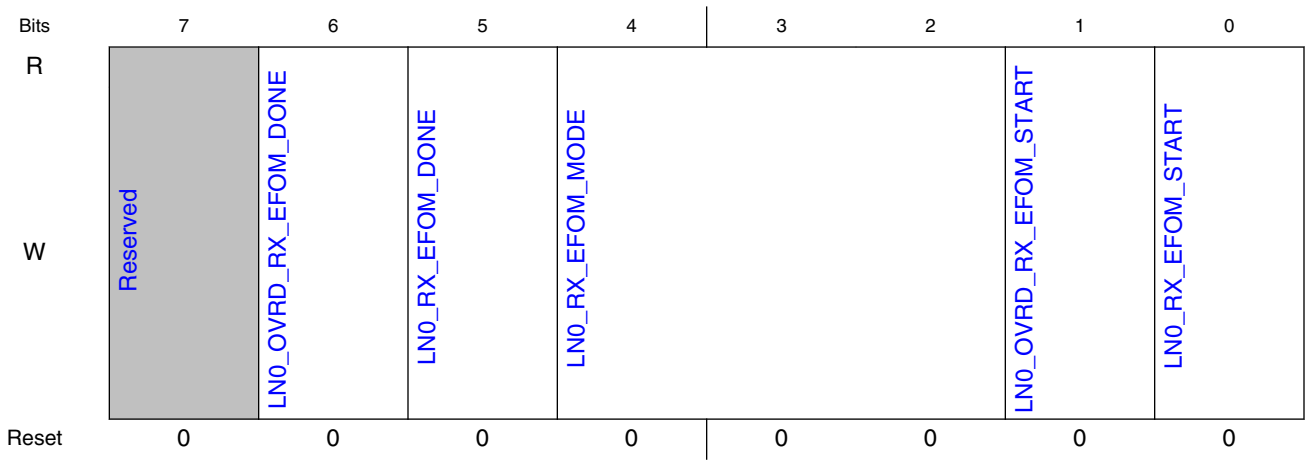
Field	Function
7-0 LN0_RX_EFOM_FEEDBACK__7_0	

11.4.3.1.306 (TRSV_REG0B3)

11.4.3.1.306.1 Offset

Register	Offset
TRSV_REG0B3	6CCh

11.4.3.1.306.2 Diagram



11.4.3.1.306.3 Fields

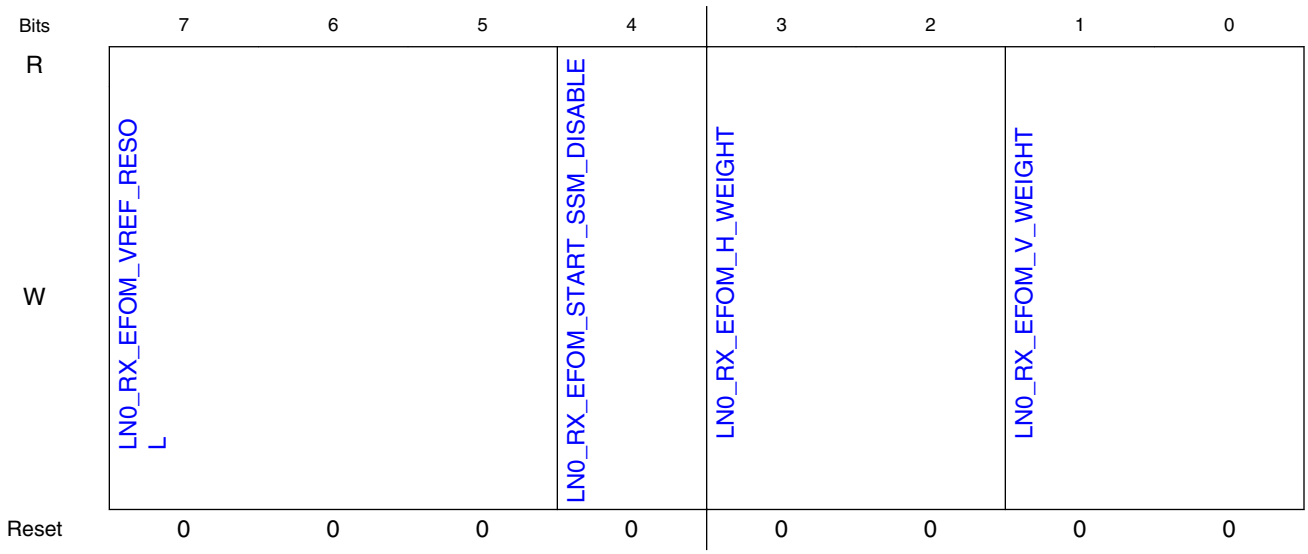
Field	Function
7 —	Reserved
6 LN0_OVRD_RX_EFOM_DONE	Override enable for rx_efom_done
5 LN0_RX_EFOM_DONE	
4-2 LN0_RX_EFOM_MODE	
1 LN0_OVRD_RX_EFOM_START	Override enable for rx_efom_start
0 LN0_RX_EFOM_START	

11.4.3.1.307 (TRSV_REG0B4)

11.4.3.1.307.1 Offset

Register	Offset
TRSV_REG0B4	6D0h

11.4.3.1.307.2 Diagram



11.4.3.1.307.3 Fields

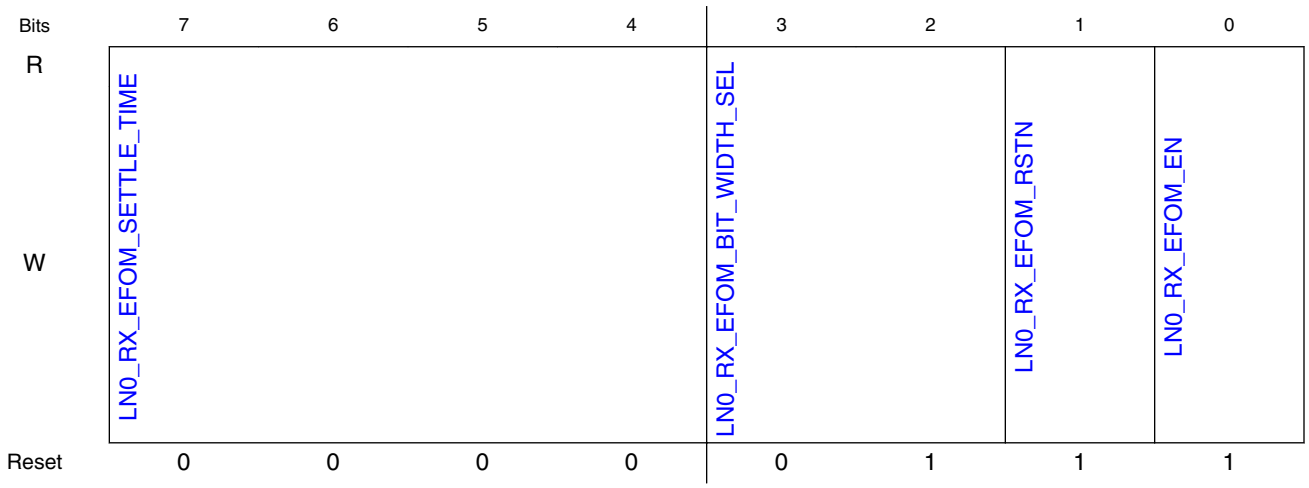
Field	Function
7-5 LN0_RX_EFOM_VREF_RESOL	
4 LN0_RX_EFOM_START_SSM_DISABLE	
3-2 LN0_RX_EFOM_H_WEIGHT	
1-0 LN0_RX_EFOM_V_WEIGHT	

11.4.3.1.308 (TRSV_REG0B5)

11.4.3.1.308.1 Offset

Register	Offset
TRSV_REG0B5	6D4h

11.4.3.1.308.2 Diagram



11.4.3.1.308.3 Fields

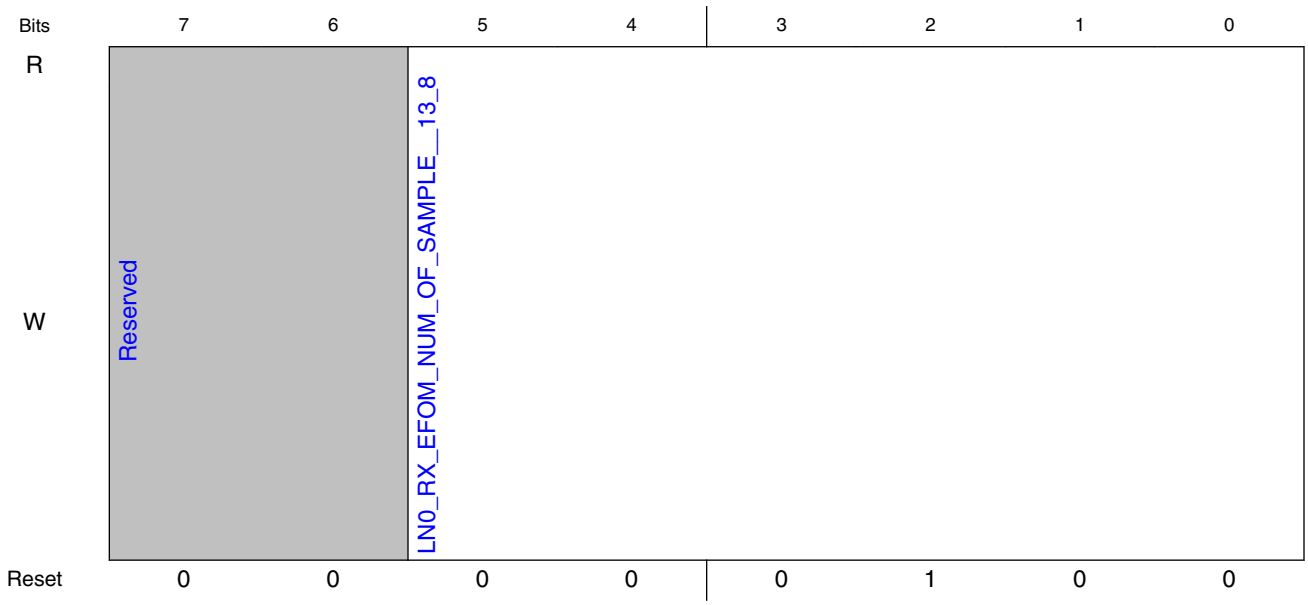
Field	Function
7-4 LN0_RX_EFOM_SETTLE_TIME	
3-2 LN0_RX_EFOM_BIT_WIDTH_SEL	
1 LN0_RX_EFOM_RSTN	
0 LN0_RX_EFOM_EN	

11.4.3.1.309 (TRSV_REG0B6)

11.4.3.1.309.1 Offset

Register	Offset
TRSV_REG0B6	6D8h

11.4.3.1.309.2 Diagram



11.4.3.1.309.3 Fields

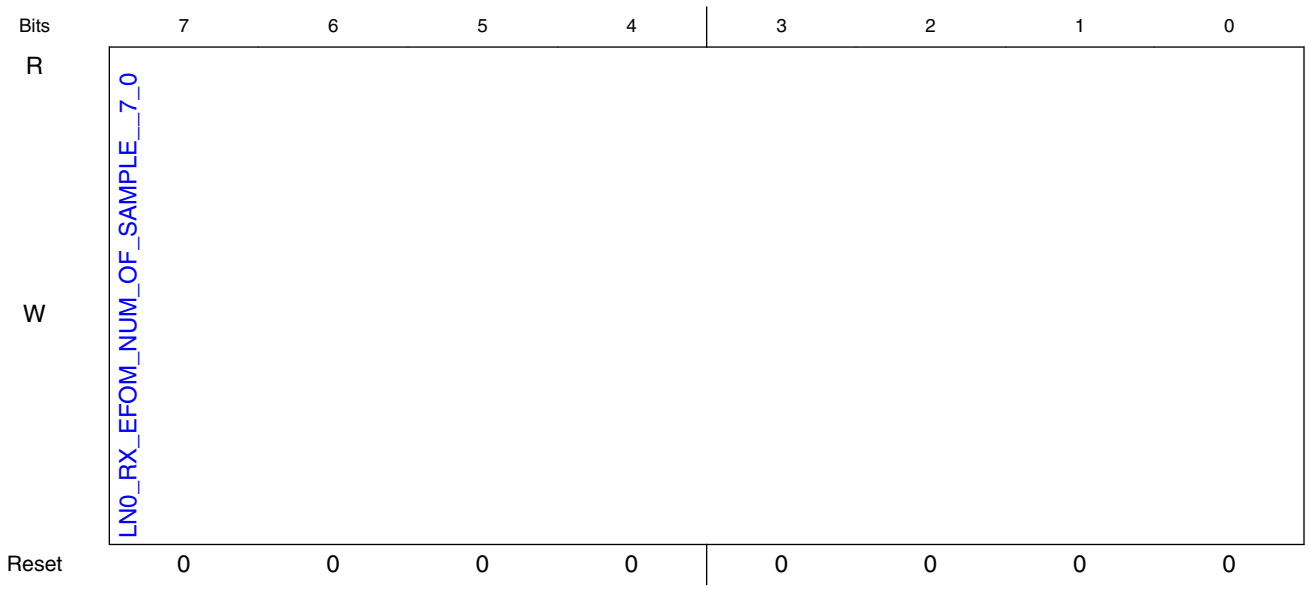
Field	Function
7-6 —	Reserved
5-0 LN0_RX_EFOM_NUM_OF_SAMPLE__13_8	

11.4.3.1.310 (TRSV_REG0B7)

11.4.3.1.310.1 Offset

Register	Offset
TRSV_REG0B7	6DCh

11.4.3.1.310.2 Diagram



11.4.3.1.310.3 Fields

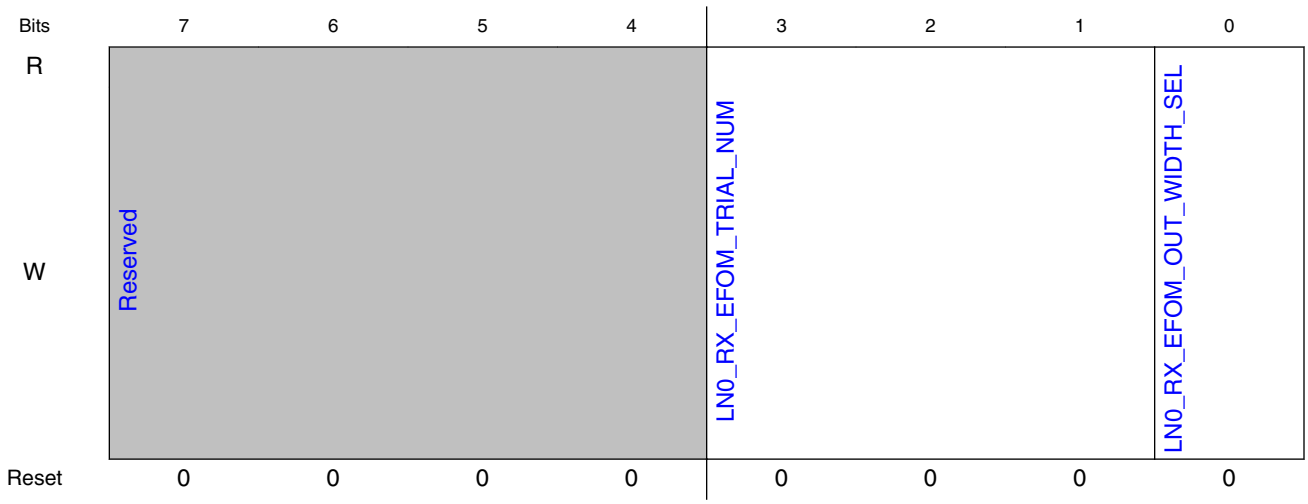
Field	Function
7-0 LN0_RX_EFOM_NUM_OF_SAMPLE__7_0	

11.4.3.1.311 (TRSV_REG0B8)

11.4.3.1.311.1 Offset

Register	Offset
TRSV_REG0B8	6E0h

11.4.3.1.311.2 Diagram



11.4.3.1.311.3 Fields

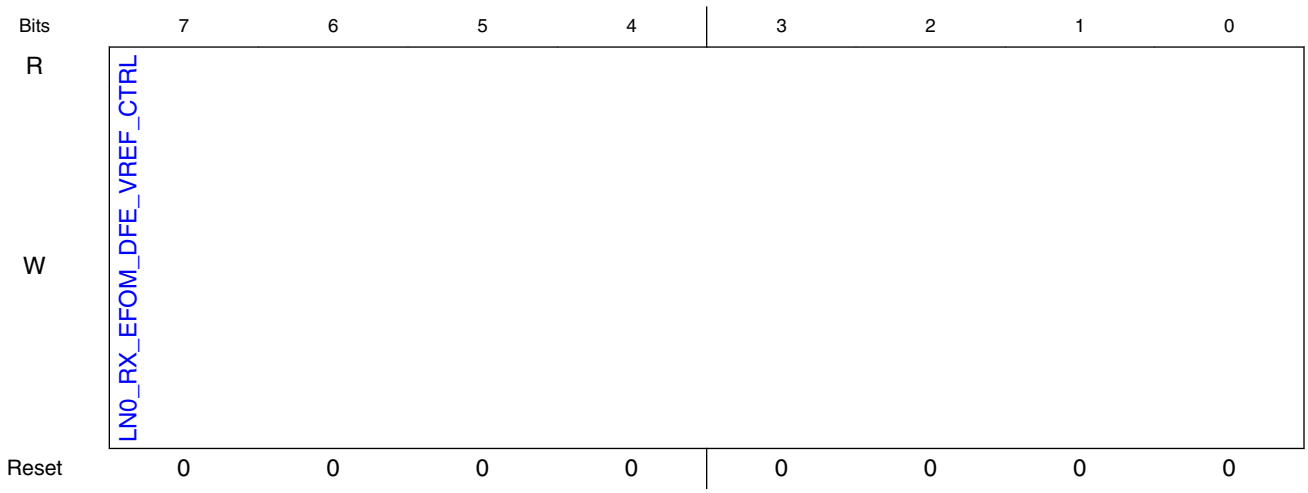
Field	Function
7-4 —	Reserved
3-1 LN0_RX_EFOM_TRIAL_NUM	
0 LN0_RX_EFOM_OUT_WIDTH_SEL	

11.4.3.1.312 (TRSV_REG0B9)

11.4.3.1.312.1 Offset

Register	Offset
TRSV_REG0B9	6E4h

11.4.3.1.312.2 Diagram



11.4.3.1.312.3 Fields

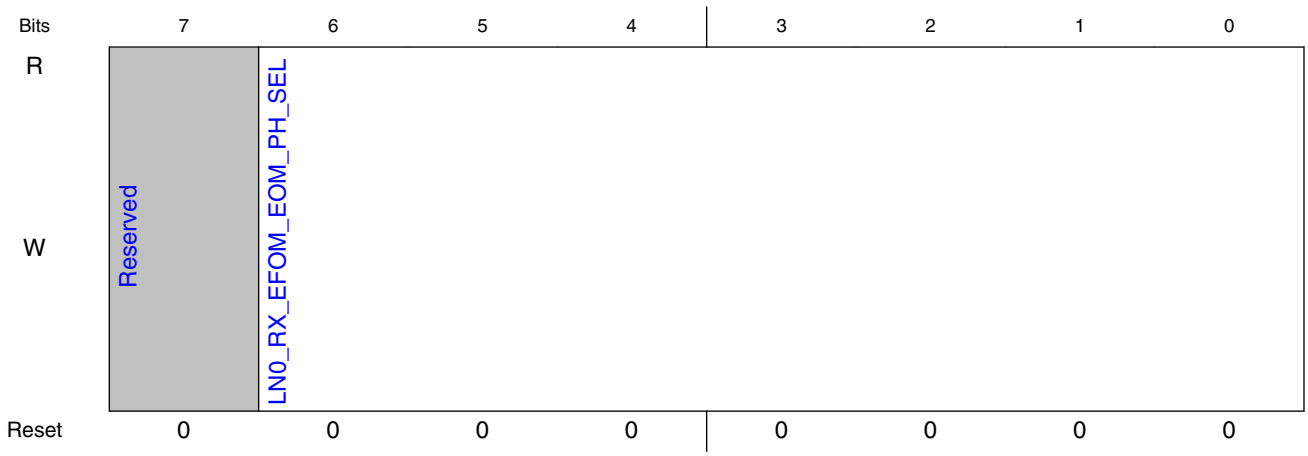
Field	Function
7-0 LN0_RX_EFOM_DFE_VREF_CTRL	

11.4.3.1.313 (TRSV_REG0BA)

11.4.3.1.313.1 Offset

Register	Offset
TRSV_REG0BA	6E8h

11.4.3.1.313.2 Diagram



11.4.3.1.313.3 Fields

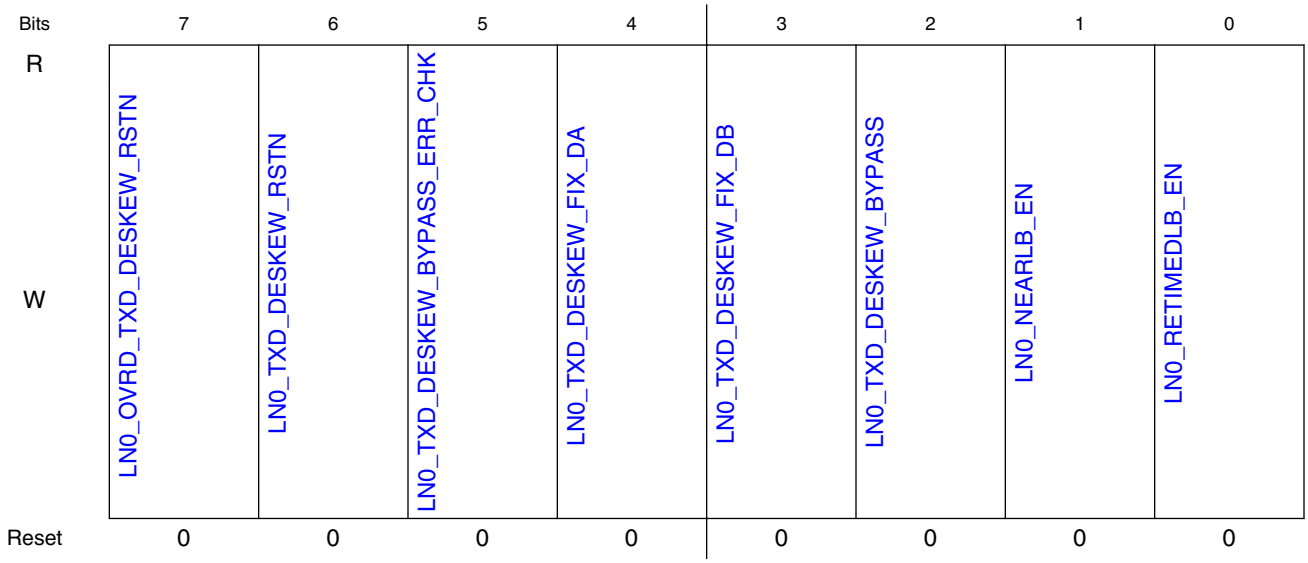
Field	Function
7 —	Reserved
6-0 LN0_RX_EFOM_EOM_PH_SEL	

11.4.3.1.314 (TRSV_REG0BB)

11.4.3.1.314.1 Offset

Register	Offset
TRSV_REG0BB	6ECh

11.4.3.1.314.2 Diagram



11.4.3.1.314.3 Fields

Field	Function
7 LN0_OVRD_TXD_DESKEW_RSTN	Override enable for txd_deskew_rstn
6 LN0_TXD_DESKEW_RSTN	
5 LN0_TXD_DESKEW_BYPASS_ERR_CHK	
4 LN0_TXD_DESKEW_FIX_DA	
3 LN0_TXD_DESKEW_FIX_DB	
2 LN0_TXD_DESKEW_BYPASS	
1 LN0_NEARLB_EN	
0	

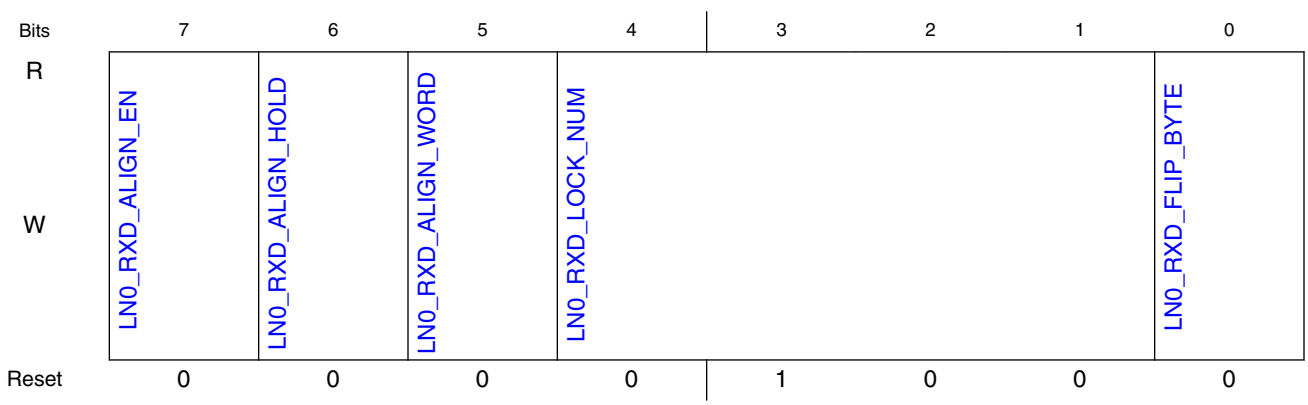
Field	Function
LN0_RETIMEDLB_EN	

11.4.3.1.315 (TRSV_REG0BC)

11.4.3.1.315.1 Offset

Register	Offset
TRSV_REG0BC	6F0h

11.4.3.1.315.2 Diagram



11.4.3.1.315.3 Fields

Field	Function
7 LN0_RXD_ALIGN_EN	
6 LN0_RXD_ALIGN_HOLD	
5 LN0_RXD_ALIGN_WORD	
4-1 LN0_RXD_LOCK_NUM	

Table continues on the next page...

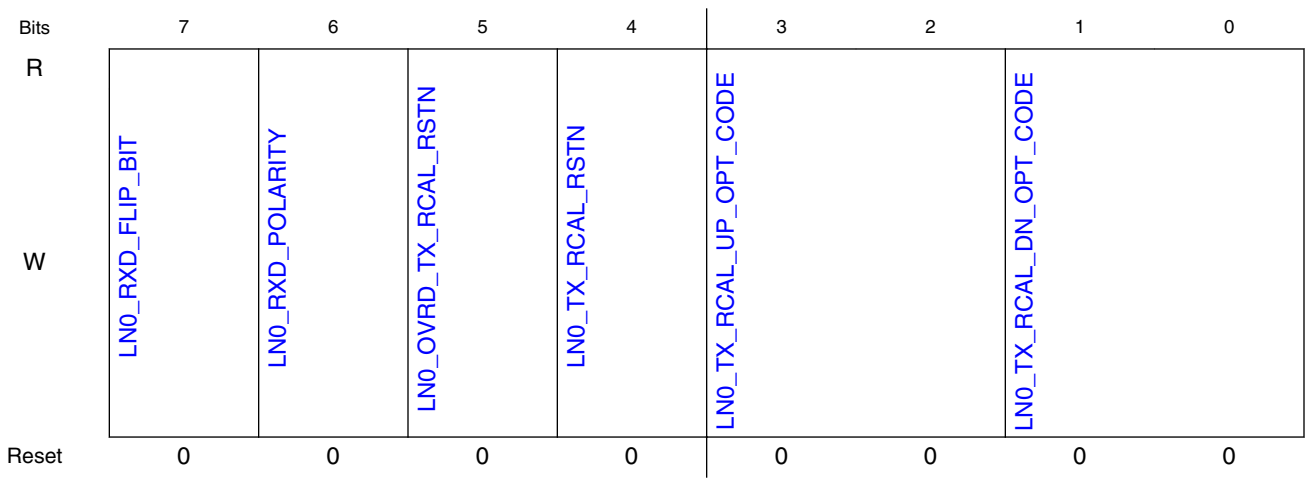
Field	Function
0 LN0_RXD_FLIP_BYTE	

11.4.3.1.316 (TRSV_REG0BD)

11.4.3.1.316.1 Offset

Register	Offset
TRSV_REG0BD	6F4h

11.4.3.1.316.2 Diagram



11.4.3.1.316.3 Fields

Field	Function
7 LN0_RXD_FLIP_BIT	
6 LN0_RXD_POLARITY	
5 LN0_OVRD_TX_RCAL_RSTN	Override enable for tx_rcal_rstn

Table continues on the next page...

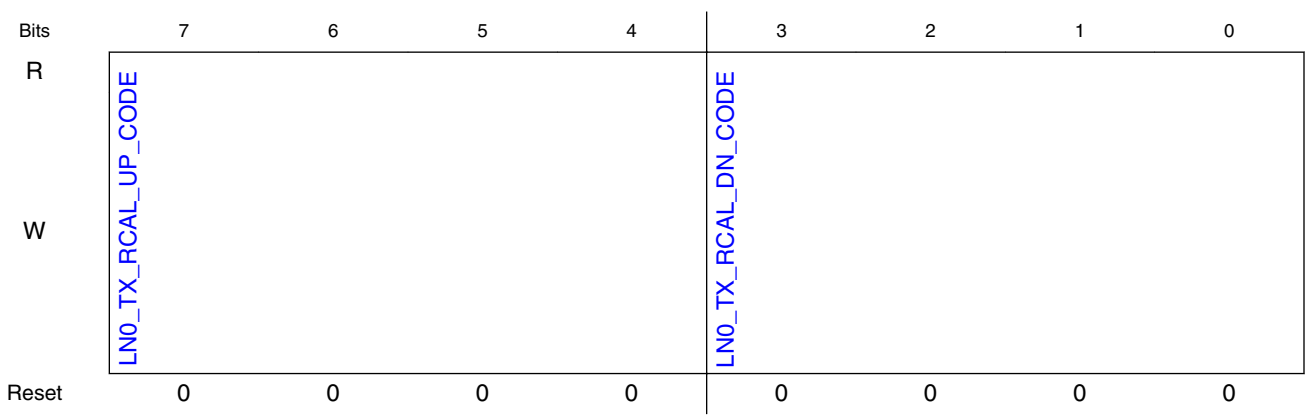
Field	Function
4 LN0_TX_RCAL_RSTN	
3-2 LN0_TX_RCAL_UP_OPT_CODE	
1-0 LN0_TX_RCAL_DN_OPT_CODE	

11.4.3.1.317 (TRSV_REG0BE)

11.4.3.1.317.1 Offset

Register	Offset
TRSV_REG0BE	6F8h

11.4.3.1.317.2 Diagram



11.4.3.1.317.3 Fields

Field	Function
7-4 LN0_TX_RCAL_UP_CODE	Termination up control bits. <3> bit is reserved bit., Default code = 011,Min code =000 and Max code is 111

Table continues on the next page...

Field	Function
3-0 LN0_TX_RCAL_DN_CODE	Termination down control bits. <3>bit is reserved bit., Default code= 011,Min code =000 and Max code is 111

11.4.3.1.318 (TRSV_REG0BF)

11.4.3.1.318.1 Offset

Register	Offset
TRSV_REG0BF	6FCh

11.4.3.1.318.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_OVRD_TX_RCAL_COMP_OUT	LN0_TX_RCAL_COMP_OUT	LN0_OVRD_TX_RCAL_DONE	LN0_TX_RCAL_DONE	LN0_OVRD_RX_RCAL_RSTN	LN0_RX_RCAL_RSTN	LN0_RX_RCAL_OPT_CODE	
Reset	0	0	0	0	0	0	0	0

11.4.3.1.318.3 Fields

Field	Function
7 LN0_OVRD_TX_RCAL_COMP_OUT	Override enable for tx_rcal_comp_out
6 LN0_TX_RCAL_COMP_OUT	

Table continues on the next page...

Field	Function
5 LN0_OVRD_TX_RCAL_DONE	Override enable for tx_rcal_done
4 LN0_TX_RCAL_DONE	Monitoring for TX RCAL done
3 LN0_OVRD_RX_RCAL_RSTN	Override enable for rx_rcal_rstn
2 LN0_RX_RCAL_RSTN	
1-0 LN0_RX_RCAL_OPT_CODE	

11.4.3.1.319 (TRSV_REG0C0)

11.4.3.1.319.1 Offset

Register	Offset
TRSV_REG0C0	700h

11.4.3.1.319.2 Diagram

Bits	7	6	5	4	3	2	1	0
R								
W	LN0_RX_RTERM_CTRL				LN0_OVRD_RX_RCAL_COMP_OUT	LN0_RX_RCAL_COMP_OUT	LN0_OVRD_RX_RCAL_DONE	LN0_RX_RCAL_DONE
Reset	0	0	0	0	0	0	0	0

11.4.3.1.319.3 Fields

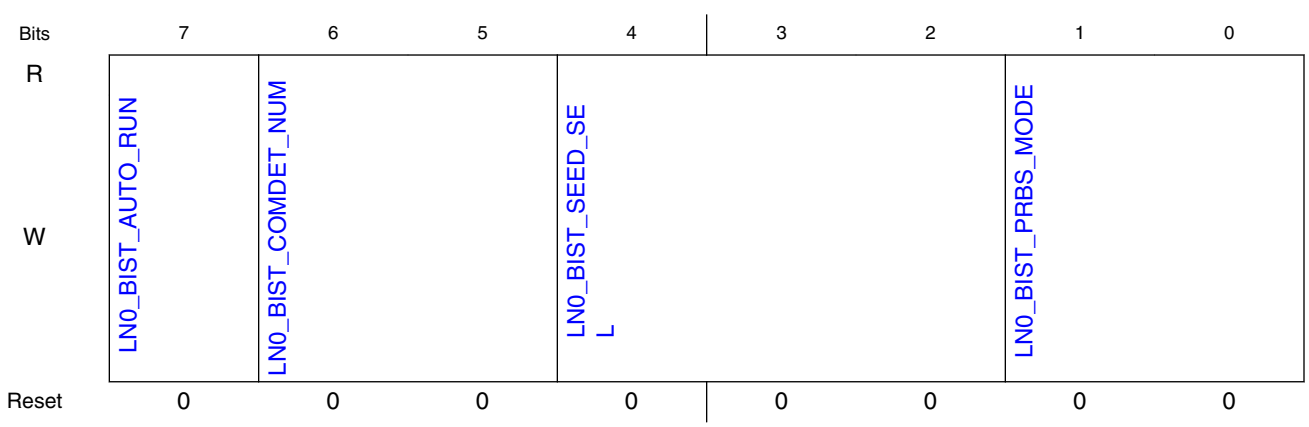
Field	Function
7-4 LN0_RX_RTERM_CTRL	Termination Calibration will send control signals to make 42.5 ohms. MSB<4>=> 1'b0=50,1'b1=42.5(default). Others<3:0> Binary weighted control for RTERM
3 LN0_OVRD_RX_RCAL_COMP_OUT	Override enable for rx_rcal_comp_out
2 LN0_RX_RCAL_COMP_OUT	
1 LN0_OVRD_RX_RCAL_DONE	Override enable for rx_rcal_done
0 LN0_RX_RCAL_DONE	RX RCAL done

11.4.3.1.320 (TRSV_REG0C1)

11.4.3.1.320.1 Offset

Register	Offset
TRSV_REG0C1	704h

11.4.3.1.320.2 Diagram



11.4.3.1.320.3 Fields

Field	Function
7 LN0_BIST_AUTO_RUN	
6-5 LN0_BIST_COMDET_NUM	
4-2 LN0_BIST_SEED_SEL	
1-0 LN0_BIST_PRBS_MODE	

11.4.3.1.321 (TRSV_REG0C2)

11.4.3.1.321.1 Offset

Register	Offset
TRSV_REG0C2	708h

11.4.3.1.321.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_BIST_EN	LN0_BIST_DATA_EN	LN0_BIST_RX_EN	LN0_BIST_RX_HOLD	LN0_BIST_RX_START	LN0_BIST_TX_EN	LN0_BIST_TX_ERRINJ	LN0_BIST_TX_START
Reset	0	0	0	0	0	0	0	0

11.4.3.1.321.3 Fields

Field	Function
7	

Table continues on the next page...

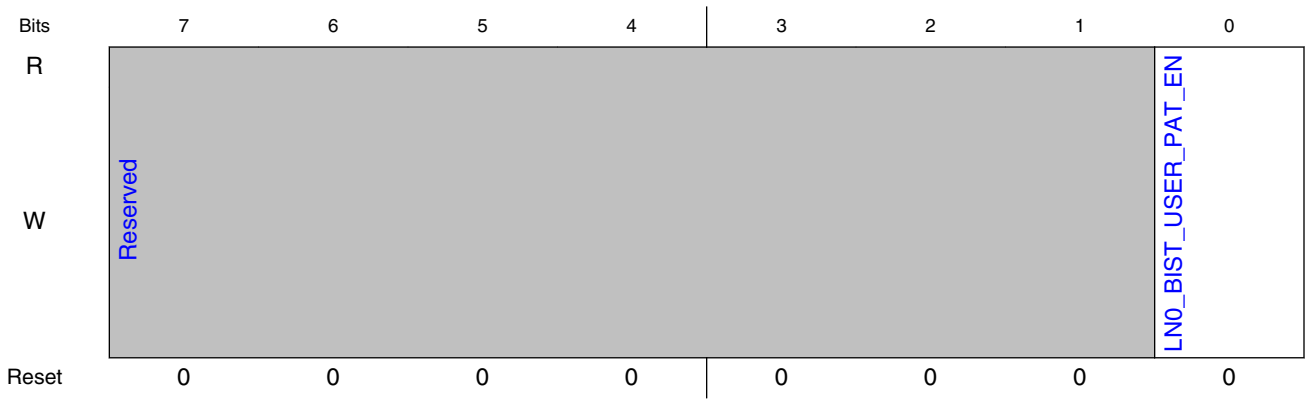
Field	Function
LN0_BIST_EN	
6 LN0_BIST_DATA_EN	
5 LN0_BIST_RX_EN	
4 LN0_BIST_RX_HOLD	
3 LN0_BIST_RX_START	
2 LN0_BIST_TX_EN	
1 LN0_BIST_TX_ERRINJ	
0 LN0_BIST_TX_START	

11.4.3.1.322 (TRSV_REG0C3)

11.4.3.1.322.1 Offset

Register	Offset
TRSV_REG0C3	70Ch

11.4.3.1.322.2 Diagram



11.4.3.1.322.3 Fields

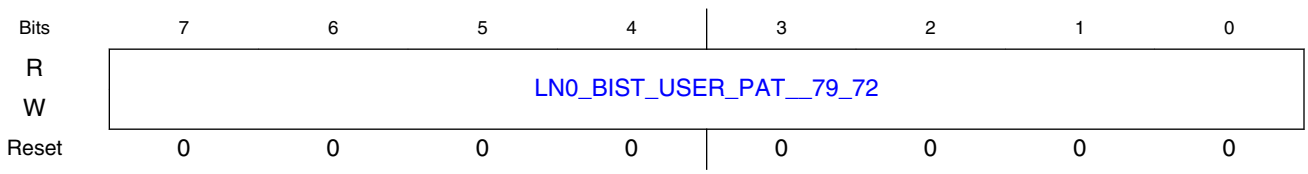
Field	Function
7-1 —	Reserved
0 LN0_BIST_USER_PAT_EN	

11.4.3.1.323 (TRSV_REG0C4)

11.4.3.1.323.1 Offset

Register	Offset
TRSV_REG0C4	710h

11.4.3.1.323.2 Diagram



11.4.3.1.323.3 Fields

Field	Function
7-0 LN0_BIST_USER_PAT__79_72	

11.4.3.1.324 (TRSV_REG0C5)

11.4.3.1.324.1 Offset

Register	Offset
TRSV_REG0C5	714h

11.4.3.1.324.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_BIST_USER_PAT__71_64							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.324.3 Fields

Field	Function
7-0 LN0_BIST_USER_PAT__71_64	

11.4.3.1.325 (TRSV_REG0C6)

11.4.3.1.325.1 Offset

Register	Offset
TRSV_REG0C6	718h

11.4.3.1.325.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_BIST_USER_PAT__63_56							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.325.3 Fields

Field	Function
7-0 LN0_BIST_USER_PAT__63_56	

11.4.3.1.326 (TRSV_REG0C7)

11.4.3.1.326.1 Offset

Register	Offset
TRSV_REG0C7	71Ch

11.4.3.1.326.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_BIST_USER_PAT__55_48							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.326.3 Fields

Field	Function
7-0 LN0_BIST_USER_PAT__55_48	

11.4.3.1.327 (TRSV_REG0C8)

11.4.3.1.327.1 Offset

Register	Offset
TRSV_REG0C8	720h

11.4.3.1.327.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_BIST_USER_PAT__47_40							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.327.3 Fields

Field	Function
7-0 LN0_BIST_USER_PAT__47_40	

11.4.3.1.328 (TRSV_REG0C9)

11.4.3.1.328.1 Offset

Register	Offset
TRSV_REG0C9	724h

11.4.3.1.328.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_BIST_USER_PAT__39_32							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.328.3 Fields

Field	Function
7-0 LN0_BIST_USER_PAT__39_32	

11.4.3.1.329 (TRSV_REG0CA)

11.4.3.1.329.1 Offset

Register	Offset
TRSV_REG0CA	728h

11.4.3.1.329.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_BIST_USER_PAT__31_24							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.329.3 Fields

Field	Function
7-0 LN0_BIST_USER_PAT__31_24	

11.4.3.1.330 (TRSV_REG0CB)

11.4.3.1.330.1 Offset

Register	Offset
TRSV_REG0CB	72Ch

11.4.3.1.330.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_BIST_USER_PAT__23_16							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.330.3 Fields

Field	Function
7-0 LN0_BIST_USER_PAT__23_16	

11.4.3.1.331 (TRSV_REG0CC)

11.4.3.1.331.1 Offset

Register	Offset
TRSV_REG0CC	730h

11.4.3.1.331.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_BIST_USER_PAT__15_8							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.331.3 Fields

Field	Function
7-0 LN0_BIST_USER_PAT__15_8	

11.4.3.1.332 (TRSV_REG0CD)

11.4.3.1.332.1 Offset

Register	Offset
TRSV_REG0CD	734h

11.4.3.1.332.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_BIST_USER_PAT__7_0							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.332.3 Fields

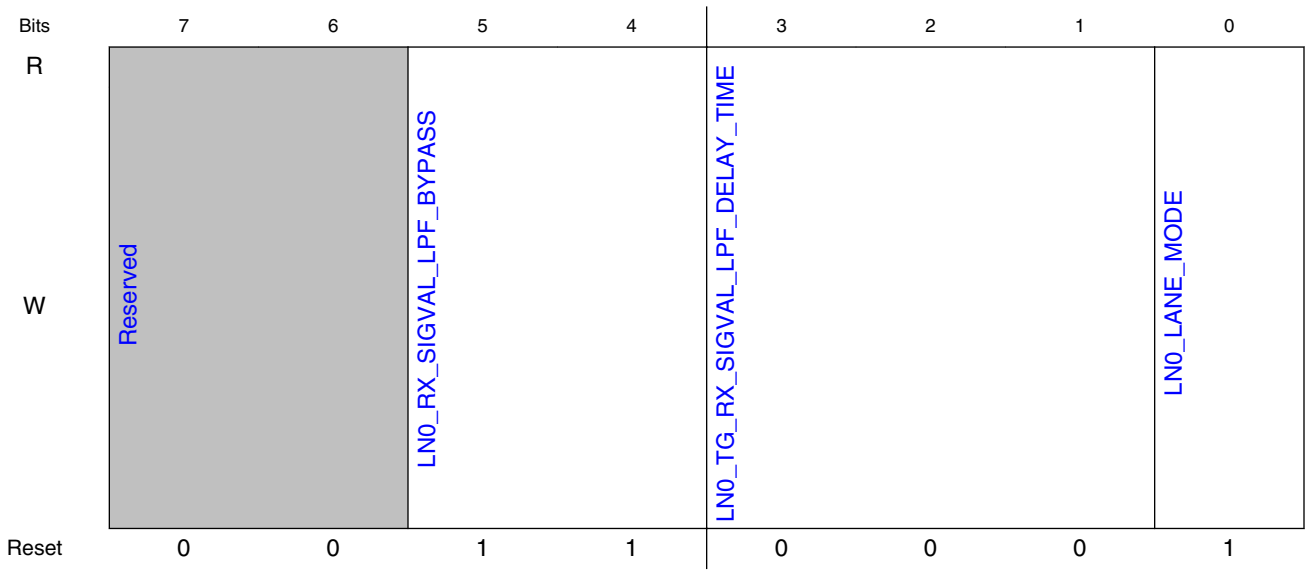
Field	Function
7-0 LN0_BIST_USER_PAT__7_0	

11.4.3.1.333 (TRSV_REG0CE)

11.4.3.1.333.1 Offset

Register	Offset
TRSV_REG0CE	738h

11.4.3.1.333.2 Diagram



11.4.3.1.333.3 Fields

Field	Function
7-6 —	Reserved
5-4 LN0_RX_SIGVAL_LPF_BYPASS	
3-1 LN0_TG_RX_SIGVAL_LPF_DELAY_TIME	
0 LN0_LANE_MODE	Lane operation mode 1 : Aggregation Mode 0 : Lane Bifurcation mode

11.4.3.1.334 (TRSV_REG0CF)

11.4.3.1.334.1 Offset

Register	Offset
TRSV_REG0CF	73Ch

11.4.3.1.334.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	Reserved	LN0_OVRD_LANE_RATE	LN0_LANE_RATE		LN0_LANE_TIMER_SEL	LN0_MISC_TX_CLK_SRC	LN0_MISC_RX_CLK_SRC	LN0_MISC_RX_CLK_INV
W								
Reset	0	0	0	0	0	0	0	0

11.4.3.1.334.3 Fields

Field	Function
7 —	Reserved
6 LN0_OVRD_LANE_RATE	Override enable for lane_rate
5-4 LN0_LANE_RATE	
3 LN0_LANE_TIMER_SEL	
2 LN0_MISC_TX_CLK_SRC	
1 LN0_MISC_RX_CLK_SRC	
0 LN0_MISC_RX_CLK_INV	

11.4.3.1.335 (TRSV_REG0D0)

11.4.3.1.335.1 Offset

Register	Offset
TRSV_REG0D0	740h

11.4.3.1.335.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	Reserved	LN0_OVRD_MISC_RX_SQHS_SIGVAL	LN0_MISC_RX_SQHS_SIGVAL	LN0_MISC_RX_DATA_CLEAR_SRC	LN0_OVRD_MISC_RX_LFPS_DET	LN0_MISC_RX_LFPS_DET	LN0_OVRD_MISC_TX_RXD_DETECTED	LN0_MISC_TX_RXD_DETECTED
W								
Reset	0	0	0	0	0	0	0	0

11.4.3.1.335.3 Fields

Field	Function
7 —	Reserved
6 LN0_OVRD_MISC_RX_SQHS_SIGVAL	Override enable for misc_rx_sqhs_sigval
5 LN0_MISC_RX_SQHS_SIGVAL	
4 LN0_MISC_RX_DATA_CLEAR_SRC	
3 LN0_OVRD_MISC_RX_LFPS_DET	Override enable for misc_rx_lfps_det

Table continues on the next page...

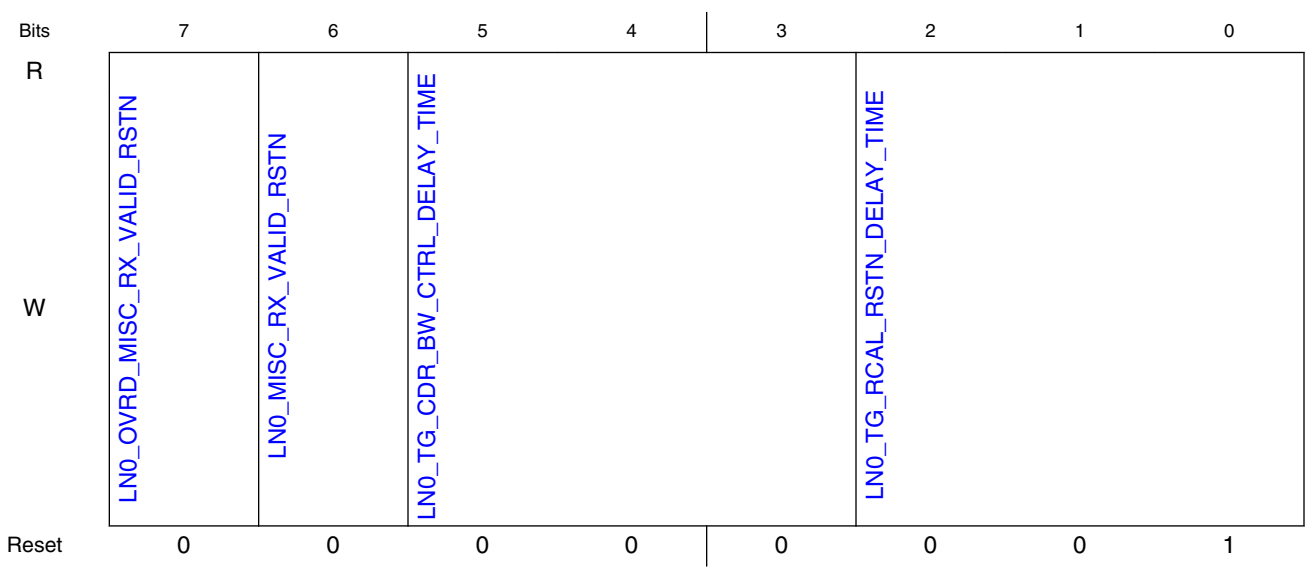
Field	Function
2 LN0_MISC_RX_LFPS_DET	
1 LN0_OVRD_MISC_TX_RXD_DETECTED	Override enable for misc_tx_rxd_detected
0 LN0_MISC_TX_RXD_DETECTED	

11.4.3.1.336 (TRSV_REG0D1)

11.4.3.1.336.1 Offset

Register	Offset
TRSV_REG0D1	744h

11.4.3.1.336.2 Diagram



11.4.3.1.336.3 Fields

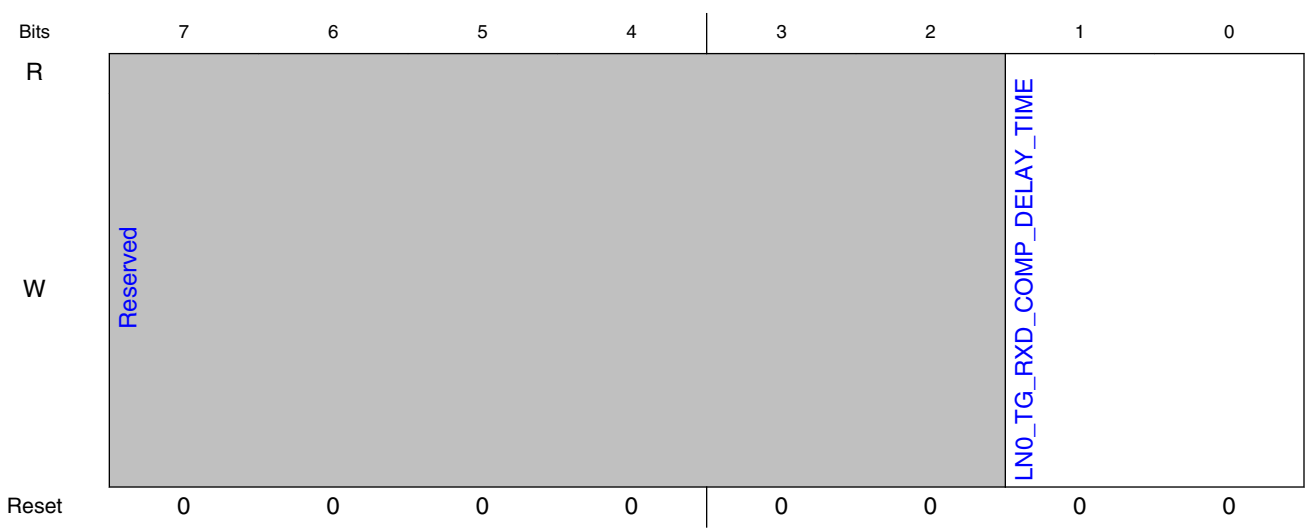
Field	Function
7 LN0_OVRD_MISC_RX_VALID_RSTN	Override enable for misc_rx_valid_rstn
6 LN0_MISC_RX_VALID_RSTN	
5-3 LN0_TG_CDR_BW_CTRL_DELAY_TIME	RX CDR bandwidth change time control
2-0 LN0_TG_RCAL_RSTN_DELAY_TIME	Rx Rcal reset delay time after PLL AFC done

11.4.3.1.337 (TRSV_REG0D2)

11.4.3.1.337.1 Offset

Register	Offset
TRSV_REG0D2	748h

11.4.3.1.337.2 Diagram



11.4.3.1.337.3 Fields

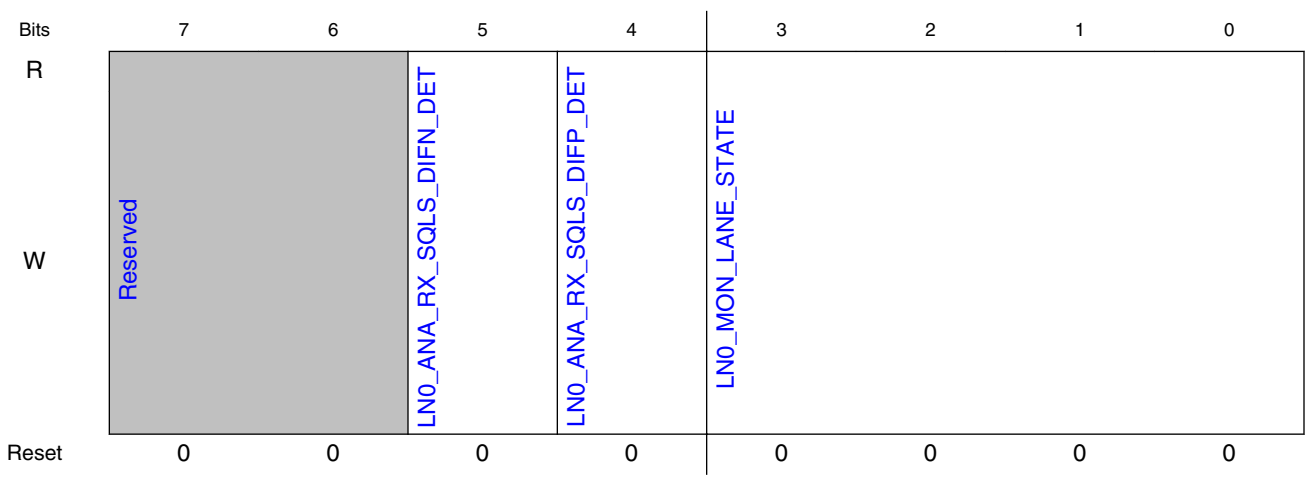
Field	Function
7-2 —	Reserved
1-0 LN0_TG_RXD_COMP_DELAY_TIME	

11.4.3.1.338 (TRSV_REG0D3)

11.4.3.1.338.1 Offset

Register	Offset
TRSV_REG0D3	74Ch

11.4.3.1.338.2 Diagram



11.4.3.1.338.3 Fields

Field	Function
7-6 —	Reserved

Table continues on the next page...

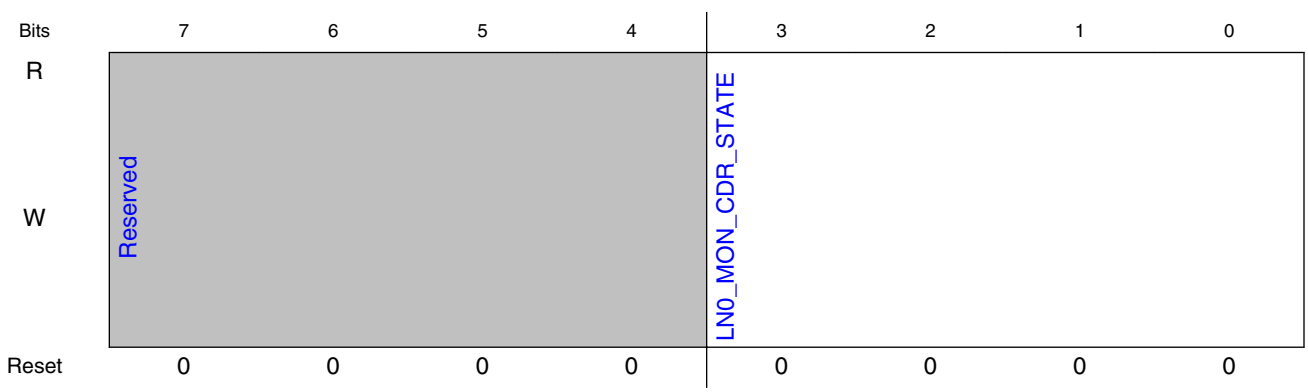
Field	Function
5 LN0_ANA_RX_SQLS_DIFN_DET	DIFN Detection signal
4 LN0_ANA_RX_SQLS_DIFP_DET	DIFP Detection signal
3-0 LN0_MON_LANE_STATE	

11.4.3.1.339 (TRSV_REG0D4)

11.4.3.1.339.1 Offset

Register	Offset
TRSV_REG0D4	750h

11.4.3.1.339.2 Diagram



11.4.3.1.339.3 Fields

Field	Function
7-4 —	Reserved
3-0	

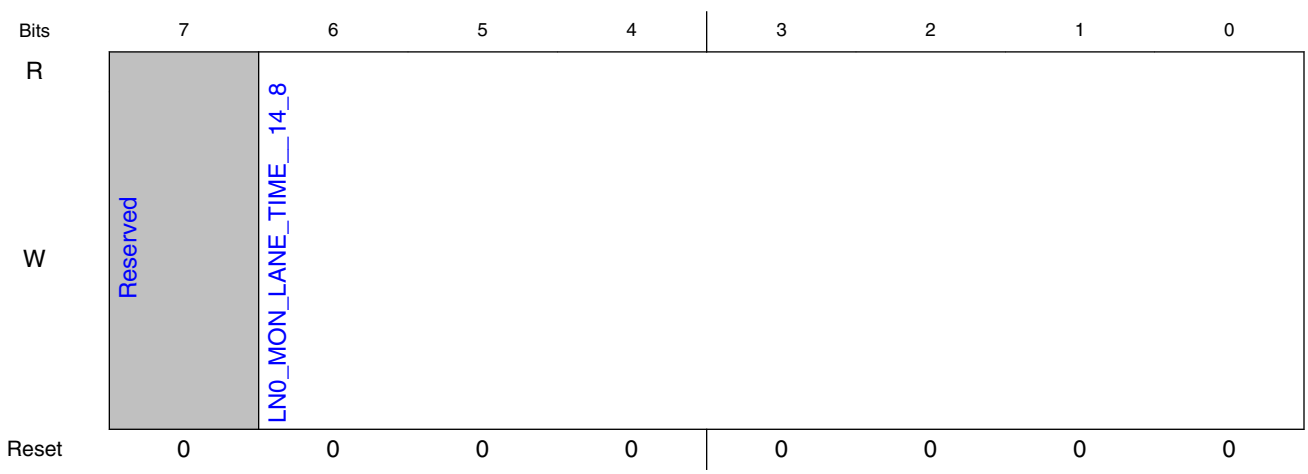
Field	Function
LN0_MON_CDR_STATE	

11.4.3.1.340 (TRSV_REG0D5)

11.4.3.1.340.1 Offset

Register	Offset
TRSV_REG0D5	754h

11.4.3.1.340.2 Diagram



11.4.3.1.340.3 Fields

Field	Function
7 —	Reserved
6-0 LN0_MON_LANE_TIME__14_8	

11.4.3.1.341 (TRSV_REG0D6)

11.4.3.1.341.1 Offset

Register	Offset
TRSV_REG0D6	758h

11.4.3.1.341.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_MON_LANE_TIME__7_0							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.341.3 Fields

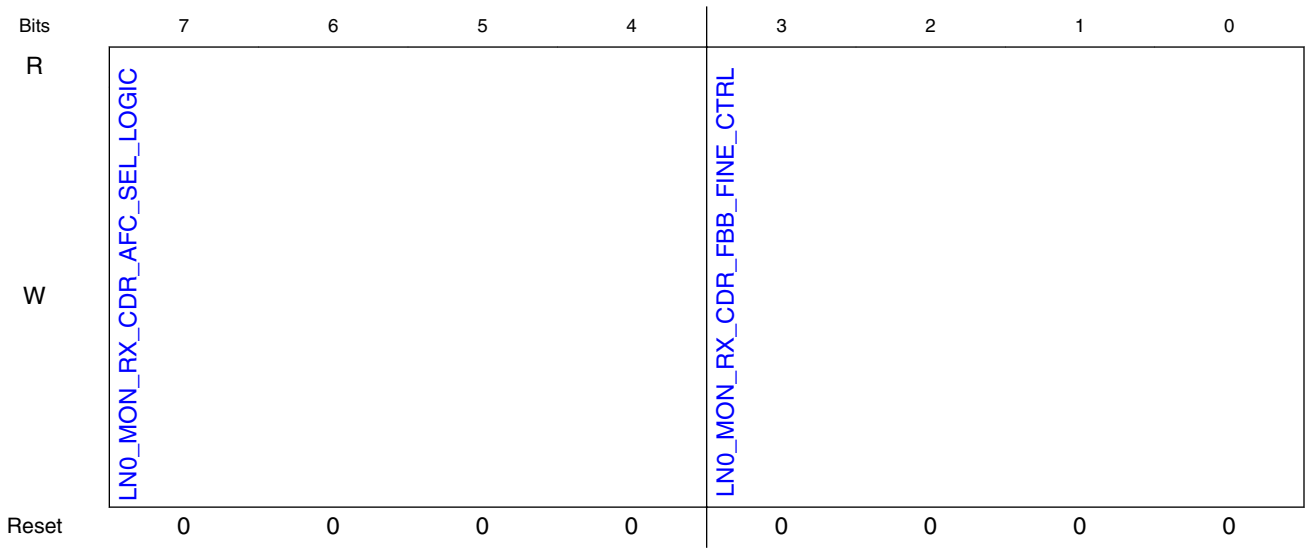
Field	Function
7-0 LN0_MON_LANE_TIME__7_0	

11.4.3.1.342 (TRSV_REG0D7)

11.4.3.1.342.1 Offset

Register	Offset
TRSV_REG0D7	75Ch

11.4.3.1.342.2 Diagram



11.4.3.1.342.3 Fields

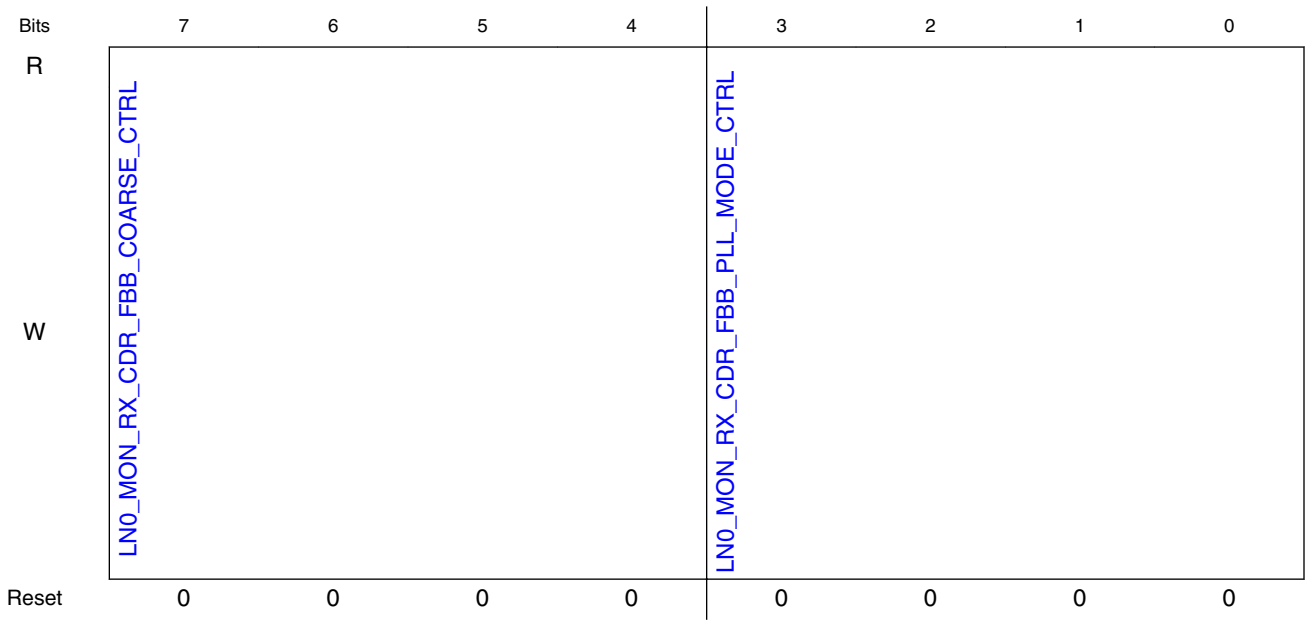
Field	Function
7-4 LN0_MON_RX_CDR_AFC_SEL_LOGIC	
3-0 LN0_MON_RX_CDR_FBB_FINE_CTRL	

11.4.3.1.343 (TRSV_REG0D8)

11.4.3.1.343.1 Offset

Register	Offset
TRSV_REG0D8	760h

11.4.3.1.343.2 Diagram



11.4.3.1.343.3 Fields

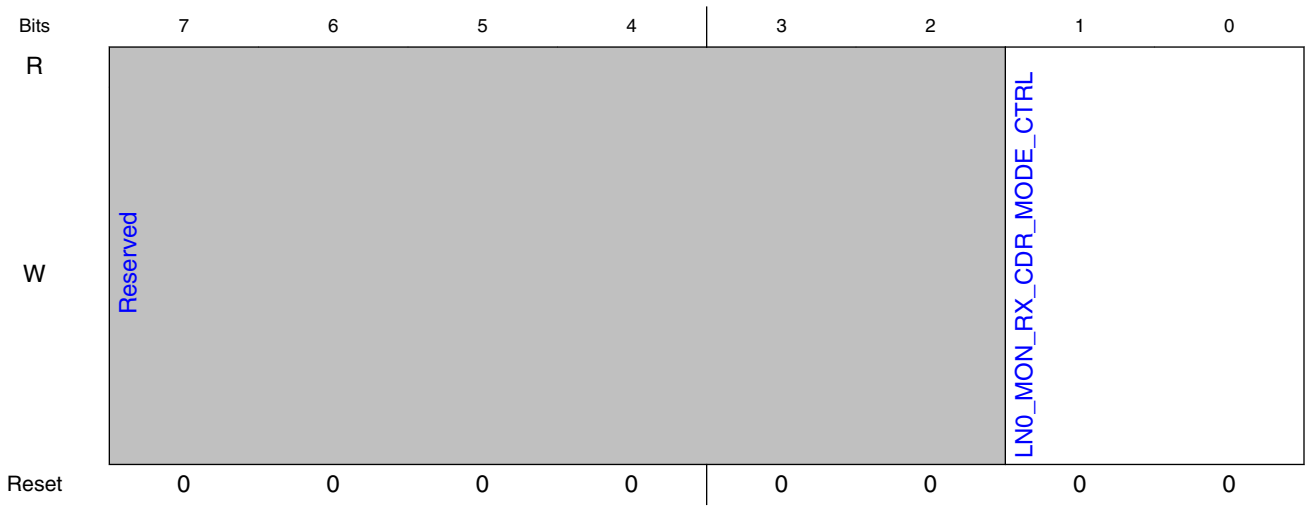
Field	Function
7-4 LN0_MON_RX_CDR_FBB_COARSE_CTRL	
3-0 LN0_MON_RX_CDR_FBB_PLL_MODE_CTRL	

11.4.3.1.344 (TRSV_REG0D9)

11.4.3.1.344.1 Offset

Register	Offset
TRSV_REG0D9	764h

11.4.3.1.344.2 Diagram



11.4.3.1.344.3 Fields

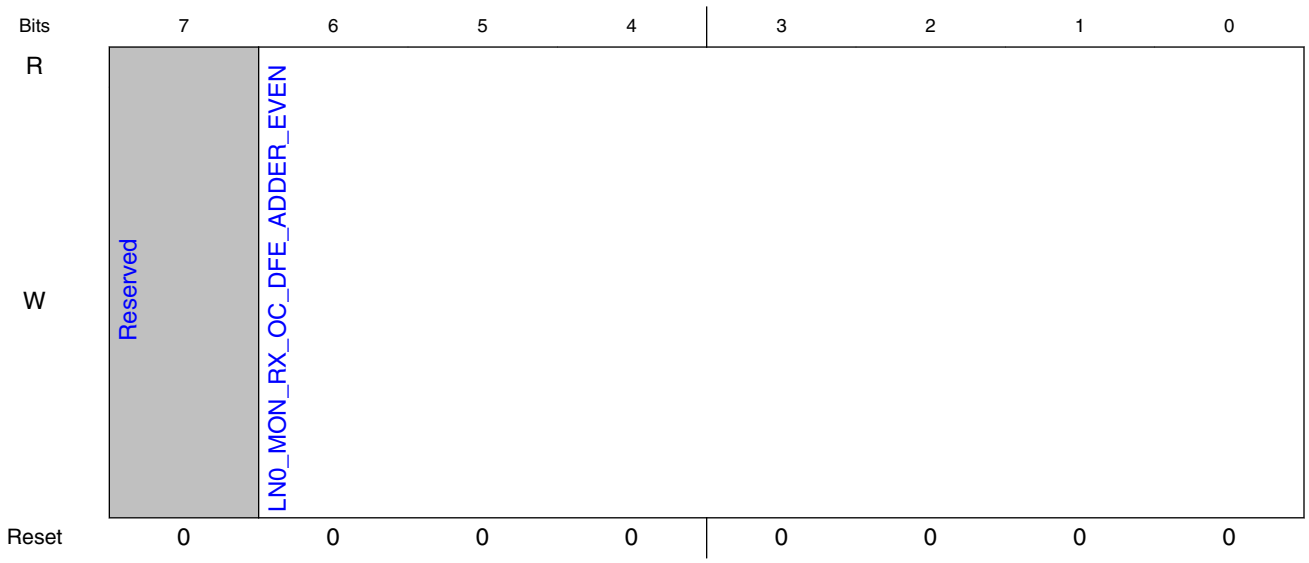
Field	Function
7-2 —	Reserved
1-0 LN0_MON_RX_CDR_MODE_CTRL	

11.4.3.1.345 (TRSV_REG0DA)

11.4.3.1.345.1 Offset

Register	Offset
TRSV_REG0DA	768h

11.4.3.1.345.2 Diagram



11.4.3.1.345.3 Fields

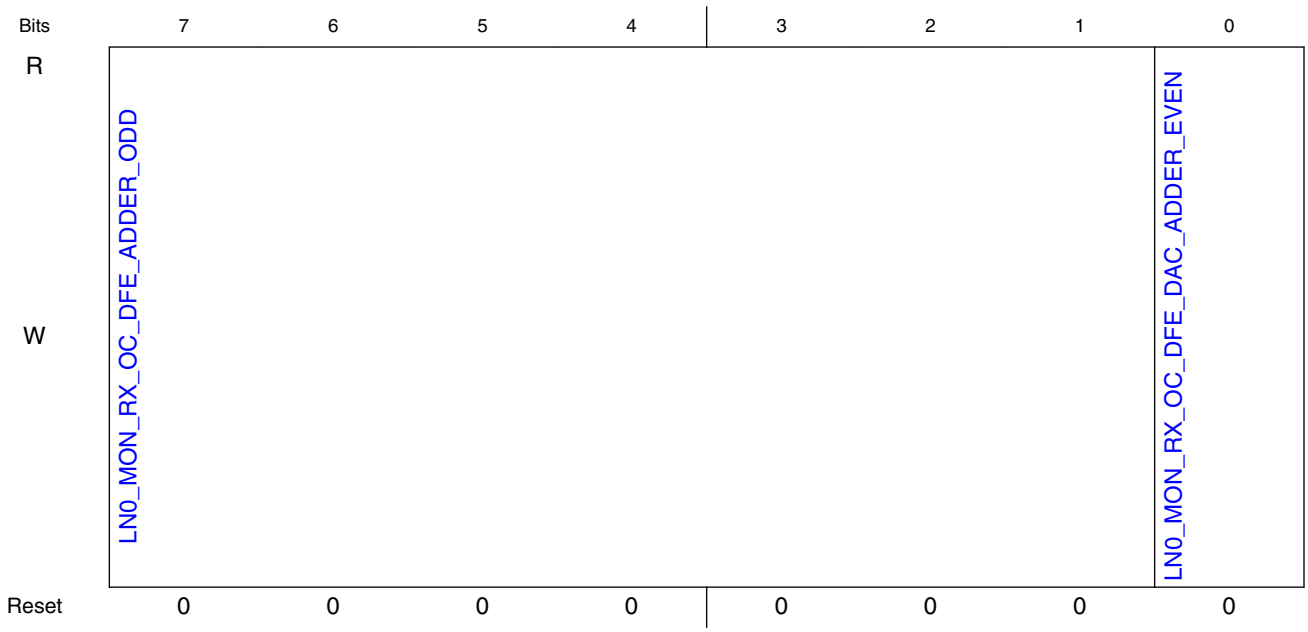
Field	Function
7 —	Reserved
6-0 LN0_MON_RX_OC_DFE_ADDER_EVEN	

11.4.3.1.346 (TRSV_REG0DB)

11.4.3.1.346.1 Offset

Register	Offset
TRSV_REG0DB	76Ch

11.4.3.1.346.2 Diagram



11.4.3.1.346.3 Fields

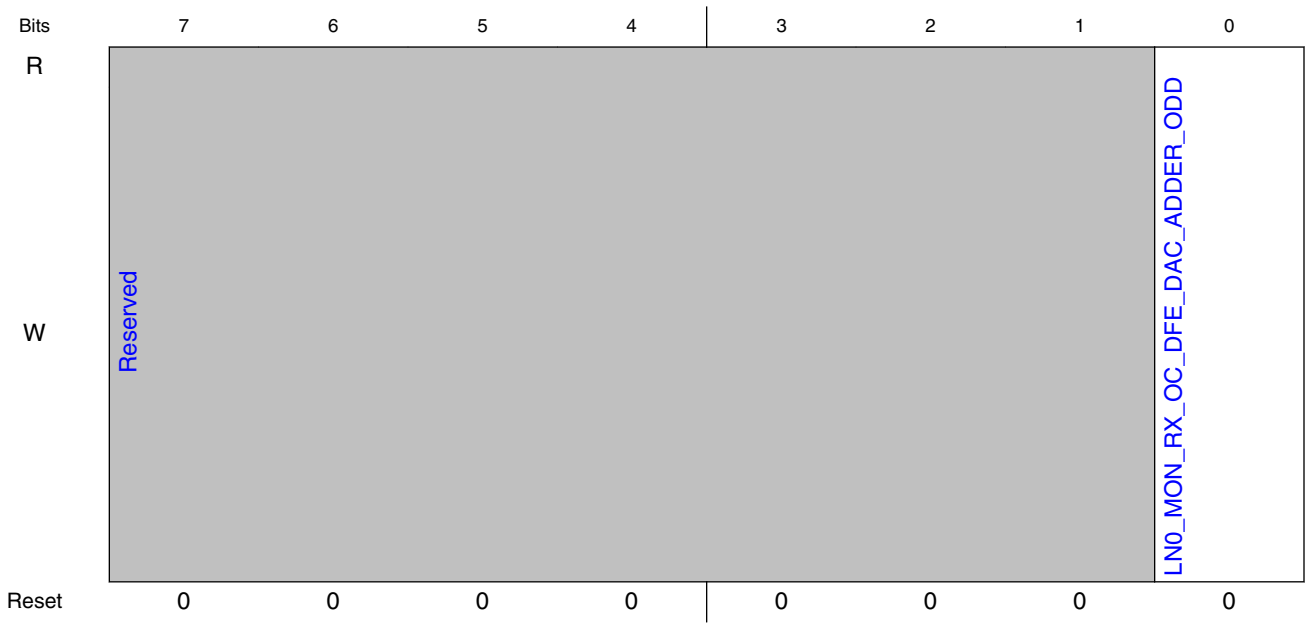
Field	Function
7-1 LN0_MON_RX_OC_DFE_ADDER_ODD	
0 LN0_MON_RX_OC_DFE_DAC_ADDER_EVEN	

11.4.3.1.347 (TRSV_REG0DC)

11.4.3.1.347.1 Offset

Register	Offset
TRSV_REG0DC	770h

11.4.3.1.347.2 Diagram



11.4.3.1.347.3 Fields

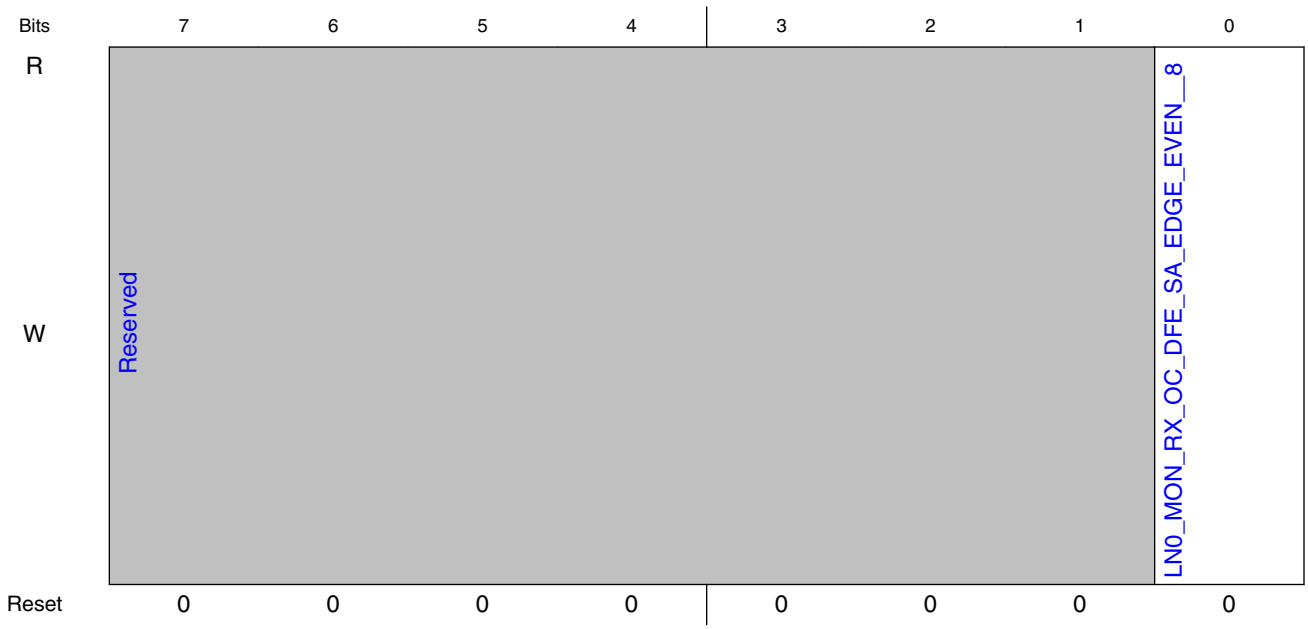
Field	Function
7-1 —	Reserved
0 LN0_MON_RX_OC_DFE_DAC_ADDER_ODD	

11.4.3.1.348 (TRSV_REG0DD)

11.4.3.1.348.1 Offset

Register	Offset
TRSV_REG0DD	774h

11.4.3.1.348.2 Diagram



11.4.3.1.348.3 Fields

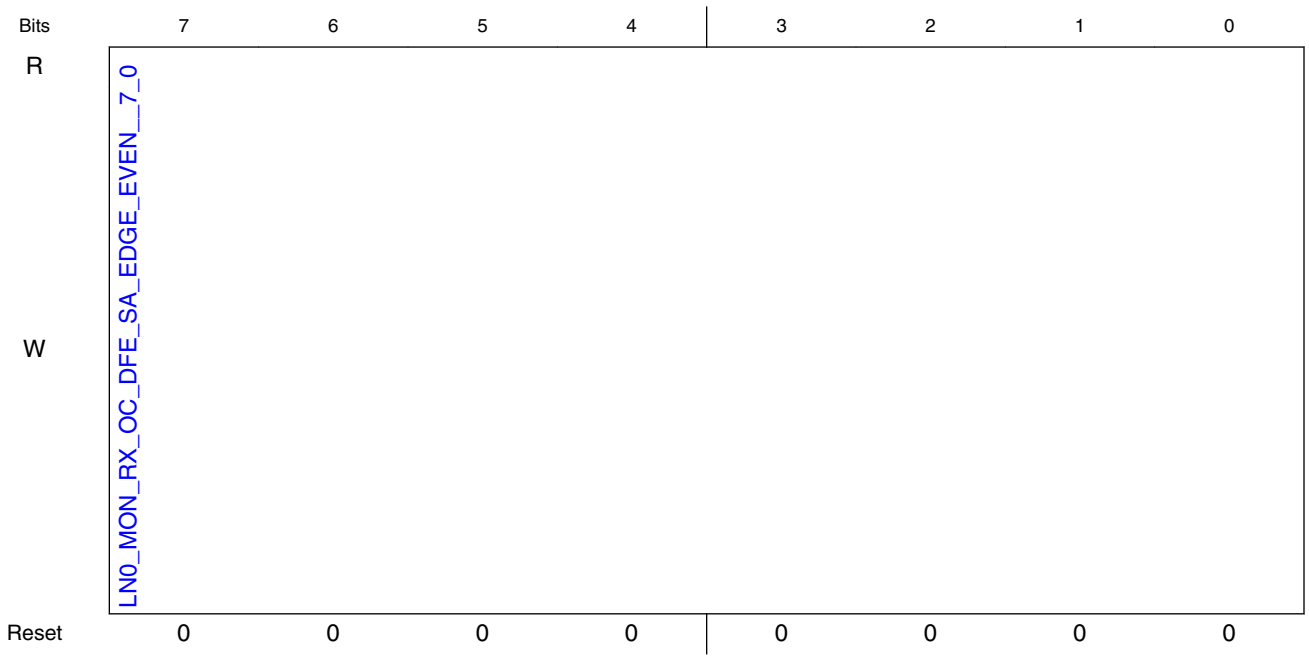
Field	Function
7-1 —	Reserved
0 LN0_MON_RX_OC_DFE_SA_EDGE_EVEN__8	

11.4.3.1.349 (TRSV_REG0DE)

11.4.3.1.349.1 Offset

Register	Offset
TRSV_REG0DE	778h

11.4.3.1.349.2 Diagram



11.4.3.1.349.3 Fields

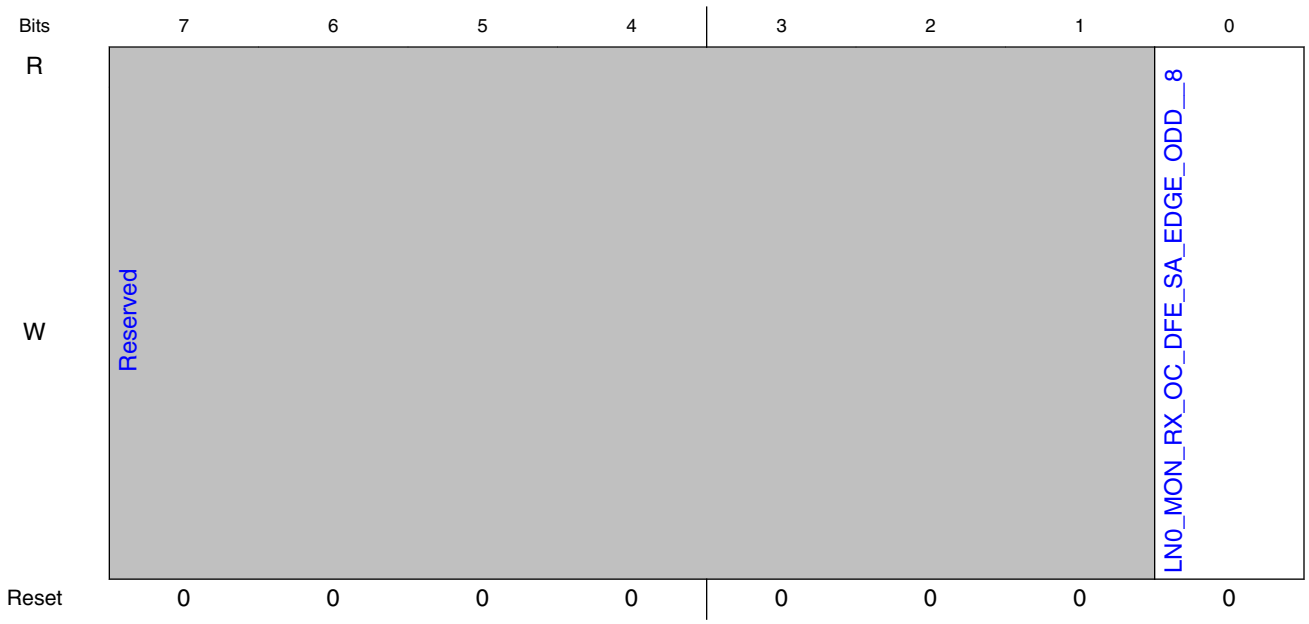
Field	Function
7-0 LN0_MON_RX_OC_DFE_SA_EDGE_EVEN__7_0	

11.4.3.1.350 (TRSV_REG0DF)

11.4.3.1.350.1 Offset

Register	Offset
TRSV_REG0DF	77Ch

11.4.3.1.350.2 Diagram



11.4.3.1.350.3 Fields

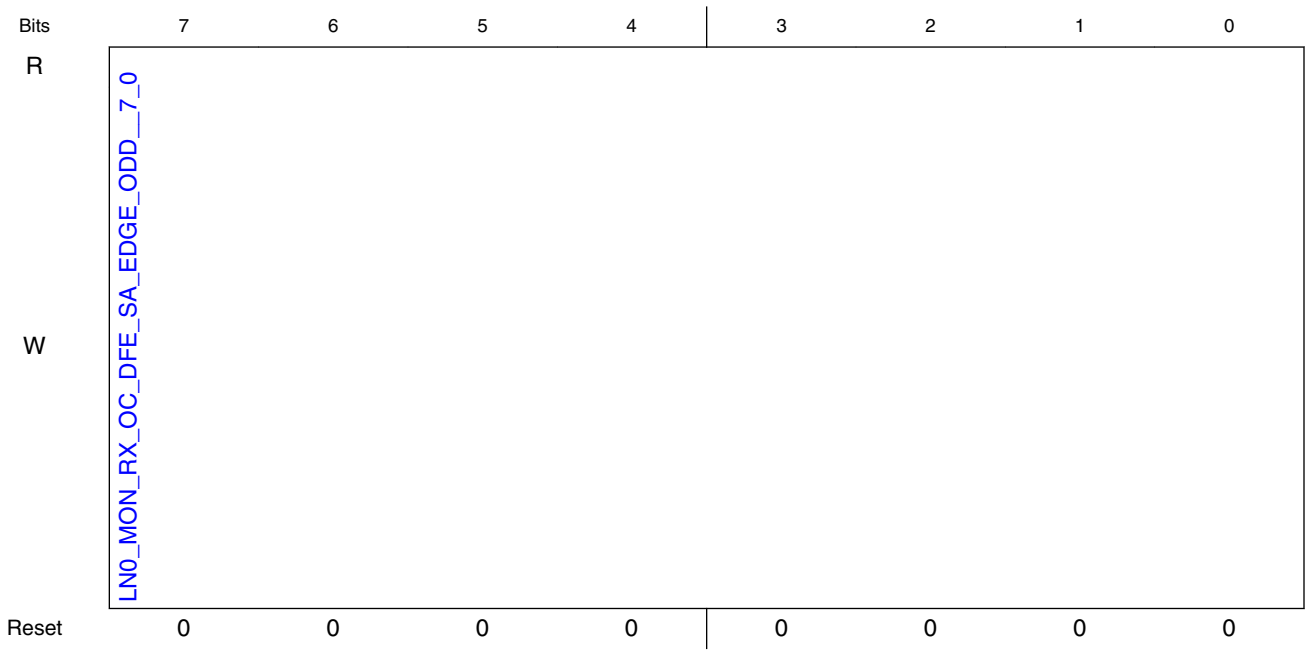
Field	Function
7-1 —	Reserved
0 LN0_MON_RX_OC_DFE_SA_EDGE_ODD__8	

11.4.3.1.351 (TRSV_REG0E0)

11.4.3.1.351.1 Offset

Register	Offset
TRSV_REG0E0	780h

11.4.3.1.351.2 Diagram



11.4.3.1.351.3 Fields

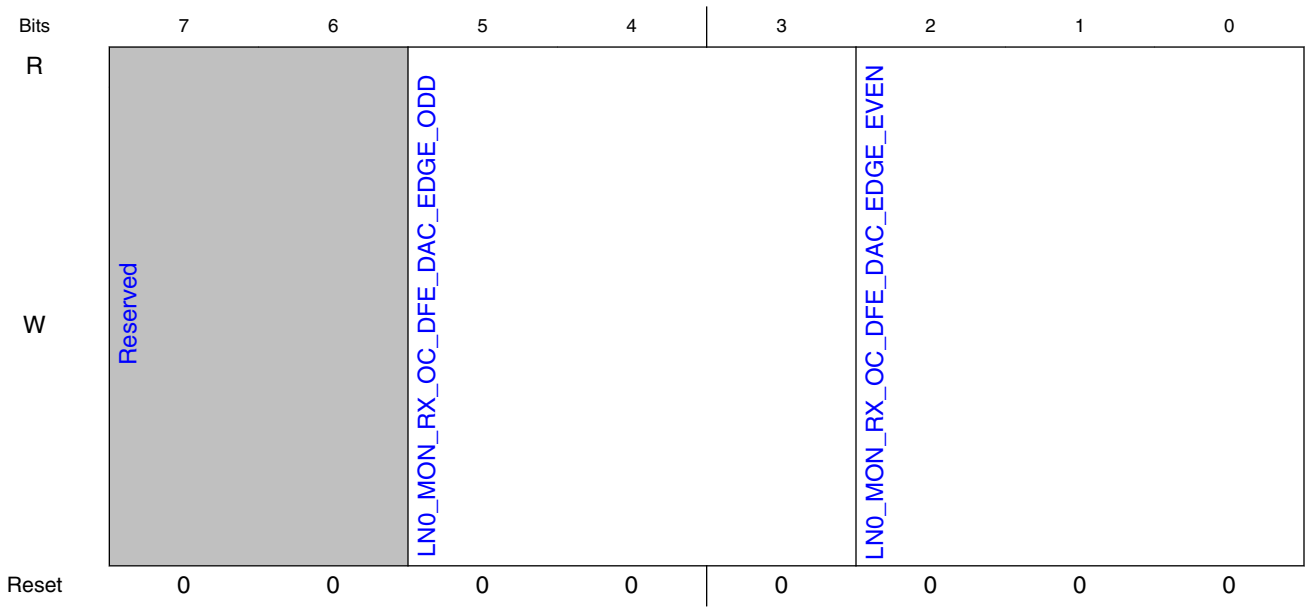
Field	Function
7-0 LN0_MON_RX_OC_DFE_SA_EDGE_ODD__7_0	

11.4.3.1.352 (TRSV_REG0E1)

11.4.3.1.352.1 Offset

Register	Offset
TRSV_REG0E1	784h

11.4.3.1.352.2 Diagram



11.4.3.1.352.3 Fields

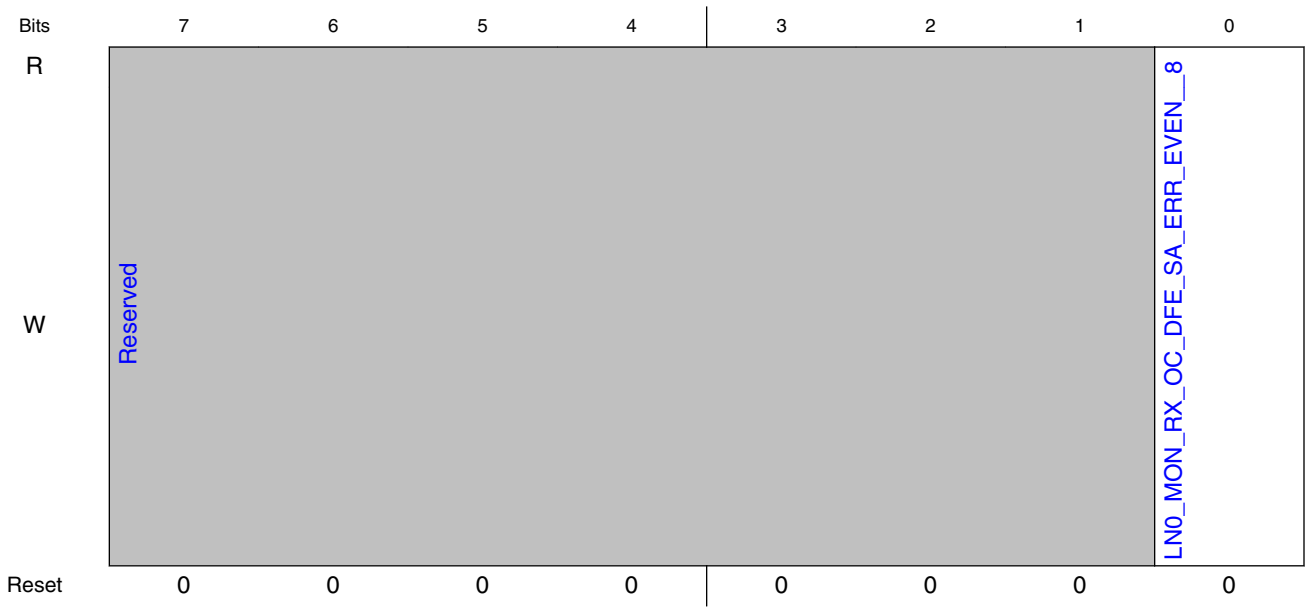
Field	Function
7-6 —	Reserved
5-3 LN0_MON_RX_OC_DFE_DAC_EDGE_ODD	
2-0 LN0_MON_RX_OC_DFE_DAC_EDGE_EVEN	

11.4.3.1.353 (TRSV_REG0E2)

11.4.3.1.353.1 Offset

Register	Offset
TRSV_REG0E2	788h

11.4.3.1.353.2 Diagram



11.4.3.1.353.3 Fields

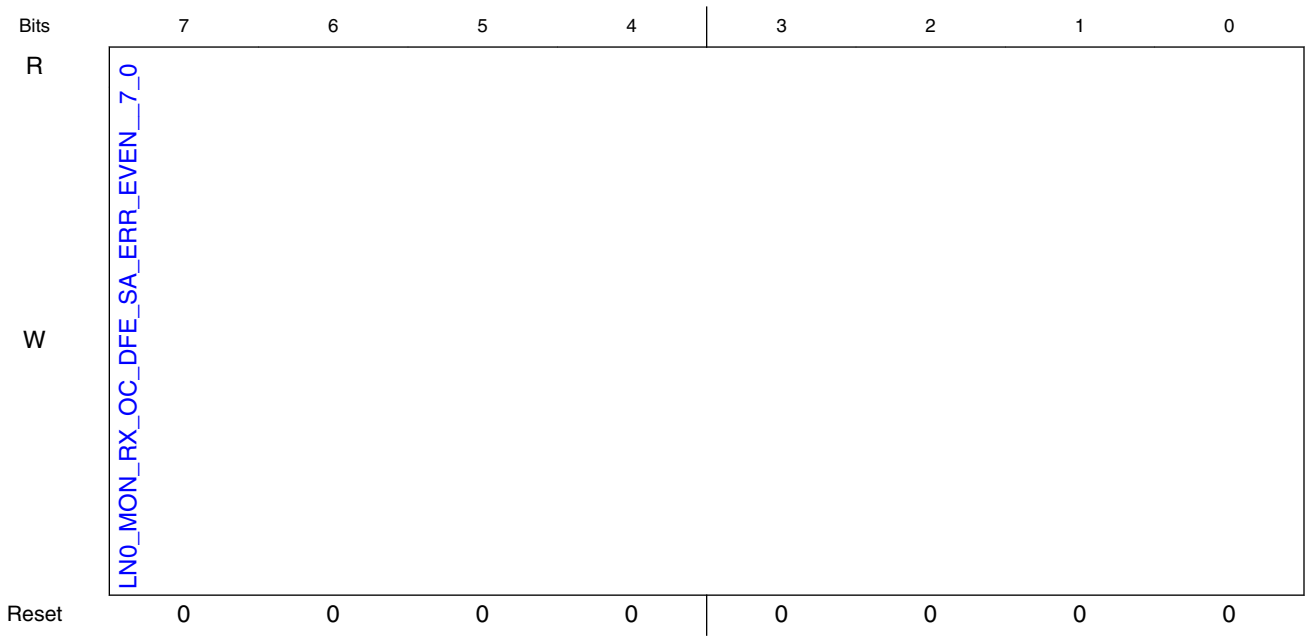
Field	Function
7-1 —	Reserved
0 LN0_MON_RX_OC_DFE_SA_ERR_EVEN__8	

11.4.3.1.354 (TRSV_REG0E3)

11.4.3.1.354.1 Offset

Register	Offset
TRSV_REG0E3	78Ch

11.4.3.1.354.2 Diagram



11.4.3.1.354.3 Fields

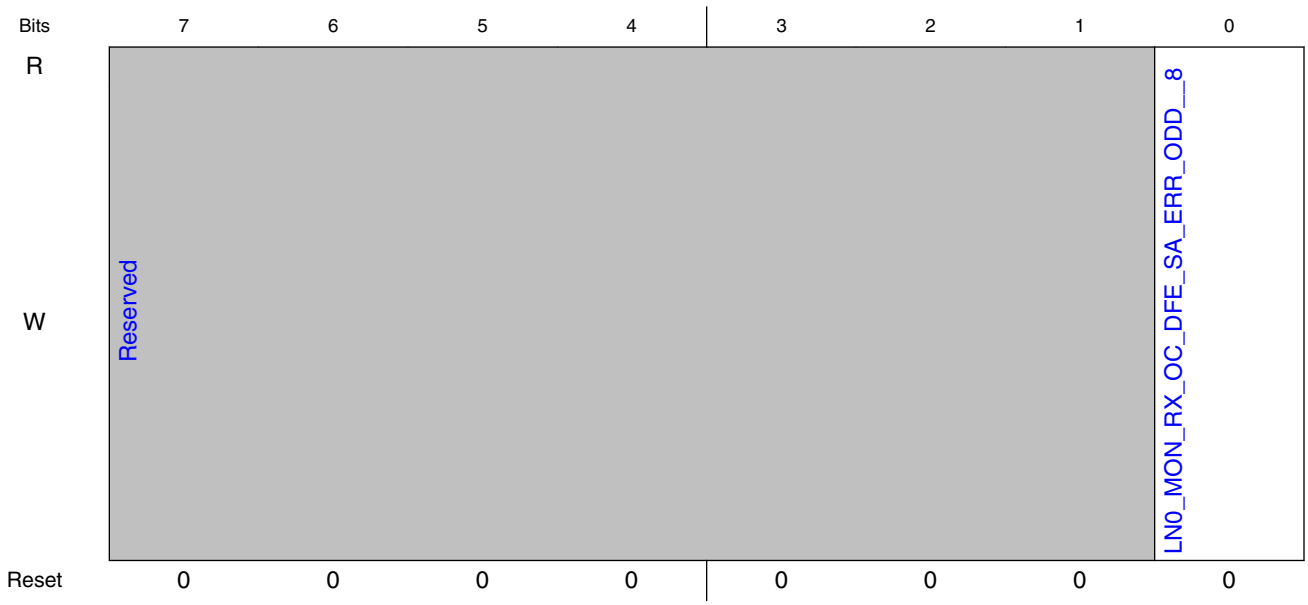
Field	Function
7-0 LN0_MON_RX_OC_DFE_SA_ERR_EVEN__7_0	

11.4.3.1.355 (TRSV_REG0E4)

11.4.3.1.355.1 Offset

Register	Offset
TRSV_REG0E4	790h

11.4.3.1.355.2 Diagram



11.4.3.1.355.3 Fields

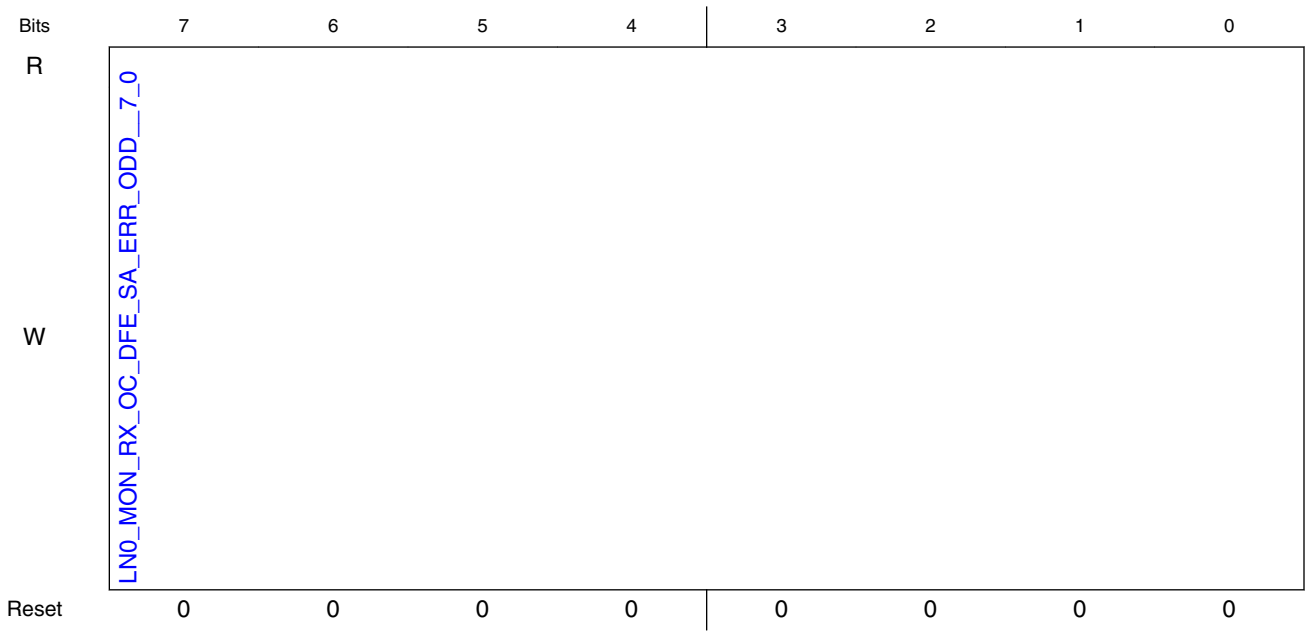
Field	Function
7-1 —	Reserved
0 LN0_MON_RX_OC_DFE_SA_ERR_ODD__8	

11.4.3.1.356 (TRSV_REG0E5)

11.4.3.1.356.1 Offset

Register	Offset
TRSV_REG0E5	794h

11.4.3.1.356.2 Diagram



11.4.3.1.356.3 Fields

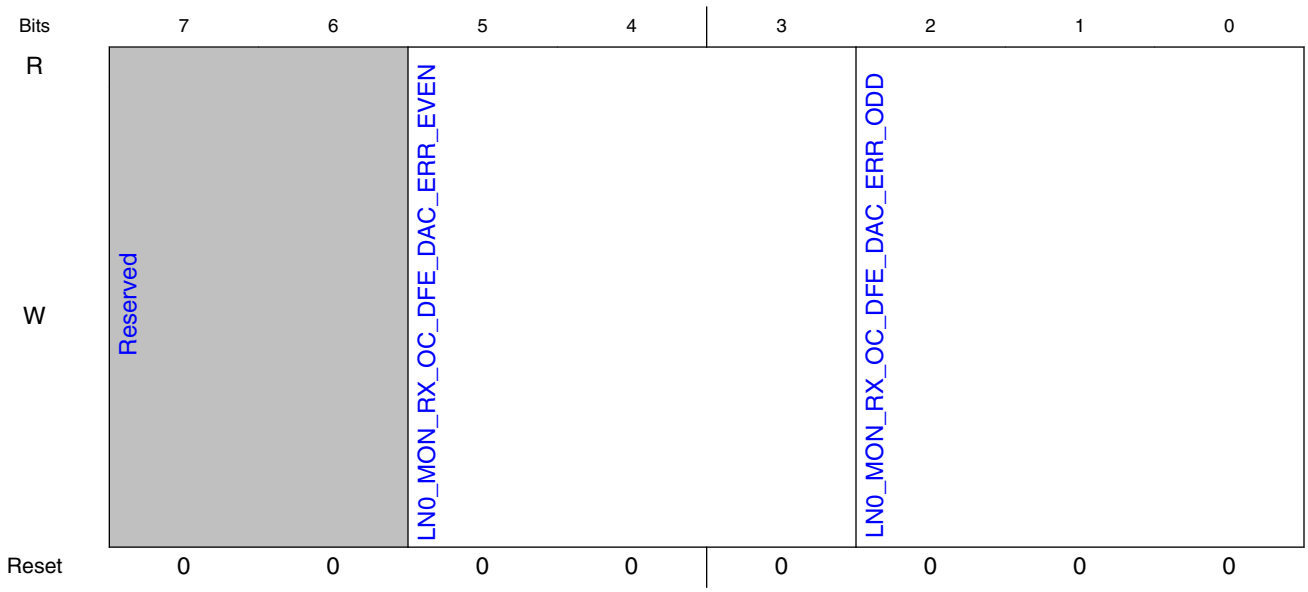
Field	Function
7-0 LN0_MON_RX_OC_DFE_SA_ERR_ODD__7_0	

11.4.3.1.357 (TRSV_REG0E6)

11.4.3.1.357.1 Offset

Register	Offset
TRSV_REG0E6	798h

11.4.3.1.357.2 Diagram



11.4.3.1.357.3 Fields

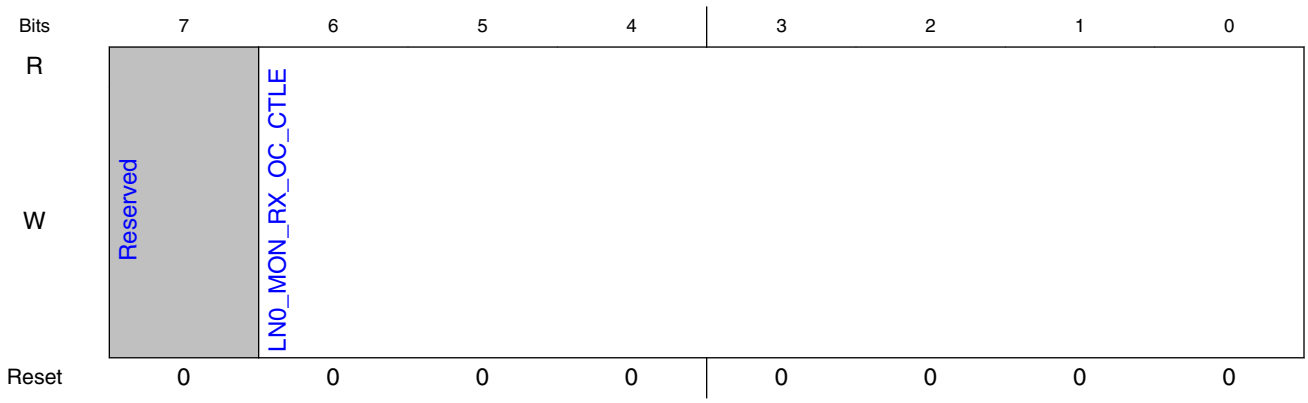
Field	Function
7-6 —	Reserved
5-3 LN0_MON_RX_OC_DFE_DAC_ERR_EVEN	
2-0 LN0_MON_RX_OC_DFE_DAC_ERR_ODD	

11.4.3.1.358 (TRSV_REG0E7)

11.4.3.1.358.1 Offset

Register	Offset
TRSV_REG0E7	79Ch

11.4.3.1.358.2 Diagram



11.4.3.1.358.3 Fields

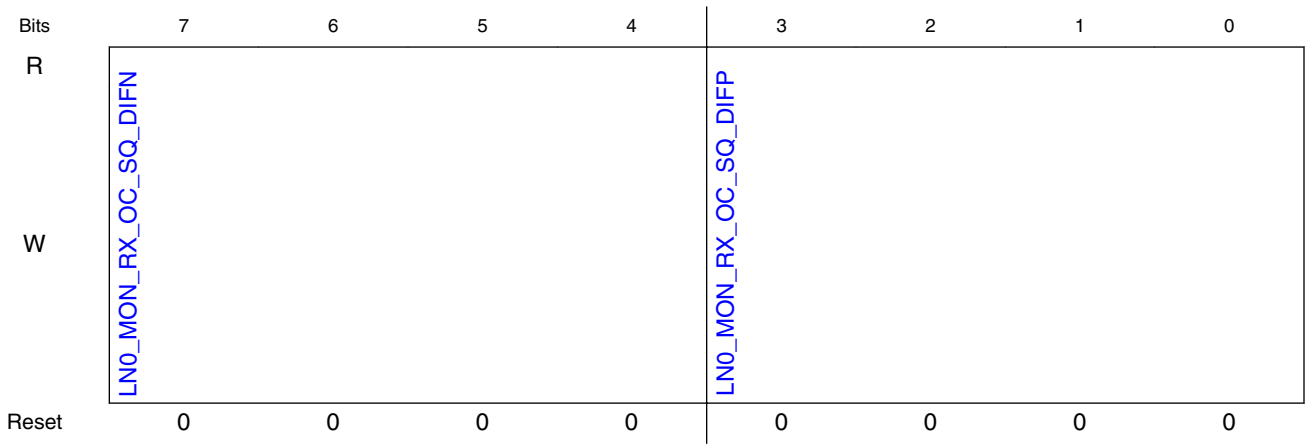
Field	Function
7 —	Reserved
6-0 LN0_MON_RX_OC_CTLE	

11.4.3.1.359 (TRSV_REG0E8)

11.4.3.1.359.1 Offset

Register	Offset
TRSV_REG0E8	7A0h

11.4.3.1.359.2 Diagram



11.4.3.1.359.3 Fields

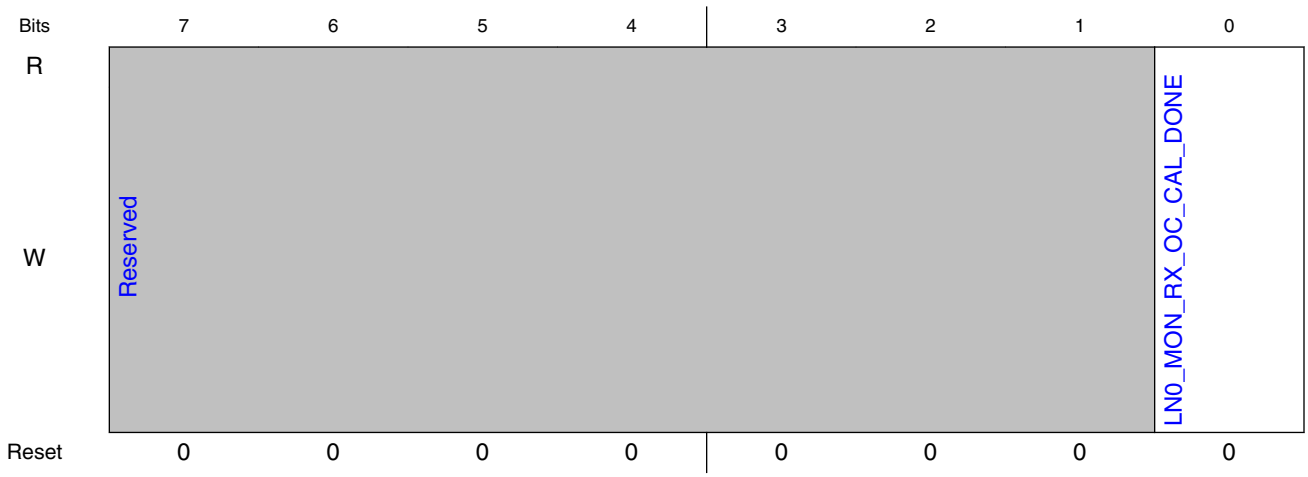
Field	Function
7-4 LN0_MON_RX_OC_SQ_DIFN	
3-0 LN0_MON_RX_OC_SQ_DIFP	

11.4.3.1.360 (TRSV_REG0E9)

11.4.3.1.360.1 Offset

Register	Offset
TRSV_REG0E9	7A4h

11.4.3.1.360.2 Diagram



11.4.3.1.360.3 Fields

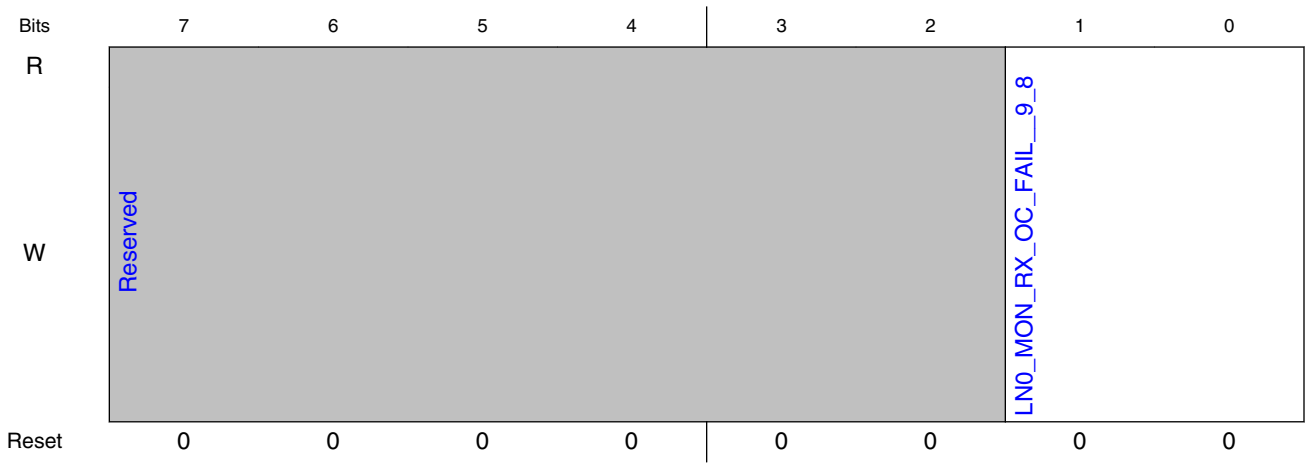
Field	Function
7-1 —	Reserved
0 LN0_MON_RX_OC_CAL_DONE	

11.4.3.1.361 (TRSV_REG0EA)

11.4.3.1.361.1 Offset

Register	Offset
TRSV_REG0EA	7A8h

11.4.3.1.361.2 Diagram



11.4.3.1.361.3 Fields

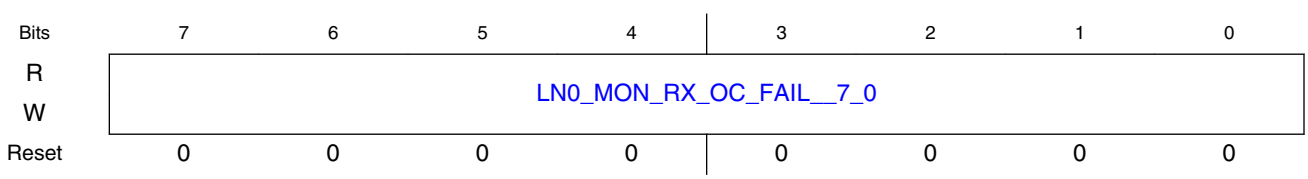
Field	Function
7-2 —	Reserved
1-0 LN0_MON_RX_OC_FAIL__9_8	

11.4.3.1.362 (TRSV_REG0EB)

11.4.3.1.362.1 Offset

Register	Offset
TRSV_REG0EB	7ACh

11.4.3.1.362.2 Diagram



11.4.3.1.362.3 Fields

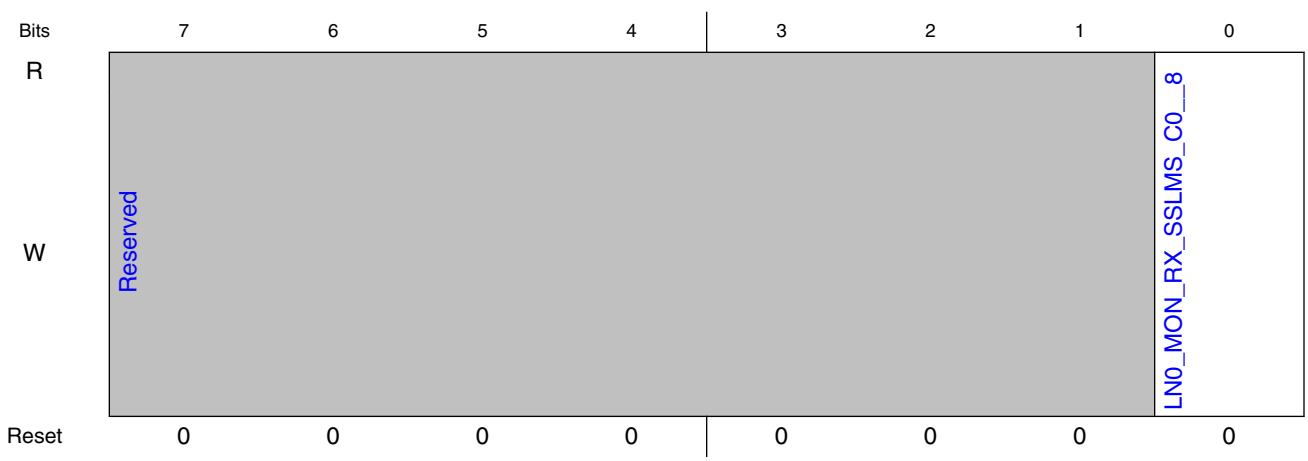
Field	Function
7-0 LN0_MON_RX_OC_FAIL__7_0	

11.4.3.1.363 (TRSV_REG0EC)

11.4.3.1.363.1 Offset

Register	Offset
TRSV_REG0EC	7B0h

11.4.3.1.363.2 Diagram



11.4.3.1.363.3 Fields

Field	Function
7-1 —	Reserved
0 LN0_MON_RX_SSLMS_C0__8	

11.4.3.1.364 (TRSV_REG0ED)

11.4.3.1.364.1 Offset

Register	Offset
TRSV_REG0ED	7B4h

11.4.3.1.364.2 Diagram

Bits	7	6	5	4	3	2	1	0
R W	LN0_MON_RX_SSLMS_C0__7_0							
Reset	0	0	0	0	0	0	0	0

11.4.3.1.364.3 Fields

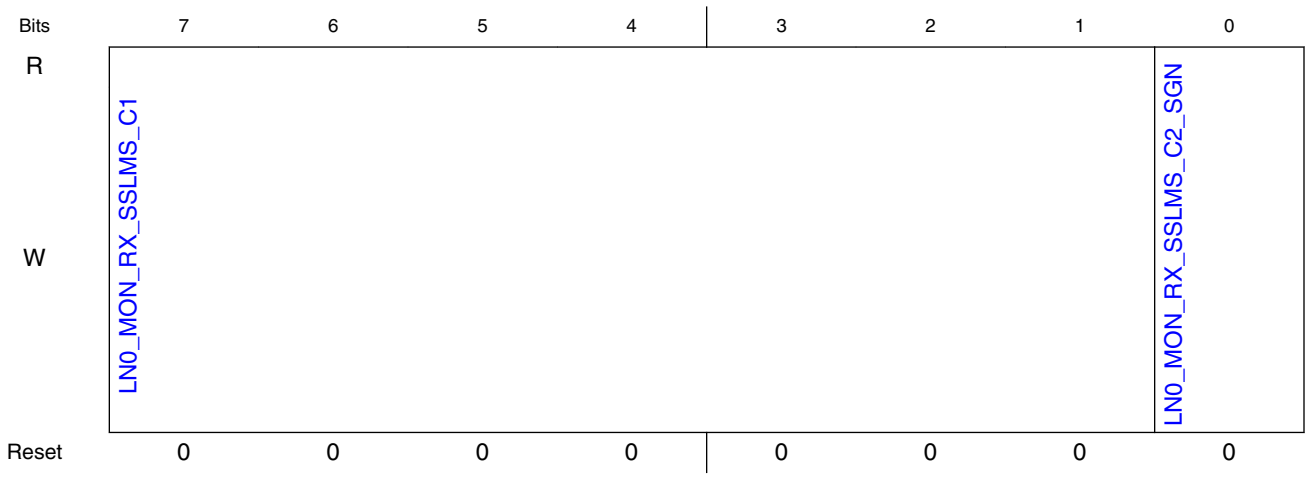
Field	Function
7-0 LN0_MON_RX_SSLMS_C0__7_0	

11.4.3.1.365 (TRSV_REG0EE)

11.4.3.1.365.1 Offset

Register	Offset
TRSV_REG0EE	7B8h

11.4.3.1.365.2 Diagram



11.4.3.1.365.3 Fields

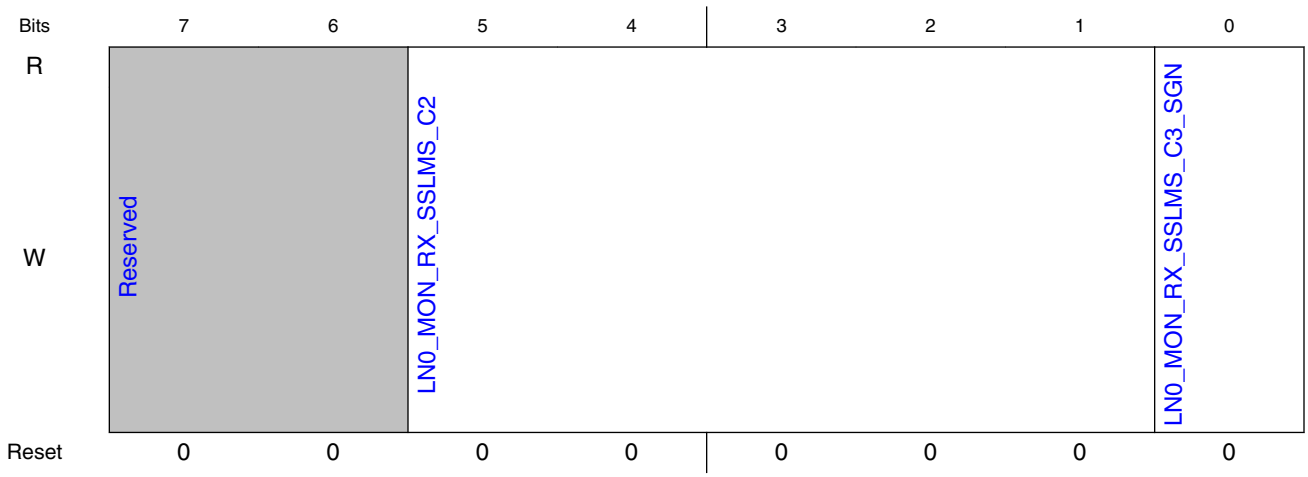
Field	Function
7-1 LN0_MON_RX_SSLMS_C1	
0 LN0_MON_RX_SSLMS_C2_SGN	

11.4.3.1.366 (TRSV_REG0EF)

11.4.3.1.366.1 Offset

Register	Offset
TRSV_REG0EF	7BCh

11.4.3.1.366.2 Diagram



11.4.3.1.366.3 Fields

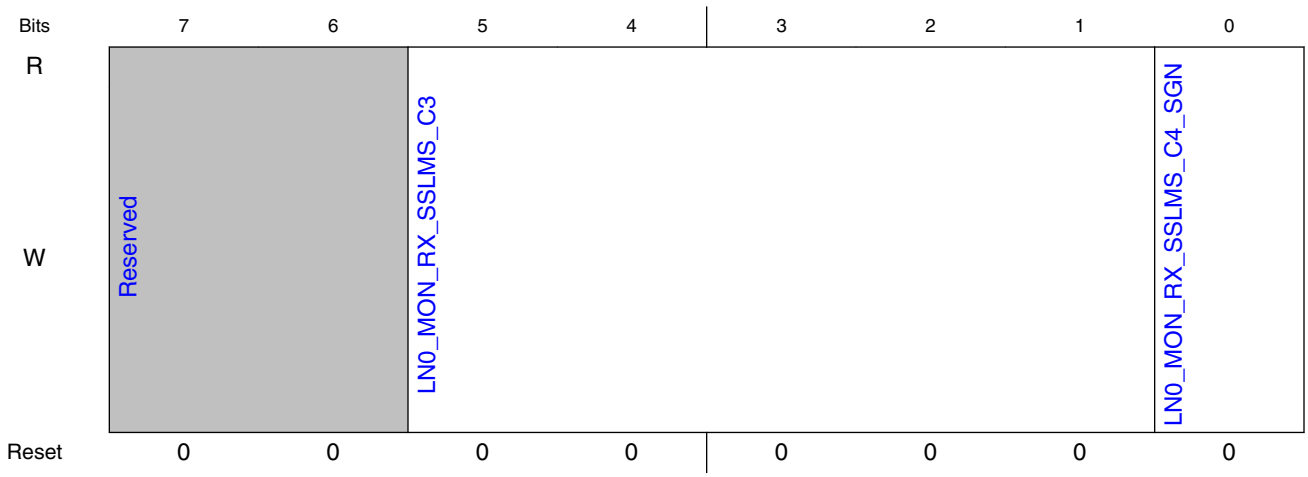
Field	Function
7-6 —	Reserved
5-1 LN0_MON_RX_SSLMS_C2	
0 LN0_MON_RX_SSLMS_C3_SGN	

11.4.3.1.367 (TRSV_REG0F0)

11.4.3.1.367.1 Offset

Register	Offset
TRSV_REG0F0	7C0h

11.4.3.1.367.2 Diagram



11.4.3.1.367.3 Fields

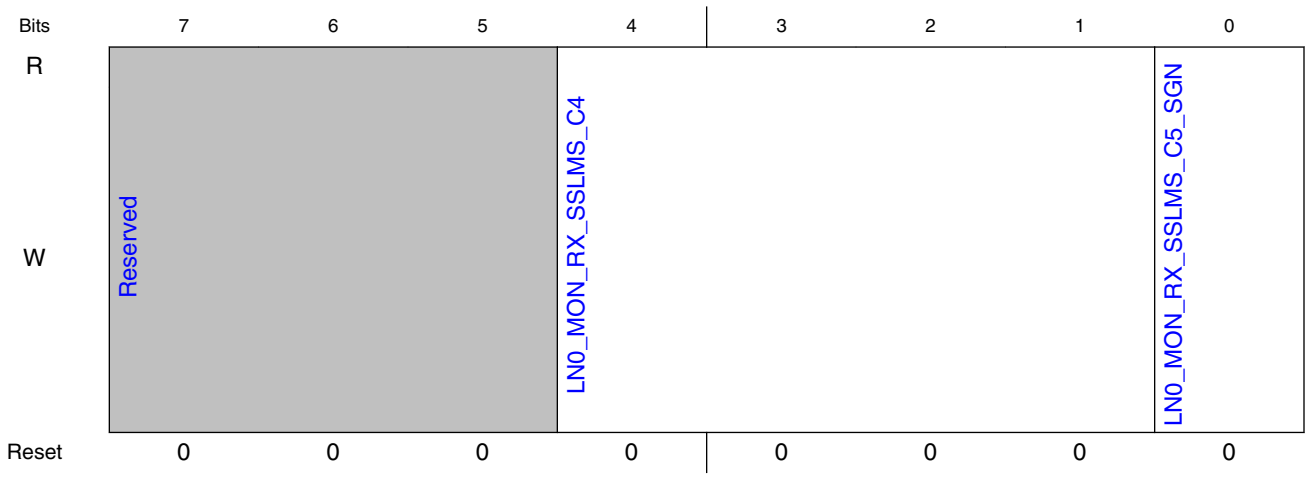
Field	Function
7-6 —	Reserved
5-1 LN0_MON_RX_SSLMS_C3	
0 LN0_MON_RX_SSLMS_C4_SGN	

11.4.3.1.368 (TRSV_REG0F1)

11.4.3.1.368.1 Offset

Register	Offset
TRSV_REG0F1	7C4h

11.4.3.1.368.2 Diagram



11.4.3.1.368.3 Fields

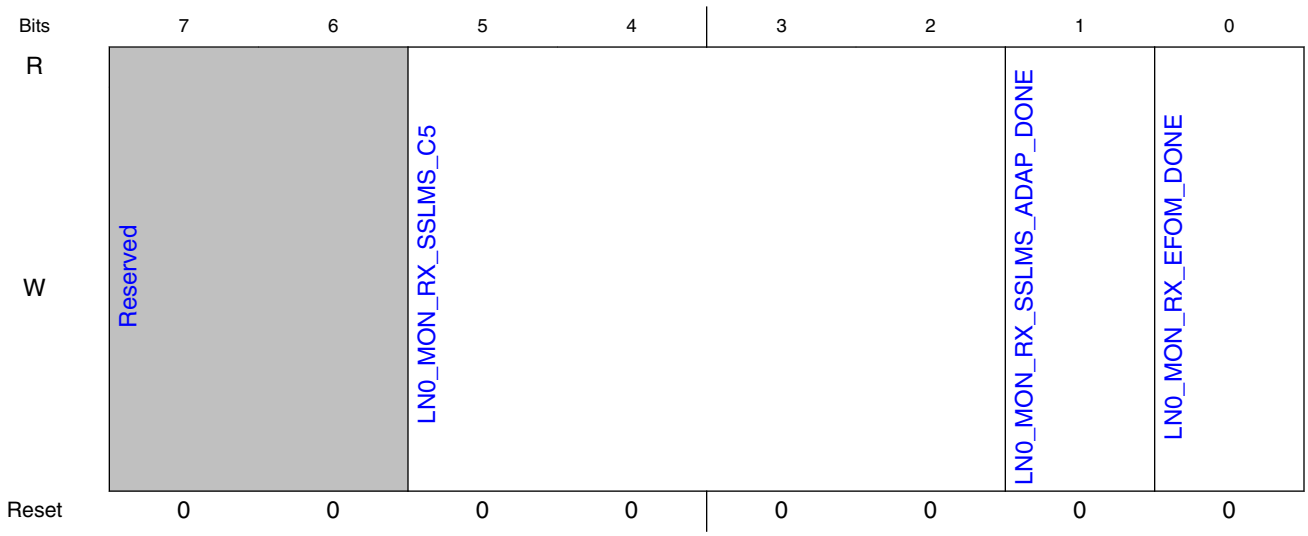
Field	Function
7-5 —	Reserved
4-1 LN0_MON_RX_SSLMS_C4	
0 LN0_MON_RX_SSLMS_C5_SGN	

11.4.3.1.369 (TRSV_REG0F2)

11.4.3.1.369.1 Offset

Register	Offset
TRSV_REG0F2	7C8h

11.4.3.1.369.2 Diagram



11.4.3.1.369.3 Fields

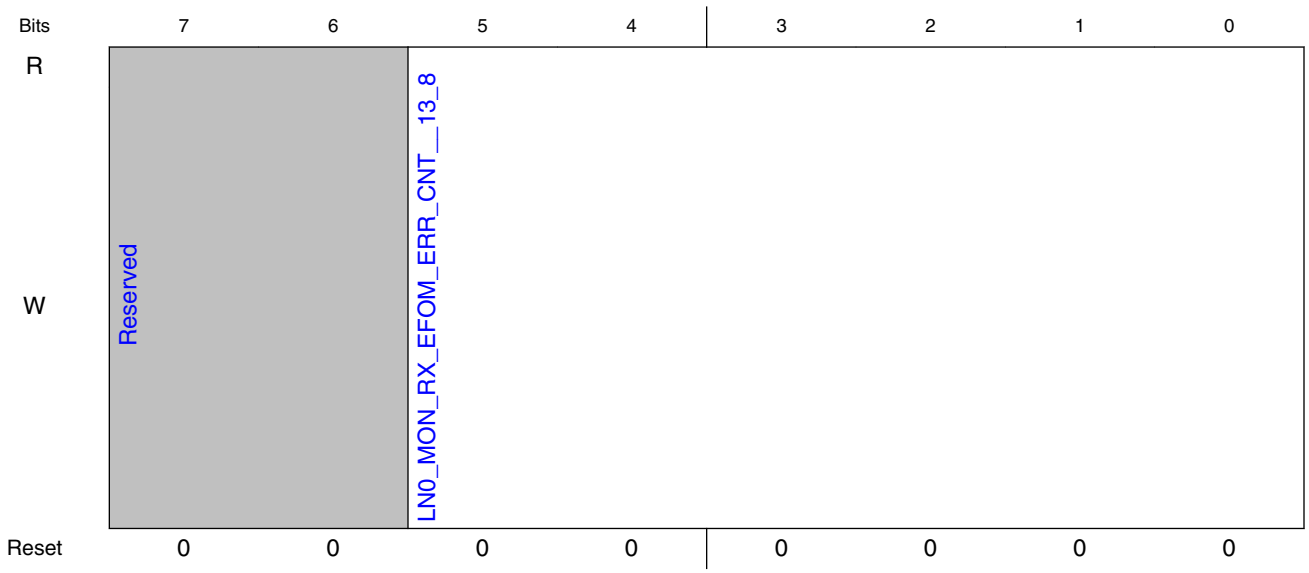
Field	Function
7-6 —	Reserved
5-2 LN0_MON_RX_SSLMS_C5	
1 LN0_MON_RX_SSLMS_ADAP_DONE	
0 LN0_MON_RX_EFOM_DONE	

11.4.3.1.370 (TRSV_REG0F3)

11.4.3.1.370.1 Offset

Register	Offset
TRSV_REG0F3	7CCh

11.4.3.1.370.2 Diagram



11.4.3.1.370.3 Fields

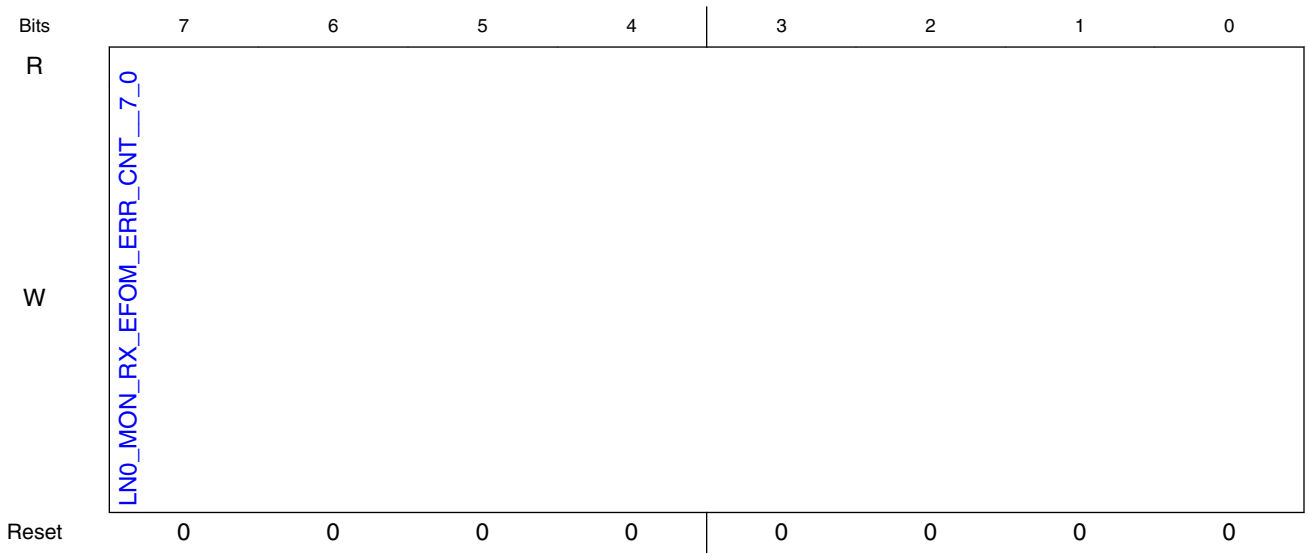
Field	Function
7-6 —	Reserved
5-0 LN0_MON_RX_EFOM_ERR_CNT__13_8	

11.4.3.1.371 (TRSV_REG0F4)

11.4.3.1.371.1 Offset

Register	Offset
TRSV_REG0F4	7D0h

11.4.3.1.371.2 Diagram



11.4.3.1.371.3 Fields

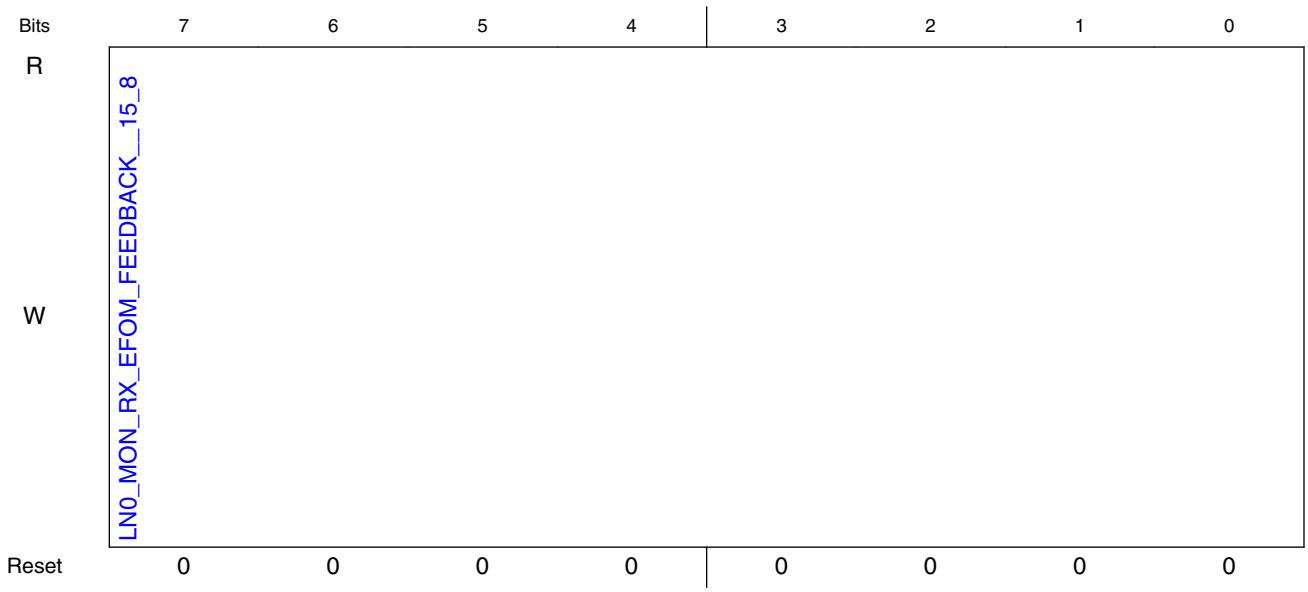
Field	Function
7-0 LN0_MON_RX_EFOM_ERR_CNT__7_0	

11.4.3.1.372 (TRSV_REG0F5)

11.4.3.1.372.1 Offset

Register	Offset
TRSV_REG0F5	7D4h

11.4.3.1.372.2 Diagram



11.4.3.1.372.3 Fields

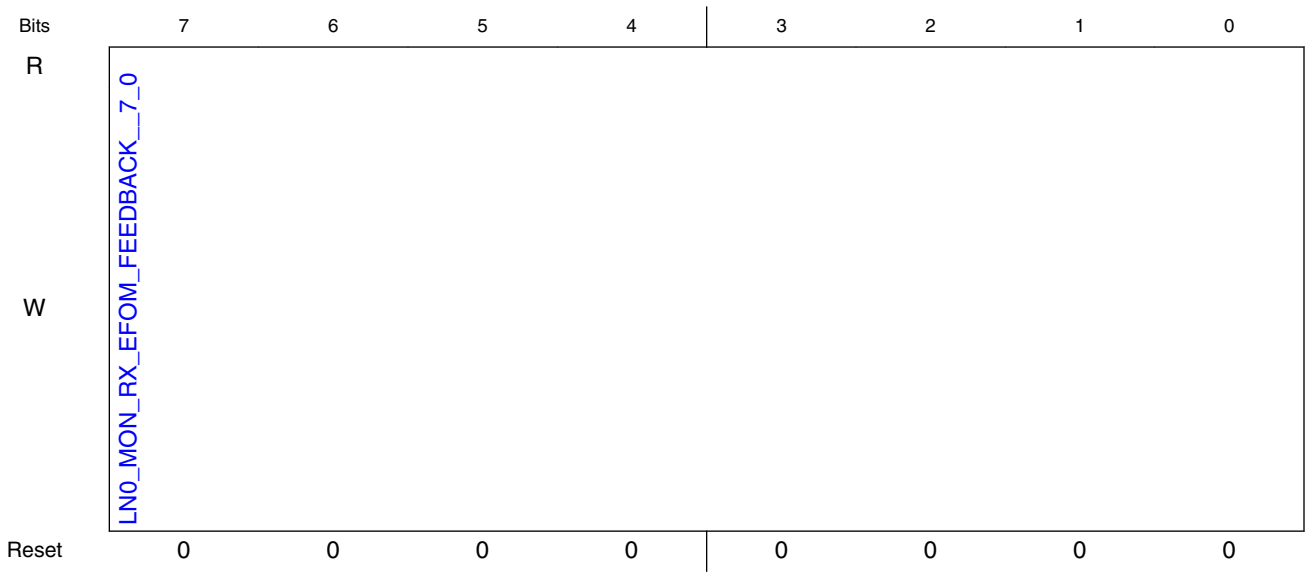
Field	Function
7-0 LN0_MON_RX_EFOM_FEEDBACK__15_8	

11.4.3.1.373 (TRSV_REG0F6)

11.4.3.1.373.1 Offset

Register	Offset
TRSV_REG0F6	7D8h

11.4.3.1.373.2 Diagram



11.4.3.1.373.3 Fields

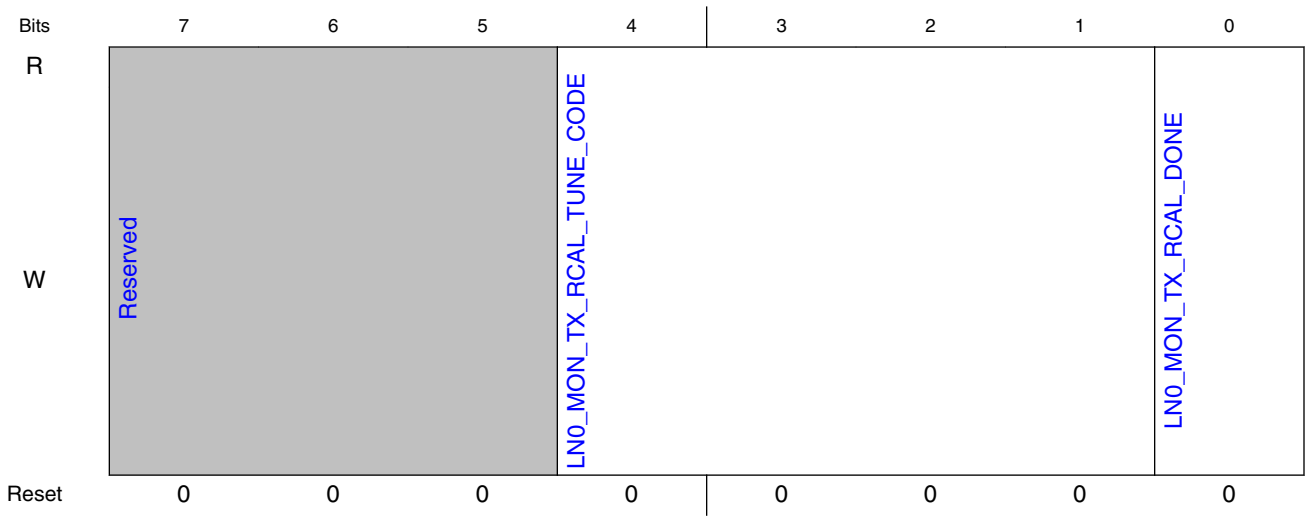
Field	Function
7-0 LN0_MON_RX_EFOM_FEEDBACK__7_0	

11.4.3.1.374 (TRSV_REG0F7)

11.4.3.1.374.1 Offset

Register	Offset
TRSV_REG0F7	7DCh

11.4.3.1.374.2 Diagram



11.4.3.1.374.3 Fields

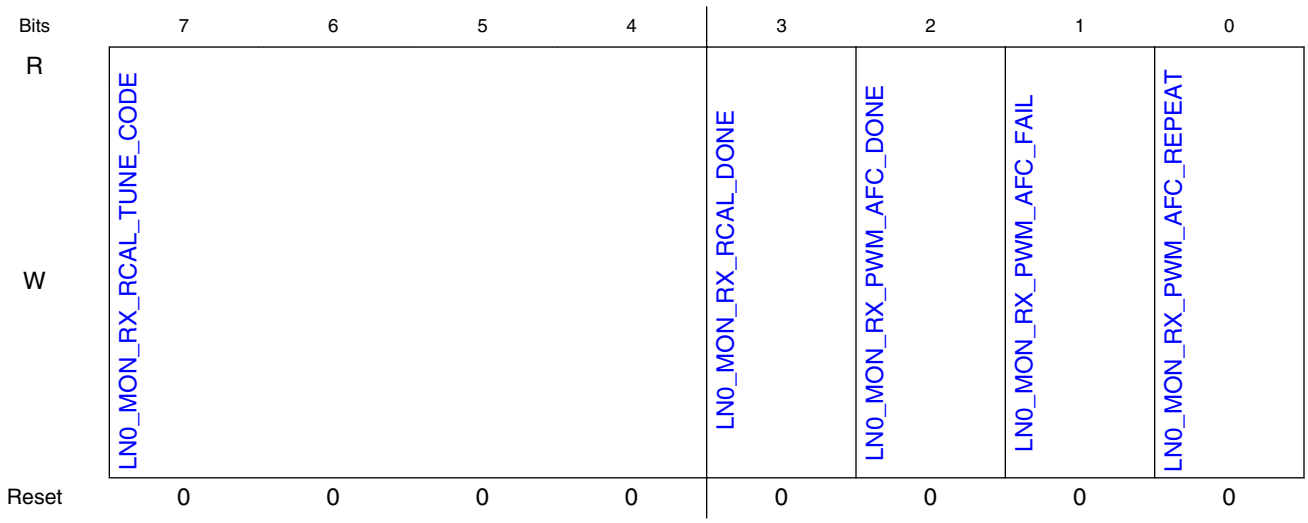
Field	Function
7-5 —	Reserved
4-1 LN0_MON_TX_RCAL_TUNE_CODE	
0 LN0_MON_TX_RCAL_DONE	

11.4.3.1.375 (TRSV_REG0F8)

11.4.3.1.375.1 Offset

Register	Offset
TRSV_REG0F8	7E0h

11.4.3.1.375.2 Diagram



11.4.3.1.375.3 Fields

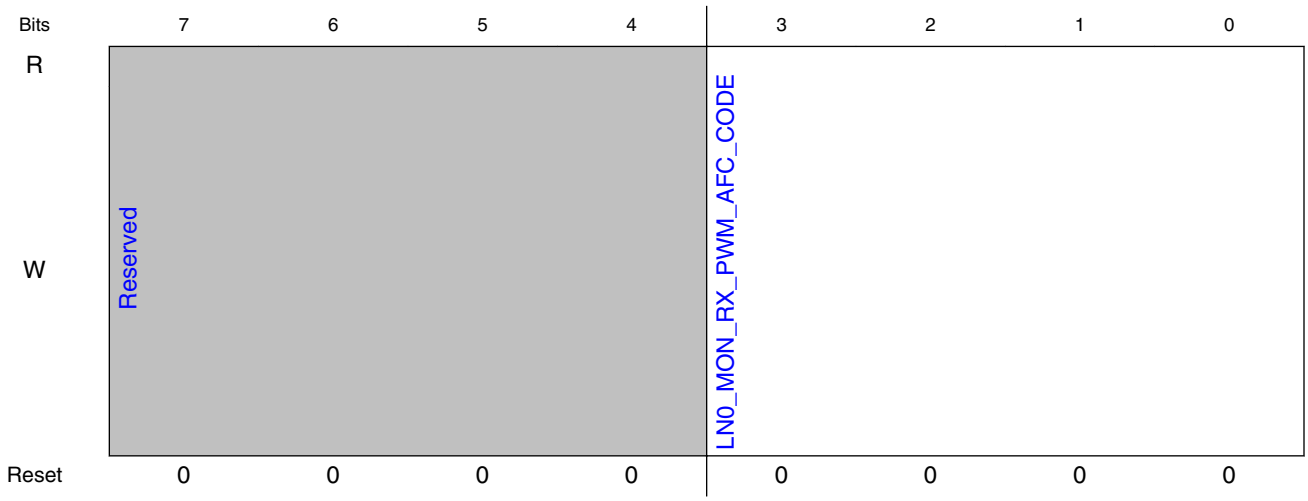
Field	Function
7-4 LN0_MON_RX_RCAL_TUNE_CODE	
3 LN0_MON_RX_RCAL_DONE	
2 LN0_MON_RX_PWM_AFC_DONE	
1 LN0_MON_RX_PWM_AFC_FAIL	
0 LN0_MON_RX_PWM_AFC_REPEAT	

11.4.3.1.376 (TRSV_REG0F9)

11.4.3.1.376.1 Offset

Register	Offset
TRSV_REG0F9	7E4h

11.4.3.1.376.2 Diagram



11.4.3.1.376.3 Fields

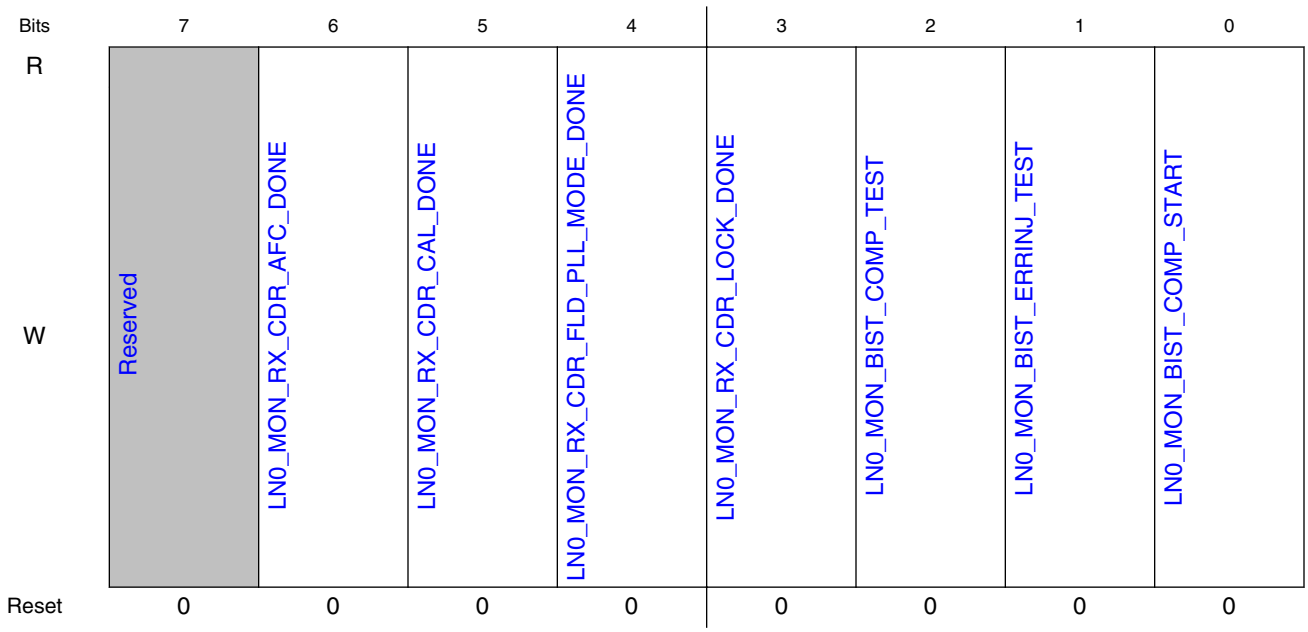
Field	Function
7-4 —	Reserved
3-0 LN0_MON_RX_PWM_AFC_CODE	

11.4.3.1.377 (TRSV_REG0FA)

11.4.3.1.377.1 Offset

Register	Offset
TRSV_REG0FA	7E8h

11.4.3.1.377.2 Diagram



11.4.3.1.377.3 Fields

Field	Function
7 —	Reserved
6 LN0_MON_RX_CDR_AFC_DONE	
5 LN0_MON_RX_CDR_CAL_DONE	
4 LN0_MON_RX_CDR_FLD_PLL_MODE_DONE	
3 LN0_MON_RX_CDR_LOCK_DONE	
2 LN0_MON_BIST_COMP_TEST	
1	

Table continues on the next page...

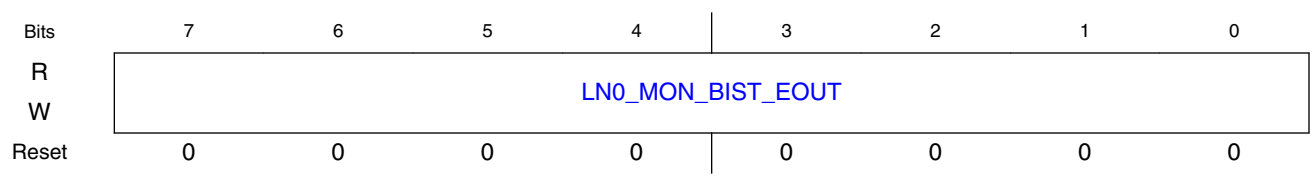
Field	Function
LN0_MON_BIST_ERRINJ_TEST	
0 LN0_MON_BIST_COMP_START	

11.4.3.1.378 (TRSV_REG0FB)

11.4.3.1.378.1 Offset

Register	Offset
TRSV_REG0FB	7ECh

11.4.3.1.378.2 Diagram



11.4.3.1.378.3 Fields

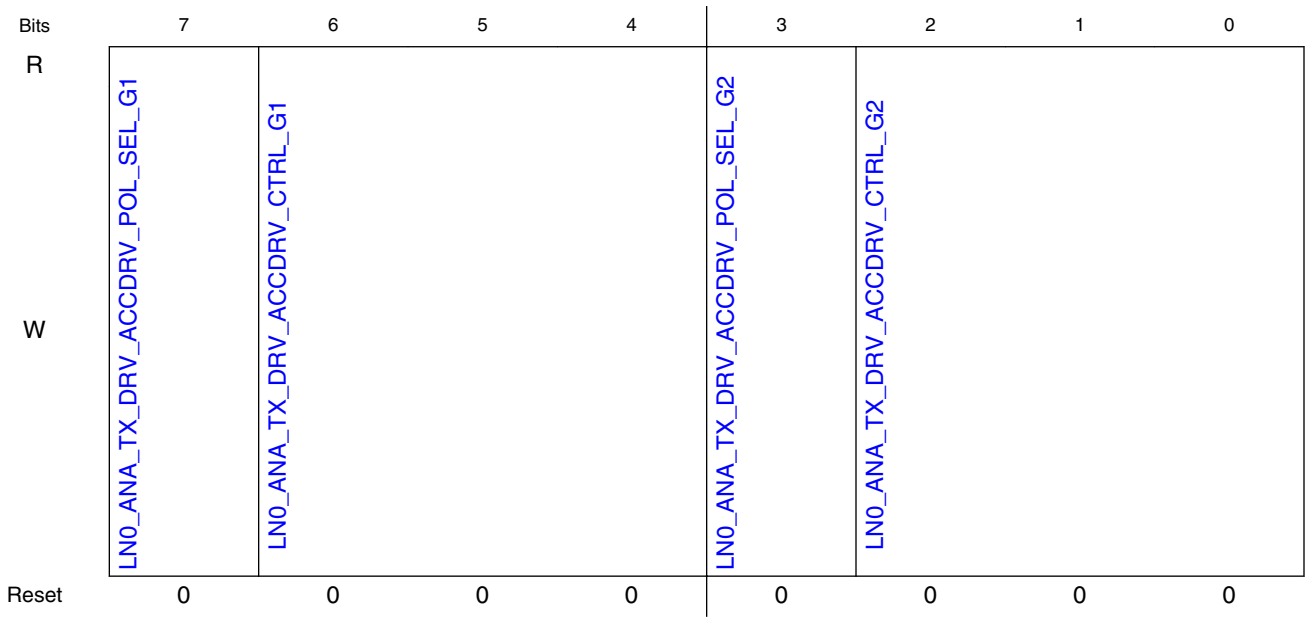
Field	Function
7-0 LN0_MON_BIST_EOUT	

11.4.3.1.379 (TRSV_REG0FC)

11.4.3.1.379.1 Offset

Register	Offset
TRSV_REG0FC	7F0h

11.4.3.1.379.2 Diagram



11.4.3.1.379.3 Fields

Field	Function
7 LN0_ANA_TX_DRV_ACCDRV_POL_SEL_G1	
6-4 LN0_ANA_TX_DRV_ACCDRV_CTRL_G1	
3 LN0_ANA_TX_DRV_ACCDRV_POL_SEL_G2	
2-0 LN0_ANA_TX_DRV_ACCDRV_CTRL_G2	

11.4.3.1.380 (TRSV_REG0FD)

11.4.3.1.380.1 Offset

Register	Offset
TRSV_REG0FD	7F4h

11.4.3.1.380.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	LN0_ANA_TX_DRV_ACCDRV_POL_SEL_G3	LN0_ANA_TX_DRV_ACCDRV_CTRL_G3			LN0_ANA_TX_DRV_ACCDRV_POL_SEL_G4	LN0_ANA_TX_DRV_ACCDRV_CTRL_G4		
W								
Reset	0	0	0	0	0	0	0	0

11.4.3.1.380.3 Fields

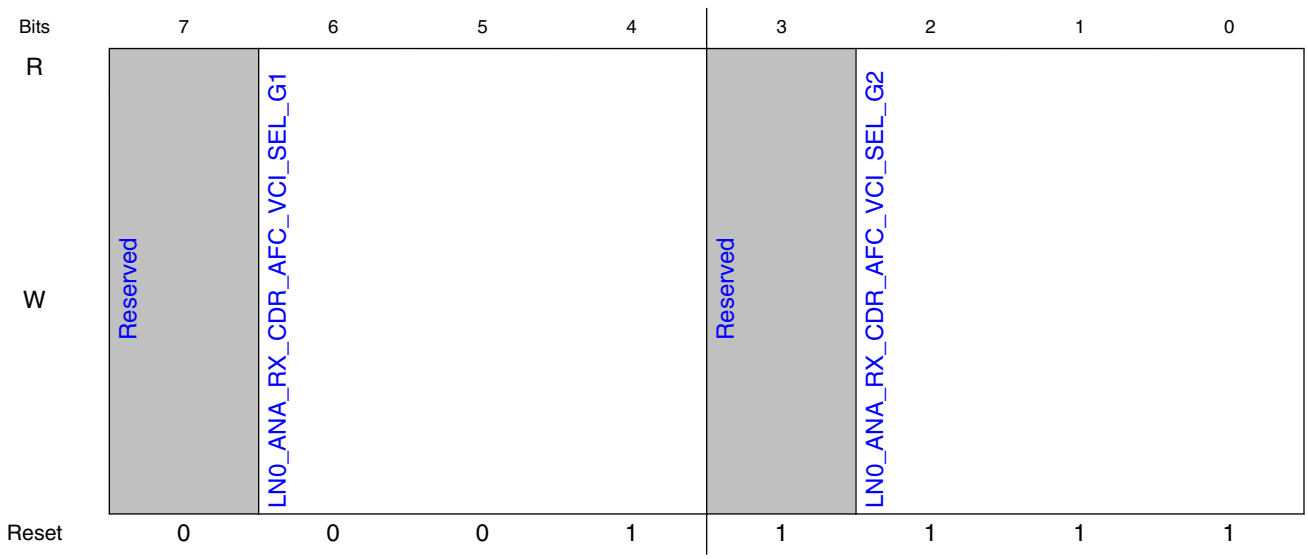
Field	Function
7 LN0_ANA_TX_DRV_ACCDRV_POL_SEL_G3	
6-4 LN0_ANA_TX_DRV_ACCDRV_CTRL_G3	
3 LN0_ANA_TX_DRV_ACCDRV_POL_SEL_G4	
2-0 LN0_ANA_TX_DRV_ACCDRV_CTRL_G4	

11.4.3.1.381 (TRSV_REG0FE)

11.4.3.1.381.1 Offset

Register	Offset
TRSV_REG0FE	7F8h

11.4.3.1.381.2 Diagram



11.4.3.1.381.3 Fields

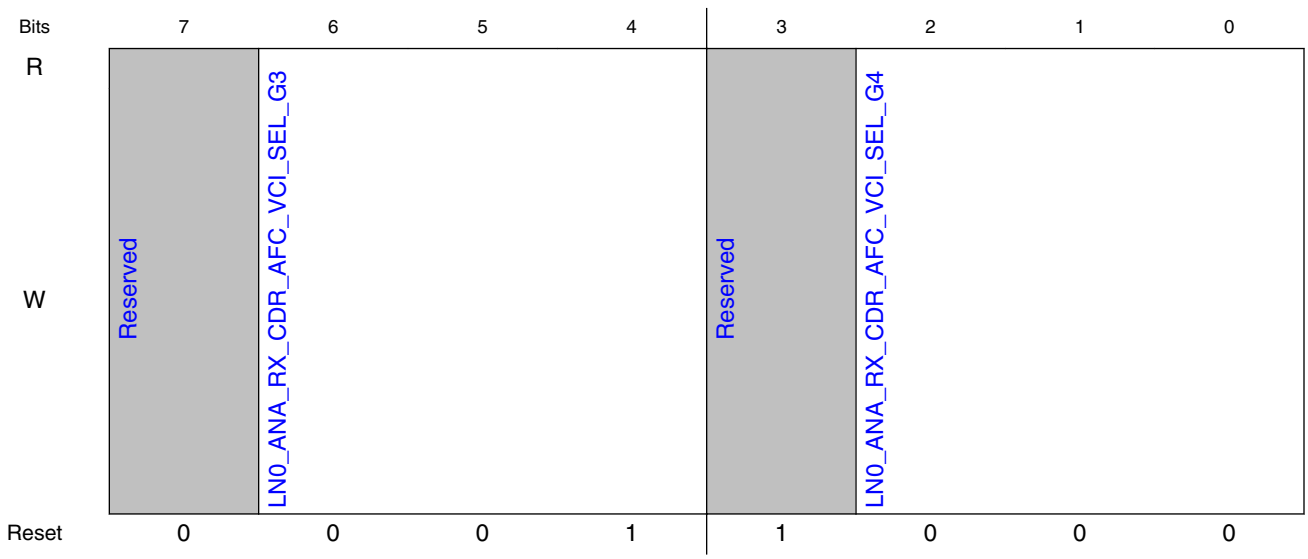
Field	Function
7 —	Reserved
6-4 LN0_ANA_RX_CDR_AFC_VCI_SEL_G1	
3 —	Reserved
2-0 LN0_ANA_RX_CDR_AFC_VCI_SEL_G2	

11.4.3.1.382 (TRSV_REG0FF)

11.4.3.1.382.1 Offset

Register	Offset
TRSV_REG0FF	7FCh

11.4.3.1.382.2 Diagram



11.4.3.1.382.3 Fields

Field	Function
7 —	Reserved
6-4 LN0_ANA_RX_CDR_AFC_VCI_SEL_G3	
3 —	Reserved
2-0 LN0_ANA_RX_CDR_AFC_VCI_SEL_G4	

11.5 Ethernet MAC (ENET)

11.5.1 Introduction

The MAC-NET core, in conjunction with a 10/100/1000-Mbit/s MAC, implements layer 3 network acceleration functions. These functions are designed to accelerate the processing of various common networking protocols, such as IP, TCP, UDP, and ICMP, providing wire speed services to client applications.

11.5.2 Overview

The core implements a triple-speed 10/100/1000-Mbit/s Ethernet MAC compliant with the IEEE802.3-2002 standard. The MAC layer provides compatibility with half- or full-duplex 10/100-Mbit/s and full-duplex gigabit Ethernet LANs.

The MAC operation is fully programmable and can be used in Network Interface Card (NIC), bridging, or switching applications. The core implements the remote network monitoring (RMON) counters according to IETF RFC 2819.

The core also implements a hardware acceleration block to optimize the performance of network controllers providing TCP/IP, UDP, and ICMP protocol services. The acceleration block performs critical functions in hardware, which are typically implemented with large software overhead.

The core implements programmable embedded FIFOs that can provide buffering on the receive path for lossless flow control.

Advanced power management features are available with magic packet detection and programmable power-down modes.

A unified DMA (uDMA), internal to the ENET module, optimizes data transfer between the ENET core and the SoC, and supports an enhanced buffer descriptor programming model to support IEEE 1588 functionality.

The programmable Ethernet MAC with IEEE 1588 integrates a standard IEEE 802.3 Ethernet MAC with a time-stamping module. The IEEE 1588 standard provides accurate clock synchronization for distributed control nodes for industrial automation applications.

11.5.2.1 Features

The MAC-NET core includes the following features.

11.5.2.1.1 Ethernet MAC features

- Implements the full 802.3 specification with preamble/SFD generation, frame padding generation, CRC generation and checking
- Supports zero-length preamble
- Dynamically configurable to support 10/100-Mbit/s and gigabit operation
- Supports 10/100 Mbit/s full-duplex and configurable half-duplex operation
- Supports gigabit full-duplex operation
- Compliant with the AMD magic packet detection with interrupt for node remote power management
- Seamless interface to commercial ethernet PHY devices via one of the following:
  - a 4-bit Media Independent Interface (MII) operating at 2.5/25 MHz.
  - a 4-bit non-standard MII-Lite (MII without the CRS and COL signals) operating at 2.5/25 MHz.
  - a 2-bit Reduced MII (RMII) operating at 50 MHz.
  - a (double data rate) 4-bit Reduced GMII (RGMII) operating at 125 MHz.
- Simple 64-Bit FIFO user-application interface
- CRC-32 checking at full speed with optional forwarding of the frame check sequence (FCS) field to the client
- CRC-32 generation and append on transmit or forwarding of user application provided FCS selectable on a per-frame basis
- In full-duplex mode:
  - Implements automated pause frame (802.3 x31A) generation and termination, providing flow control without user application intervention
  - Pause quanta used to form pause frames — dynamically programmable
  - Pause frame generation additionally controllable by user application offering flexible traffic flow control
  - Optional forwarding of received pause frames to the user application
  - Implements standard flow-control mechanism
- In half-duplex mode: provides full collision support, including jamming, backoff, and automatic retransmission
- Supports VLAN-tagged frames according to IEEE 802.1Q
- Programmable MAC address: Insertion on transmit; discards frames with mismatching destination address on receive (except broadcast and pause frames)
- Programmable promiscuous mode support to omit MAC destination address checking on receive
- Multicast and unicast address filtering on receive based on 64-entry hash table, reducing higher layer processing load

- Programmable frame maximum length providing support for any standard or proprietary frame length
- Statistics indicators for frame traffic and errors (alignment, CRC, length) and pause frames providing for IEEE 802.3 basic and mandatory management information database (MIB) package and remote network monitoring (RFC 2819)
- Simple handshake user application FIFO interface with fully programmable depth and threshold levels
- Provides separate status word for each received frame on the user interface providing information such as frame length, frame type, VLAN tag, and error information
- Multiple internal loopback options
- MDIO master interface for PHY device configuration and management supports two programmable MDIO base addresses, and standard (IEEE 802.3 Clause 22) and extended (Clause 45) MDIO frame formats
- Supports legacy FEC buffer descriptors
- Interrupt coalescing reduces the number of interrupts generated by the MAC, reducing CPU loading
- Traffic-shaping bandwidth distribution supports credit-based and round-robin-based policies. Either policy can be combined with time-based shaping.
- AVB (Audio Video Bridging, IEEE 802.1Qav) features:
  - Credit-based bandwidth distribution policy can be combined with time-based shaping
  - AVB endpoint talker and listener support
  - Support for arbitration between different priority traffic (for example, AVB class A, AVB class B, and non-AVB)

11.5.2.1.2 IP protocol performance optimization features

- Operates on TCP/IP and UDP/IP and ICMP/IP protocol data or IP header only
- Enables wire-speed processing
- Supports IPv4 and IPv6
- Transparent passing of frames of other types and protocols
- Supports VLAN tagged frames according to IEEE 802.1q with transparent forwarding of VLAN tag and control field
- Automatic IP-header and payload (protocol specific) checksum calculation and verification on receive
- Automatic IP-header and payload (protocol specific) checksum generation and automatic insertion on transmit configurable on a per-frame basis
- Supports IP and TCP, UDP, ICMP data for checksum generation and checking

- Supports full header options for IPv4 and TCP protocol headers
- Provides IPv6 support to datagrams with base header only — datagrams with extension headers are passed transparently unmodifed/unchecked
- Provides statistics information for received IP and protocol errors
- Configurable automatic discard of erroneous frames
- Configurable automatic host-to-network (RX) and network-to-host (TX) byte order conversion for IP and TCP/UDP/ICMP headers within the frame
- Configurable padding remove for short IP datagrams on receive
- Configurable Ethernet payload alignment to allow for 32-bit word-aligned header and payload processing
- Programmable store-and-forward operation with clock and rate decoupling FIFOs

11.5.2.1.3 IEEE 1588 features

- Supports all IEEE 1588 frames.
- Allows reference clock to be chosen independently of network speed.
- Software-programmable precise time-stamping of ingress and egress frames
- Timer monitoring capabilities for system calibration and timing accuracy management
- Precise time-stamping of external events with programmable interrupt generation
- Programmable event and interrupt generation for external system control
- Supports hardware- and software-controllable timer synchronization.
- Provides a 4-channel IEEE 1588 timer. Each channel supports input capture and output compare using the 1588 counter.

11.5.2.2 Block diagram

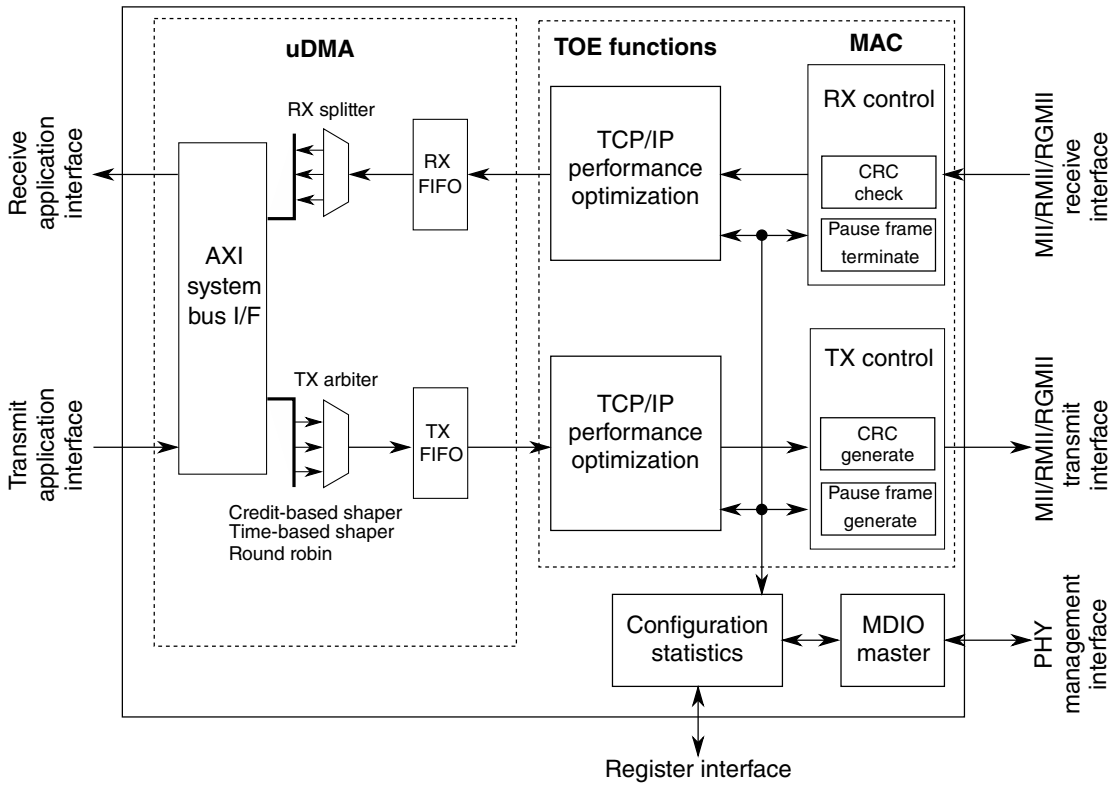


Figure 11-68. Ethernet MAC-NET core block diagram

11.5.3 External Signals

The table found here describes the external signals of ENET.

11.5.4 Clocks

The table found here describes the clock sources for ENET.

11.5.5 Memory map/register definition

ENET registers must be read or written with 32-bit accesses. Non-32 bit accesses will terminate with an error.

Reserved bits should be written with 0 and ignored on read. Unused registers read zero and a write has no effect.

This table shows Ethernet registers organization.

Table 11-101. Register map summary

Offset Address	Section	Description
0x0000 – 0x01FF	Configuration	Core control and status registers
0x0200 – 0x03FF	Statistics counters	MIB and Remote Network Monitoring (RFC 2819) registers
0x0400 – 0x0430	1588 control	1588 adjustable timer (TSM) and 1588 frame control
0x0600 – 0x07FC	Capture/Compare block	Registers for the Capture/Compare block

ENET memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BE_0004	Interrupt Event Register (ENET_EIR)	32	w1c	0000_0000h	11.5.5.1/ 3703
30BE_0008	Interrupt Mask Register (ENET_EIMR)	32	R/W	0000_0000h	11.5.5.2/ 3707
30BE_0010	Receive Descriptor Active Register - Ring 0 (ENET_RDAR)	32	R/W	0000_0000h	11.5.5.3/ 3711
30BE_0014	Transmit Descriptor Active Register - Ring 0 (ENET_TDAR)	32	R/W	0000_0000h	11.5.5.4/ 3711
30BE_0024	Ethernet Control Register (ENET_ECR)	32	R/W	See section	11.5.5.5/ 3713
30BE_0040	MII Management Frame Register (ENET_MMFR)	32	R/W	0000_0000h	11.5.5.6/ 3715
30BE_0044	MII Speed Control Register (ENET_MSCR)	32	R/W	0000_0000h	11.5.5.7/ 3716
30BE_0064	MIB Control Register (ENET_MIBC)	32	R/W	C000_0000h	11.5.5.8/ 3718
30BE_0084	Receive Control Register (ENET_RCR)	32	R/W	05EE_0001h	11.5.5.9/ 3719
30BE_00C4	Transmit Control Register (ENET_TCR)	32	R/W	0000_0000h	11.5.5.10/ 3722
30BE_00E4	Physical Address Lower Register (ENET_PALR)	32	R/W	0000_0000h	11.5.5.11/ 3724
30BE_00E8	Physical Address Upper Register (ENET_PAUR)	32	R/W	0000_8808h	11.5.5.12/ 3724
30BE_00EC	Opcode/Pause Duration Register (ENET_OPD)	32	R/W	0001_0000h	11.5.5.13/ 3725
30BE_00F0	Transmit Interrupt Coalescing Register (ENET_TXIC0)	32	R/W	0000_0000h	11.5.5.14/ 3725
30BE_00F4	Transmit Interrupt Coalescing Register (ENET_TXIC1)	32	R/W	0000_0000h	11.5.5.14/ 3725
30BE_00F8	Transmit Interrupt Coalescing Register (ENET_TXIC2)	32	R/W	0000_0000h	11.5.5.14/ 3725
30BE_0100	Receive Interrupt Coalescing Register (ENET_RXIC0)	32	R/W	0000_0000h	11.5.5.15/ 3726

Table continues on the next page...

ENET memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BE_0104	Receive Interrupt Coalescing Register (ENET_RXIC1)	32	R/W	0000_0000h	11.5.5.15/ 3726
30BE_0108	Receive Interrupt Coalescing Register (ENET_RXIC2)	32	R/W	0000_0000h	11.5.5.15/ 3726
30BE_0118	Descriptor Individual Upper Address Register (ENET_IAUR)	32	R/W	0000_0000h	11.5.5.16/ 3727
30BE_011C	Descriptor Individual Lower Address Register (ENET_IALR)	32	R/W	0000_0000h	11.5.5.17/ 3728
30BE_0120	Descriptor Group Upper Address Register (ENET_GAUR)	32	R/W	0000_0000h	11.5.5.18/ 3728
30BE_0124	Descriptor Group Lower Address Register (ENET_GALR)	32	R/W	0000_0000h	11.5.5.19/ 3729
30BE_0144	Transmit FIFO Watermark Register (ENET_TFWR)	32	R/W	0000_0000h	11.5.5.20/ 3729
30BE_0160	Receive Descriptor Ring 1 Start Register (ENET_RDSR1)	32	R/W	0000_0000h	11.5.5.21/ 3730
30BE_0164	Transmit Buffer Descriptor Ring 1 Start Register (ENET_TDSR1)	32	R/W	0000_0000h	11.5.5.22/ 3731
30BE_0168	Maximum Receive Buffer Size Register - Ring 1 (ENET_MRBR1)	32	R/W	0000_0000h	11.5.5.23/ 3732
30BE_016C	Receive Descriptor Ring 2 Start Register (ENET_RDSR2)	32	R/W	0000_0000h	11.5.5.24/ 3733
30BE_0170	Transmit Buffer Descriptor Ring 2 Start Register (ENET_TDSR2)	32	R/W	0000_0000h	11.5.5.25/ 3733
30BE_0174	Maximum Receive Buffer Size Register - Ring 2 (ENET_MRBR2)	32	R/W	0000_0000h	11.5.5.26/ 3734
30BE_0180	Receive Descriptor Ring 0 Start Register (ENET_RDSR)	32	R/W	0000_0000h	11.5.5.27/ 3735
30BE_0184	Transmit Buffer Descriptor Ring 0 Start Register (ENET_TDSR)	32	R/W	0000_0000h	11.5.5.28/ 3736
30BE_0188	Maximum Receive Buffer Size Register - Ring 0 (ENET_MRBR)	32	R/W	0000_0000h	11.5.5.29/ 3736
30BE_0190	Receive FIFO Section Full Threshold (ENET_RSFL)	32	R/W	0000_0000h	11.5.5.30/ 3737
30BE_0194	Receive FIFO Section Empty Threshold (ENET_RSEM)	32	R/W	0000_0000h	11.5.5.31/ 3738
30BE_0198	Receive FIFO Almost Empty Threshold (ENET_RAEM)	32	R/W	0000_0004h	11.5.5.32/ 3738
30BE_019C	Receive FIFO Almost Full Threshold (ENET_RAFL)	32	R/W	0000_0004h	11.5.5.33/ 3739
30BE_01A0	Transmit FIFO Section Empty Threshold (ENET_TSEM)	32	R/W	0000_0000h	11.5.5.34/ 3739
30BE_01A4	Transmit FIFO Almost Empty Threshold (ENET_TAEM)	32	R/W	0000_0004h	11.5.5.35/ 3740

Table continues on the next page...

ENET memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BE_01A8	Transmit FIFO Almost Full Threshold (ENET_TAFL)	32	R/W	0000_0008h	11.5.5.36/ 3740
30BE_01AC	Transmit Inter-Packet Gap (ENET_TIPG)	32	R/W	0000_000Ch	11.5.5.37/ 3741
30BE_01B0	Frame Truncation Length (ENET_FTRL)	32	R/W	0000_07FFh	11.5.5.38/ 3741
30BE_01C0	Transmit Accelerator Function Configuration (ENET_TACC)	32	R/W	0000_0000h	11.5.5.39/ 3742
30BE_01C4	Receive Accelerator Function Configuration (ENET_RACC)	32	R/W	0000_0000h	11.5.5.40/ 3743
30BE_01C8	Receive Classification Match Register for Class n (ENET_RCMR1)	32	R/W	0000_0000h	11.5.5.41/ 3744
30BE_01CC	Receive Classification Match Register for Class n (ENET_RCMR2)	32	R/W	0000_0000h	11.5.5.41/ 3744
30BE_01D8	DMA Class Based Configuration (ENET_DMA1CFG)	32	R/W	0000_0000h	11.5.5.42/ 3745
30BE_01DC	DMA Class Based Configuration (ENET_DMA2CFG)	32	R/W	0000_0000h	11.5.5.42/ 3745
30BE_01E0	Receive Descriptor Active Register - Ring 1 (ENET_RDAR1)	32	R/W	0000_0000h	11.5.5.43/ 3747
30BE_01E4	Transmit Descriptor Active Register - Ring 1 (ENET_TDAR1)	32	R/W	0000_0000h	11.5.5.44/ 3748
30BE_01E8	Receive Descriptor Active Register - Ring 2 (ENET_RDAR2)	32	R/W	0000_0000h	11.5.5.45/ 3749
30BE_01EC	Transmit Descriptor Active Register - Ring 2 (ENET_TDAR2)	32	R/W	0000_0000h	11.5.5.46/ 3750
30BE_01F0	QOS Scheme (ENET_QOS)	32	R/W	0000_0000h	11.5.5.47/ 3750
30BE_0200	Reserved Statistic Register (ENET_RMON_T_DROP)	32	R	0000_0000h	11.5.5.48/ 3752
30BE_0204	Tx Packet Count Statistic Register (ENET_RMON_T_PACKETS)	32	R	0000_0000h	11.5.5.49/ 3752
30BE_0208	Tx Broadcast Packets Statistic Register (ENET_RMON_T_BC_PKT)	32	R	0000_0000h	11.5.5.50/ 3753
30BE_020C	Tx Multicast Packets Statistic Register (ENET_RMON_T_MC_PKT)	32	R	0000_0000h	11.5.5.51/ 3753
30BE_0210	Tx Packets with CRC/Align Error Statistic Register (ENET_RMON_T_CRC_ALIGN)	32	R	0000_0000h	11.5.5.52/ 3754
30BE_0214	Tx Packets Less Than Bytes and Good CRC Statistic Register (ENET_RMON_T_UNDERSIZE)	32	R	0000_0000h	11.5.5.53/ 3754
30BE_0218	Tx Packets GT MAX_FL bytes and Good CRC Statistic Register (ENET_RMON_T_OVERSIZE)	32	R	0000_0000h	11.5.5.54/ 3754
30BE_021C	Tx Packets Less Than 64 Bytes and Bad CRC Statistic Register (ENET_RMON_T_FRAG)	32	R	0000_0000h	11.5.5.55/ 3755

Table continues on the next page...

ENET memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BE_0220	Tx Packets Greater Than MAX_FL bytes and Bad CRC Statistic Register (ENET_RMON_T_JAB)	32	R	0000_0000h	11.5.5.56/ 3755
30BE_0224	Tx Collision Count Statistic Register (ENET_RMON_T_COL)	32	R	0000_0000h	11.5.5.57/ 3756
30BE_0228	Tx 64-Byte Packets Statistic Register (ENET_RMON_T_P64)	32	R	0000_0000h	11.5.5.58/ 3756
30BE_022C	Tx 65- to 127-byte Packets Statistic Register (ENET_RMON_T_P65TO127)	32	R	0000_0000h	11.5.5.59/ 3756
30BE_0230	Tx 128- to 255-byte Packets Statistic Register (ENET_RMON_T_P128TO255)	32	R	0000_0000h	11.5.5.60/ 3757
30BE_0234	Tx 256- to 511-byte Packets Statistic Register (ENET_RMON_T_P256TO511)	32	R	0000_0000h	11.5.5.61/ 3757
30BE_0238	Tx 512- to 1023-byte Packets Statistic Register (ENET_RMON_T_P512TO1023)	32	R	0000_0000h	11.5.5.62/ 3758
30BE_023C	Tx 1024- to 2047-byte Packets Statistic Register (ENET_RMON_T_P1024TO2047)	32	R	0000_0000h	11.5.5.63/ 3758
30BE_0240	Tx Packets Greater Than 2048 Bytes Statistic Register (ENET_RMON_T_P_GTE2048)	32	R	0000_0000h	11.5.5.64/ 3759
30BE_0244	Tx Octets Statistic Register (ENET_RMON_T_OCTETS)	32	R	0000_0000h	11.5.5.65/ 3759
30BE_0248	Reserved Statistic Register (ENET_IEEE_T_DROP)	32	R	0000_0000h	11.5.5.66/ 3759
30BE_024C	Frames Transmitted OK Statistic Register (ENET_IEEE_T_FRAME_OK)	32	R	0000_0000h	11.5.5.67/ 3760
30BE_0250	Frames Transmitted with Single Collision Statistic Register (ENET_IEEE_T_1COL)	32	R	0000_0000h	11.5.5.68/ 3760
30BE_0254	Frames Transmitted with Multiple Collisions Statistic Register (ENET_IEEE_T_MCOL)	32	R	0000_0000h	11.5.5.69/ 3761
30BE_0258	Frames Transmitted after Deferral Delay Statistic Register (ENET_IEEE_T_DEF)	32	R	0000_0000h	11.5.5.70/ 3761
30BE_025C	Frames Transmitted with Late Collision Statistic Register (ENET_IEEE_T_LCOL)	32	R	0000_0000h	11.5.5.71/ 3761
30BE_0260	Frames Transmitted with Excessive Collisions Statistic Register (ENET_IEEE_T_EXCOL)	32	R	0000_0000h	11.5.5.72/ 3762
30BE_0264	Frames Transmitted with Tx FIFO Underrun Statistic Register (ENET_IEEE_T_MACERR)	32	R	0000_0000h	11.5.5.73/ 3762
30BE_0268	Frames Transmitted with Carrier Sense Error Statistic Register (ENET_IEEE_T_CSERR)	32	R	0000_0000h	11.5.5.74/ 3763
30BE_026C	Reserved Statistic Register (ENET_IEEE_T_SQE)	32	R (reads 0)	0000_0000h	11.5.5.75/ 3763
30BE_0270	Flow Control Pause Frames Transmitted Statistic Register (ENET_IEEE_T_FDXFC)	32	R	0000_0000h	11.5.5.76/ 3763
30BE_0274	Octet Count for Frames Transmitted w/o Error Statistic Register (ENET_IEEE_T_OCTETS_OK)	32	R	0000_0000h	11.5.5.77/ 3764

Table continues on the next page...

ENET memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BE_0284	Rx Packet Count Statistic Register (ENET_RMON_R_PACKETS)	32	R	0000_0000h	11.5.5.78/ 3764
30BE_0288	Rx Broadcast Packets Statistic Register (ENET_RMON_R_BC_PKT)	32	R	0000_0000h	11.5.5.79/ 3765
30BE_028C	Rx Multicast Packets Statistic Register (ENET_RMON_R_MC_PKT)	32	R	0000_0000h	11.5.5.80/ 3765
30BE_0290	Rx Packets with CRC/Align Error Statistic Register (ENET_RMON_R_CRC_ALIGN)	32	R	0000_0000h	11.5.5.81/ 3765
30BE_0294	Rx Packets with Less Than 64 Bytes and Good CRC Statistic Register (ENET_RMON_R_UNDERSIZE)	32	R	0000_0000h	11.5.5.82/ 3766
30BE_0298	Rx Packets Greater Than MAX_FL and Good CRC Statistic Register (ENET_RMON_R_OVERSIZE)	32	R	0000_0000h	11.5.5.83/ 3766
30BE_029C	Rx Packets Less Than 64 Bytes and Bad CRC Statistic Register (ENET_RMON_R_FRAG)	32	R	0000_0000h	11.5.5.84/ 3767
30BE_02A0	Rx Packets Greater Than MAX_FL Bytes and Bad CRC Statistic Register (ENET_RMON_R_JAB)	32	R	0000_0000h	11.5.5.85/ 3767
30BE_02A4	Reserved Statistic Register (ENET_RMON_R_RESVD_0)	32	R (reads 0)	0000_0000h	11.5.5.86/ 3767
30BE_02A8	Rx 64-Byte Packets Statistic Register (ENET_RMON_R_P64)	32	R	0000_0000h	11.5.5.87/ 3768
30BE_02AC	Rx 65- to 127-Byte Packets Statistic Register (ENET_RMON_R_P65TO127)	32	R	0000_0000h	11.5.5.88/ 3768
30BE_02B0	Rx 128- to 255-Byte Packets Statistic Register (ENET_RMON_R_P128TO255)	32	R	0000_0000h	11.5.5.89/ 3769
30BE_02B4	Rx 256- to 511-Byte Packets Statistic Register (ENET_RMON_R_P256TO511)	32	R	0000_0000h	11.5.5.90/ 3769
30BE_02B8	Rx 512- to 1023-Byte Packets Statistic Register (ENET_RMON_R_P512TO1023)	32	R	0000_0000h	11.5.5.91/ 3769
30BE_02BC	Rx 1024- to 2047-Byte Packets Statistic Register (ENET_RMON_R_P1024TO2047)	32	R	0000_0000h	11.5.5.92/ 3770
30BE_02C0	Rx Packets Greater than 2048 Bytes Statistic Register (ENET_RMON_R_P_GTE2048)	32	R	0000_0000h	11.5.5.93/ 3770
30BE_02C4	Rx Octets Statistic Register (ENET_RMON_R_OCTETS)	32	R	0000_0000h	11.5.5.94/ 3771
30BE_02C8	Frames not Counted Correctly Statistic Register (ENET_IEEE_R_DROP)	32	R	0000_0000h	11.5.5.95/ 3771
30BE_02CC	Frames Received OK Statistic Register (ENET_IEEE_R_FRAME_OK)	32	R	0000_0000h	11.5.5.96/ 3771
30BE_02D0	Frames Received with CRC Error Statistic Register (ENET_IEEE_R_CRC)	32	R	0000_0000h	11.5.5.97/ 3772
30BE_02D4	Frames Received with Alignment Error Statistic Register (ENET_IEEE_R_ALIGN)	32	R	0000_0000h	11.5.5.98/ 3772
30BE_02D8	Receive FIFO Overflow Count Statistic Register (ENET_IEEE_R_MACERR)	32	R	0000_0000h	11.5.5.99/ 3773

Table continues on the next page...

ENET memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BE_02DC	Flow Control Pause Frames Received Statistic Register (ENET_IEEE_R_FDXFC)	32	R	0000_0000h	11.5.5.100/ 3773
30BE_02E0	Octet Count for Frames Received without Error Statistic Register (ENET_IEEE_R_OCTETS_OK)	32	R	0000_0000h	11.5.5.101/ 3773
30BE_0400	Adjustable Timer Control Register (ENET_ATCR)	32	R/W	0000_0000h	11.5.5.102/ 3774
30BE_0404	Timer Value Register (ENET_ATVR)	32	R/W	0000_0000h	11.5.5.103/ 3776
30BE_0408	Timer Offset Register (ENET_ATOFF)	32	R/W	0000_0000h	11.5.5.104/ 3776
30BE_040C	Timer Period Register (ENET_ATPER)	32	R/W	3B9A_CA00h	11.5.5.105/ 3777
30BE_0410	Timer Correction Register (ENET_ATCOR)	32	R/W	0000_0000h	11.5.5.106/ 3777
30BE_0414	Time-Stamping Clock Period Register (ENET_ATINC)	32	R/W	0000_0000h	11.5.5.107/ 3778
30BE_0418	Timestamp of Last Transmitted Frame (ENET_ATSTMP)	32	R	0000_0000h	11.5.5.108/ 3778
30BE_0604	Timer Global Status Register (ENET_TGSR)	32	R/W	0000_0000h	11.5.5.109/ 3779
30BE_0608	Timer Control Status Register (ENET_TCSR0)	32	R/W	0000_0000h	11.5.5.110/ 3780
30BE_060C	Timer Compare Capture Register (ENET_TCCR0)	32	R/W	0000_0000h	11.5.5.111/ 3781
30BE_0610	Timer Control Status Register (ENET_TCSR1)	32	R/W	0000_0000h	11.5.5.110/ 3780
30BE_0614	Timer Compare Capture Register (ENET_TCCR1)	32	R/W	0000_0000h	11.5.5.111/ 3781
30BE_0618	Timer Control Status Register (ENET_TCSR2)	32	R/W	0000_0000h	11.5.5.110/ 3780
30BE_061C	Timer Compare Capture Register (ENET_TCCR2)	32	R/W	0000_0000h	11.5.5.111/ 3781
30BE_0620	Timer Control Status Register (ENET_TCSR3)	32	R/W	0000_0000h	11.5.5.110/ 3780
30BE_0624	Timer Compare Capture Register (ENET_TCCR3)	32	R/W	0000_0000h	11.5.5.111/ 3781

11.5.5.1 Interrupt Event Register (ENET_EIR)

When an event occurs that sets a bit in EIR, an interrupt occurs if the corresponding bit in the interrupt mask register (EIMR) is also set. Writing a 1 to an EIR bit clears it; writing 0 has no effect. This register is cleared upon hardware reset.

NOTE

TxBD[INT] and RxBD[INT] must be set to 1 to allow setting the corresponding EIR register flags in enhanced mode, ENET_ECR[EN1588] = 1. Legacy mode does not require these flags to be enabled.

Address: 30BE_0000h base + 4h offset = 30BE_0004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	BABR	BABT	GRA	TXF	TXB	RXF	RXB	MII	EBERR	LC	RL	UN	PLR	WAKEUP	TS_AVAIL
W		w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TS_TIMER	RXFLUSH_2	RXFLUSH_1	RXFLUSH_0					TXF2	TXB2	RXF2	RXB2	TXF1	TXB1	RXF1	RXB1
W	w1c	w1c	w1c	w1c	0			0	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_EIR field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 BABR	Babbling Receive Error Indicates a frame was received with length in excess of RCR[MAX_FL] bytes.
29 BABT	Babbling Transmit Error Indicates the transmitted frame length exceeds RCR[MAX_FL] bytes. Usually this condition is caused when a frame that is too long is placed into the transmit data buffer(s). Truncation does not occur.
28 GRA	Graceful Stop Complete This interrupt is asserted after the transmitter is put into a pause state after completion of the frame currently being transmitted. See Graceful Transmit Stop (GTS) for conditions that lead to graceful stop.

Table continues on the next page...

ENET_EIR field descriptions (continued)

Field	Description
	NOTE: The GRA interrupt is asserted only when the TX transitions into the stopped state. If this bit is cleared by writing 1 and the TX is still stopped, the bit is not set again.
27 TXF	Transmit Frame Interrupt Indicates a frame has been transmitted and the last corresponding buffer descriptor has been updated.
26 TXB	Transmit Buffer Interrupt Indicates a transmit buffer descriptor has been updated.
25 RXF	Receive Frame Interrupt Indicates a frame has been received and the last corresponding buffer descriptor has been updated.
24 RXB	Receive Buffer Interrupt Indicates a receive buffer descriptor is not the last in the frame has been updated.
23 MII	MII Interrupt. Indicates that the MII has completed the data transfer requested.
22 EBERR	Ethernet Bus Error Indicates a system bus error occurred when a uDMA transaction is underway. When this bit is set, ECR[ETHEREN] is cleared, halting frame processing by the MAC. When this occurs, software must ensure proper actions, possibly resetting the system, to resume normal operation.
21 LC	Late Collision Indicates a collision occurred beyond the collision window (slot time) in half-duplex mode. The frame truncates with a bad CRC and the remainder of the frame is discarded.
20 RL	Collision Retry Limit Indicates a collision occurred on each of 16 successive attempts to transmit the frame. The frame is discarded without being transmitted and transmission of the next frame commences. This error can only occur in half-duplex mode.
19 UN	Transmit FIFO Underrun Indicates the transmit FIFO became empty before the complete frame was transmitted. **NOTE:** In situations where the device has various masters generating high traffic, a FIFO underrun can occur on the transmit FIFO. To avoid transmit FIFO underrun, store and forward can be enabled in ENET_TFWR[STRFWD]. See STRFWD. Also, a higher priority can be set for ENET traffic using available means on the central bus fabric connecting the ENET module.
18 PLR	Payload Receive Error Indicates a frame was received with a payload length error. See Frame Length/Type Verification: Payload Length Check for more information.
17 WAKEUP	Node Wakeup Request Indication Read-only status bit to indicate that a magic packet has been detected. Will act only if ECR[MAGICEN] is set.
16 TS_AVAIL	Transmit Timestamp Available Indicates that the timestamp of the last transmitted timing frame is available in the ATSTMP register.
15 TS_TIMER	Timestamp Timer

Table continues on the next page...

ENET_EIR field descriptions (continued)

Field	Description
	The adjustable timer reached the period event. A period event interrupt can be generated if ATCR[PEREN] is set and the timer wraps according to the periodic setting in the ATPER register. Set the timer period value before setting ATCR[PEREN].
14 RXFLUSH_2	RX DMA Ring 2 flush indication. This ring's RX frame has been flushed due to either RDAR2[RDAR] or RxBD[E] being clear and only if QOS[RXFLUSH_2] is enabled.
13 RXFLUSH_1	RX DMA Ring 1 flush indication. This ring's RX frame has been flushed due to either RDAR1[RDAR] or RxBD[E] being clear and only if QOS[RXFLUSH_1] is enabled.
12 RXFLUSH_0	RX DMA Ring 0 flush indication. . This ring's RX frame has been flushed due to either RDAR[RDAR] or RxBD[E] being clear and only if QOS[RXFLUSH_0] is enabled.
11–9 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
8 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
7 TXF2	Transmit frame interrupt, class 2 This bit indicates that a frame has been transmitted and the last corresponding buffer descriptor has been updated (ring/class 2).
6 TXB2	Transmit buffer interrupt, class 2 This field indicates that a transmit buffer descriptor has been updated (ring/class 2).
5 RXF2	Receive frame interrupt, class 2 This field indicates that a frame has been received and the last corresponding buffer descriptor has been updated (ring/class 2).
4 RXB2	Receive buffer interrupt, class 2 This field indicates that a receive buffer descriptor, that not the last in the frame, has been updated (ring/class 2).
3 TXF1	Transmit frame interrupt, class 1 This bit indicates that a frame has been transmitted and the last corresponding buffer descriptor has been updated (ring/class 1).
2 TXB1	Transmit buffer interrupt, class 1 This field indicates that a transmit buffer descriptor has been updated (ring/class 1).
1 RXF1	Receive frame interrupt, class 1 This field indicates that a frame has been received and the last corresponding buffer descriptor has been updated (ring/class 1).
0 RXB1	Receive buffer interrupt, class 1 This field indicates that a receive buffer descriptor, that not the last in the frame, has been updated (ring/class 1).

11.5.5.2 Interrupt Mask Register (ENET_EIMR)

EIMR controls which interrupt events are allowed to generate actual interrupts. A hardware reset clears this register. If the corresponding bits in the EIR and EIMR registers are set, an interrupt is generated. The interrupt signal remains asserted until a 1 is written to the EIR field (write 1 to clear) or a 0 is written to the EIMR field.

Address: 30BE_0000h base + 8h offset = 30BE_0008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R		BABR	BABT	GRA	TXF	TXB	RXF	RXB	MII	EBERR	LC	RL	UN	PLR	WAKEUP	TS_AVAIL
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TS_TIMER	RXFLUSH_2	RXFLUSH_1	RXFLUSH_0					TXF2	TXB2	RXF2	RXB2	TXF1	TXB1	RXF1	RXB1
W					0			0								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_EIMR field descriptions

Field	Description
31 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
30 BABR	BABR Interrupt Mask Corresponds to interrupt source EIR[BABR] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR BABR field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared. 0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
29 BABT	BABT Interrupt Mask Corresponds to interrupt source EIR[BABT] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR BABT field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared. 0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
28 GRA	GRA Interrupt Mask

Table continues on the next page...

ENET_EIMR field descriptions (continued)

Field	Description
	Corresponds to interrupt source EIR[GRA] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR GRA field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared. 0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
27 TXF	TXF Interrupt Mask Corresponds to interrupt source EIR[TXF] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR TXF field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared. 0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
26 TXB	TXB Interrupt Mask Corresponds to interrupt source EIR[TXB] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR TXF field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared. 0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
25 RXF	RXF Interrupt Mask Corresponds to interrupt source EIR[RXF] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR RXF field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
24 RXB	RXB Interrupt Mask Corresponds to interrupt source EIR[RXB] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR RXB field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
23 MII	MII Interrupt Mask Corresponds to interrupt source EIR[MII] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR MII field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
22 EBERR	EBERR Interrupt Mask Corresponds to interrupt source EIR[EBERR] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR EBERR field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
21 LC	LC Interrupt Mask Corresponds to interrupt source EIR[LC] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The

Table continues on the next page...

ENET_EIMR field descriptions (continued)

Field	Description
	corresponding EIR LC field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
20 RL	RL Interrupt Mask Corresponds to interrupt source EIR[RL] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR RL field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
19 UN	UN Interrupt Mask Corresponds to interrupt source EIR[UN] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR UN field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
18 PLR	PLR Interrupt Mask Corresponds to interrupt source EIR[PLR] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR PLR field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
17 WAKEUP	WAKEUP Interrupt Mask Corresponds to interrupt source EIR[WAKEUP] register and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR WAKEUP field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
16 TS_AVAIL	TS_AVAIL Interrupt Mask Corresponds to interrupt source EIR[TS_AVAIL] register and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR TS_AVAIL field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
15 TS_TIMER	TS_TIMER Interrupt Mask Corresponds to interrupt source EIR[TS_TIMER] register and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR TS_TIMER field reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
14 RXFLUSH_2	Corresponds to interrupt source EIR[RXFLUSH_2] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[RXFLUSH_2] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
13 RXFLUSH_1	Corresponds to interrupt source EIR[RXFLUSH_1] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[RXFLUSH_1] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
12 RXFLUSH_0	Corresponds to interrupt source EIR[RXFLUSH_0] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[RXFLUSH_0] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
11–9 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.

Table continues on the next page...

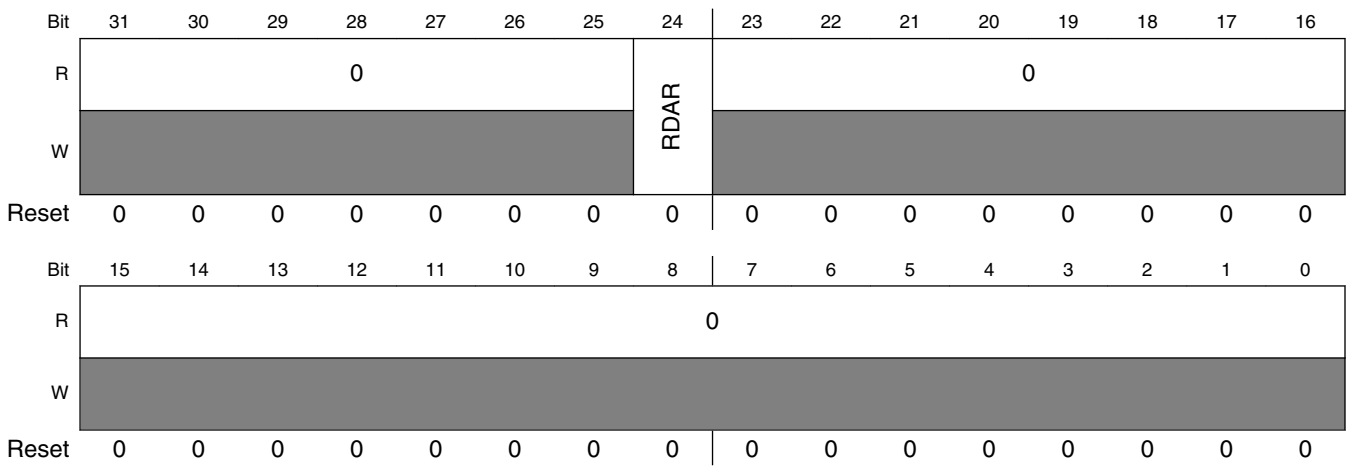
ENET_EIMR field descriptions (continued)

Field	Description
8 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
7 TXF2	Transmit frame interrupt, class 2 Corresponds to interrupt source EIR[TXF2] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[TXF2] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
6 TXB2	Transmit buffer interrupt, class 2 Corresponds to interrupt source EIR[TXB2] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[TXB2] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
5 RXF2	Receive frame interrupt, class 2 Corresponds to interrupt source EIR[RXF2] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[RXF2] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
4 RXB2	Receive buffer interrupt, class 2 Corresponds to interrupt source EIR[RXB2] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[RXB2] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
3 TXF1	Transmit frame interrupt, class 1 Corresponds to interrupt source EIR[TXF1] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[TXF1] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
2 TXB1	Transmit buffer interrupt, class 1 Corresponds to interrupt source EIR[TXB1] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[TXB1] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
1 RXF1	Receive frame interrupt, class 1 Corresponds to interrupt source EIR[RXF1] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[RXF1] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
0 RXB1	Receive buffer interrupt, class 1 Corresponds to interrupt source EIR[RXB1] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[RXB1] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.

11.5.5.3 Receive Descriptor Active Register - Ring 0 (ENET_RDAR)

RDAR is a command register, written by the user, to indicate that the receive descriptor ring 0 has been updated, that is, that the driver produced empty receive buffers with the empty bit set.

Address: 30BE_0000h base + 10h offset = 30BE_0010h



ENET_RDAR field descriptions

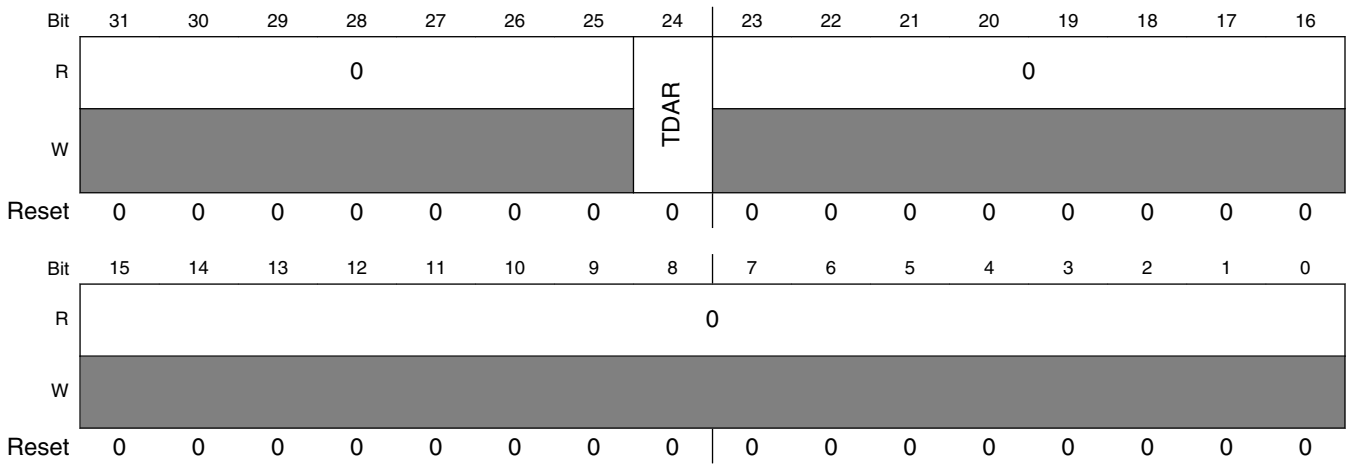
Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 RDAR	Receive Descriptor Active Always set to 1 when this register is written, regardless of the value written. This field is cleared by the MAC device when no additional empty descriptors remain in the receive ring. It is also cleared when ECR[ETHEREN] transitions from set to cleared or when ECR[RESET] is set.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.4 Transmit Descriptor Active Register - Ring 0 (ENET_TDAR)

The TDAR is a command register that the user writes to indicate that the transmit descriptor ring 0 has been updated, that is, that transmit buffers have been produced by the driver with the ready bit set in the buffer descriptor.

The TDAR register is cleared at reset, when ECR[ETHEREN] transitions from set to cleared, or when ECR[RESET] is set.

Address: 30BE_0000h base + 14h offset = 30BE_0014h



ENET_TDAR field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 TDAR	Transmit Descriptor Active Always set to 1 when this register is written, regardless of the value written. This bit is cleared by the MAC device when no additional ready descriptors remain in the transmit ring. Also cleared when ECR[ETHEREN] transitions from set to cleared or when ECR[RESET] is set.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.5 Ethernet Control Register (ENET_ECR)

ECR is a read/write user register, though hardware may also alter fields in this register. It controls many of the high level features of the Ethernet MAC, including legacy FEC support through the EN1588 field.

Address: 30BE_0000h base + 24h offset = 30BE_0024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved														Reserved	
Reset	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				SVLANDBL	VLANUSE2ND	SVLANEN	DBSWP	Reserved	DBGEN	SPEED	EN1588	SLEEP	MAGICEN	ETHEREN	RESET
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_ECR field descriptions

Field	Description
31–18 Reserved	This field is reserved. Always write 01110000000000b to this field.
17–12 Reserved	This field is reserved. Always write 0 to this field.
11 SVLANDBL	S-VLAN double tag If enabled, S-VLAN detection requires a double-tagged frame to define a frame as being a VLAN frame. The following rules apply: • If the first tag is the S-VLAN type, it must be followed by a second tag with the C-VLAN type to declare the frame as VLAN frame. • If the first tag is the S-VLAN type but no 2nd tag follows, it is not considered a VLAN frame but instead treated as an untagged frame. • If the first tag is the C-VLAN type, it is considered a VLAN frame as normal. **NOTE:** VLANUSE2ND can be used to determine from which tag the data should be extracted. This applies only if SVLAN_EN = 1, ignored otherwise.
10 VLANUSE2ND	VLAN use second tag 0 Always extract data from the first VLAN tag if it exists. 1 When a double-tagged frame is detected, the data of the second tag is extracted for further processing. A double-tagged frame is defined as: • The first tag can be a C-VLAN or a S-VLAN (if SVLAN_ENA = 1) • The second tag must be a C-VLAN

Table continues on the next page...

ENET_ECR field descriptions (continued)

Field	Description
9 SVLANEN	S-VLAN enable Enable additional detection of S-VLAN tag according to IEEE802.1Q. 0 Only the EtherType 0x8100 will be considered for VLAN detection. 1 The EtherType 0x88a8 will be considered in addition to 0x8100 (C-VLAN) to identify a VLAN frame in receive. When a VLAN frame is identified, the two bytes following the VLAN type are extracted and used by the classification match comparators, RCMRn.
8 DBSWP	Descriptor Byte Swapping Enable Swaps the byte locations of the buffer descriptors. **NOTE:** This field resets to 0 and must not be changed. 0 The buffer descriptor bytes are not swapped to support big-endian devices. 1 The buffer descriptor bytes are swapped to support little-endian devices.
7 Reserved	This field is reserved. Always write 0 to this field.
6 DBGEN	Debug Enable Enables the MAC to enter hardware freeze mode when the device enters debug mode. 0 MAC continues operation in debug mode. 1 MAC enters hardware freeze mode when the processor is in debug mode.
5 SPEED	Selects between 10/100-Mbit/s and 1000-Mbit/s modes of operation. 0 10/100-Mbit/s mode 1 1000-Mbit/s mode
4 EN1588	EN1588 Enable Enables enhanced functionality of the MAC. 0 Legacy FEC buffer descriptors and functions enabled. 1 Enhanced frame time-stamping functions enabled.
3 SLEEP	Sleep Mode Enable 0 Normal operating mode. 1 Sleep mode.
2 MAGICEN	Magic Packet Detection Enable Enables/disables magic packet detection. **NOTE:** MAGICEN is relevant only if the SLEEP field is set. If MAGICEN is set, changing the SLEEP field enables/disables sleep mode and magic packet detection. **NOTE:** EIMR[WAKEUP] must be written to one if Magic packet wakeup is programed to wake up the chip from low power mode. 0 Magic detection logic disabled. 1 The MAC core detects magic packets and asserts EIR[WAKEUP] when a frame is detected.
1 ETHEREN	Ethernet Enable

Table continues on the next page...

ENET_ECR field descriptions (continued)

Field	Description
	Enables/disables the Ethernet MAC. When the MAC is disabled, the buffer descriptors for an aborted transmit frame are not updated. The uDMA, buffer descriptor, and FIFO control logic are reset, including the buffer descriptor and FIFO pointers. Hardware clears this field under the following conditions: • RESET is set by software • An error condition causes the EBERR field to set. **NOTE:** • ETHEREN must be set at the very last step during ENET configuration/setup/initialization, only *after* all other ENET-related registers have been configured. • If ETHEREN is cleared to 0 by software then next time ETHEREN is set, the EIR interrupts must cleared to 0 due to previous pending interrupts. 0 Reception immediately stops and transmission stops after a bad CRC is appended to any currently transmitted frame. 1 MAC is enabled, and reception and transmission are possible.
0 RESET	Ethernet MAC Reset When this field is set, it clears the ETHEREN field.

11.5.5.6 MII Management Frame Register (ENET_MMFR)

Writing to MMFR triggers a management frame transaction to the PHY device unless MSCR is programmed to zero.

If MSCR is changed from zero to non-zero during a write to MMFR, an MII frame is generated with the data previously written to the MMFR. This allows MMFR and MSCR to be programmed in either order if MSCR is currently zero.

If the MMFR register is written while frame generation is in progress, the frame contents are altered. Software must use the EIR[MII] interrupt indication to avoid writing to the MMFR register while frame generation is in progress.

Address: 30BE_0000h base + 40h offset = 30BE_0040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ST		OP		PA					RA					TA		DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_MMFR field descriptions

Field	Description
31–30 ST	Start Of Frame Delimiter See Table 11-140 (Clause 22) or Table 11-142 (Clause 45) for correct value.
29–28 OP	Operation Code

Table continues on the next page...

ENET_MMFR field descriptions (continued)

Field	Description
	See Table 11-140 (Clause 22) or Table 11-142 (Clause 45) for correct value.
27–23 PA	PHY Address See Table 11-140 (Clause 22) or Table 11-142 (Clause 45) for correct value.
22–18 RA	Register Address See Table 11-140 (Clause 22) or Table 11-142 (Clause 45) for correct value.
17–16 TA	Turn Around This field must be programmed to 10 to generate a valid MII management frame.
DATA	Management Frame Data This is the field for data to be written to or read from the PHY register.

11.5.5.7 MII Speed Control Register (ENET_MSCR)

MSCR provides control of the MII clock (MDC pin) frequency and allows a preamble drop on the MII management frame.

The MII_SPEED field must be programmed with a value to provide an MDC frequency of less than or equal to 2.5 MHz to be compliant with the IEEE 802.3 MII specification. The MII_SPEED must be set to a non-zero value to source a read or write management frame. After the management frame is complete, the MSCR register may optionally be cleared to turn off MDC. The MDC signal generated has a 50% duty cycle except when MII_SPEED changes during operation. This change takes effect following a rising or falling edge of MDC.

For example, if the internal module clock (that is, peripheral bus clock) is 25 MHz, programming MII_SPEED to 0x4 results in an MDC as given in the following equation:

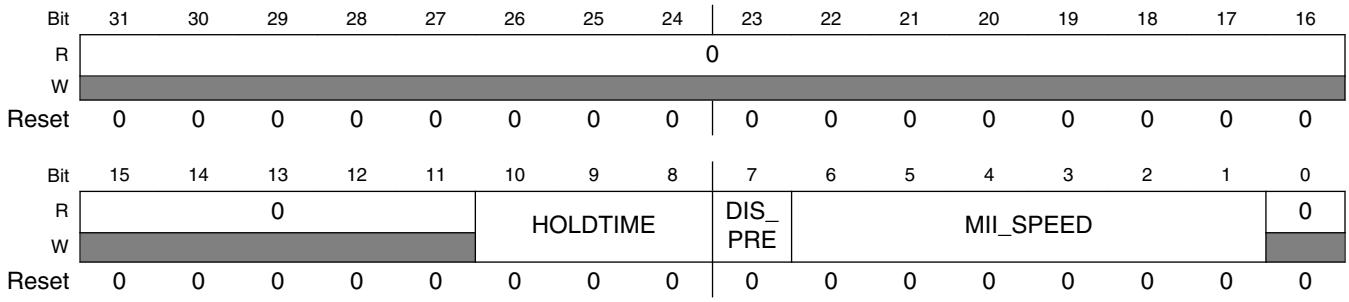
MII clock frequency = 25 MHz / ((4 + 1) x 2) = 2.5 MHz

The following table shows the optimum values for MII_SPEED as a function of IPS bus clock frequency.

Table 11-102. Programming Examples for MSCR

Internal module clock frequency	MSCR [MII_SPEED]	MDC frequency
25 MHz	0x4	2.50 MHz
33 MHz	0x6	2.36 MHz
40 MHz	0x7	2.50 MHz
50 MHz	0x9	2.50 MHz
66 MHz	0xD	2.36 MHz

Address: 30BE_0000h base + 44h offset = 30BE_0044h



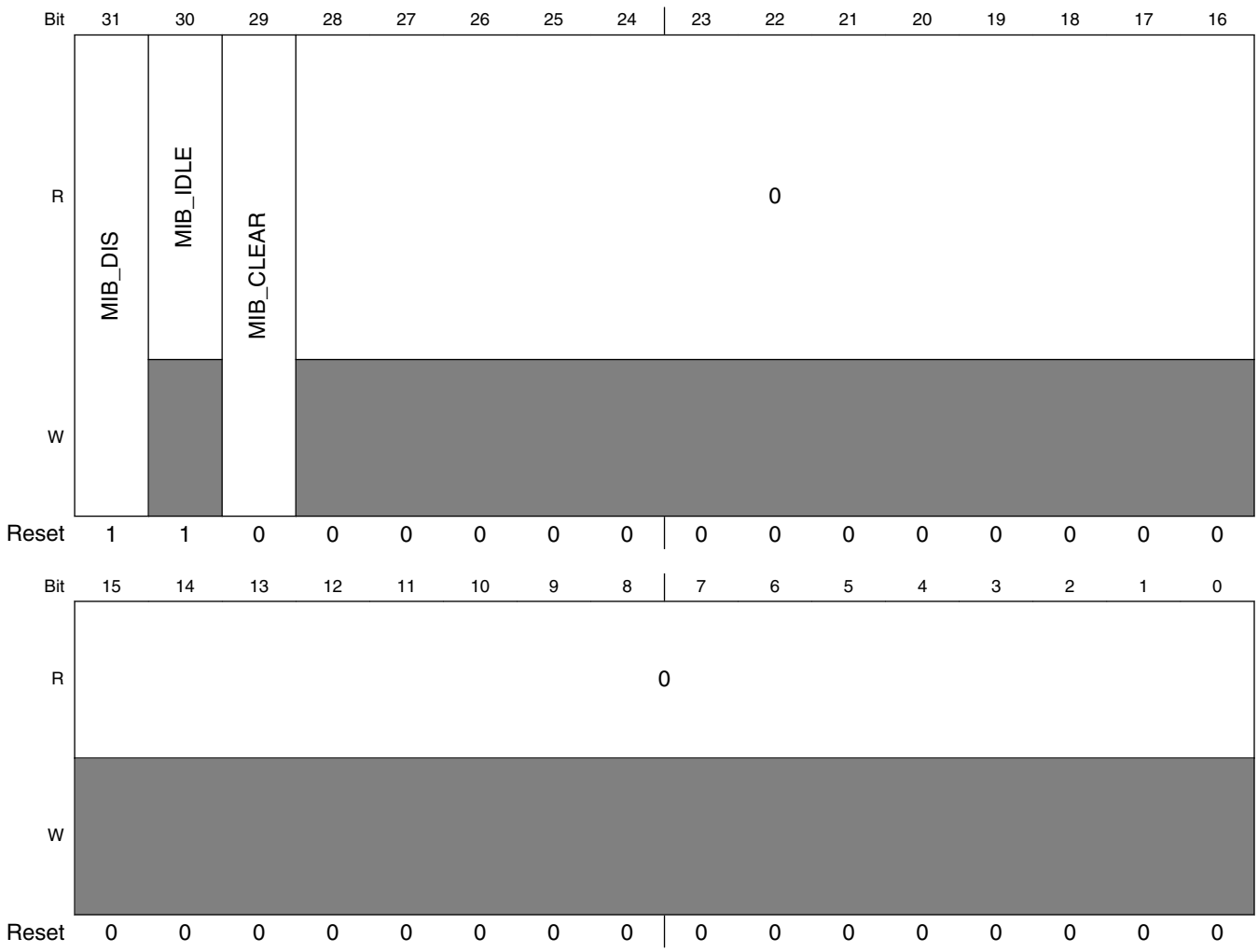
ENET_MSCR field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 HOLDTIME	Hold time On MDIO Output IEEE802.3 clause 22 defines a minimum of 10 ns for the hold time on the MDIO output. Depending on the host bus frequency, the setting may need to be increased. 000 1 internal module clock cycle 001 2 internal module clock cycles 010 3 internal module clock cycles 111 8 internal module clock cycles
7 DIS_PRE	Disable Preamble Enables/disables prepending a preamble to the MII management frame. The MII standard allows the preamble to be dropped if the attached PHY devices do not require it. 0 Preamble enabled. 1 Preamble (32 ones) is not prepended to the MII management frame.
6–1 MII_SPEED	MII Speed Controls the frequency of the MII management interface clock (MDC) relative to the internal module clock. A value of 0 in this field turns off MDC and leaves it in low voltage state. Any non-zero value results in the MDC frequency of: 1/((MII_SPEED + 1) x 2) of the internal module clock frequency
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.8 MIB Control Register (ENET_MIBC)

MIBC is a read/write register controlling and observing the state of the MIB block. Access this register to disable the MIB block operation or clear the MIB counters. The MIB_DIS field resets to 1.

Address: 30BE_0000h base + 64h offset = 30BE_0064h



ENET_MIBC field descriptions

Field	Description
31 MIB_DIS	Disable MIB Logic If this control field is set, 0 MIB logic is enabled. 1 MIB logic is disabled. The MIB logic halts and does not update any MIB counters.

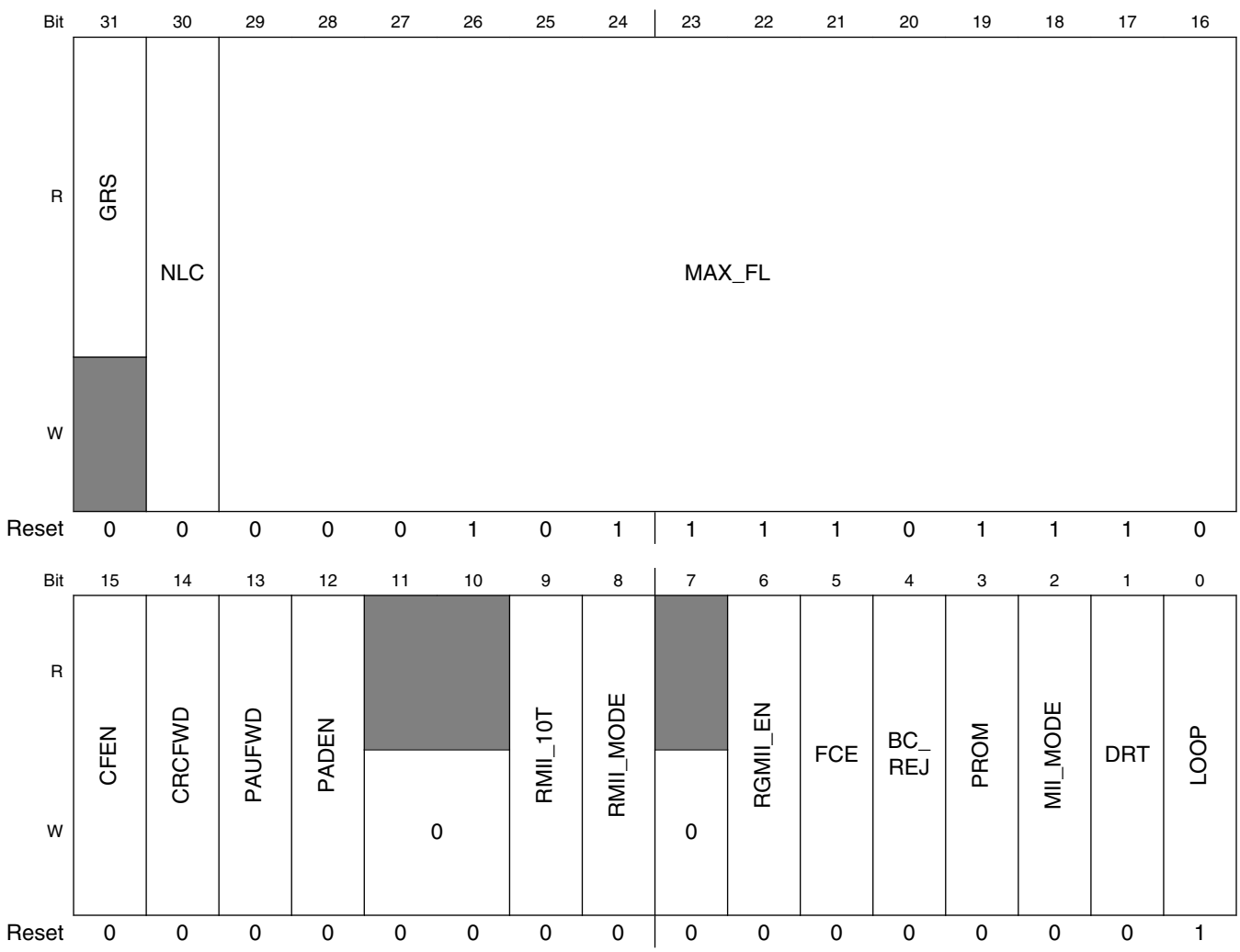
Table continues on the next page...

ENET_MIBC field descriptions (continued)

Field	Description
30 MIB_IDLE	MIB Idle 0 The MIB block is updating MIB counters. 1 The MIB block is not currently updating any MIB counters.
29 MIB_CLEAR	MIB Clear **NOTE:** This field is not self-clearing. To clear the MIB counters set and then clear this field. 0 See note above. 1 All statistics counters are reset to 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.9 Receive Control Register (ENET_RCR)

Address: 30BE_0000h base + 84h offset = 30BE_0084h



ENET_RCR field descriptions

Field	Description
31 GRS	Graceful Receive Stopped Read-only status indicating that the MAC receive datapath is stopped.
30 NLC	Payload Length Check Disable Enables/disables a payload length check. 0 The payload length check is disabled. 1 The core checks the frame's payload length with the frame length/type field. Errors are indicated in the EIR[PLR] field.
29–16 MAX_FL	Maximum Frame Length Resets to decimal 1518. Length is measured starting at DA and includes the CRC at the end of the frame. Transmit frames longer than MAX_FL cause the BABT interrupt to occur. Receive frames longer than MAX_FL cause the BABR interrupt to occur and set the LG field in the end of frame receive buffer descriptor. The recommended default value to be programmed is 1518 or 1522 if VLAN tags are supported.
15 CFEN	MAC Control Frame Enable Enables/disables the MAC control frame. 0 MAC control frames with any opcode other than 0x0001 (pause frame) are accepted and forwarded to the client interface. 1 MAC control frames with any opcode other than 0x0001 (pause frame) are silently discarded.
14 CRCFWD	Terminate/Forward Received CRC Specifies whether the CRC field of received frames is transmitted or stripped. **NOTE:** If padding function is enabled (PADEN = 1), CRCFWD is ignored and the CRC field is checked and always terminated and removed. 0 The CRC field of received frames is transmitted to the user application. 1 The CRC field is stripped from the frame.
13 PAUFWD	Terminate/Forward Pause Frames Specifies whether pause frames are terminated or forwarded. 0 Pause frames are terminated and discarded in the MAC. 1 Pause frames are forwarded to the user application.
12 PADEN	Enable Frame Padding Remove On Receive Specifies whether the MAC removes padding from received frames. 0 No padding is removed on receive by the MAC. 1 Padding is removed from received frames.
11–10 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
9 RMII_10T	Enables 10-Mbit/s mode of the RMII or RGMII . 0 100-Mbit/s operation. 1 10-Mbit/s operation.

Table continues on the next page...

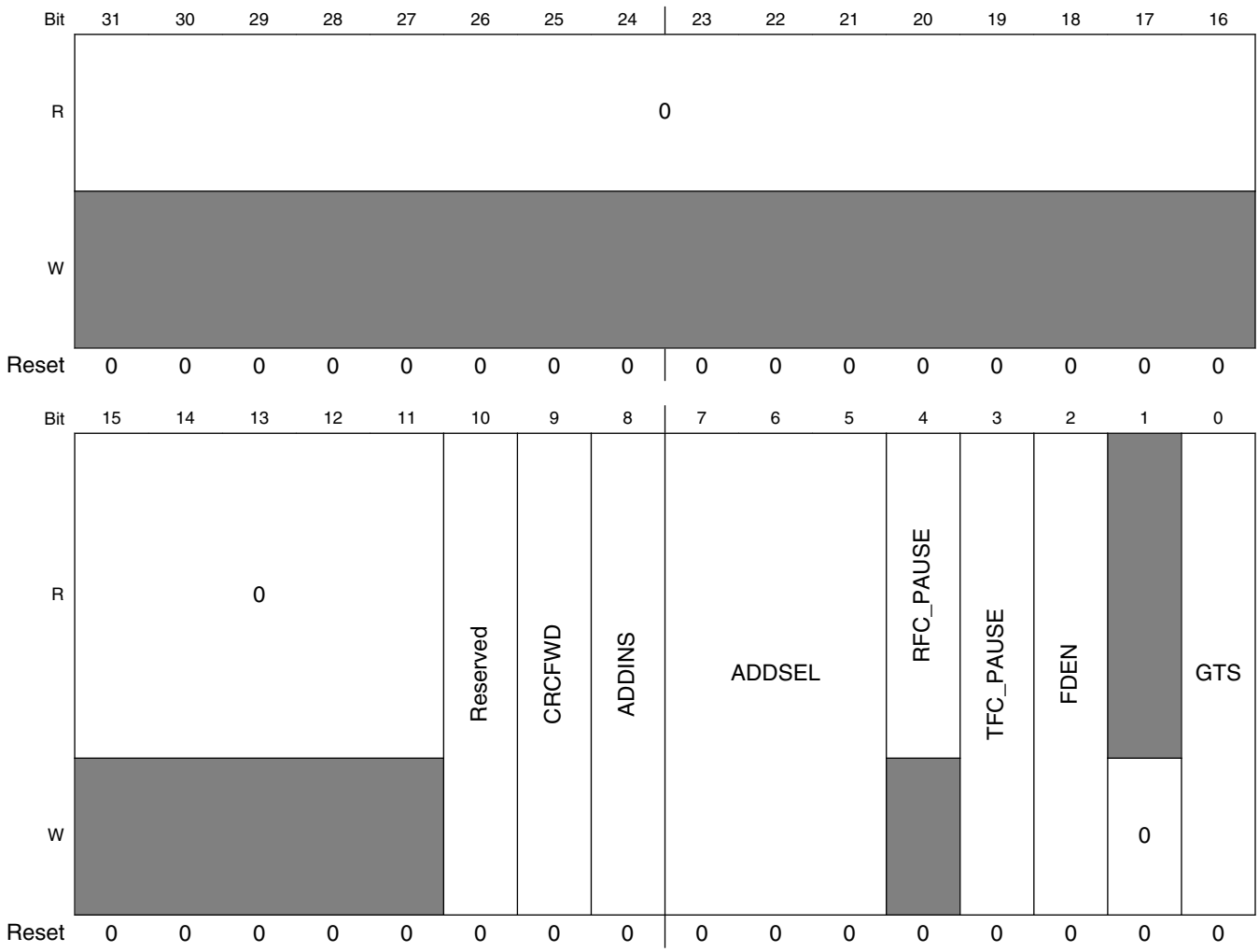
ENET_RCR field descriptions (continued)

Field	Description
8 RMII_MODE	RMII Mode Enable Specifies whether the MAC is configured for MII mode or RMII operation , when ECR[SPEED] is cleared . **NOTE:** Do not set both RCR[RGMII_EN] and RCR[RMII_MODE]. 0 MAC configured for MII mode. 1 MAC configured for RMII operation.
7 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
6 RGMII_EN	RGMII Mode Enable **NOTE:** Do not set both RCR[RGMII_EN] and RCR[RMII_MODE]. 0 MAC configured for non-RGMII operation 1 MAC configured for RGMII operation. If ECR[SPEED] is set, the MAC is in RGMII 1000-Mbit/s mode. If ECR[SPEED] is cleared, the MAC is in RGMII 10/100-Mbit/s mode.
5 FCE	Flow Control Enable If set, the receiver detects PAUSE frames. Upon PAUSE frame detection, the transmitter stops transmitting data frames for a given duration.
4 BC_REJ	Broadcast Frame Reject If set, frames with destination address (DA) equal to 0xFFFF_FFFF_FFFF are rejected unless the PROM field is set. If BC_REJ and PROM are set, frames with broadcast DA are accepted and the MISS (M) is set in the receive buffer descriptor.
3 PROM	Promiscuous Mode All frames are accepted regardless of address matching. 0 Disabled. 1 Enabled.
2 MII_MODE	Media Independent Interface Mode This field must always be set. 0 Reserved. 1 MII or RMII mode, as indicated by the RMII_MODE field.
1 DRT	Disable Receive On Transmit 0 Receive path operates independently of transmit (i.e., full-duplex mode). Can also be used to monitor transmit activity in half-duplex mode. 1 Disable reception of frames while transmitting. (Normally used for half-duplex mode.)
0 LOOP	Internal Loopback This is an MII internal loopback, therefore MII_MODE must be written to 1 and RMII_MODE must be written to 0. 0 Loopback disabled. 1 Transmitted frames are looped back internal to the device and transmit MII output signals are not asserted. DRT must be cleared.

11.5.5.10 Transmit Control Register (ENET_TCR)

TCR is read/write and configures the transmit block. This register is cleared at system reset. FDEN can only be modified when ECR[ETHEREN] is cleared.

Address: 30BE_0000h base + C4h offset = 30BE_00C4h



ENET_TCR field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 Reserved	This field is reserved. This field is read/write and must be set to 0.
9 CRCFWD	Forward Frame From Application With CRC

Table continues on the next page...

ENET_TCR field descriptions (continued)

Field	Description
	0 TxBD[TC] controls whether the frame has a CRC from the application. 1 The transmitter does not append any CRC to transmitted frames, as it is expecting a frame with CRC from the application.
8 ADDINS	Set MAC Address On Transmit 0 The source MAC address is not modified by the MAC. 1 The MAC overwrites the source MAC address with the programmed MAC address according to ADDSEL.
7–5 ADDSEL	Source MAC Address Select On Transmit If ADDINS is set, indicates the MAC address that overwrites the source MAC address. 000 Node MAC address programmed on PADDR1/2 registers. 100 Reserved. 101 Reserved. 110 Reserved.
4 RFC_PAUSE	Receive Frame Control Pause This status field is set when a full-duplex flow control pause frame is received and the transmitter pauses for the duration defined in this pause frame. This field automatically clears when the pause duration is complete.
3 TFC_PAUSE	Transmit Frame Control Pause Pauses frame transmission. When this field is set, EIR[GRA] is set. With transmission of data frames stopped, the MAC transmits a MAC control PAUSE frame. Next, the MAC clears TFC_PAUSE and resumes transmitting data frames. If the transmitter pauses due to user assertion of GTS or reception of a PAUSE frame, the MAC may continue transmitting a MAC control PAUSE frame. 0 No PAUSE frame transmitted. 1 The MAC stops transmission of data frames after the current transmission is complete.
2 FDEN	Full-Duplex Enable If this field is set, frames transmit independent of carrier sense and collision inputs. Only modify this bit when ECR[ETHEREN] is cleared.
1 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
0 GTS	Graceful Transmit Stop When this field is set, MAC stops transmission after any frame currently transmitted is complete and EIR[GRA] is set. If frame transmission is not currently underway, the GRA interrupt is asserted immediately. After transmission finishes, clear GTS to restart. The next frame in the transmit FIFO is then transmitted. If an early collision occurs during transmission when GTS is set, transmission stops after the collision. The frame is transmitted again after GTS is cleared. There may be old frames in the transmit FIFO that transmit when GTS is reasserted. To avoid this, clear ECR[ETHEREN] following the GRA interrupt.

11.5.5.11 Physical Address Lower Register (ENET_PALR)

PALR contains the lower 32 bits (bytes 0, 1, 2, 3) of the 48-bit address used in the address recognition process to compare with the destination address (DA) field of receive frames with an individual DA. In addition, this register is used in bytes 0 through 3 of the six-byte source address field when transmitting PAUSE frames.

Address: 30BE_0000h base + E4h offset = 30BE_00E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PADDR1																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_PALR field descriptions

Field	Description
PADDR1	Pause Address Bytes 0 (bits 31:24), 1 (bits 23:16), 2 (bits 15:8), and 3 (bits 7:0) of the 6-byte individual address are used for exact match and the source address field in PAUSE frames.

11.5.5.12 Physical Address Upper Register (ENET_PAUR)

PAUR contains the upper 16 bits (bytes 4 and 5) of the 48-bit address used in the address recognition process to compare with the destination address (DA) field of receive frames with an individual DA. In addition, this register is used in bytes 4 and 5 of the six-byte source address field when transmitting PAUSE frames. Bits 15:0 of PAUR contain a constant type field (0x8808) for transmission of PAUSE frames.

Address: 30BE_0000h base + E8h offset = 30BE_00E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PADDR2																TYPE															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0

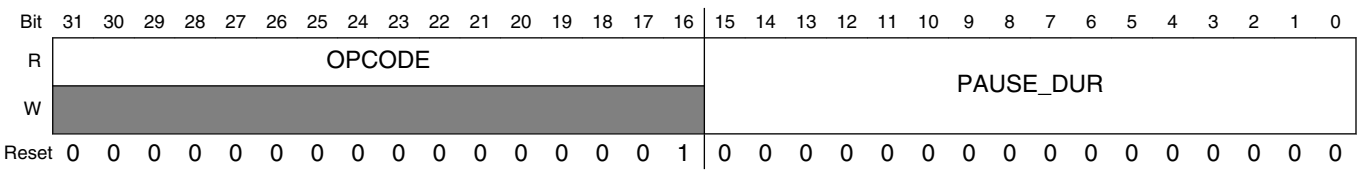
ENET_PAUR field descriptions

Field	Description
31–16 PADDR2	Bytes 4 (bits 31:24) and 5 (bits 23:16) of the 6-byte individual address used for exact match, and the source address field in PAUSE frames.
TYPE	Type Field In PAUSE Frames These fields have a constant value of 0x8808.

11.5.5.13 Opcode/Pause Duration Register (ENET_OPD)

OPD is read/write accessible. This register contains the 16-bit opcode and 16-bit pause duration fields used in transmission of a PAUSE frame. The opcode field is a constant value, 0x0001. When another node detects a PAUSE frame, that node pauses transmission for the duration specified in the pause duration field. The lower 16 bits of this register are not reset and you must initialize it.

Address: 30BE_0000h base + ECh offset = 30BE_00ECh



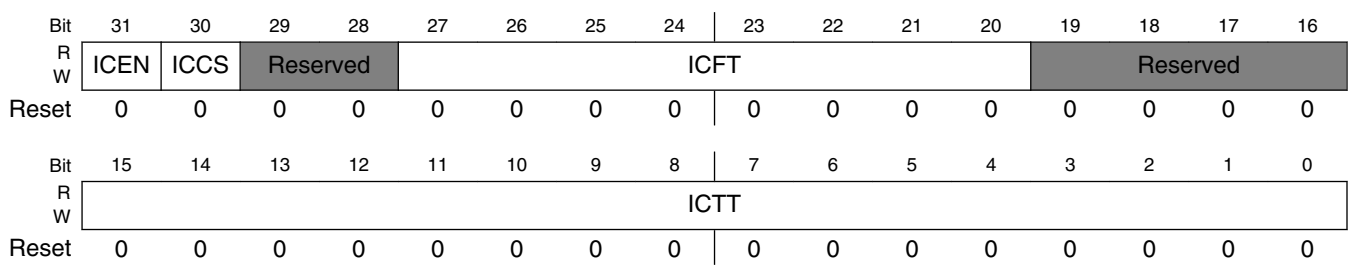
ENET_OPD field descriptions

Field	Description
31–16 OPCODE	Opcode Field In PAUSE Frames These fields have a constant value of 0x0001.
PAUSE_DUR	Pause Duration Pause duration field used in PAUSE frames.

11.5.5.14 Transmit Interrupt Coalescing Register (ENET_TXIC*n*)

See Interrupt coalescence for more information.

Address: 30BE_0000h base + F0h offset + (4d × i), where i=0d to 2d



ENET_TXIC*n* field descriptions

Field	Description
31 ICEN	Interrupt Coalescing Enable

Table continues on the next page...

ENET_TXIC*n* field descriptions (continued)

Field	Description
	0 Disable Interrupt coalescing. 1 Enable Interrupt coalescing.
30 ICCS	Interrupt Coalescing Timer Clock Source Select 0 Use MII/GMII TX clocks. 1 Use ENET system clock.
29–28 Reserved	This field must be set to 0. This field is reserved.
27–20 ICFT	Interrupt coalescing frame count threshold This value determines the number of frames needed to be transmitted for raising an interrupt. Frame counter restarts after reaching this threshold value or after the expiring of the coalescing timer. Must be greater than zero to avoid unpredictable behavior.
19–16 Reserved	This field must be set to 0. This field is reserved.
ICTT	Interrupt coalescing timer threshold Interrupt coalescing timer threshold in units of 64 clock periods. This value determines the maximum amount of time after transmitting a frame before raising an interrupt. The threshold timer is disabled after expiring or number of frame transmission defined by ICFT and starts again upon transmission of the next first frame. Must be greater than zero to avoid unpredictable behavior.

11.5.5.15 Receive Interrupt Coalescing Register (ENET_RXIC*n*)

See Interrupt coalescence for more information.

Address: 30BE_0000h base + 100h offset + (4d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ICEN	ICCS	Reserved						ICFT				Reserved			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ICTT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RXIC*n* field descriptions

Field	Description
31 ICEN	Interrupt Coalescing Enable 0 Disable Interrupt coalescing. 1 Enable Interrupt coalescing.

Table continues on the next page...

ENET_RXIC*n* field descriptions (continued)

Field	Description
30 ICCS	Interrupt Coalescing Timer Clock Source Select 0 Use MII/GMII TX clocks. 1 Use ENET system clock.
29–28 Reserved	This field must be set to 0. This field is reserved.
27–20 ICFT	Interrupt coalescing frame count threshold This value determines the number of frames needed to be received for raising an interrupt. Frame counter restarts after reaching this threshold value or after the expiring of the coalescing timer. Must be greater than zero to avoid unpredictable behavior.
19–16 Reserved	This field must be set to 0. This field is reserved.
ICTT	Interrupt coalescing timer threshold Interrupt coalescing timer threshold in units of 64 clock periods. This value determines the maximum amount of time after receiving a frame before raising an interrupt. The threshold timer is disabled after expiring or number of frame reception defined by ICFT and starts again upon reception of the next first frame. Must be greater than zero to avoid unpredictable behavior.

11.5.5.16 Descriptor Individual Upper Address Register (ENET_IAUR)

IAUR contains the upper 32 bits of the 64-bit individual address hash table. The address recognition process uses this table to check for a possible match with the destination address (DA) field of receive frames with an individual DA. This register is not reset and you must initialize it.

Address: 30BE_0000h base + 118h offset = 30BE_0118h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IADDR1																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

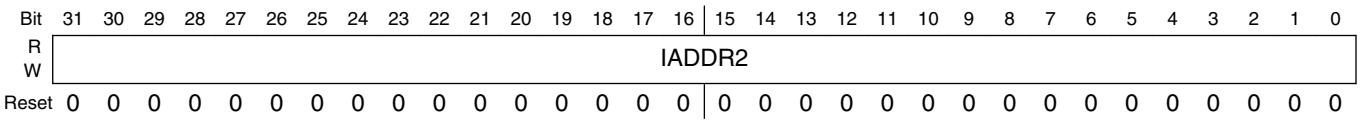
ENET_IAUR field descriptions

Field	Description
IADDR1	Contains the upper 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a unicast address. Bit 31 of IADDR1 contains hash index bit 63. Bit 0 of IADDR1 contains hash index bit 32.

11.5.5.17 Descriptor Individual Lower Address Register (ENET_IALR)

IALR contains the lower 32 bits of the 64-bit individual address hash table. The address recognition process uses this table to check for a possible match with the DA field of receive frames with an individual DA. This register is not reset and you must initialize it.

Address: 30BE_0000h base + 11Ch offset = 30BE_011Ch



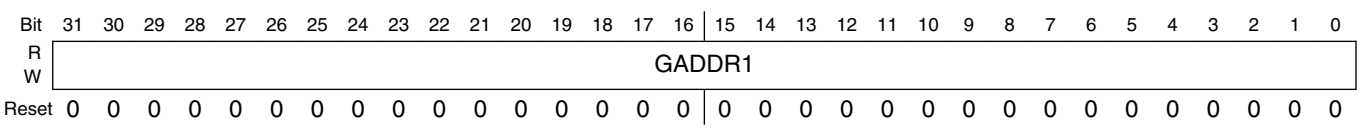
ENET_IALR field descriptions

Field	Description
IADDR2	Contains the lower 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a unicast address. Bit 31 of IADDR2 contains hash index bit 31. Bit 0 of IADDR2 contains hash index bit 0.

11.5.5.18 Descriptor Group Upper Address Register (ENET_GAUR)

GAUR contains the upper 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. You must initialize this register.

Address: 30BE_0000h base + 120h offset = 30BE_0120h



ENET_GAUR field descriptions

Field	Description
GADDR1	Contains the upper 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. Bit 31 of GADDR1 contains hash index bit 63. Bit 0 of GADDR1 contains hash index bit 32.

11.5.5.19 Descriptor Group Lower Address Register (ENET_GALR)

GALR contains the lower 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. You must initialize this register.

Address: 30BE_0000h base + 124h offset = 30BE_0124h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GADDR2																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_GALR field descriptions

Field	Description
GADDR2	Contains the lower 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. Bit 31 of GADDR2 contains hash index bit 31. Bit 0 of GADDR2 contains hash index bit 0.

11.5.5.20 Transmit FIFO Watermark Register (ENET_TFWR)

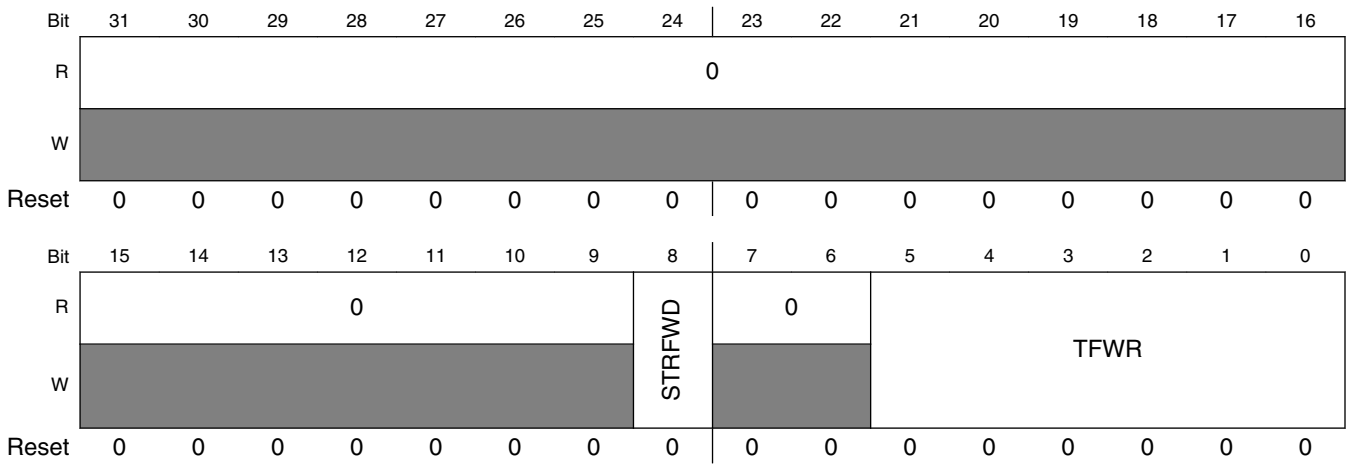
If TFWR[STRFWD] is cleared, TFWR[TFWR] controls the amount of data required in the transmit FIFO before transmission of a frame can begin. This allows you to minimize transmit latency (TFWR = 00 or 01) or allow for larger bus access latency (TFWR = 11) due to contention for the system bus. Setting the watermark to a high value minimizes the risk of transmit FIFO underrun due to contention for the system bus. The byte counts associated with the TFWR field may need to be modified to match a given system requirement, for example, worst-case bus access latency by the transmit data uDMA channel.

When the FIFO level reaches the value the TFWR field and when the STR_FWD is set to ‘0’, the MAC transmit control logic starts frame transmission even before the end-of-frame is available in the FIFO (cut-through operation).

If a complete frame has a size smaller than the threshold programmed with TFWR, the MAC also transmits the Frame to the line.

To enable store and forward on the Transmit path, set STR_FWD to ‘1’. In this case, the MAC starts to transmit data only when a complete frame is stored in the Transmit FIFO.

Address: 30BE_0000h base + 144h offset = 30BE_0144h



ENET_TFWR field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 STRFWD	Store And Forward Enable 0 Reset. The transmission start threshold is programmed in TFWR[TFWR]. 1 Enabled.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TFWR	Transmit FIFO Write If TFWR[STRFWD] is cleared, this field indicates the number of bytes, in steps of 64 bytes, written to the transmit FIFO before transmission of a frame begins. **NOTE:** If a frame with less than the threshold is written, it is still sent independently of this threshold setting. The threshold is relevant only if the frame is larger than the threshold given. 000000 64 bytes written. 000001 64 bytes written. 000010 128 bytes written. 000011 192 bytes written. ... ... 111111 4032 bytes written.

11.5.5.21 Receive Descriptor Ring 1 Start Register (ENET_RDSR1)

RDSR1 points to the beginning of circular receive buffer descriptor queue 1 in external memory. This pointer must be 64-bit aligned (bits 2–0 must be zero); however, for optimal performance the pointer should be 512-bit aligned, that is, evenly divisible by 64.

NOTE

This register must be initialized prior to operation.

Address: 30BE_0000h base + 160h offset = 30BE_0160h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R_DES_START															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	R_DES_START														0	
W														0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RDSR1 field descriptions

Field	Description
31–3 R_DES_START	Pointer to the beginning of the receive buffer descriptor queue 1.
2 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.22 Transmit Buffer Descriptor Ring 1 Start Register (ENET_TDSR1)

TDSR1 provides a pointer to the beginning of the circular transmit buffer descriptor queue 1 in external memory. This pointer must be 64-bit aligned (bits 2–0 must be zero); however, for optimal performance the pointer should be 512-bit aligned, that is, evenly divisible by 64.

NOTE

This register must be initialized prior to operation.

Address: 30BE_0000h base + 164h offset = 30BE_0164h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	X_DES_START															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	X_DES_START														0	
W														0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_TDSR1 field descriptions

Field	Description
31–3 X_DES_START	Pointer to the beginning of transmit buffer descriptor queue 1.

Table continues on the next page...

ENET_TDSR1 field descriptions (continued)

Field	Description
2 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.23 Maximum Receive Buffer Size Register - Ring 1 (ENET_MRBR1)

MRBR1 is a user-programmable register that dictates the maximum size of all ring-1 receive buffers. This value should take into consideration that the receive CRC is always written into the last receive buffer.

- To allow one maximum size frame per buffer, MRBR1 must be set to RCR[MAX_FL] or larger.
- R_BUF_SIZE is concatentated with the four least-significant bits of this register and are used as the maximum receive buffer size.
- To properly align the buffer, MRBR1 must be evenly divisible by 64. To ensure this, set the lower two bits of R_BUF_ZIZE to zero. The lower four bits are already set to zero by the device.
- To minimize bus usage (descriptor fetches), set MRBR1 greater than or equal to 256 bytes.

NOTE

This register must be initialized before operation.

Address: 30BE_0000h base + 168h offset = 30BE_0168h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																					R_BUF_SIZE							0			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_MRBR1 field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–4 R_BUF_SIZE	Receive buffer size in bytes. This value, concatenated with the four least-significant bits of this register (which are always zero), is the effective maximum receive buffer size.
Reserved	This field, which is always zero, is the four least-significant bits of the maximum receive buffer size. This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.24 Receive Descriptor Ring 2 Start Register (ENET_RDSR2)

RDSR points to the beginning of circular receive buffer descriptor queue 2 in external memory. This pointer must be 64-bit aligned (bits 2–0 must be zero); however, for optimal performance the pointer should be 512-bit aligned, that is, evenly divisible by 64.

NOTE

This register must be initialized prior to operation

Address: 30BE_0000h base + 16Ch offset = 30BE_016Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R_DES_START															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	R_DES_START														0	
W														0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RDSR2 field descriptions

Field	Description
31–3 R_DES_START	Pointer to the beginning of receive buffer descriptor queue 2.
2 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.25 Transmit Buffer Descriptor Ring 2 Start Register (ENET_TDSR2)

TDSR2 provides a pointer to the beginning of circular transmit buffer descriptor queue 2 in external memory. This pointer must be 64-bit aligned (bits 2–0 must be zero); however, for optimal performance the pointer should be 512-bit aligned, that is, evenly divisible by 64.

NOTE

This register must be initialized prior to operation

Address: 30BE_0000h base + 170h offset = 30BE_0170h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	X_DES_START															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	X_DES_START														0	
W														0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_TDSR2 field descriptions

Field	Description
31–3 X_DES_START	Pointer to the beginning of transmit buffer descriptor queue 2.
2 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.26 Maximum Receive Buffer Size Register - Ring 2 (ENET_MRBR2)

MRBR2 is a user-programmable register that dictates the maximum size of all ring-2 receive buffers. This value should take into consideration that the receive CRC is always written into the last receive buffer.

- To allow one maximum size frame per buffer, MRBR2 must be set to RCR[MAX_FL] or larger.
- R_BUF_SIZE is concatentated with the four least-significant bits of this register and are used as the maximum receive buffer size.
- To properly align the buffer, MRBR2 must be evenly divisible by 64. To ensure this, set the lower two bits of R_BUF_SIZE to zero. The lower four bits are already set to zero by the device.
- To minimize bus usage (descriptor fetches), set MRBR2 greater than or equal to 256 bytes.

NOTE

This register must be initialized prior to operation

Address: 30BE_0000h base + 174h offset = 30BE_0174h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																R_BUF_SIZE												0			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_MRBR2 field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–4 R_BUF_SIZE	Receive buffer size in bytes. This value, concatenated with the four least-significant bits of this register (which are always zero), is the effective maximum receive buffer size.
Reserved	This field, which is always zero, is the four least-significant bits of the maximum receive buffer size. This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.27 Receive Descriptor Ring 0 Start Register (ENET_RDSR)

RDSR points to the beginning of the circular receive buffer descriptor queue in external memory. This pointer must be 64-bit aligned (bits 2–0 must be zero); however, for optimal performance the pointer should be 512-bit aligned, that is, evenly divisible by 64.

NOTE

This register must be initialized prior to operation

Address: 30BE_0000h base + 180h offset = 30BE_0180h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R_DES_START															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	R_DES_START														0	
W														0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RDSR field descriptions

Field	Description
31–3 R_DES_START	Pointer to the beginning of the receive buffer descriptor queue. 0
2 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.28 Transmit Buffer Descriptor Ring 0 Start Register (ENET_TDSR)

TDSR provides a pointer to the beginning of the circular transmit buffer descriptor queue 0 in external memory. This pointer must be 64-bit aligned (bits 2–0 must be zero); however, for optimal performance the pointer should be 512-bit aligned, that is, evenly divisible by 64.

NOTE

This register must be initialized prior to operation.

Address: 30BE_0000h base + 184h offset = 30BE_0184h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	X_DES_START															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	X_DES_START														0	
W														0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_TDSR field descriptions

Field	Description
31–3 X_DES_START	Pointer to the beginning of the transmit buffer descriptor queue.
2 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.29 Maximum Receive Buffer Size Register - Ring 0 (ENET_MRBR)

The MRBR is a user-programmable register that dictates the maximum size of all ring-0 receive buffers. This value should take into consideration that the receive CRC is always written into the last receive buffer.

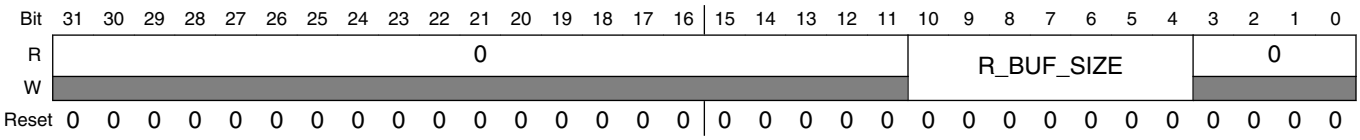
- R_BUF_SIZE is concatentated with the four least-significant bits of this register and are used as the maximum receive buffer size.
- To allow one maximum size frame per buffer, MRBR must be set to RCR[MAX_FL] or larger.

- To properly align the buffer, MRBR must be evenly divisible by 64. To ensure this, set the lower two bits of R_BUF_SIZE to zero. The lower four bits of this register are already set to zero by the device.
- To minimize bus usage (descriptor fetches), set MRBR greater than or equal to 256 bytes.

NOTE

This register must be initialized before operation.

Address: 30BE_0000h base + 188h offset = 30BE_0188h

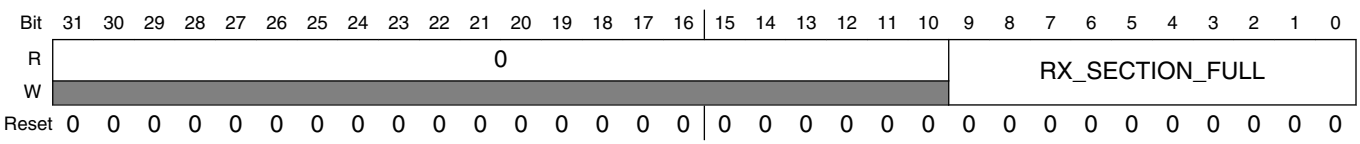


ENET_MRBR field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–4 R_BUF_SIZE	Receive buffer size in bytes. This value, concatenated with the four least-significant bits of this register (which are always zero), is the effective maximum receive buffer size.
Reserved	This field, which is always zero, is the four least-significant bits of the maximum receive buffer size. This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.30 Receive FIFO Section Full Threshold (ENET_RSFL)

Address: 30BE_0000h base + 190h offset = 30BE_0190h

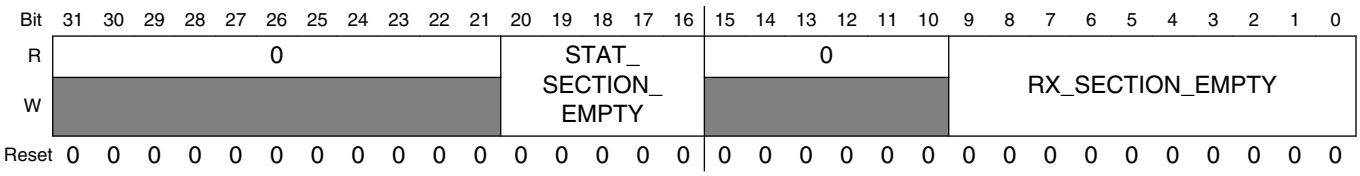


ENET_RSFL field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RX_SECTION_ FULL	Value Of Receive FIFO Section Full Threshold Value, in 64-bit words, of the receive FIFO section full threshold. Clear this field to enable store and forward on the RX FIFO. When programming a value greater than 0 (cut-through operation), it must be greater than RAEM[RX_ALMOST_EMPTY]. When the FIFO level reaches the value in this field, data is available in the Receive FIFO (cut-through operation).

11.5.5.31 Receive FIFO Section Empty Threshold (ENET_RSEM)

Address: 30BE_0000h base + 194h offset = 30BE_0194h

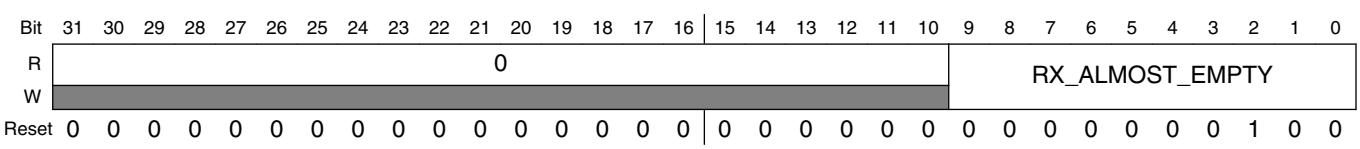


ENET_RSEM field descriptions

Field	Description
31–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20–16 STAT_ SECTION_ EMPTY	RX Status FIFO Section Empty Threshold Defines number of frames in the receive FIFO, independent of its size, that can be accepted. If the limit is reached, reception will continue normally, however a pause frame will be triggered to indicate a possible congestion to the remote device to avoid FIFO overflow. A value of 0 disables automatic pause frame generation
15–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RX_SECTION_ EMPTY	Value Of The Receive FIFO Section Empty Threshold Value, in 64-bit words, of the receive FIFO section empty threshold. When the FIFO has reached this level, a pause frame will be issued. A value of 0 disables automatic pause frame generation. When the FIFO level goes below the value programmed in this field, an XON pause frame is issued to indicate the FIFO congestion is cleared to the remote Ethernet client. **NOTE:** The section-empty threshold indications from both FIFOs are OR'ed to cause XOFF pause frame generation.

11.5.5.32 Receive FIFO Almost Empty Threshold (ENET_RAEM)

Address: 30BE_0000h base + 198h offset = 30BE_0198h



ENET_RAEM field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

ENET_RAEM field descriptions (continued)

Field	Description
RX_ALMOST_EMPTY	Value Of The Receive FIFO Almost Empty Threshold Value, in 64-bit words, of the receive FIFO almost empty threshold. When the FIFO level reaches the value programmed in this field and the end-of-frame has not been received for the frame yet, the core receive read control stops FIFO read (and subsequently stops transferring data to the MAC client application). It continues to deliver the frame, if again more data than the threshold or the end-of-frame is available in the FIFO. A minimum value of 4 should be set.

11.5.5.33 Receive FIFO Almost Full Threshold (ENET_RAFL)

Address: 30BE_0000h base + 19Ch offset = 30BE_019Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																						RX_ALMOST_FULL									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

ENET_RAFL field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RX_ALMOST_FULL	Value Of The Receive FIFO Almost Full Threshold Value, in 64-bit words, of the receive FIFO almost full threshold. When the FIFO level comes close to the maximum, so that there is no more space for at least RX_ALMOST_FULL number of words, the MAC stops writing data in the FIFO and truncates the received frame to avoid FIFO overflow. The corresponding error status will be set when the frame is delivered to the application. A minimum value of 4 should be set.

11.5.5.34 Transmit FIFO Section Empty Threshold (ENET_TSEM)

Address: 30BE_0000h base + 1A0h offset = 30BE_01A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																						TX_SECTION_EMPTY									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_TSEM field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TX_SECTION_EMPTY	Value Of The Transmit FIFO Section Empty Threshold

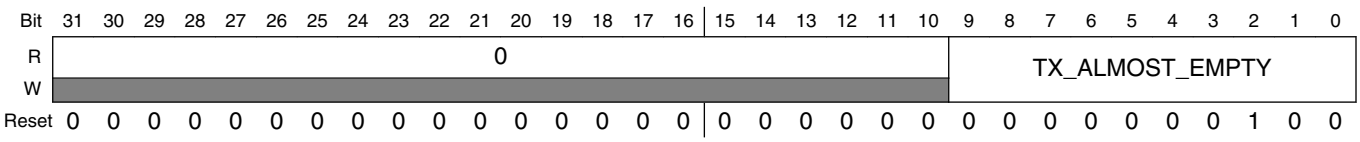
Table continues on the next page...

ENET_TSEM field descriptions (continued)

Field	Description
	Value, in 64-bit words, of the transmit FIFO section empty threshold. See Transmit FIFO for more information.

11.5.5.35 Transmit FIFO Almost Empty Threshold (ENET_TAEM)

Address: 30BE_0000h base + 1A4h offset = 30BE_01A4h

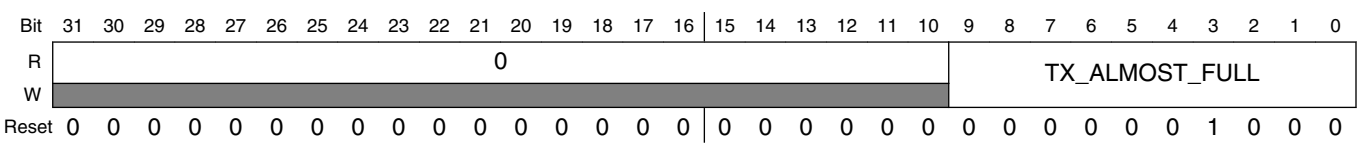


ENET_TAEM field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TX_ALMOST_EMPTY	Value of Transmit FIFO Almost Empty Threshold Value, in 64-bit words, of the transmit FIFO almost empty threshold. When the FIFO level reaches the value programmed in this field, and no end-of-frame is available for the frame, the MAC transmit logic, to avoid FIFO underflow, stops reading the FIFO and transmits a frame with an MII error indication. See Transmit FIFO for more information. A minimum value of 4 should be set.

11.5.5.36 Transmit FIFO Almost Full Threshold (ENET_TAFL)

Address: 30BE_0000h base + 1A8h offset = 30BE_01A8h



ENET_TAFL field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TX_ALMOST_FULL	Value Of The Transmit FIFO Almost Full Threshold Value, in 64-bit words, of the transmit FIFO almost full threshold. A minimum value of six is required . A recommended value of at least 8 should be set allowing a latency of two clock cycles to the application. If more latency is required the value can be increased as necessary (latency = TAFL - 5).

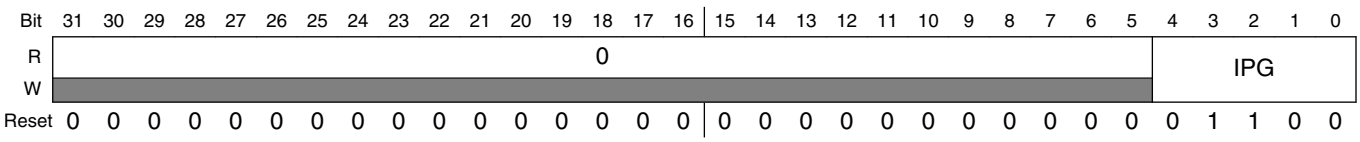
Table continues on the next page...

ENET_TAFL field descriptions (continued)

Field	Description
	When the FIFO level comes close to the maximum, so that there is no more space for at least TX_ALMOST_FULL number of words, the pin ff_tx_rdy is deasserted. If the application does not react on this signal, the FIFO write control logic, to avoid FIFO overflow, truncates the current frame and sets the error status. As a result, the frame will be transmitted with an GMII/MII error indication. See Transmit FIFO for more information. **NOTE:** A FIFO overflow is a fatal error and requires a global reset on the transmit datapath or at least deassertion of ETHEREN.

11.5.5.37 Transmit Inter-Packet Gap (ENET_TIPG)

Address: 30BE_0000h base + 1ACh offset = 30BE_01ACh

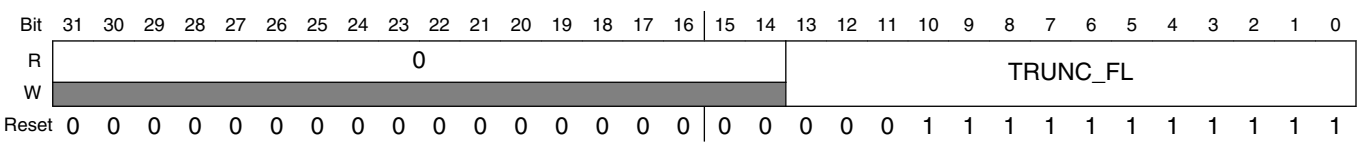


ENET_TIPG field descriptions

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
IPG	Transmit Inter-Packet Gap Indicates the IPG, in bytes, between transmitted frames. Valid values range from 8 to 26. If the written value is less than 8 or greater than 26, the internal (effective) IPG is 12. **NOTE:** The IPG value read will be the value that was written, even if it is out of range.

11.5.5.38 Frame Truncation Length (ENET_FTRL)

Address: 30BE_0000h base + 1B0h offset = 30BE_01B0h



ENET_FTRL field descriptions

Field	Description
31–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TRUNC_FL	Frame Truncation Length

Table continues on the next page...

ENET_FTRL field descriptions (continued)

Field	Description
	Indicates the value a receive frame is truncated, if it is greater than this value. Must be greater than or equal to RCR[MAX_FL]. **NOTE:** Truncation happens at TRUNC_FL. However, when truncation occurs, the application (FIFO) may receive less data, guaranteeing that it never receives more than the set limit.

11.5.5.39 Transmit Accelerator Function Configuration (ENET_TACC)

TACC controls accelerator actions when sending frames. The register can be changed before or after each frame, but it must remain unmodified during frame writes into the transmit FIFO.

The TFWR[STRFWD] field must be set to use the checksum feature.

Address: 30BE_0000h base + 1C0h offset = 30BE_01C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W									0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R												PROCHK	IPCHK			SHIFT16
W						0								0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_TACC field descriptions

Field	Description
31–5 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
4 PROCHK	Enables insertion of protocol checksum. 0 Checksum not inserted. 1 If an IP frame with a known protocol is transmitted, the checksum is inserted automatically into the frame. The checksum field must be cleared. The other frames are not modified.
3 IPCHK	Enables insertion of IP header checksum. 0 Checksum is not inserted. 1 If an IP frame is transmitted, the checksum is inserted automatically. The IP header checksum field must be cleared. If a non-IP frame is transmitted the frame is not modified.

Table continues on the next page...

ENET_TACC field descriptions (continued)

Field	Description
2–1 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
0 SHIFT16	TX FIFO Shift-16 0 Disabled. 1 Indicates to the transmit data FIFO that the written frames contain two additional octets before the frame data. This means the actual frame begins at bit 16 of the first word written into the FIFO. This function allows putting the frame payload on a 32-bit boundary in memory, as the 14-byte Ethernet header is extended to a 16-byte header.

11.5.5.40 Receive Accelerator Function Configuration (ENET_RACC)

Address: 30BE_0000h base + 1C4h offset = 30BE_01C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									SHIFT16	LINEDIS				PRODIS	IPDIS	PADREM
W	0										0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RACC field descriptions

Field	Description
31–8 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
7 SHIFT16	RX FIFO Shift-16 When this field is set, the actual frame data starts at bit 16 of the first word read from the RX FIFO aligning the Ethernet payload on a 32-bit boundary. **NOTE:** This function only affects the FIFO storage and has no influence on the statistics, which use the actual length of the frame received. 0 Disabled. 1 Instructs the MAC to write two additional bytes in front of each frame received into the RX FIFO.
6 LINEDIS	Enable Discard Of Frames With MAC Layer Errors 0 Frames with errors are not discarded. 1 Any frame received with a CRC, length, or PHY error is automatically discarded and not forwarded to the user application interface.

Table continues on the next page...

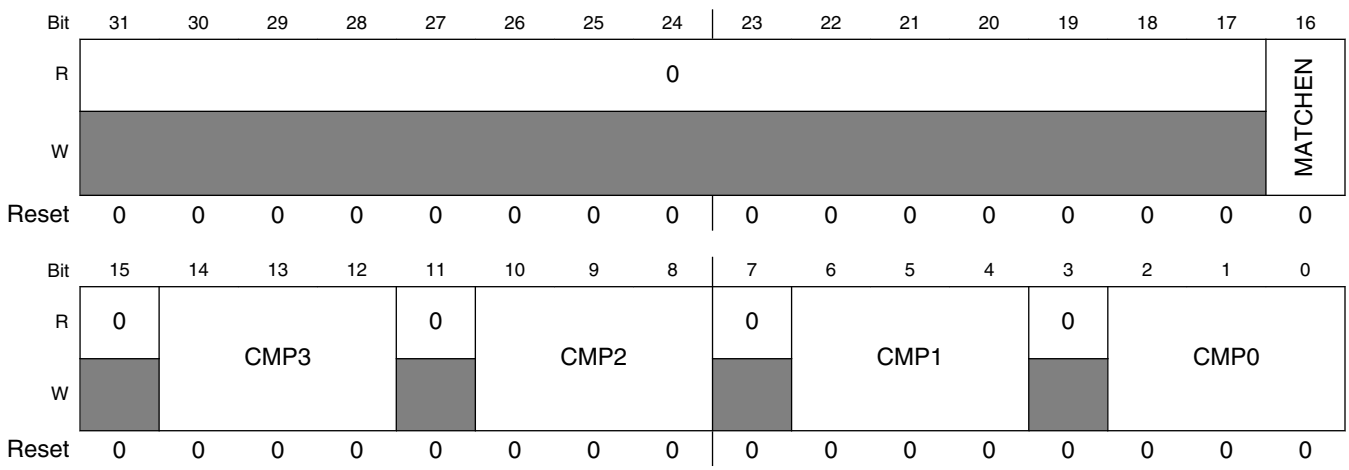
ENET_RACC field descriptions (continued)

Field	Description
5–3 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
2 PRODIS	Enable Discard Of Frames With Wrong Protocol Checksum 0 Frames with wrong checksum are not discarded. 1 If a TCP/IP, UDP/IP, or ICMP/IP frame is received that has a wrong TCP, UDP, or ICMP checksum, the frame is discarded. Discarding is only available when the RX FIFO operates in store and forward mode (RSFL cleared).
1 IPDIS	Enable Discard Of Frames With Wrong IPv4 Header Checksum 0 Frames with wrong IPv4 header checksum are not discarded. 1 If an IPv4 frame is received with a mismatching header checksum, the frame is discarded. IPv6 has no header checksum and is not affected by this setting. Discarding is only available when the RX FIFO operates in store and forward mode (RSFL cleared).
0 PADREM	Enable Padding Removal For Short IP Frames 0 Padding not removed. 1 Any bytes following the IP payload section of the frame are removed from the frame.

11.5.5.41 Receive Classification Match Register for Class n (ENET_RCMR*n*)

This match register allows specifying up to four priorities, which are tested (OR'ed) simultaneously. The match detection uses the extracted VLAN field according to the rules for VLAN detection configured through the ECR register. If both match registers, RCMR1 and RCMR2, report a match at the same time, only the class 1 match is indicated as the final result.

Address: 30BE_0000h base + 1C8h offset + (4d × i), where i=0d to 1d



ENET_RCMR*n* field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 MATCHEN	Match Enable **NOTE:** A comparison is done only on incoming VLAN frames. If no VLAN frame is received no match will occur. If both match registers have overlapping compare values and hence can match both on the same frame, only class 1 will be indicated and the class 2 match is ignored. 0 Disabled (default): no compares will occur and the classification indicator for this class will never assert. 1 The register contents are valid and a comparison with all compare values is done when a VLAN frame is received.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–12 CMP3	Compare 3 Fourth value to compare against. If unused it must be set to the same value as CMP0.
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 CMP2	Compare 2 Third value to compare against. If unused it must be set to the same value as CMP0.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 CMP1	Compare 1 Second value to compare against. If unused it must be set to the same value as CMP0.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CMP0	Compare 0 A three-bit value that will be compared with the frame's VLAN priority field (if a VLAN frame is received). All four compare values, CMP0..3, will be used in parallel. If any of the values match, a match for the class is reported (if MATCHEN is 1). **NOTE:** To implement a single priority match, all four compare values must be set to the same value.

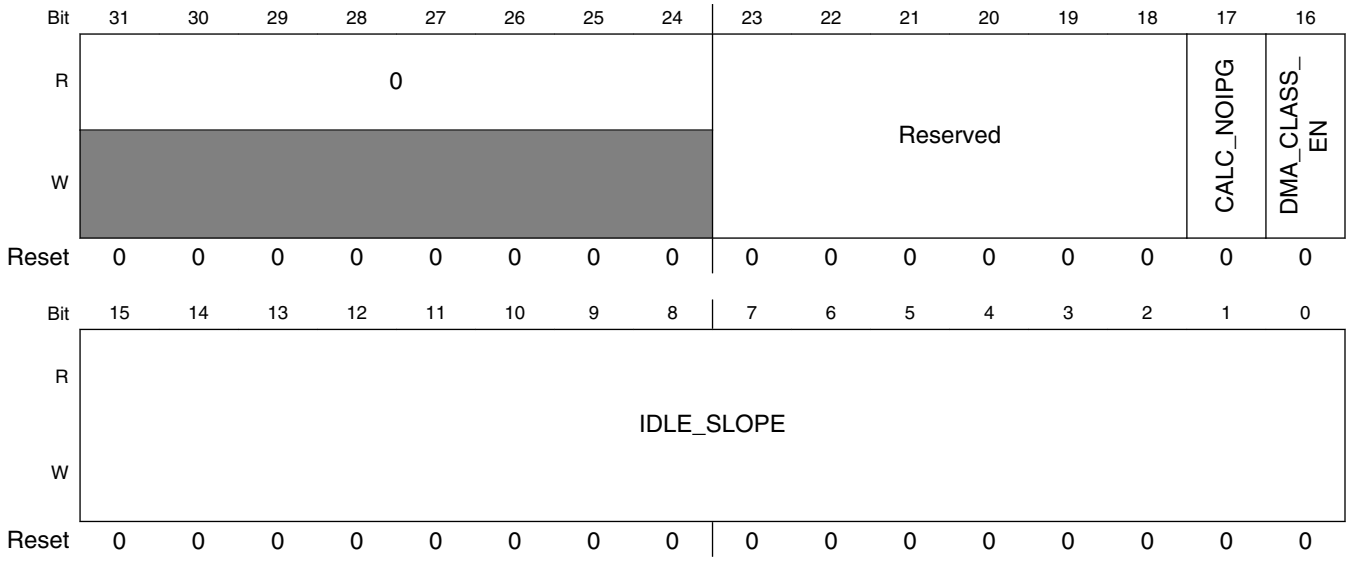
11.5.5.42 DMA Class Based Configuration (ENET_DMA*n*CFG)

The DMA class based configuration registers are used to configure the DMA controller interface to support the additional class 1 (buffer descriptor ring 1) and class 2 (buffer descriptor ring 2) traffic and define configuration options such as bandwidth allocation as needed.

NOTE

The registers are cleared when ECR[ETHEREN] becomes 0.

Address: 30BE_0000h base + 1D8h offset + (4d × i), where i=0d to 1d



ENET_DMA*n*CFG field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–18 Reserved	This field is reserved. **NOTE:** Write only zeroes to this field.
17 CALC_NOIPG	Calculate no IPG Disable inclusion of IPG bytes for bandwidth calculations. 0 The traffic shaper function should consider 12 octets of IPG in addition to the frame data transferred for a frame when doing bandwidth calculations. This is the default. 1 Addition of 12 bytes for the IPG should be omitted when calculating the bandwidth (for traffic shaping, when writing a frame into the transmit FIFO, the shaper will usually consider 12 bytes of IPG for every frame as part of the bandwidth allocated by the frame. This addition can be suppressed, meaning short frames will become more bandwidth than large frames due to the relation of data to IPG overhead).
16 DMA_CLASS_EN	DMA class enable 0 The DMA controller's channel for the class is not used. **NOTE:** Disabling the DMA controller of a class also requires disabling the class match comparator for the class (see registers RCMRn). When class 1 and class 2 queues are disabled then their frames will be placed in queue 0. 1 Enable the DMA controller to support the corresponding descriptor ring for this class of traffic.
IDLE_SLOPE	Idle slope 16-bit value to define the per class idle slope setting used by the credit based shaper defining allocated bandwidth for the class. This value is used to calculate the BW (bandwidth) fraction, given by the equation, BW fraction = 1/(1+512/IDLE_SLOPE). Idle slope is restricted to certain values. For values less than 128, idle slope = 2^n, where n = 0, 1, 2, ...6. For values equal to or greater than 128, idle slope = 128×m, where m = 1, 2, 3, ...12.

Table continues on the next page...

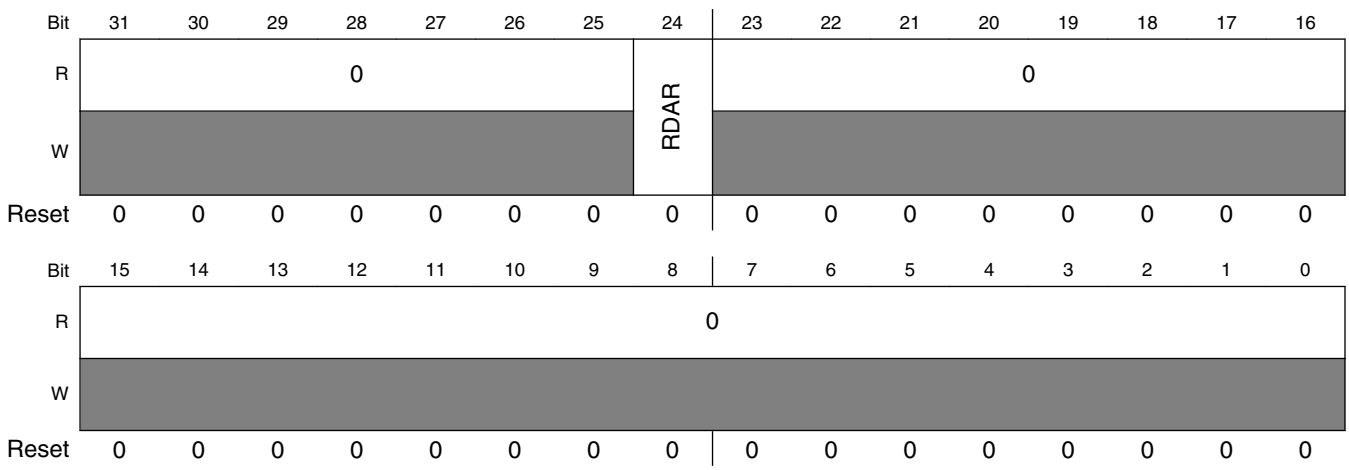
ENET_DMA*n*CFG field descriptions (continued)

Field	Description
	Example 1. BW fraction = 0.20 = 1/(1+(512/128)); therefore idleslope = 128. Example 2. BW fraction = 0.33 = 1/(1+(512/256)); therefore idleslope = 256. Example 3. BW fraction = 0.75 = 1/(1+(512/1536)); therefore idleslope = 1536. **NOTE:** For AVB applications, the BW fraction of class 1 and class 2 combined must not exceed .75.

11.5.5.43 Receive Descriptor Active Register - Ring 1 (ENET_RDAR1)

RDAR1 is a command register, written by the user, to indicate that the receive descriptor ring 1 has been updated, that is, that the driver produced empty receive buffers with the empty bit set.

Address: 30BE_0000h base + 1E0h offset = 30BE_01E0h



ENET_RDAR1 field descriptions

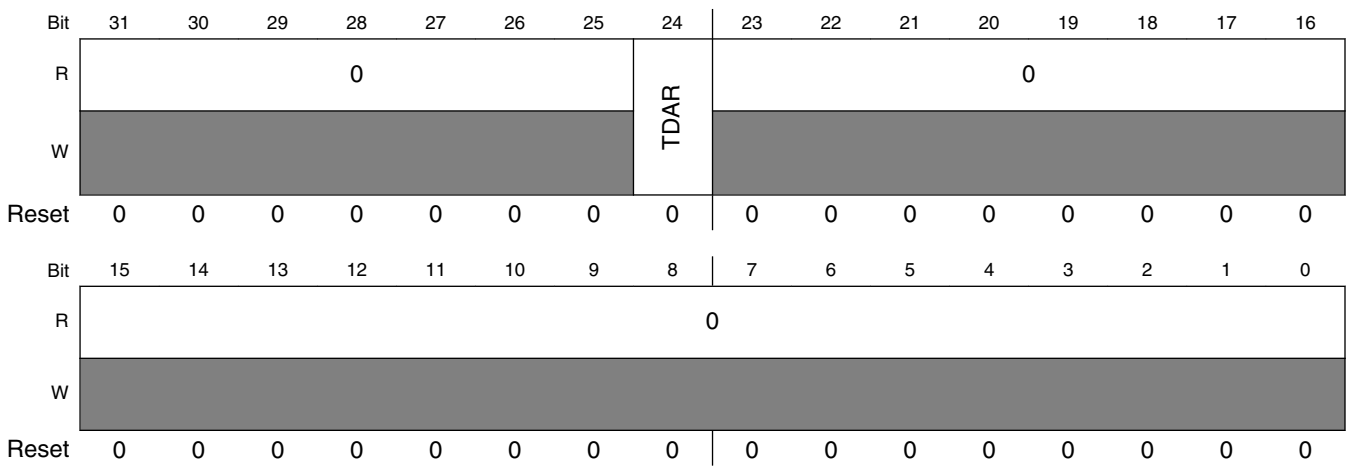
Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 RDAR	Receive Descriptor Active Always set to 1 when this register is written, regardless of the value written. This field is cleared by the MAC device when no additional empty descriptors remain in the receive ring. It is also cleared when ECR[ETHEREN] transitions from set to cleared or when ECR[RESET] is set.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.44 Transmit Descriptor Active Register - Ring 1 (ENET_TDAR1)

TDAR1 is a command register that the user writes to indicate that transmit descriptor ring 1 has been updated, that is, that transmit buffers have been produced by the driver with the ready bit set in the buffer descriptor.

The TDAR register is cleared at reset, when ECR[ETHEREN] transitions from set to cleared, or when ECR[RESET] is set.

Address: 30BE_0000h base + 1E4h offset = 30BE_01E4h



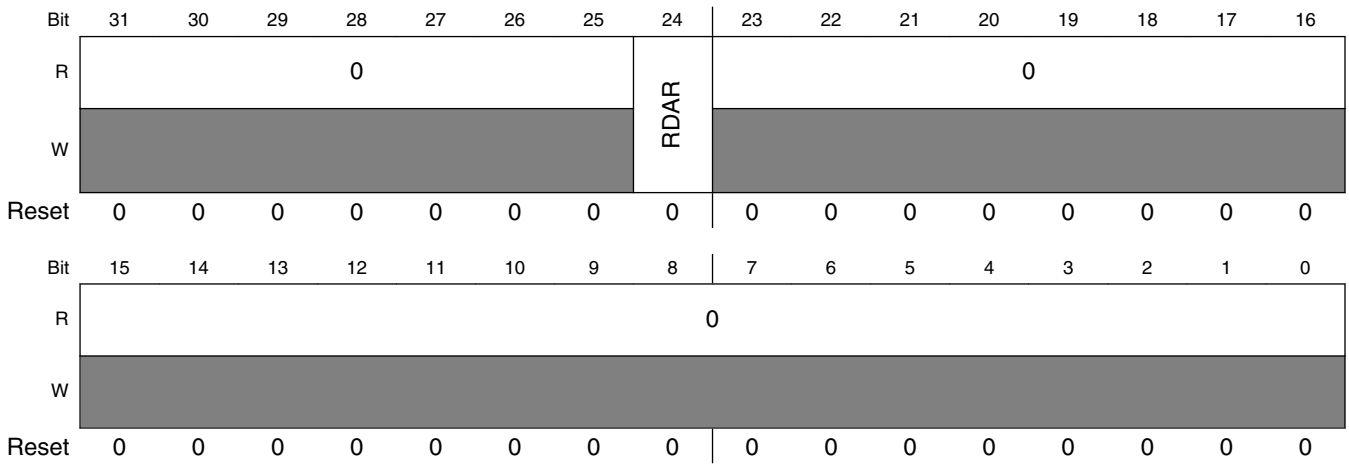
ENET_TDAR1 field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 TDAR	Transmit Descriptor Active Always set to 1 when this register is written, regardless of the value written. This bit is cleared by the MAC device when no additional ready descriptors remain in the transmit ring. Also cleared when ECR[ETHEREN] transitions from set to cleared or when ECR[RESET] is set.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.45 Receive Descriptor Active Register - Ring 2 (ENET_RDAR2)

RDAR2 is a command register, written by the user, to indicate that the receive descriptor ring 2 has been updated, that is, that the driver produced empty receive buffers with the empty bit set.

Address: 30BE_0000h base + 1E8h offset = 30BE_01E8h



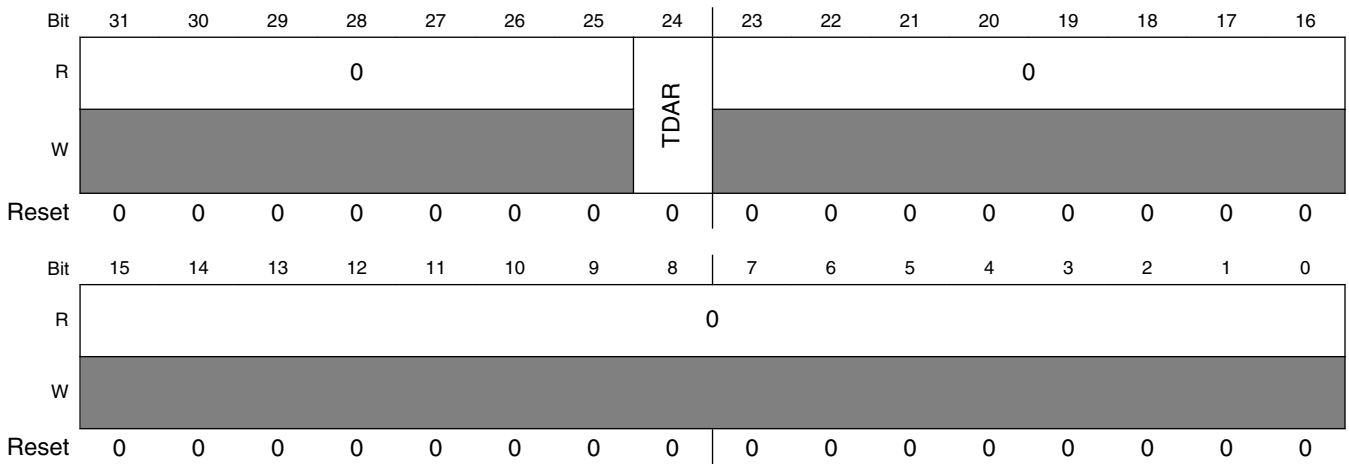
ENET_RDAR2 field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 RDAR	Receive Descriptor Active Always set to 1 when this register is written, regardless of the value written. This field is cleared by the MAC device when no additional empty descriptors remain in the receive ring. It is also cleared when ECR[ETHEREN] transitions from set to cleared or when ECR[RESET] is set.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.46 Transmit Descriptor Active Register - Ring 2 (ENET_TDAR2)

TDAR2 is a command register that the user writes to indicate that transmit descriptor ring 2 has been updated, that is, that transmit buffers have been produced by the driver with the ready bit set in the buffer descriptor.

Address: 30BE_0000h base + 1ECh offset = 30BE_01ECh



ENET_TDAR2 field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 TDAR	Transmit Descriptor Active Always set to 1 when this register is written, regardless of the value written. This bit is cleared by the MAC device when no additional ready descriptors remain in the transmit ring. Also cleared when ECR[ETHEREN] transitions from set to cleared or when ECR[RESET] is set.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

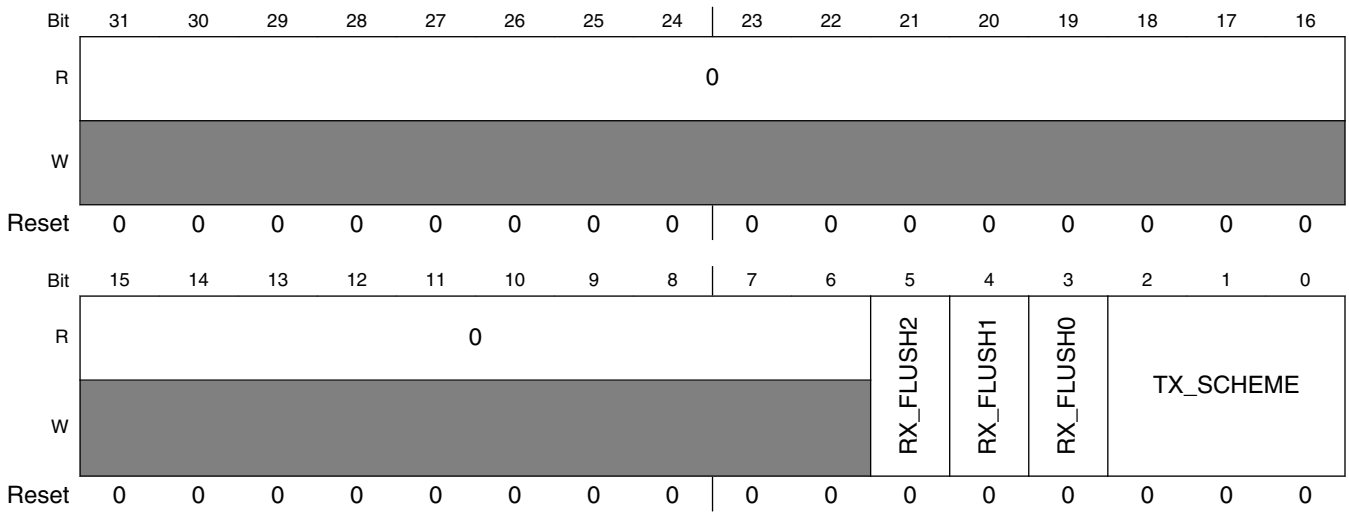
11.5.5.47 QOS Scheme (ENET_QOS)

This register sets the QOS scheme.

NOTE

When both class 1 and class 2 are disabled, RX flushing for these rings must also be disabled.

Address: 30BE_0000h base + 1F0h offset = 30BE_01F0h



ENET_QOS field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 RX_FLUSH2	RX Flush Ring 2 Enable or disable RX Flush for ring 2. See Receive flushing. 0 Disable 1 Enable
4 RX_FLUSH1	RX Flush Ring 1 Enable or disable RX Flush for ring 1. See Receive flushing. 0 Disable 1 Enable
3 RX_FLUSH0	RX Flush Ring 0 Enable or disable RX Flush for ring 0. See Receive flushing. 0 Disable 1 Enable
TX_SCHEME	TX scheme configuration Configuration information for DMA to select transmitter queue selection/arbitration scheme. 000 Credit-based scheme 001 Round-robin scheme 010-111 Reserved

11.5.5.48 Reserved Statistic Register (ENET_RMON_T_DROP)

Address: 30BE_0000h base + 200h offset = 30BE_0200h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_DROP field descriptions

Field	Description
Reserved	This read-only field always has the value 0. This field is reserved.

11.5.5.49 Tx Packet Count Statistic Register (ENET_RMON_T_PACKETS)

Address: 30BE_0000h base + 204h offset = 30BE_0204h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TXPKTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_PACKETS field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Packet count Transmit packet count

11.5.5.50 Tx Broadcast Packets Statistic Register (ENET_RMON_T_BC_PKT)

RMON Tx Broadcast Packets

Address: 30BE_0000h base + 208h offset = 30BE_0208h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TXPKTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_BC_PKT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Broadcast packets

11.5.5.51 Tx Multicast Packets Statistic Register (ENET_RMON_T_MC_PKT)

Address: 30BE_0000h base + 20Ch offset = 30BE_020Ch

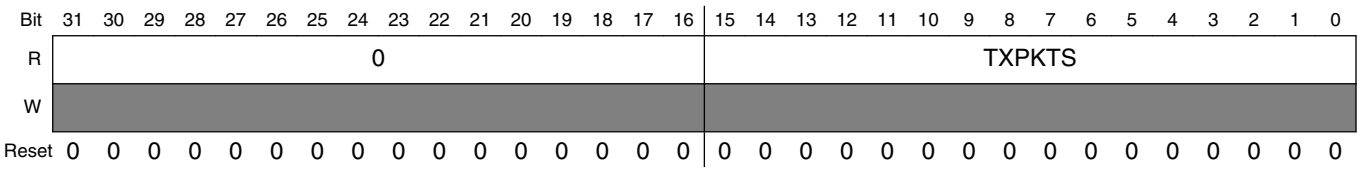
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TXPKTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_MC_PKT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Multicast packets

11.5.5.52 Tx Packets with CRC/Align Error Statistic Register (ENET_RMON_T_CRC_ALIGN)

Address: 30BE_0000h base + 210h offset = 30BE_0210h

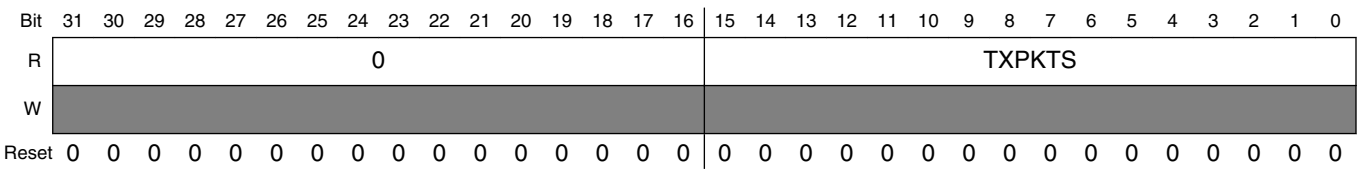


ENET_RMON_T_CRC_ALIGN field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Packets with CRC/align error

11.5.5.53 Tx Packets Less Than Bytes and Good CRC Statistic Register (ENET_RMON_T_UNDERSIZE)

Address: 30BE_0000h base + 214h offset = 30BE_0214h

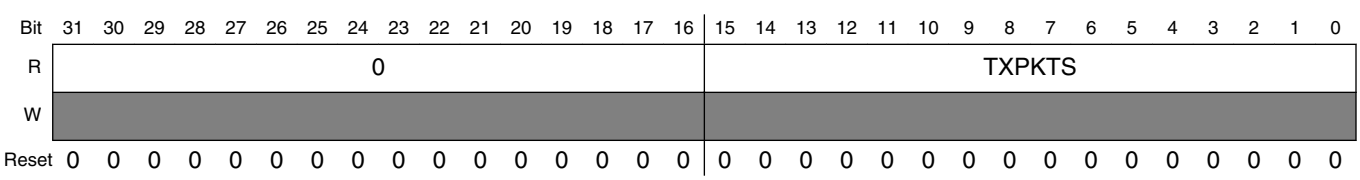


ENET_RMON_T_UNDERSIZE field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Number of transmit packets less than 64 bytes with good CRC

11.5.5.54 Tx Packets GT MAX_FL bytes and Good CRC Statistic Register (ENET_RMON_T_OVERSIZE)

Address: 30BE_0000h base + 218h offset = 30BE_0218h



ENET_RMON_T_OVERSIZE field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Number of transmit packets greater than MAX_FL bytes with good CRC

11.5.5.55 Tx Packets Less Than 64 Bytes and Bad CRC Statistic Register (ENET_RMON_T_FRAG)

.

Address: 30BE_0000h base + 21Ch offset = 30BE_021Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TXPKTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_FRAG field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Number of packets less than 64 bytes with bad CRC

11.5.5.56 Tx Packets Greater Than MAX_FL bytes and Bad CRC Statistic Register (ENET_RMON_T_JAB)

Address: 30BE_0000h base + 220h offset = 30BE_0220h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TXPKTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_JAB field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Number of transmit packets greater than MAX_FL bytes and bad CRC

11.5.5.57 Tx Collision Count Statistic Register (ENET_RMON_T_COL)

Address: 30BE_0000h base + 224h offset = 30BE_0224h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TXPKTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_COL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Number of transmit collisions

11.5.5.58 Tx 64-Byte Packets Statistic Register (ENET_RMON_T_P64)

.

Address: 30BE_0000h base + 228h offset = 30BE_0228h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TXPKTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_P64 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Number of 64-byte transmit packets

11.5.5.59 Tx 65- to 127-byte Packets Statistic Register (ENET_RMON_T_P65TO127)

Address: 30BE_0000h base + 22Ch offset = 30BE_022Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TXPKTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_P65TO127 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Number of 65- to 127-byte transmit packets

11.5.5.60 Tx 128- to 255-byte Packets Statistic Register (ENET_RMON_T_P128TO255)

Address: 30BE_0000h base + 230h offset = 30BE_0230h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TXPKTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_P128TO255 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Number of 128- to 255-byte transmit packets

11.5.5.61 Tx 256- to 511-byte Packets Statistic Register (ENET_RMON_T_P256TO511)

Address: 30BE_0000h base + 234h offset = 30BE_0234h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TXPKTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_P256TO511 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Number of 256- to 511-byte transmit packets

11.5.5.62 Tx 512- to 1023-byte Packets Statistic Register (ENET_RMON_T_P512TO1023)

.

Address: 30BE_0000h base + 238h offset = 30BE_0238h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TXPKTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_P512TO1023 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Number of 512- to 1023-byte transmit packets

11.5.5.63 Tx 1024- to 2047-byte Packets Statistic Register (ENET_RMON_T_P1024TO2047)

Address: 30BE_0000h base + 23Ch offset = 30BE_023Ch

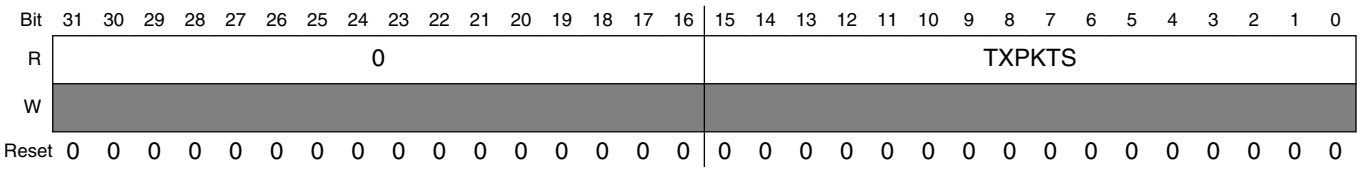
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TXPKTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_T_P1024TO2047 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Number of 1024- to 2047-byte transmit packets

11.5.5.64 Tx Packets Greater Than 2048 Bytes Statistic Register (ENET_RMON_T_P_GTE2048)

Address: 30BE_0000h base + 240h offset = 30BE_0240h

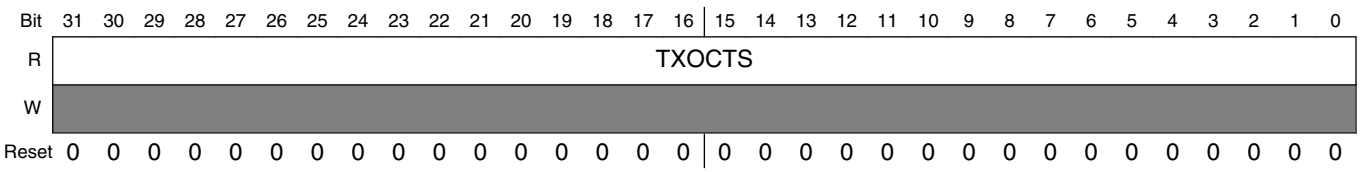


ENET_RMON_T_P_GTE2048 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXPKTS	Number of transmit packets greater than 2048 bytes

11.5.5.65 Tx Octets Statistic Register (ENET_RMON_T_OCTETS)

Address: 30BE_0000h base + 244h offset = 30BE_0244h

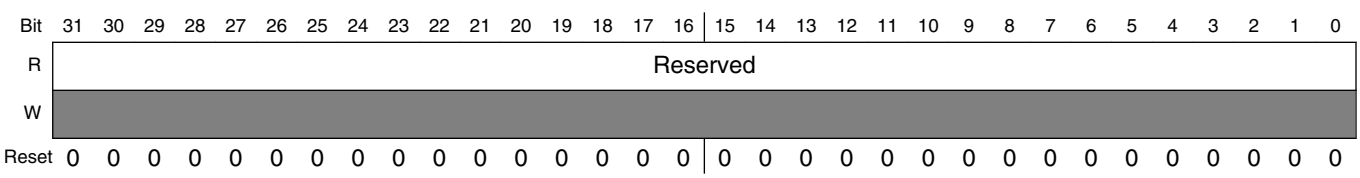


ENET_RMON_T_OCTETS field descriptions

Field	Description
TXOCTS	Number of transmit octets

11.5.5.66 Reserved Statistic Register (ENET_IEEE_T_DROP)

Address: 30BE_0000h base + 248h offset = 30BE_0248h

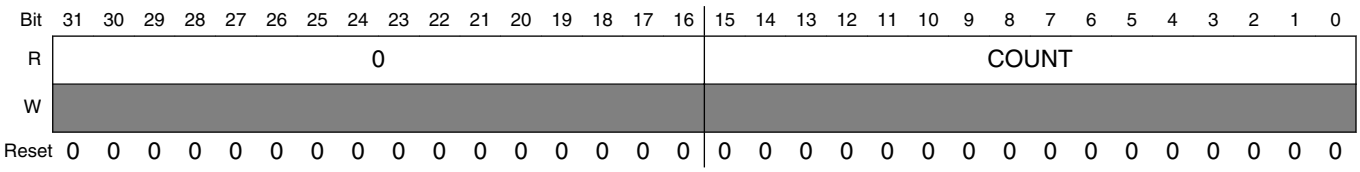


ENET_IEEE_T_DROP field descriptions

Field	Description
Reserved	This read-only field always has the value 0. This field is reserved.

11.5.5.67 Frames Transmitted OK Statistic Register (ENET_IEEE_T_FRAME_OK)

Address: 30BE_0000h base + 24Ch offset = 30BE_024Ch

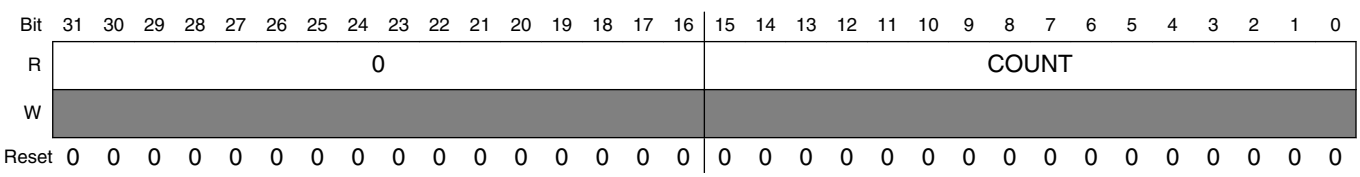


ENET_IEEE_T_FRAME_OK field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of frames transmitted OK **NOTE:** Does not increment for the broadcast frames when broadcast reject is enabled and promiscuous mode is disabled within the receive control register (RCR).

11.5.5.68 Frames Transmitted with Single Collision Statistic Register (ENET_IEEE_T_1COL)

Address: 30BE_0000h base + 250h offset = 30BE_0250h



ENET_IEEE_T_1COL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of frames transmitted with one collision

11.5.5.69 Frames Transmitted with Multiple Collisions Statistic Register (ENET_IEEE_T_MCOL)

Address: 30BE_0000h base + 254h offset = 30BE_0254h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_IEEE_T_MCOL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of frames transmitted with multiple collisions

11.5.5.70 Frames Transmitted after Deferral Delay Statistic Register (ENET_IEEE_T_DEF)

Address: 30BE_0000h base + 258h offset = 30BE_0258h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_IEEE_T_DEF field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of frames transmitted with deferral delay

11.5.5.71 Frames Transmitted with Late Collision Statistic Register (ENET_IEEE_T_LCOL)

Address: 30BE_0000h base + 25Ch offset = 30BE_025Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_IEEE_T_LCOL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of frames transmitted with late collision

11.5.5.72 Frames Transmitted with Excessive Collisions Statistic Register (ENET_IEEE_T_EXCOL)

Address: 30BE_0000h base + 260h offset = 30BE_0260h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_IEEE_T_EXCOL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of frames transmitted with excessive collisions

11.5.5.73 Frames Transmitted with Tx FIFO Underrun Statistic Register (ENET_IEEE_T_MACERR)

Address: 30BE_0000h base + 264h offset = 30BE_0264h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_IEEE_T_MACERR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of frames transmitted with transmit FIFO underrun

11.5.5.74 Frames Transmitted with Carrier Sense Error Statistic Register (ENET_IEEE_T_CSERR)

Address: 30BE_0000h base + 268h offset = 30BE_0268h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_IEEE_T_CSERR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of frames transmitted with carrier sense error

11.5.5.75 Reserved Statistic Register (ENET_IEEE_T_SQE)

Address: 30BE_0000h base + 26Ch offset = 30BE_026Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_IEEE_T_SQE field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	This read-only field is reserved and always has the value 0. **NOTE:** Counter not implemented as no SQE information is available.

11.5.5.76 Flow Control Pause Frames Transmitted Statistic Register (ENET_IEEE_T_FDXFC)

Address: 30BE_0000h base + 270h offset = 30BE_0270h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_IEEE_T_FDXFC field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of flow-control pause frames transmitted

11.5.5.77 Octet Count for Frames Transmitted w/o Error Statistic Register (ENET_IEEE_T_OCTETS_OK)

Address: 30BE_0000h base + 274h offset = 30BE_0274h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNT																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_IEEE_T_OCTETS_OK field descriptions

Field	Description
COUNT	Octet count for frames transmitted without error **NOTE** Counts total octets (includes header and FCS fields).

11.5.5.78 Rx Packet Count Statistic Register (ENET_RMON_R_PACKETS)

Address: 30BE_0000h base + 284h offset = 30BE_0284h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_R_PACKETS field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of packets received

11.5.5.79 Rx Broadcast Packets Statistic Register (ENET_RMON_R_BC_PKT)

Address: 30BE_0000h base + 288h offset = 30BE_0288h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_R_BC_PKT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of receive broadcast packets

11.5.5.80 Rx Multicast Packets Statistic Register (ENET_RMON_R_MC_PKT)

Address: 30BE_0000h base + 28Ch offset = 30BE_028Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_R_MC_PKT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of receive multicast packets

11.5.5.81 Rx Packets with CRC/Align Error Statistic Register (ENET_RMON_R_CRC_ALIGN)

Address: 30BE_0000h base + 290h offset = 30BE_0290h

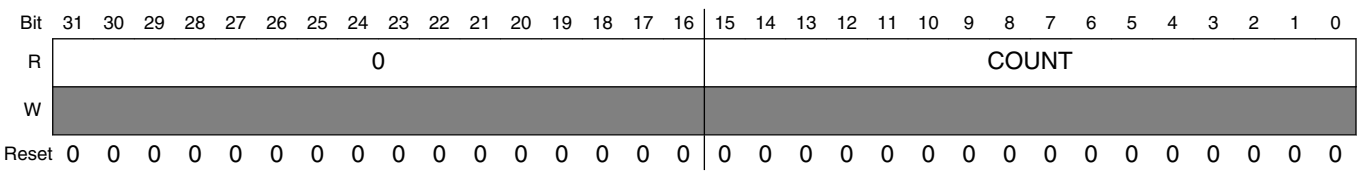
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_R_CRC_ALIGN field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of receive packets with CRC or align error

11.5.5.82 Rx Packets with Less Than 64 Bytes and Good CRC Statistic Register (ENET_RMON_R_UNDERSIZE)

Address: 30BE_0000h base + 294h offset = 30BE_0294h

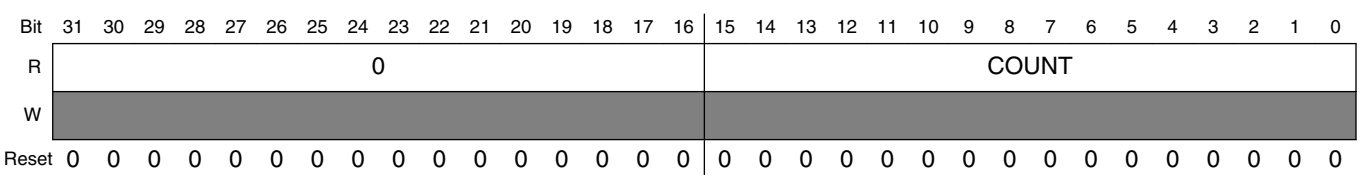


ENET_RMON_R_UNDERSIZE field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of receive packets with less than 64 bytes and good CRC

11.5.5.83 Rx Packets Greater Than MAX_FL and Good CRC Statistic Register (ENET_RMON_R_OVERSIZE)

Address: 30BE_0000h base + 298h offset = 30BE_0298h



ENET_RMON_R_OVERSIZE field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of receive packets greater than MAX_FL and good CRC

11.5.5.84 Rx Packets Less Than 64 Bytes and Bad CRC Statistic Register (ENET_RMON_R_FRAG)

Address: 30BE_0000h base + 29Ch offset = 30BE_029Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_R_FRAG field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of receive packets with less than 64 bytes and bad CRC

11.5.5.85 Rx Packets Greater Than MAX_FL Bytes and Bad CRC Statistic Register (ENET_RMON_R_JAB)

Address: 30BE_0000h base + 2A0h offset = 30BE_02A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_R_JAB field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of receive packets greater than MAX_FL and bad CRC

11.5.5.86 Reserved Statistic Register (ENET_RMON_R_RESVD_0)

Address: 30BE_0000h base + 2A4h offset = 30BE_02A4h

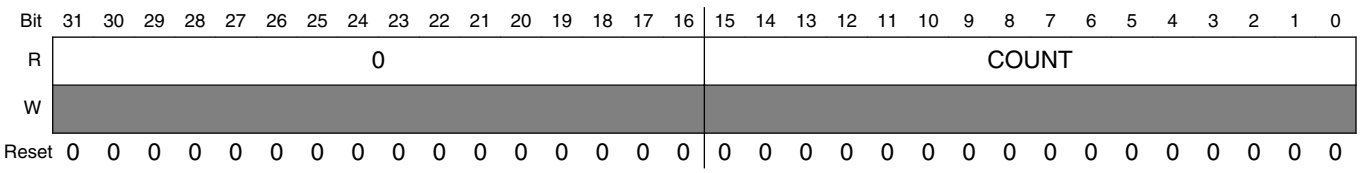
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																0															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_R_RESVD_0 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.5.5.87 Rx 64-Byte Packets Statistic Register (ENET_RMON_R_P64)

Address: 30BE_0000h base + 2A8h offset = 30BE_02A8h

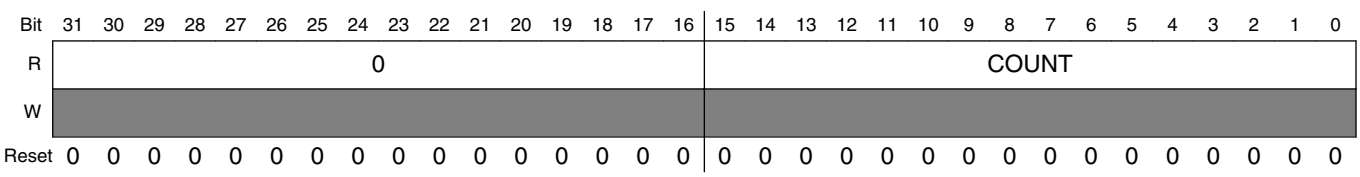


ENET_RMON_R_P64 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of 64-byte receive packets

11.5.5.88 Rx 65- to 127-Byte Packets Statistic Register (ENET_RMON_R_P65TO127)

Address: 30BE_0000h base + 2ACh offset = 30BE_02ACh



ENET_RMON_R_P65TO127 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of 65- to 127-byte recieve packets

11.5.5.89 Rx 128- to 255-Byte Packets Statistic Register (ENET_RMON_R_P128TO255)

Address: 30BE_0000h base + 2B0h offset = 30BE_02B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_R_P128TO255 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of 128- to 255-byte recieve packets

11.5.5.90 Rx 256- to 511-Byte Packets Statistic Register (ENET_RMON_R_P256TO511)

Address: 30BE_0000h base + 2B4h offset = 30BE_02B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_R_P256TO511 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of 256- to 511-byte recieve packets

11.5.5.91 Rx 512- to 1023-Byte Packets Statistic Register (ENET_RMON_R_P512TO1023)

Address: 30BE_0000h base + 2B8h offset = 30BE_02B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_R_P512TO1023 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of 512- to 1023-byte recieve packets

11.5.5.92 Rx 1024- to 2047-Byte Packets Statistic Register (ENET_RMON_R_P1024TO2047)

Address: 30BE_0000h base + 2BCh offset = 30BE_02BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_R_P1024TO2047 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of 1024- to 2047-byte recieve packets

11.5.5.93 Rx Packets Greater than 2048 Bytes Statistic Register (ENET_RMON_R_P_GTE2048)

Address: 30BE_0000h base + 2C0h offset = 30BE_02C0h

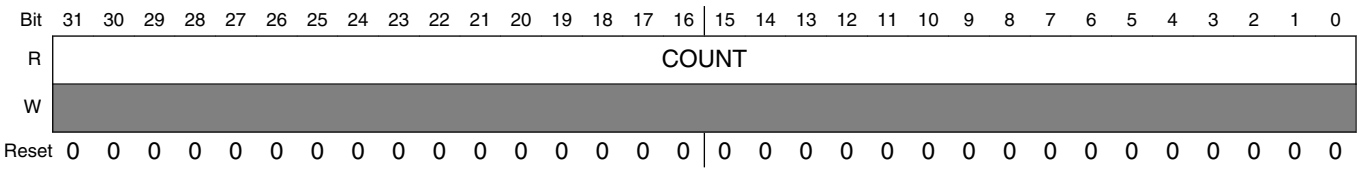
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_RMON_R_P_GTE2048 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of greater-than-2048-byte recieve packets

11.5.5.94 Rx Octets Statistic Register (ENET_RMON_R_OCTETS)

Address: 30BE_0000h base + 2C4h offset = 30BE_02C4h



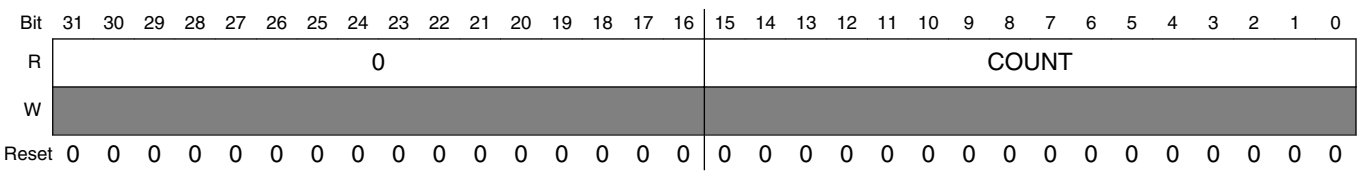
ENET_RMON_R_OCTETS field descriptions

Field	Description
COUNT	Number of receive octets

11.5.5.95 Frames not Counted Correctly Statistic Register (ENET_IEEE_R_DROP)

Counter increments if a frame with invalid or missing SFD character is detected and has been dropped. None of the other counters increments if this counter increments.

Address: 30BE_0000h base + 2C8h offset = 30BE_02C8h

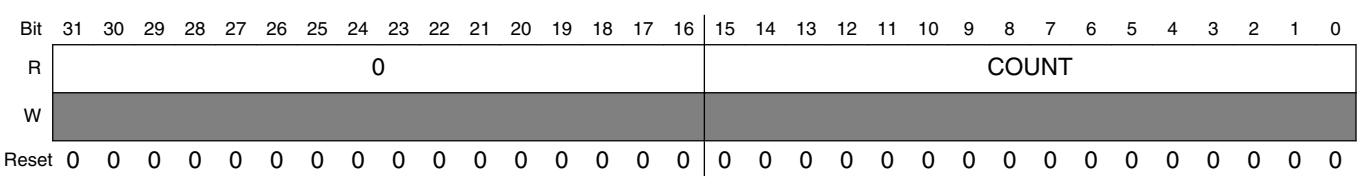


ENET_IEEE_R_DROP field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Frame count

11.5.5.96 Frames Received OK Statistic Register (ENET_IEEE_R_FRAME_OK)

Address: 30BE_0000h base + 2CCh offset = 30BE_02CCh

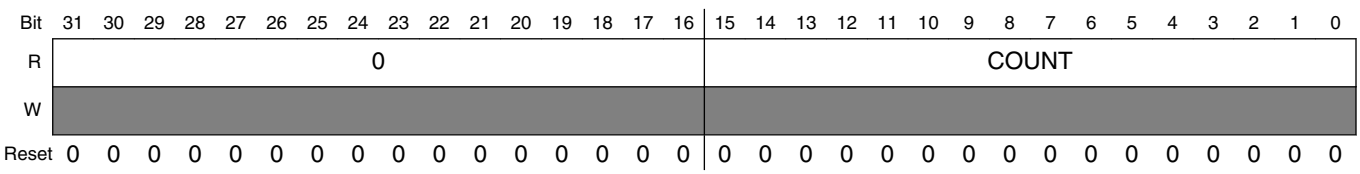


ENET_IEEE_R_FRAME_OK field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of frames received OK

11.5.5.97 Frames Received with CRC Error Statistic Register (ENET_IEEE_R_CRC)

Address: 30BE_0000h base + 2D0h offset = 30BE_02D0h

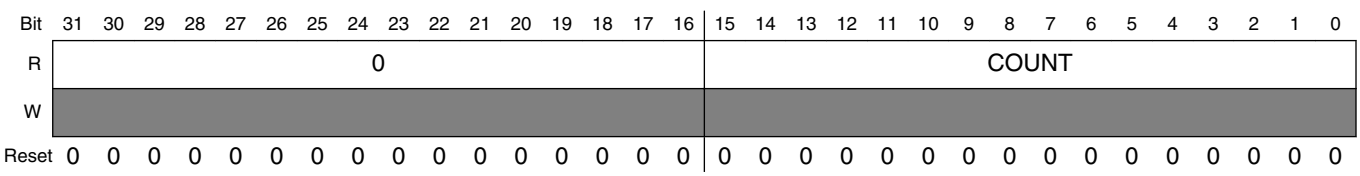


ENET_IEEE_R_CRC field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of frames received with CRC error

11.5.5.98 Frames Received with Alignment Error Statistic Register (ENET_IEEE_R_ALIGN)

Address: 30BE_0000h base + 2D4h offset = 30BE_02D4h

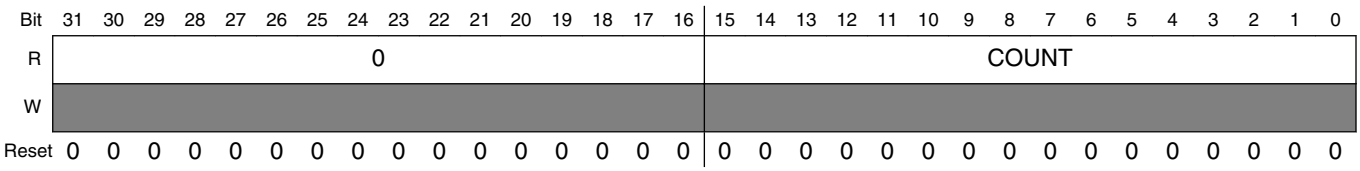


ENET_IEEE_R_ALIGN field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of frames received with alignment error

11.5.5.99 Receive FIFO Overflow Count Statistic Register (ENET_IEEE_R_MACERR)

Address: 30BE_0000h base + 2D8h offset = 30BE_02D8h

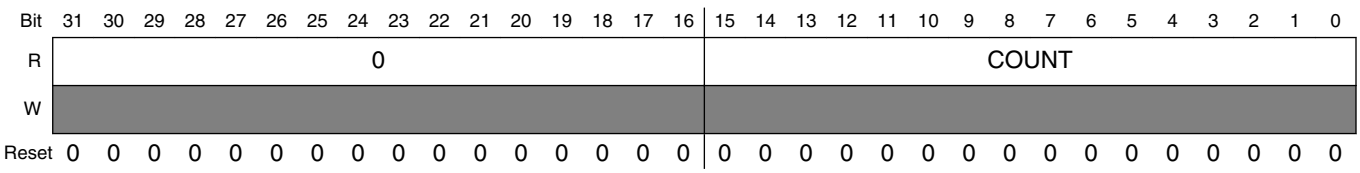


ENET_IEEE_R_MACERR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Receive FIFO overflow count

11.5.5.100 Flow Control Pause Frames Received Statistic Register (ENET_IEEE_R_FDXFC)

Address: 30BE_0000h base + 2DCh offset = 30BE_02DCh

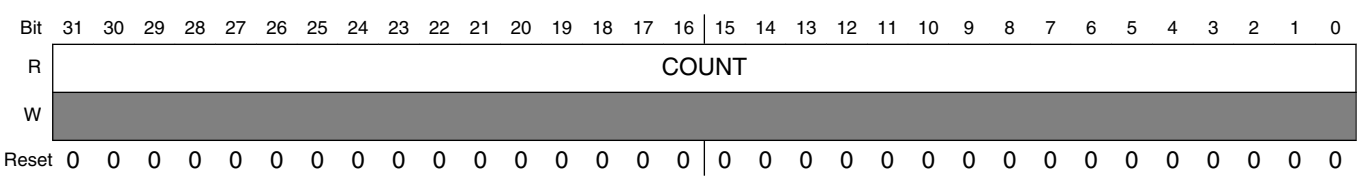


ENET_IEEE_R_FDXFC field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Number of flow-control pause frames received

11.5.5.101 Octet Count for Frames Received without Error Statistic Register (ENET_IEEE_R_OCTETS_OK)

Address: 30BE_0000h base + 2E0h offset = 30BE_02E0h



ENET_IEEE_R_OCTETS_OK field descriptions

Field	Description
COUNT	Number of octets for frames received without error **NOTE:** Counts total octets (includes header and FCS fields). Does not increment for the broadcast frames when broadcast reject is enabled and promiscuous mode is disabled within the receive control register (RCR).

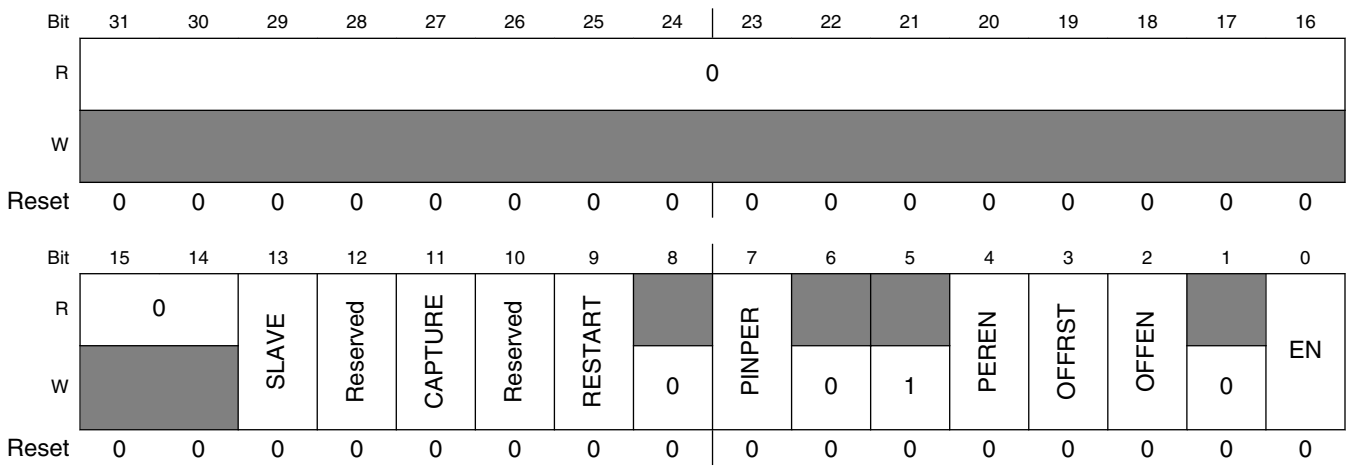
11.5.5.102 Adjustable Timer Control Register (ENET_ATCR)

ATCR command fields can trigger the corresponding events directly. It is not necessary to preserve any of the configuration fields when a command field is set in the register, that is, no read-modify-write is required.

NOTE

The CAPTURE and RESTART fields and bits 12 and 10 must be 0 in order to write to the other fields in this register.

Address: 30BE_0000h base + 400h offset = 30BE_0400h



ENET_ATCR field descriptions

Field	Description
31–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 SLAVE	Enable Timer Slave Mode 0 The timer is active and all configuration fields in this register are relevant. 1 The internal timer is disabled and the externally provided timer value is used. All other fields, except CAPTURE, in this register have no effect. CAPTURE can still be used to capture the current timer value.

Table continues on the next page...

ENET_ATCR field descriptions (continued)

Field	Description
12 Reserved	This field is reserved. Always write 0 to this field.
11 CAPTURE	Capture Timer Value When this field is set, all other fields are ignored during a write. This field automatically clears to 0 after the command completes. **NOTE:** To ensure that the correct time value is read from the ATVR register, a minimum amount of time must elapse from issuing this command to reading the ATVR register. This minimum time is defined by the greater of either six register clock cycles or six 1588/timestamp clock cycles. 0 No effect. 1 The current time is captured and can be read from the ATVR register.
10 Reserved	This field is reserved. Always write 0 to this field.
9 RESTART	Reset Timer Resets the timer to zero. This has no effect on the counter enable. If the counter is enabled when this field is set, the timer is reset to zero and starts counting from there. When set, all other fields are ignored during a write. This field automatically clears to 0 after the command completes. RESTART should be used when the timer is enabled. **NOTE:** The Reset Timer command requires at least 6 clock cycles of either the register clock or the 1588/timestamp clock, whichever is greater, to complete.
8 Reserved	This field is reserved.
7 PINPER	Enables event signal output assertion on period event. **NOTE:** Not all devices contain the event signal output. See the chip configuration details. 0 Disable. 1 Enable.
6 Reserved	This field is reserved.
5 Reserved	This field is reserved. **NOTE:** This field must be written always with one.
4 PEREN	Enable Periodical Event 0 Disable. 1 A period event interrupt can be generated (EIR[TS_TIMER]) and the event signal output is asserted when the timer wraps around according to the periodic setting ATPER. The timer period value must be set before setting this bit. **NOTE:** Not all devices contain the event signal output. See the chip configuration details.
3 OFFRST	Reset Timer On Offset Event 0 The timer is not affected and no action occurs, besides clearing OFFEN, when the offset is reached. 1 If OFFEN is set, the timer resets to zero when the offset setting is reached. The offset event does not cause a timer interrupt.
2 OFFEN	Enable One-Shot Offset Event

Table continues on the next page...

ENET_ATCR field descriptions (continued)

Field	Description
	0 Disable. 1 The timer can be reset to zero when the given offset time is reached (offset event). The field is cleared when the offset event is reached, so no further event occurs until the field is set again. The timer offset value must be set before setting this field.
1 Reserved	This field is reserved.
0 EN	Enable Timer 0 The timer stops at the current value. 1 The timer starts incrementing.

11.5.5.103 Timer Value Register (ENET_ATVR)

Address: 30BE_0000h base + 404h offset = 30BE_0404h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ATIME																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_ATVR field descriptions

Field	Description
ATIME	A write sets the timer. A read returns the last captured value. To read the current value, issue a capture command (i.e., set ATCR[CAPTURE]) prior to reading this register.

11.5.5.104 Timer Offset Register (ENET_ATOFF)

Address: 30BE_0000h base + 408h offset = 30BE_0408h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OFFSET																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_ATOFF field descriptions

Field	Description
OFFSET	Offset value for one-shot event generation. When the timer reaches the value, an event can be generated to reset the counter. If the increment value in ATINC is given in true nanoseconds, this value is also given in true nanoseconds.

11.5.5.105 Timer Period Register (ENET_ATPER)

Address: 30BE_0000h base + 40Ch offset = 30BE_040Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PERIOD																															
Reset	0	0	1	1	1	0	1	1	1	0	0	1	1	0	1	0	1	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0

ENET_ATPER field descriptions

Field	Description
PERIOD	Value for generating periodic events. Each instance the timer reaches this value, the period event occurs and the timer restarts. If the increment value in ATINC is given in true nanoseconds, this value is also given in true nanoseconds. The value should be initialized to 1,000,000,000 (1×10^9) to represent a timer wrap around of one second. The increment value set in ATINC should be set to the true nanoseconds of the period of clock ts_clk, hence implementing a true 1 second counter. **NOTE:** The value of PERIOD has the following constraint: 2^{32} – ENET_ATINC[INC_COR] – 3×ENET_ATINC[INC] ≥ PERIOD > 0.

11.5.5.106 Timer Correction Register (ENET_ATCOR)

Address: 30BE_0000h base + 410h offset = 30BE_0410h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	COR														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

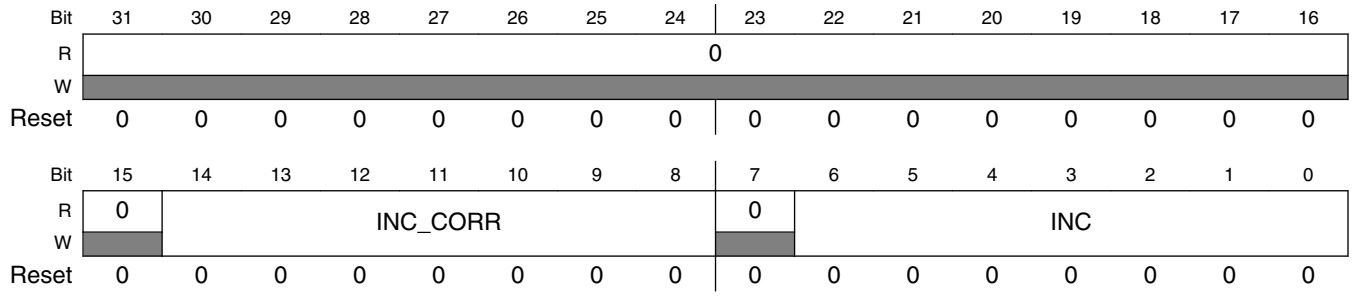
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_ATCOR field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COR	Correction Counter Wrap-Around Value Defines after how many timer clock cycles (ts_clk) the correction counter should be reset and trigger a correction increment on the timer. The amount of correction is defined in ATINC[INC_CORR]. A value of 0 disables the correction counter and no corrections occur. **NOTE:** This value is given in clock cycles, not in nanoseconds as all other values.

11.5.5.107 Time-Stamping Clock Period Register (ENET_ATINC)

Address: 30BE_0000h base + 414h offset = 30BE_0414h

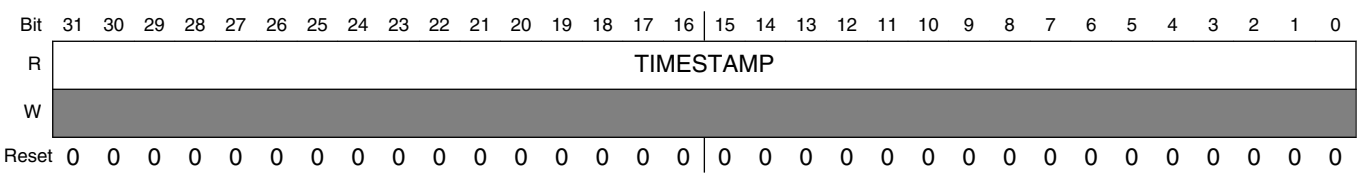


ENET_ATINC field descriptions

Field	Description
31–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–8 INC_CORR	Correction Increment Value This value is added every time the correction timer expires (every clock cycle given in ATCOR). A value less than INC slows down the timer. A value greater than INC speeds up the timer.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
INC	Clock Period Of The Timestamping Clock (ts_clk) In Nanoseconds The timer increments by this amount each clock cycle. For example, set to 10 for 100 MHz, 8 for 125 MHz, 5 for 200 MHz. **NOTE:** For highest precision, use a value that is an integer fraction of the period set in ATPER.

11.5.5.108 Timestamp of Last Transmitted Frame (ENET_ATSTMP)

Address: 30BE_0000h base + 418h offset = 30BE_0418h



ENET_ATSTMP field descriptions

Field	Description
TIMESTAMP	Timestamp of the last frame transmitted by the core that had TxBD[TS] set . This register is only valid when EIR[TS_AVAIL] is set.

11.5.5.109 Timer Global Status Register (ENET_TGSR)

Address: 30BE_0000h base + 604h offset = 30BE_0604h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

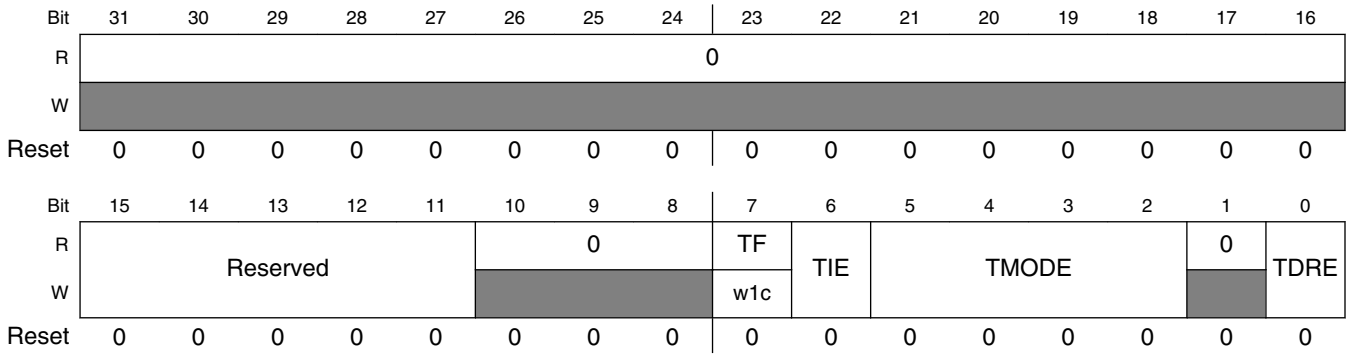
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												TF3	TF2	TF1	TF0
W													w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_TGSR field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 TF3	Copy Of Timer Flag For Channel 3 0 Timer Flag for Channel 3 is clear 1 Timer Flag for Channel 3 is set
2 TF2	Copy Of Timer Flag For Channel 2 0 Timer Flag for Channel 2 is clear 1 Timer Flag for Channel 2 is set
1 TF1	Copy Of Timer Flag For Channel 1 0 Timer Flag for Channel 1 is clear 1 Timer Flag for Channel 1 is set
0 TF0	Copy Of Timer Flag For Channel 0 0 Timer Flag for Channel 0 is clear 1 Timer Flag for Channel 0 is set

11.5.5.110 Timer Control Status Register (ENET_TCSR*n*)

Address: 30BE_0000h base + 608h offset + (8d × i), where i=0d to 3d



ENET_TCSR*n* field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–11 Reserved	This field is reserved. This field must be written with 0.
10–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 TF	Timer Flag Sets when input capture or output compare occurs. This flag is double buffered between the module clock and 1588 clock domains. When this field is 1, it can be cleared to 0 by writing 1 to it. 0 Input Capture or Output Compare has not occurred. 1 Input Capture or Output Compare has occurred.
6 TIE	Timer Interrupt Enable 0 Interrupt is disabled 1 Interrupt is enabled
5–2 TMODE	Timer Mode Updating the Timer Mode field takes a few cycles to register because it is synchronized to the 1588 clock. The version of Timer Mode returned on a read is from the 1588 clock domain. When changing Timer Mode, always disable the channel and read this register to verify the channel is disabled first. 0000 Timer Channel is disabled. 0001 Timer Channel is configured for Input Capture on rising edge. 0010 Timer Channel is configured for Input Capture on falling edge. 0011 Timer Channel is configured for Input Capture on both edges. 0100 Timer Channel is configured for Output Compare - software only. 0101 Timer Channel is configured for Output Compare - toggle output on compare. 0110 Timer Channel is configured for Output Compare - clear output on compare. 0111 Timer Channel is configured for Output Compare - set output on compare. 1000 Reserved

Table continues on the next page...

ENET_TCSR*n* field descriptions (continued)

Field	Description
	1010 Timer Channel is configured for Output Compare - clear output on compare, set output on overflow. 10X1 Timer Channel is configured for Output Compare - set output on compare, clear output on overflow. 110X Reserved 1110 Timer Channel is configured for Output Compare - pulse output low on compare for one 1588-clock cycle. 1111 Timer Channel is configured for Output Compare - pulse output high on compare for one 1588-clock cycle.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 TDRE	Timer DMA Request Enable 0 DMA request is disabled 1 DMA request is enabled

11.5.5.111 Timer Compare Capture Register (ENET_TCCR*n*)

Address: 30BE_0000h base + 60Ch offset + (8d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TCC																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENET_TCCR*n* field descriptions

Field	Description
TCC	Timer Capture Compare This register is double buffered between the module clock and 1588 clock domains. When configured for compare, the 1588 clock domain updates with the value in the module clock domain whenever the Timer Channel is first enabled and on each subsequent compare. Write to this register with the first compare value before enabling the Timer Channel. When the Timer Channel is enabled, write the second compare value either immediately, or at least before the first compare occurs. After each compare, write the next compare value before the previous compare occurs and before clearing the Timer Flag. The compare occurs one 1588 clock cycle after the IEEE 1588 Counter increments past the compare value in the 1588 clock domain. If the compare value is less than the value of the 1588 Counter when the Timer Channel is first enabled, then the compare does not occur until following the next overflow of the 1588 Counter. If the compare value is greater than the IEEE 1588 Counter when the 1588 Counter overflows, or the compare value is less than the value of the IEEE 1588 Counter after the overflow, then the compare occurs one 1588 clock cycle following the overflow. When configured for capture, the value of the IEEE 1588 Counter is captured into the 1588 clock domain and then updated into the module clock domain, provided the Timer Flag is clear. Always read the capture value before clearing the Timer Flag.

11.5.6 Functional description

This section provides a complete functional description of the MAC-NET core.

11.5.6.1 Ethernet MAC frame formats

The IEEE 802.3 standard defines the Ethernet frame format as follows:

- Minimum length of 64 bytes
- Maximum length of 1518 bytes excluding the preamble and the start frame delimiter (SFD) bytes

An Ethernet frame consists of the following fields:

- Seven bytes preamble
- Start frame delimiter (SFD)
- Two address fields
- Length or type field
- Data field
- Frame check sequence (CRC value)
- Extension field is defined only for Gigabit Ethernet half-duplex implementations and is not supported by the MAC core

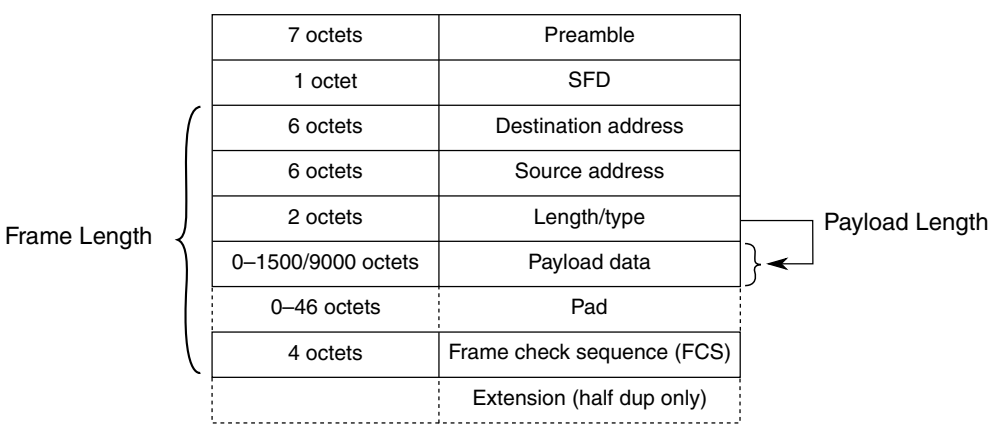


Figure 11-69. MAC frame format overview

Optionally, MAC frames can be VLAN-tagged with an additional four-byte field inserted between the MAC source address and the type/length field. VLAN tagging is defined by the IEEE P802.1q specification. VLAN-tagged frames have a maximum length of 1522 bytes, excluding the preamble and the SFD bytes.

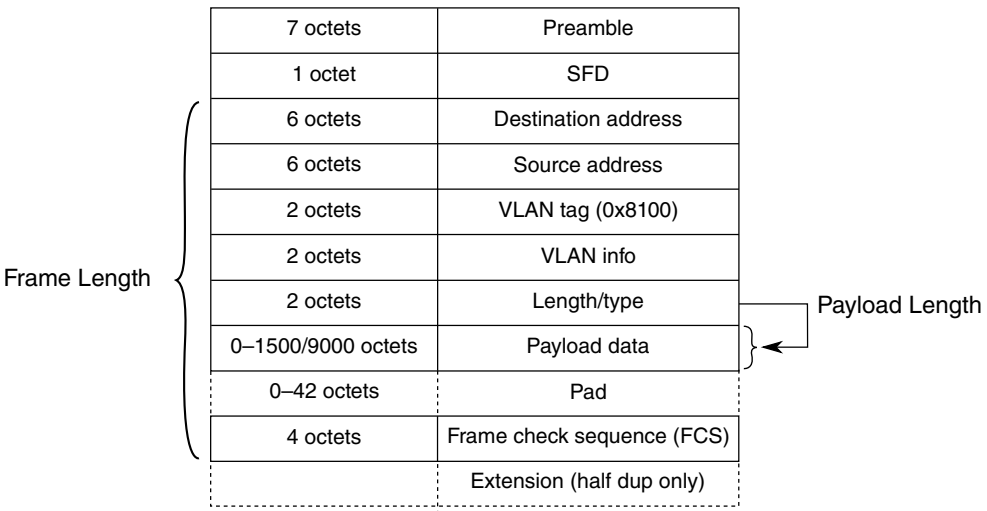


Figure 11-70. VLAN-tagged MAC frame format overview

Table 11-103. MAC frame definition

Term	Description
Frame length	Defines the length, in octets, of the complete frame without preamble and SFD. A frame has a valid length if it contains at least 64 octets and does not exceed the programmed maximum length.
Payload length	The length/type field indicates the length of the frame's payload section. The most significant byte is sent/received first. • If the length/type field is set to a value less than 46, the payload is padded so that the minimum frame length requirement (64 bytes) is met. For VLAN-tagged frames, a value less than 42 indicates a padded frame. • If the length/type field is set to a value larger than the programmed frame maximum length (e.g. 1518) it is interpreted as a type field.
Destination and source address	48-bit MAC addresses. The least significant byte is sent/received first and the first two least significant bits of the MAC address distinguish MAC frames, as detailed in MAC address check.

Note

Although the IEEE specification defines a maximum frame length, the MAC core provides the flexibility to program any value for the frame maximum length.

11.5.6.1.1 Pause Frames

The receiving device generates a pause frame to indicate a congestion to the emitting device, which should stop sending data.

Pause frames are indicated by the length/type set to 0x8808. The two first bytes of a pause frame following the type, defines a 16-bit opcode field set to 0x0001 always. A 16-bit pause quanta is defined in the frame payload bytes 2 (P1) and 3 (P2) as defined in the following table. The P1 pause quanta byte is the most significant.

Table 11-104. Pause Frame Format (Values in Hex)

1	2	3	4	5	6	7	8	9	10	11	12	13	14
55	55	55	55	55	55	55	D5	01	80	C2	00	00	01
Preamble							SFD	Multicast Destination Address					
15	16	17	18	19	20	21	22	23	24	25	26	27 –68	
00	00	00	00	00	00	88	08	00	01	hi	lo	00	
Source Address						Type		Opcode		P1	P2	pad (42)	
69	70	71	72										
26	6B	AE	0A										
CRC-32													

There is no payload length field found within a pause frame and a pause frame is always padded with 42 bytes (0x00).

If a pause frame with a pause value greater than zero (XOFF condition) is received, the MAC stops transmitting data as soon the current frame transfer is completed. The MAC stops transmitting data for the value defined in pause quanta. One pause quanta fraction refers to 512 bit times.

If a pause frame with a pause value of zero (XON condition) is received, the transmitter is allowed to send data immediately (see Full-duplex flow control operation for details).

11.5.6.1.2 Magic packets

A magic packet is a unicast, multicast, or broadcast packet, which carries a defined sequence in the payload section.

Magic packets are received and inspected only under specific conditions as described in Magic packet detection.

The defined sequence to decode a magic packet is formed with a synchronization stream which consists of six consecutive 0xFF bytes, and is followed by sequence of sixteen consecutive unicast MAC addresses of the node to be awakened.

This sequence can be located anywhere in the magic packet payload. The magic packet is formed with a standard Ethernet header, optional padding, and CRC.

11.5.6.2 IP and higher layers frame format

The following sections use the term datagram to describe the protocol specific data unit that is found within the payload section of its container entity.

For example, an IP datagram specifies the payload section of an Ethernet frame. A TCP datagram specifies the payload section within an IP datagram.

11.5.6.2.1 Ethernet types

IP datagrams are carried in the payload section of an Ethernet frame. The Ethernet frame type/length field discriminates several datagram types.

The following table lists the types of interest:

Table 11-105. Ethernet type value examples

Type	Description
0x8100	VLAN-tagged frame. The actual type is found 4 octets later in the frame.
0x0800	IPv4
0x0806	ARP
0x86DD	IPv6

11.5.6.2.2 IPv4 datagram format

The following figure shows the IP Version 4 (IPv4) header, which is located at the beginning of an IP datagram. It is organized in 32-bit words. The first byte sent/received is the leftmost byte of the first word (in other words, version/IHL field).

The IP header can contain further options, which are always padded if necessary to guarantee the payload following the header is aligned to a 32-bit boundary.

The IP header is immediately followed by the payload, which can contain further protocol headers (for example, TCP or UDP, as indicated by the protocol field value). The complete IP datagram is transported in the payload section of an Ethernet frame.

Table 11-106. IPv4 header format

31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
Version	IHL	TOS		Length			
Fragment ID				Flags	Fragment offset		
TTL		Protocol		Header checksum			
Source address							
Destination address							
Options							

Table 11-107. IPv4 header fields

Field name	Description
Version	4-bit IP version information. 0x4 for IPv4 frames.
IHL	4-bit Internet header length information. Determines number of 32-bit words found within the IP header. If no options are present, the default value is 0x5.
TOS	Type of service/DiffServ field.
Length	Total length of the datagram in bytes, including all octets of header and payload.
Fragment ID, flags, fragment offset	Fields used for IP fragmentation.
TTL	Time-to-live. In effect, is decremented at each router arrival. If zero, datagram must be discarded.
Protocol	Identifier of protocol that follows in the datagram.
Header checksum	Checksum of IP header. For computational purposes, this field's value is zero.
Source address	Source IP address.
Destination address	Destination IP address.

11.5.6.2.3 IPv6 datagram format

The following figure shows the IP version 6 (IPv6) header, which is located at the beginning of an IP datagram. It is organized in 32-bit words and has a fixed length of ten words (40 bytes). The next header field identifies the type of the header that follows the IPv6 header. It is defined similar to the protocol identifier within IPv4, with new definitions for identifying extension headers. These headers can be inserted between the IPv6 header and the protocol header, which will shift the protocol header accordingly.The accelerator currently only supports IPv6 without extension headers (in other words, the next header specifies TCP, UDP, or IMCP).

The first byte sent/received is the leftmost byte of the first word (in other words, version/ traffic class fields).

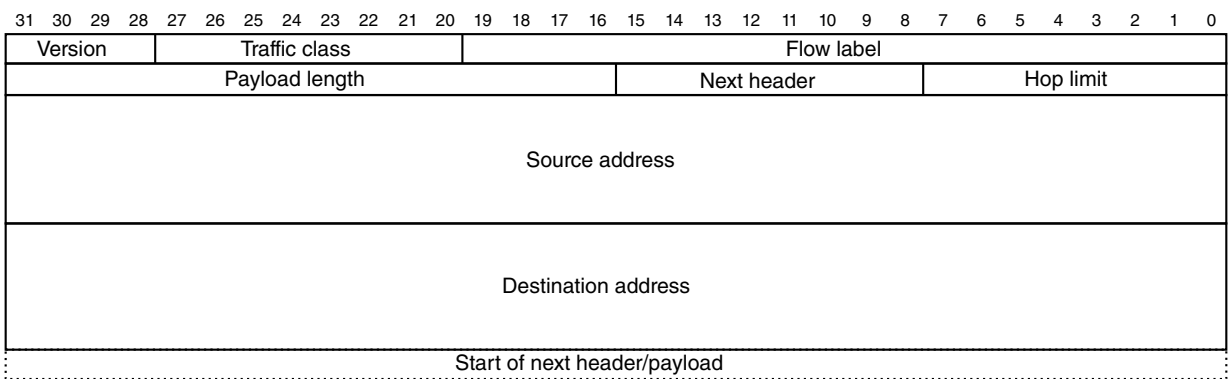


Figure 11-71. IPv6 header format

Table 11-108. IPv6 header fields

Field name	Description
Version	4-bit IP version information. 0x6 for all IPv6 frames.
Traffic class	8-bit field defining the traffic class.
Flow label	20-bit flow label identifying frames of the same flow.
Payload length	16-bit length of the datagram payload in bytes. It includes all octets following the IPv6 header.
Next header	Identifies the header that follows the IPv6 header. This can be the protocol header or any IPv6 defined extension header.
Hop limit	Hop counter, decremented by one by each station that forwards the frame. If hop limit is 0 the frame must be discarded.
Source address	128-bit IPv6 source address.
Destination address	128-bit IPv6 destination address.

11.5.6.2.4 Internet Control Message Protocol (ICMP) datagram format

An internet control message protocol (ICMP) is found following the IP header, if the protocol identifier is 1. The ICMP datagram has a four-octet header followed by additional message data.

Table 11-109. ICMP header format

31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
Type		Code		Checksum			
ICMP message data							

Table 11-110. IP header fields

Field name	Description
Type	8-bit type information
Code	8-bit code that is related to the message type
Checksum	16-bit one's complement checksum over the complete ICMP datagram

11.5.6.2.5 User Datagram Protocol (UDP) datagram format

A user datagram protocol header is found after the IP header, when the protocol identifier is 17.

The payload of the datagram is after the UDP header. The header byte order follows the conventions given for the IP header above.

Table 11-111. UDP header format

31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
Source port				Destination port			
Length				Checksum			

Table 11-112. UDP header fields

Field name	Description
Source port	Source application port
Destination port	Destination application port
Length	Length of user data which immediately follows the header, including the UDP header (that is, minimum value is 8)
Checksum	Checksum over the complete datagram and some IP header information

11.5.6.2.6 TCP datagram format

A TCP header is found following the IP header, when the protocol identifier has a value of 6.

The TCP payload immediately follows the TCP header.

Table 11-113. TCP header format

31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
Source port				Destination port			
Sequence number							
Acknowledgement number							
Offset	Reserved		Flags	Window			
Checksum				Urgent pointer			
Options							

Table 11-114. TCP header fields

Field name	Description
Source port	Source application port
Destination port	Destination application port
Sequence number	Transmit sequence number
Ack. number	Receive sequence number
Offset	Data offset, which is number of 32-bit words within TCP header — if no options selected, defaults to value of 5
Flags	URG, ACK, PSH, RST, SYN, FIN flags

Table continues on the next page...

Table 11-114. TCP header fields (continued)

Field name	Description
Window	TCP receive window size information
Checksum	Checksum over the complete datagram (TCP header and data) and IP header information
Options	Additional 32-bit words for protocol options

11.5.6.3 IEEE 1588 message formats

The following sections describe the IEEE 1588 message formats.

11.5.6.3.1 Transport encapsulation

The precision time protocol (PTP) datagrams are encapsulated in Ethernet frames using the UDP/IP transport mechanism, or optionally, with the newer 1588v2 directly in Ethernet frames (layer 2).

Typically, multicast addresses are used to allow efficient distribution of the synchronization messages.

11.5.6.3.1.1 UDP/IP

The 1588 messages (v1 and v2) can be transported using UDP/IP multicast messages.

Table 11-115 shows IP multicast groups defined for PTP. The table also shows their respective MAC layer multicast address mapping according to RFC 1112 (last three octets of IP follow the fixed value of 01-00-5E).

Table 11-115. UDP/IP multicast domains

Name	IP Address	MAC Address mapping
DefaultPTPdomain	224.0.1.129	01-00-5E-00-01-81
AlternatePTPdomain1	224.0.1.130	01-00-5E-00-01-82
AlternatePTPdomain2	224.0.1.131	01-00-5E-00-01-83
AlternatePTPdomain3	224.0.1.132	01-00-5E-00-01-84

Table 11-116. UDP port numbers

Message type	UDP port	Note
Event	319	Used for SYNC and DELAY_REQUEST messages
General	320	All other messages (for example, follow-up, delay-response)

11.5.6.3.1.2 Native Ethernet (PTPv2)

In addition to using UDP/IP frames, IEEE 1588v2 defines a native Ethernet frame format that uses ethertype = 0x88F7. The payload of the Ethernet frame immediately contains the PTP datagram, starting with the PTPv2 header.

Besides others, version 2 adds a peer delay mechanism to allow delay measurements between individual point-to-point links along a path over multiple nodes. The following multicast domains are also defined in PTPv2.

Table 11-117. PTPv2 multicast domains

Name	MAC address
Normal messages	01-1B-19-00-00-00
Peer delay messages	01-80-C2-00-00-0E

11.5.6.3.2 PTP header

All PTP frames contain a common header that determines the protocol version and the type of message, which defines the remaining content of the message.

All multi-octet fields are transmitted in big-endian order (the most significant byte is transmitted/received first).

The last four bits of versionPTP are at the same position (second byte) for PTPv1 and PTPv2 headers. This allows accurate identification by inspecting the first two bytes of the message.

11.5.6.3.2.1 PTPv1 header

Table 11-118. Common PTPv1 message header

Offset	Octets	Bits							
		7	6	5	4	3	2	1	0
0	2	versionPTP = 0x0001							
2	2	versionNetwork							
4	16	subdomain							
20	1	messageType							
21	1	sourceCommunicationTechnology							
22	6	sourceUuid							
28	2	sourcePortId							
30	2	sequenceId							

Table continues on the next page...

Table 11-118. Common PTPv1 message header (continued)

Offset	Octets	Bits							
		7	6	5	4	3	2	1	0
32	1	control							
33	1	0x00							
34	2	flags							
36	4	reserved							

The type of message is encoded in the messageType and control fields as shown in Table 11-119 :

Table 11-119. PTPv1 message type identification

messageType	control	Message Name	Message
0x01	0x0	SYNC	Event message
0x01	0x1	DELAY_REQ	Event message
0x02	0x2	FOLLOW_UP	General message
0x02	0x3	DELAY_RESP	General message
0x02	0x4	MANAGEMENT	General message
other	other	—	Reserved

The field sequenceId is used to non-ambiguously identify a message.

11.5.6.3.2.2 PTPv2 header

Table 11-120. Common PTPv2 message header

Offset	Octets	Bits							
		7	6	5	4	3	2	1	0
0	1	transportSpecific				messageId			
1	1	reserved				versionPTP = 0x2			
2	2	messageLength							
4	1	domainNumber							
5	1	reserved							
6	2	flags							
8	8	correctionField							
16	4	reserved							
20	10	sourcePortIdentity							
30	2	sequenceId							
32	1	control							
33	1	logMeanMessageInterval							

The type of message is encoded in the field messageId as follows:

Table 11-121. PTPv2 message type identification

messageId	Message name	Message
0x0	SYNC	Event message
0x1	DELAY_REQ	Event message
0x2	PATH_DELAY_REQ	Event message
0x3	PATH_DELAY_RESP	Event message
0x4–0x7	—	Reserved
0x8	FOLLOW_UP	General message
0x9	DELAY_RESP	General message
0xa	PATH_DELAY_FOLLOW_UP	General message
0xb	ANNOUNCE	General message
0xc	SIGNALING	General message
0xd	MANAGEMENT	General message

The PTPv2 flags field contains further details on the type of message, especially if one-step or two-step implementations are used. The one- or two-step implementation is controlled by the TWO_STEP bit in the first octet of the flags field as shown below. Reserved bits are cleared.

Table 11-122. PTPv2 message flags field definitions

Bit	Name	Description
0	ALTERNATE_MASTER	See IEEE 1588 Clause 17.4
1	TWO_STEP	1 Two-step clock 0 One-step clock
2	UNICAST	1 Transport layer address uses a unicast destination address 0 Multicast is used
3	—	Reserved
4	—	Reserved
5	Profile specific	
6	Profile specific	
7	—	Reserved

11.5.6.4 MAC receive

The MAC receive engine performs the following tasks:

- Check frame framing

- Remove frame preamble and frame SFD field
- Discard frame based on frame destination address field
- Terminate pause frames
- Check frame length
- Remove payload padding if it exists
- Calculate and verify CRC-32
- Write received frames in the core receive FIFO

If the MAC is programmed to operate in half-duplex mode, it will also check if the frame is received with a collision.

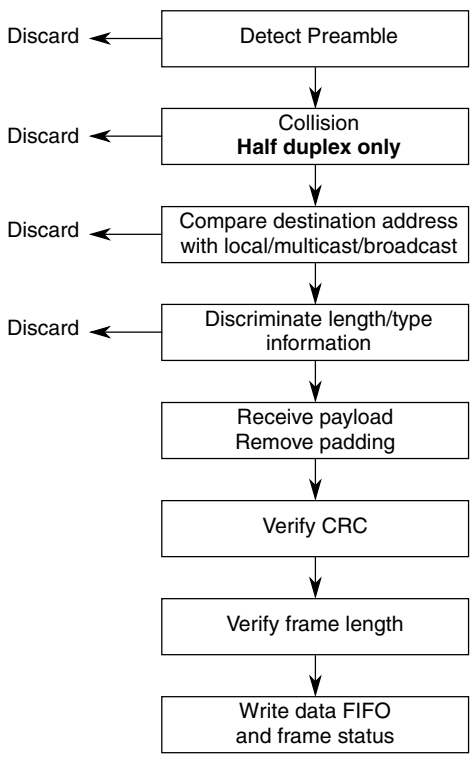


Figure 11-72. MAC receive flow

11.5.6.4.1 Collision detection in half-duplex mode

If the packet is received with a collision detected during reception of the first 64 bytes, the packet is discarded (if frame size was less than ~14 octets) or transmitted to the user application with an error and RxBD[CE] set.

11.5.6.4.2 Preamble processing

The IEEE 802.3 standard allows a maximum size of 56 bits (seven bytes) for the preamble, while the MAC core allows any preamble length, including zero length preamble.

The MAC core checks for the start frame delimiter (SFD) byte. If the next byte of the preamble, which is different from 0x55, is not 0xD5, the frame is discarded.

Although the IEEE specification dictates that the inner-packet gap should be at least 96 bits, the MAC core is designed to accept frames separated by only 64 10/100-Mbit/s operation (MII) bits.

The MAC core removes the preamble and SFD bytes.

11.5.6.4.3 MAC address check

The destination address bit 0 differentiates between multicast and unicast addresses.

- If bit 0 is 0, the MAC address is an individual (unicast) address.
- If bit 0 is 1, the MAC address defines a group (multicast) address.
- If all 48 bits of the MAC address are set, it indicates a broadcast address.

11.5.6.4.3.1 Unicast address check

If a unicast address is received, the destination MAC address is compared to the node MAC address programmed by the host in the PADDR1/2 registers.

If the destination address matches any of the programmed MAC addresses, the frame is accepted.

If Promiscuous mode is enabled (RCR[PROM] = 1) no address checking is performed and all unicast frames are accepted.

11.5.6.4.3.2 Multicast and unicast address resolution

The hash table algorithm used in the group and individual hash filtering operates as follows.

- The 48-bit destination address is mapped into one of 64 bits, represented by 64 bits in ENET*n*_GAUR/GALR (group address hash match) or ENET*n*_IAUR/IALR (individual address hash match).

- This mapping is performed by passing the 48-bit address through the on-chip 32-bit CRC generator and selecting the six most significant bits of the CRC-encoded result to generate a number between 0 and 63.
- The msb of the CRC result selects ENET*n*_GAUR (msb = 1) or ENET*n*_GALR (msb = 0).
- The five lsbs of the hash result select the bit within the selected register.
- If the CRC generator selects a bit set in the hash table, the frame is accepted; else, it is rejected.

For example, if eight group addresses are stored in the hash table and random group addresses are received, the hash table prevents roughly 56/64 (or 87.5%) of the group address frames from reaching memory. Those that do reach memory must be further filtered by the processor to determine if they truly contain one of the eight desired addresses.

The effectiveness of the hash table declines as the number of addresses increases.

The user must initialize the hash table registers. Use this CRC32 polynomial to compute the hash:

- $FCS(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x^{1} + 1$

If Promiscuous mode is enabled (ENET*n*_RCR[PROM] = 1) all unicast and multicast frames are accepted regardless of ENET*n*_GAUR/GALR and ENET*n*_IAUR/IALR settings.

11.5.6.4.3.3 Broadcast address reject

All broadcast frames are accepted if BC_REJ is cleared or ENET*n*_RCR[PROM] is set. If PROM is cleared when ENET*n*_RCR[BC_REJ] is set, all broadcast frames are rejected.

Table 11-123. Broadcast address reject programming

PROM	BC_REJ	Broadcast frames
0	0	Accepted
0	1	Rejected
1	0	Accepted
1	1	Accepted

11.5.6.4.3.4 Miss-bit implementation

For higher layer filtering purposes, RxBD[M] indicates an address miss when the MAC operates in promiscuous mode and accepts a frame that would otherwise be rejected.

If a group/individual hash or exact match does not occur and Promiscuous mode is enabled (RCR[PROM] = 1), the frame is accepted and the M bit is set in the buffer descriptor; otherwise, the frame is rejected.

This means the status bit is set in any of the following conditions during Promiscuous mode:

- A broadcast frame is received when BC_REJ is set
- A unicast is received that does not match either:
  - Node address (PALR[PADDR1] and PAUR[PADDR2])
  - Hash table for unicast (IAUR[IADDR1] and IALR[IADDR2])
- A multicast is received that does not match the GAUR[GADDR1] and GALR[GADDR2] hash table entries

11.5.6.4.4 Frame length/type verification: payload length check

If the length/type is less than 0x600 and NLC is set, the MAC checks the payload length and reports any error in the frame status word and interrupt bit PLR.

If the length/type is greater than or equal to 0x600, the MAC interprets the field as a type and no payload length check is performed.

The length check is performed on VLAN and stacked VLAN frames. If a padded frame is received, no length check can be performed due to the extended frame payload because padded frames can never have a payload length error.

11.5.6.4.5 Frame length/type verification: frame length check

When the receive frame length exceeds MAX_FL bytes, the BABR interrupt is generated and the RxBD[LG] bit is set.

The frame is not truncated unless the frame length exceeds the value programmed in ENET*n*_FTRL[TRUNC_FL]. If the frame is truncated, RxBD[TR] is set. In addition, a truncated frame always has the CRC error indication set (RxBD[CR]).

11.5.6.4.6 VLAN frames processing

VLAN frames have a length/type field set to 0x8100 immediately followed by a 16-Bit VLAN control information field.

VLAN-tagged frames are received as normal frames because the VLAN tag is not interpreted by the MAC function, and are pushed complete with the VLAN tag to the user application. If the length/type field of the VLAN-tagged frame, which is found four octets later in the frame, is less than 42, the padding is removed. In addition, the frame status word (RxBD[NO]) indicates that the current frame is VLAN tagged.

11.5.6.4.7 Pause frame termination

The receive engine terminates pause frames and does not transfer them to the receive FIFO. The quanta is extracted and sent to the MAC transmit path via a small internal clock rate decoupling asynchronous FIFO.

The quanta is written only if a correct CRC and frame length are detected by the control state machine. If not, the quanta is discarded and the MAC transmit path is not paused.

Good pause frames are ignored if ENET*n*_RCR[FCE] is cleared and are forwarded to the client interface when ENET*n*_RCR[PAUFWD] is set.

11.5.6.4.8 CRC check

The CRC-32 field is checked and forwarded to the core FIFO interface if ENET*n*_RCR[CRCFWD] is cleared and ENET*n*_RCR[PADEN] is cleared. When CRCFWD is set (regardless of PADEN), the CRC-32 field is checked and terminated (not transmitted to the FIFO).

The CRC polynomial, as specified in the 802.3 standard, is:

- $FCS(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x^{1} + 1$

The 32 bits of the CRC value are placed in the frame check sequence (FCS) field with the x^{31} term as right-most bit of the first octet. The CRC bits are thus received in the following order: x^{31}, x^{30},..., x^{1}, x^{0}.

If a CRC error is detected, the frame is marked invalid and RxBD[CR] is set.

11.5.6.4.9 Frame padding removal

When a frame is received with a payload length field set to less than 46 (42 for VLAN-tagged frames and 38 for frames with stacked VLANs), the zero padding can be removed before the frame is written into the data FIFO depending on the setting of ENET*n*_RCR[PADEN].

Note

If a frame is received with excess padding (in other words, the length field is set as mentioned above, but the frame has more than 64 octets) and padding removal is enabled, then the padding is removed as normal and no error is reported if the frame is otherwise correct (for example: good CRC, less than maximum length, and no other error).

11.5.6.4.10 Frame classification (AVB)

To support protocols such as Audio Video Bridging (AVB, IEEE 802.1Qav), normal traffic is separated from time-sensitive traffic immediately at the application interface to allow storing them in different queues for further processing. For every frame received, a classification based on VLAN priority can be performed. When a frame is received, and it contains a VLAN tag, the priority is compared against the values set in the classification match registers (see Receive Classification Match Register for Class n (ENET_RCMR*n*)) and the following figure.

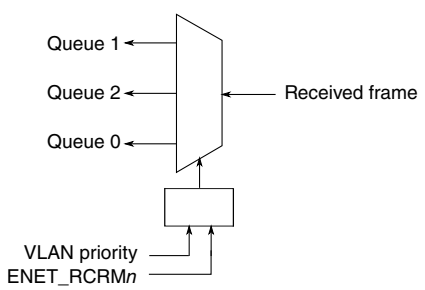


Figure 11-73. AVB frame classification

11.5.6.4.11 Receive flushing

RX flushing prevents frames in the RX FIFO from being blocked. Blocking can occur if the frame at the head of the RX FIFO cannot be forwarded because the ring it is associated with cannot accept it. This situation occurs when either the ring's RxBD[EMPTY] is not set or ENET_RDAR*n* is not set. When RX flushing is enabled, via QOS Scheme (ENET_QOS), the blocking frame will be flushed (discarded).

11.5.6.5 MAC transmit

Frame transmission starts when the transmit FIFO holds enough data.

After a transfer starts, the MAC transmit function performs the following tasks:

- Generates preamble and SFD field before frame transmission
- Generates XOFF pause frames if the receive FIFO reports a congestion or if ENET*n*_TCR[TFC_PAUSE] is set with ENET*n*_OPD[PAUSE_DUR] set to a non-zero value
- Generates XON pause frames if the receive FIFO congestion condition is cleared or if TFC_PAUSE is set with PAUSE_DUR cleared
- Suspends Ethernet frame transfer (XOFF) if a non-zero pause quanta is received from the MAC receive path
- Adds padding to the frame if required
- Calculates and appends CRC-32 to the transmitted frame
- Sends the frame with correct inter-packet gap (IPG) (deferring)

When the MAC is configured to operate in half-duplex mode, the following additional tasks are performed:

- Collision detection
- Frame retransmit after back-off timer expires

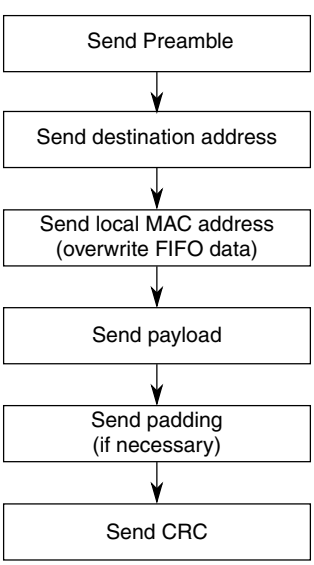


Figure 11-74. Frame transmit overview

11.5.6.5.1 Frame payload padding

The IEEE specification defines a minimum frame length of 64 bytes.

If the frame sent to the MAC from the user application has a size smaller than 60 bytes, the MAC automatically adds padding bytes (0x00) to comply with the Ethernet minimum frame length specification. Transmit padding is always performed and cannot be disabled.

If the MAC is not allowed to append a CRC (TxBD[TC] = 1), the user application is responsible for providing frames with a minimum length of 64 octets.

11.5.6.5.2 MAC address insertion

On each frame received from the core transmit FIFO interface, the source MAC address is either:

- Replaced by the address programmed in the PADDR1/2 fields (ENET*n*_TCR[ADDINS] = 1)
- Transparently forwarded to the Ethernet line (ENET*n*_TCR[ADDINS] = 0)

11.5.6.5.3 CRC-32 generation

The CRC-32 field is optionally generated and appended at the end of a frame.

The CRC polynomial, as specified in the 802.3 standard, is:

- $FCS(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1$

The 32 bits of the CRC value are placed in the FCS field so that the x^{31} term is the right-most bit of the first octet. The CRC bits are thus transmitted in the following order: x^{31}, x^{30},..., x^1, x^0.

11.5.6.5.4 Inter-packet gap (IPG)

In full-duplex mode, after frame transmission and before transmission of a new frame, an IPG (programmed in ENET*n*_TIPG) is maintained. The minimum IPG can be programmed between 8 and 26 byte-times (64 and 208 bit-times).

In half-duplex mode, the core constantly monitors the line. Actual transmission of the data onto the network occurs only if it has been idle for a 96-bit time period, and any back-off time requirements have been satisfied. In accordance with the standard, the core begins to measure the IPG from CRS/GMII_CRS de-assertion.

11.5.6.5.5 Collision detection and handling — half-duplex operation only

A collision occurs on a half-duplex network when concurrent transmissions from two or more nodes take place. During transmission, the core monitors the line condition and detects a collision when the PHY device asserts COL.

When the core detects a collision while transmitting, it stops transmission of the data and transmits a 32-bit jam pattern. If the collision is detected during the preamble or the SFD transmission, the jam pattern is transmitted after completing the SFD, which results in a minimum 96-bit fragment. The jam pattern is a fixed pattern that is not compared to the actual frame CRC, and has a very low probability (0.532) of having a jam pattern identical to the CRC.

If a collision occurs before transmission of 64 bytes (including preamble and SFD), the MAC core waits for the backoff period and retransmits the packet data (stored in a 64-byte re-transmit buffer) that has already been sent on the line. The backoff period is generated from a pseudo-random process (truncated binary exponential backoff).

If a collision occurs after transmission of 64 bytes (including preamble and SFD), the MAC discards the remainder of the frame, optionally sets the LC interrupt bit, and sets TxBD[LCE].

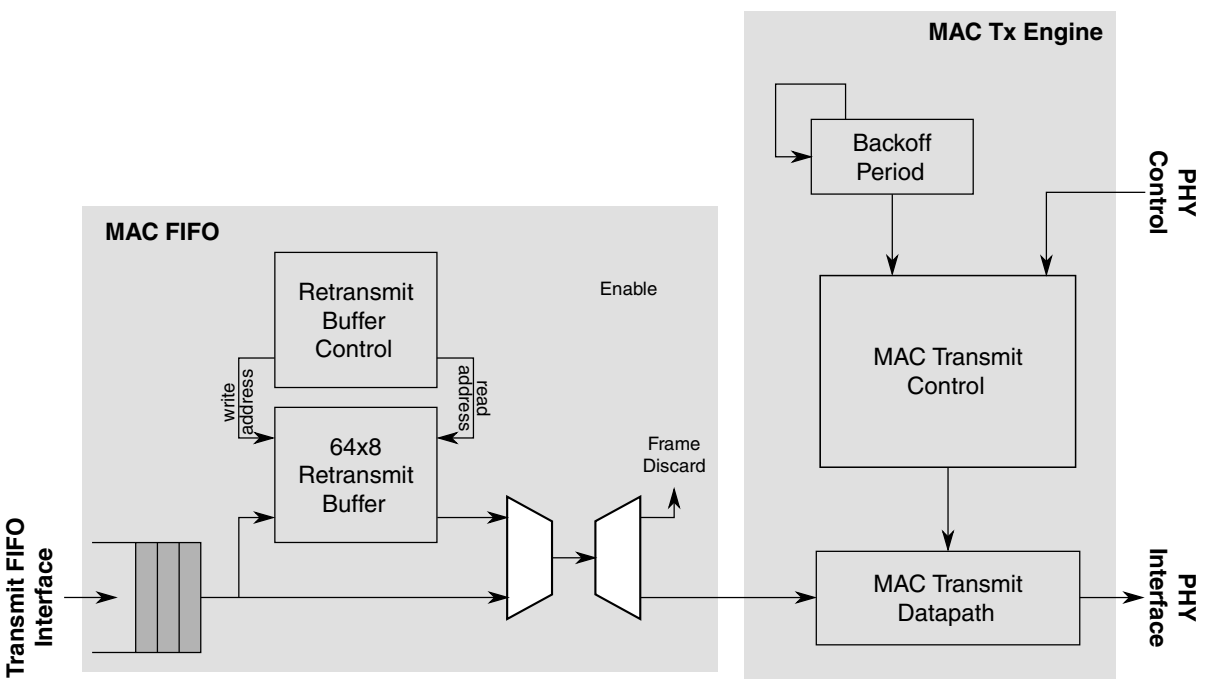


Figure 11-75. Packet re-transmit overview

The backoff time is represented by an integer multiple of slot times. One slot is equal to a 512-bit time period. The number of the delay slot times, before the n^{th} re-transmission attempt, is chosen as a uniformly-distributed random integer in the range:

- $0 < r < 2^k$
- k = min(*n*, N); where *n* is the number of retransmissions and N = 10

For example, after the first collision, the backoff period is 0 or 1 slot time. If a collision occurs on the first retransmission, the backoff period is 0, 1, 2, or 3, and so on.

The maximum backoff time (in 512-bit time slots) is limited by N = 10 as specified in the IEEE 802.3 standard.

If a collision occurs after 16 consecutive retransmissions, the core reports an excessive collision condition (ENET*n*_EIR[RL] interrupt field and TxBD[EE]) and discards the current packet from the FIFO.

In networks violating the standard requirements, a collision may occur after transmission of the first 64 bytes. In this case, the core stops the current packet transmission and discards the rest of the packet from the transmit FIFO. The core resumes transmission with the next packet available in the core transmit FIFO.

warning

Ethernet PHYs that support the SQE Test, or "heartbeat," feature must disable this feature. When this feature is enabled, the PHY asserts the collision signal after a frame is transmitted to indicate to the ENET that the PHY's collision logic is working. This may cause data corruption in the next frame from the ENET. This corrupted frame contains up to 21 zero bytes which start somewhere within the MAC destination address field. The ENET, however, will still generate a good FCS (CRC-32) but with corrupted data.

11.5.6.5.6 Rate limiting / traffic shaping support

The MAC-NET supports two methods to optimize frame traffic for either time-sensitive AVB frames (Class A and Class B) or best-effort non-AVB frames:

- Round-robin scheme
- Credit-based traffic shaping

To ensure that sufficient bandwidth is allocated for the AVB frames, the credit-based shaper must be used. Either method can be combined with a time-based shaper to ensure that a frame is always transmitted in its correct time slot.

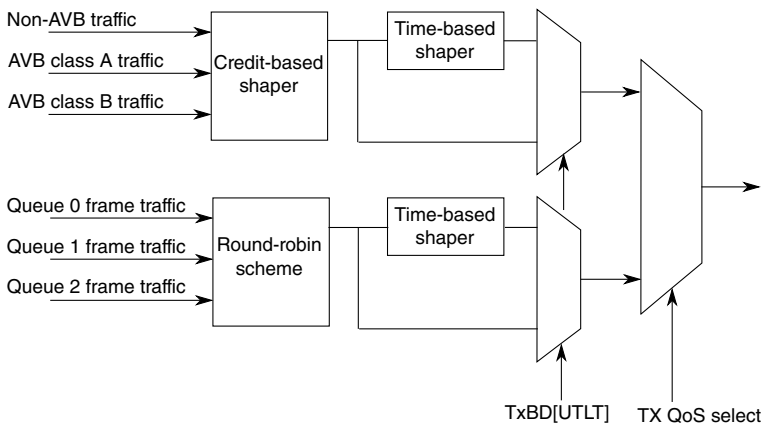


Figure 11-76. Transmit arbitration schemes

11.5.6.5.6.1 Round-robin policy

In the round-robin QoS scheme, each queue is given an equal opportunity to transmit one frame. For example, if queue *n* has a frame to transmit, the queue transmits its frame. After queue *n* has transmitted its frame, or if queue *n* does not have a frame to transmit, queue *n*+1 is then allowed to transmit its frame, and so on.

11.5.6.5.6.2 Credit-based shaper

The AVB credit based shaper acts independently, per class, to control the bandwidth distribution between normal traffic and time-sensitive traffic with respect to the total link bandwidth available. As per the IEEE 802.1Q, the maximum bandwidth distribution that can be allocated for the time-sensitive frames is 75%. The following example uses 70% for AVB frames and 30% for non-AVB frames:

Table 11-124. Bandwidth allocation example

AVB Class A	50%
AVB Class B	20%
non-AVB frames	30%
Total	100%

See DMA Class Based Configuration (ENET_DMA*n*CFG) for information on how to allocate bandwidth. The following figure shows how frame traffic is controlled within the MAC-NET core.

11.5.6.5.6.3 Time-based shaper

The time-based shaper enables the user to specify when a frame can be transmitted. It is always used in combination with either the round-robin scheme or the credit-based shaper. Use of the time-based shaper can ensure that frames are always transmitted in the correct time slot.

11.5.6.5.6.3.1 Time-based shaper example

Time-based shaping involves the following procedure:

1. Read the current value of the timer. See Timer Value Register (ENET_ATVR) and Adjustable Timer Control Register (ENET_ATCR).
2. Calculate the frame launch time and write this value to the TLT field of the frame's TxBD. See Enhanced transmit buffer descriptor.
3. Instruct the ENET to use the TLT by setting TxBD[UTLT].

The frame will be fetched and transmitted only if TxBD[TLT] is less than the current value of the timer. In other words, the timer must be past, that is, be greater than, the transmit launch time before the frame will be fetched and transmitted.

The transmit launch time must be not be greater than the current value of ENET_ATVR + (0.5 × ENET_ATPER). This means the application can not prepare frames with launch times beyond ENET_ATVR + (0.5 × ENET_ATPER). Because ENET_ATVR wraps, calculate TLT using the equation

(ENET_ATVR + (0.5 × ENET_ATPER)) mod ENET_ATPER

For example, if ENET_ATPER is set to the recommended value of 1,000,000,000 ns (one second), then the user must not prepare future frames with launch times greater than (500,000,000 + ENET_ATVR).

As a specific example, if the current value of ENET_ATVR is 100,000, then you can prepare the following frames:

1. Frame 1 TxBD[TLT] = 100,000ns + 125,000ns
2. Frame 2 TxBD[TLT] = 100,000ns + 250,000ns
3. And so on

11.5.6.6 Full-duplex flow control operation

Three conditions are handled by the core's flow control engine:

- Remote device congestion — The remote device connected to the same Ethernet segment as the core reports congestion and requests that the core stop sending data.

- Core FIFO congestion — When the core's receive FIFO reaches a user-programmable threshold (RX section empty), the core sends a pause frame back to the remote device requesting the data transfer to stop.
- Local device congestion — Any device connected to the core can request (typically, via the host processor) the remote device to stop transmitting data.

11.5.6.6.1 Remote device congestion

When the MAC transmit control gets a valid pause quanta from the receive path and if ENET*n*_RCR[FCE] is set, the MAC transmit logic:

- Completes the transfer of the current frame.
- Stops sending data for the amount of time specified by the pause quanta in 512 bit time increments.
- Sets ENET*n*_TCR[RFC_PAUSE].

Frame transfer resumes when the time specified by the quanta expires and if no new quanta value is received, or if a new pause frame with a quanta value set to 0x0000 is received. The MAC also resets RFC_PAUSE to zero.

If ENET*n*_RCR[FCE] cleared, the MAC ignores received pause frames.

Optionally and independent of ENET*n*_RCR[FCE], pause frames are forwarded to the client interface if PAUFWD is set.

11.5.6.6.2 Local device/FIFO congestion

The MAC transmit engine generates pause frames when the local receive FIFO is not able to receive more than a pre-defined number of words (FIFO programmable threshold) or when pause frame generation is requested by the local host processor.

- To generate a pause frame, the host processor sets ENET*n*_TCR[TFC_PAUSE]. A single pause frame is generated when the current frame transfer is completed and TFC_PAUSE is automatically cleared. Optionally, an interrupt is generated.

- An XOFF pause frame is generated when the receive FIFO asserts its section empty flag (internal). An XOFF pause frame is generated automatically, when the current frame transfer completes.
- An XON pause frame is generated when the receive FIFO deasserts its section empty flag (internal). An XON pause frame is generated automatically, when the current frame transfer completes.

When an XOFF pause frame is generated, the pause quanta (payload byte P1 and P2) is filled with the value programmed in ENET*n*_OPD[PAUSE_DUR].

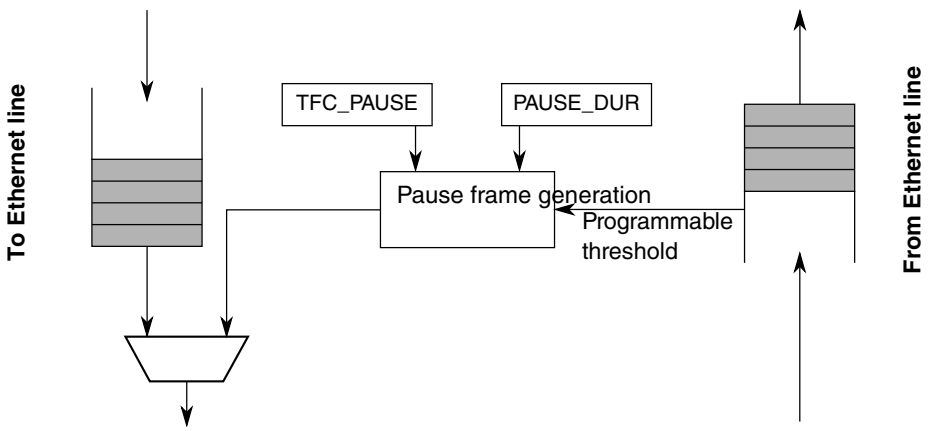


Figure 11-77. Pause frame generation overview

Note

Although the flow control mechanism should prevent any FIFO overflow on the MAC core receive path, the core receive FIFO is protected. When an overflow is detected on the receive FIFO, the current frame is truncated with an error indication set in the frame status word. The frame should subsequently be discarded by the user application.

11.5.6.7 Magic packet detection

Magic packet detection wakes a node that is put in power-down mode by the node management agent. Magic packet detection is supported only if the MAC is configured in sleep mode.

11.5.6.7.1 Sleep mode

To put the MAC in Sleep mode, set ENET*n*_ECR[SLEEP]. At the same time ENET*n*_ECR[MAGICEN] should also be set to enable magic packet detection.

In addition, if ENET is enabled, write 1 to ENET*n*_ECR[SLEEP] before entering into low power mode.

When the MAC is in Sleep mode:

- The transmit logic is disabled.
- The FIFO receive/transmit functions are disabled.
- The receive logic is kept in Normal mode, but it ignores all traffic from the line except magic packets. They are detected so that a remote agent can wake the node.

11.5.6.7.2 Magic packet detection

The core is designed to detect magic packets (see Magic packets) with the destination address set to:

- Any multicast address
- The broadcast address
- The unicast address programmed in PADDR1/2

When a magic packet is detected, EIR[WAKEUP] is set and none of the statistic registers are incremented.

11.5.6.7.3 Wakeup

When a magic packet is detected, indicated by ENET*n*_EIR[WAKEUP], ENET*n*_ECR[SLEEP] should be cleared to resume normal operation of the MAC. Clearing the SLEEP bit automatically masks ENET*n*_ECR[MAGICEN], disabling magic packet detection.

11.5.6.8 IP accelerator functions

The following sections describe the IP accelerator functions.

11.5.6.8.1 Checksum calculation

The IP and ICMP, TCP, UDP checksums are calculated with one's complement arithmetic summing up 16-bit values.

- For ICMP, the checksum is calculated over the complete ICMP datagram, in other words without IP header.
- For TCP and UDP, the checksums contain the header and data sections and values from the IP header, which can be seen as a pseudo-header that is not actually present in the data stream.

Table 11-125. IPv4 pseudo-header for checksum calculation

31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
Source address							
Destination address							
Zero		Protocol		TCP/UDP length			

Table 11-126. IPv6 pseudo-header for checksum calculation

31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
Source address							
Destination address							
TCP/UDP length							
Zero						Next header	

The TCP/UDP length value is the length of the TCP or UDP datagram, which is equal to the payload of an IP datagram. It is derived by subtracting the IP header length from the complete IP datagram length that is given in the IP header (IPv4), or directly taken from the IP header (IPv6). The protocol field is the corresponding value from the IP header. The Zero fields are all zeroes.

For IPv6, the complete 128-bit addresses are considered. The next header value identifies the upper layer protocol as either TCP or UDP. It may differ from the next header value of the IPv6 header if extension headers are inserted before the protocol header.

The checksum calculation uses 16-bit words in network byte order: The first byte sent/received is the MSB, and the second byte sent/received is the LSB of the 16-bit value to add to the checksum. If the frame ends on an odd number of bytes, a zero byte is appended for checksum calculation only, and is not actually transmitted.

11.5.6.8.2 Additional padding processing

According to IEEE 802.3, any Ethernet frame must have a minimum length of 64 octets.

The MAC usually removes padding on receive when a frame with length information is received. Because IP frames have a type value instead of length, the MAC does not remove padding for short IP frames, as it is not aware of the frame contents.

The IP accelerator function can be configured to remove the Ethernet padding bytes that might follow the IP datagram.

On transmit, the MAC automatically adds padding as necessary to fill any frame to a 64-byte length.

11.5.6.8.3 32-bit Ethernet payload alignment

The data FIFOs allow inserting two additional arbitrary bytes in front of a frame. This extends the 14-byte Ethernet header to a 16-byte header, which leads to alignment of the Ethernet payload, following the Ethernet header, on a 32-bit boundary.

This function can be enabled for transmit and receive independently with the corresponding SHIFT16 bits in the ENET*n*_TACC and ENET*n*_RACC registers.

When enabled, the valid frame data is arranged as shown in Table 11-127.

Table 11-127. 64-bit interface data structure with SHIFT16 enabled

63 56	55 48	47 40	39 32	31 24	23 16	15 8	7 0
Byte 5	Byte 4	Byte 3	Byte 2	Byte 1	Byte 0	Any value	Any value
Byte 13	Byte 12	Byte 11	Byte 10	Byte 9	Byte 8	Byte 7	Byte 6
...							

11.5.6.8.3.1 Receive processing

When ENET*n*_RACC[SHIFT16] is set, each frame is received with two additional bytes in front of the frame.

The user application must ignore these first two bytes and find the first byte of the frame in bits 23–16 of the first word from the RX FIFO.

Note

SHIFT16 must be set during initialization and kept set during the complete operation, because it influences the FIFO write behavior.

11.5.6.8.3.2 Transmit processing

When ENET*n*_TACC[SHIFT16] is set, the first two bytes of the first word written (bits 15–0) are discarded immediately by the FIFO write logic.

The SHIFT16 bit can be enabled/disabled for each frame individually if required, but can be changed only between frames.

11.5.6.8.4 Received frame discard

Because the receive FIFO must be operated in store and forward mode (ENET*n*_RSFL cleared), received frames can be discarded based on the following errors:

- The MAC function receives the frame with an error:
  - The frame has an invalid payload length
  - Frame length is greater than MAX_FL
  - Frame received with a CRC-32 error
  - Frame truncated due to receive FIFO overflow
  - Frame is corrupted as PHY signaled an error (RX_ERR asserted during reception)
- An IP frame is detected and the IP header checksum is wrong
- An IP frame with a valid IP header and a valid IP header checksum is detected, the protocol is known but the protocol-specific checksum is wrong

If one of the errors occurs and the IP accelerator function is configured to discard frames (ENET*n*_RACC), the frame is automatically discarded. Statistics are maintained normally and are not affected by this discard function.

11.5.6.8.5 IPv4 fragments

When an IPv4 IP fragment frame is received, only the IP header is inspected and its checksum verified. 32-bit alignment operates the same way on fragments as it does on normal IP frames, as specified above.

The IP fragment frame payload is not inspected for any protocol headers. As such, a protocol header would only exist in the very first fragment. To assist in protocol-specific checksum verification, the one's-complement sum is calculated on the IP payload (all bytes following the IP header) and provided with the frame status word.

The frame fragment status field (RxBD[FRAG]) is set to indicate a fragment reception, and the one's-complement sum of the IP payload is available in RxBD[Payload checksum].

Note

After all fragments have been received and reassembled, the application software can take advantage of the payload checksum delivered with the frame's status word to calculate the protocol-specific checksum of the datagram.

For example, if a TCP payload is delivered by multiple IP fragments, the application software can calculate the pseudo-header checksum value from the first fragment, and add the payload checksums delivered with the status for all fragments to verify the TCP datagram checksum.

11.5.6.8.6 IPv6 support

The following sections describe the IPv6 support.

11.5.6.8.6.1 Receive processing

An Ethernet frame of type 0x86DD identifies an IP Version 6 frame (IPv6) frame. If an IPv6 frame is received, the first IP header is inspected (first ten words), which is available in every IPv6 frame.

If the receive SHIFT16 function is enabled, the IP header is aligned on a 32-bit boundary allowing more efficient processing (see 32-bit Ethernet payload alignment).

For TCP and UDP datagrams, the pseudo-header checksum calculation is performed and verified.

To assist in protocol-specific checksum verification, the one's-complement sum is always calculated on the IP payload (all bytes following the IP header) and provided with the frame status word. For example, if extension headers were present, their sums can be subtracted in software from the checksum to isolate the TCP/UDP datagram checksum, if required.

11.5.6.8.6.2 Transmit processing

For IPv6 transmission, the SHIFT16 function is supported to process 32-bit aligned datagrams.

IPv6 has no IP header checksum; therefore, the IP checksum insertion configuration is ignored.

The protocol checksum is inserted only if the next header of the IP header is a known protocol (TCP, UDP, or ICMP). If a known protocol is detected, the checksum over all bytes following the IP header is calculated and inserted in the correct position.

The pseudo-header checksum calculation is performed for TCP and UDP datagrams accordingly.

11.5.6.9 Resets and stop controls

The following sections describe the resets and stop controls.

11.5.6.9.1 Hardware reset

To reset the Ethernet module, set ENET*n*_ECR[RESET].

11.5.6.9.2 Soft reset

When ENET*n*_ECR[ETHER_EN] is cleared during operation, the following occurs:

- uDMA, buffer descriptor, and FIFO control logic are reset, including the buffer descriptor and FIFO pointers.
- A currently ongoing transmit is terminated by asserting GMII/TXER to the PHY.
- A currently ongoing transmit FIFO write from the application is terminated by stopping the write to the FIFO, and all further data from the application is ignored. All subsequent writes are ignored until re-enabled.
- A currently ongoing receive FIFO read is terminated. The RxBD has arbitrary values in this case.

11.5.6.9.3 Hardware freeze

When the processor enters debug mode and ECR[DBGEN] is set, the MAC enters a freeze state where it stops all transmit and receive activities gracefully.

The following happens when the MAC enters hardware freeze:

- A currently ongoing receive transaction on the receive application interface is completed as normal. No further frames are read from the FIFO.
- A currently ongoing transmit transaction on the transmit application interface is completed as normal (in other words, until writing end-of-packet (EOP)).

- A currently ongoing frame receive is completed normally, after which no further frames are accepted from the MII/GMII.
- A currently ongoing frame transmit is completed normally, after which no further frames are transmitted.

11.5.6.9.4 Graceful stop

During a graceful stop, any currently ongoing transactions are completed normally and no further frames are accepted. The MAC can resume from a graceful stop without the need for a reset (for example, clearing ETHER_EN is not required).

The following conditions lead to a graceful stop of the MAC transmit or receive datapaths.

11.5.6.9.4.1 Graceful transmit stop (GTS)

When gracefully stopped, the MAC is no longer reading frame data from the transmit FIFO and has completed any ongoing transmission.

In any of the following conditions, the transmit datapath stops after an ongoing frame transmission has been completed normally.

- ENET*n*_TCR[GTS] is set by software.
- ENET*n*_TCR[TFC_PAUSE] is set by software requesting a pause frame transmission. The status (and register bit) is cleared after the pause frame has been sent.
- A pause frame was received stopping the transmitter. The stopped situation is terminated when the pause timer expires or a pause frame with zero quanta is received.
- MAC is placed in Sleep mode by software or the processor entering Stop mode (see Sleep mode).
- The MAC is in Hardware Freeze mode.

When the transmitter has reached its stopped state, the following events occur:

- The GRA interrupt is asserted, when transitioned into stopped.
- In Hardware Freeze mode, the GRA interrupt does not wait for the application write completion and asserts when the transmit state machine (in other words, line side of TX FIFO) reaches its stopped state.

11.5.6.9.4.2 Graceful receive stop (GRS)

When gracefully stopped, the MAC is no longer writing frames into the receive FIFO.

The receive datapath stops after any ongoing frame reception has been completed normally, if any of the following conditions occur:

- MAC is placed in Sleep mode either by the software or the processor is in Stop mode). The MAC continues to receive frames and search for magic packets if enabled (see Magic packet detection). However, no frames are written into the receive FIFO, and therefore are not forwarded to the application.
- The MAC is in Hardware Freeze mode. The MAC does not accept any frames from the MII/GMII.

When the receive datapath is stopped, the following events occur:

- If the RX is in the stopped state, RCR[GRS] is set
- The GRA interrupt is asserted when the transmitter and receiver are stopped
- Any ongoing receive transaction to the application (RX FIFO read) continues normally until the frame end of package (EOP) is reached. After this, the following occurs:
  - When Sleep mode is active, all further frames are discarded, flushing the RX FIFO
  - In Hardware Freeze mode, no further frames are delivered to the application and they stay in the receive FIFO.

Note

The assertion of GRS does not wait for an ongoing FIFO read transaction on the application side of the FIFO (FIFO read).

11.5.6.9.4.3 Graceful stop interrupt (GRA)

The graceful stopped interrupt (GRA) is asserted for the following conditions:

- In Sleep mode, the interrupt asserts only after both TX and RX datapaths are stopped.
- In Hardware Freeze mode, the interrupt asserts only after both TX and RX datapaths are stopped.
- The MAC transmit datapath is stopped for any other condition (GTS, TFC_PAUSE, pause received).

The GRA interrupt is triggered only once when the stopped state is entered. If the interrupt is cleared while the stop condition persists, no further interrupt is triggered.

11.5.6.10 IEEE 1588 functions

To allow for IEEE 1588 or similar time synchronization protocol implementations, the MAC is combined with a time-stamping module to support precise time-stamping of incoming and outgoing frames. Set ENET*n*_ECR[EN1588] to enable 1588 support.

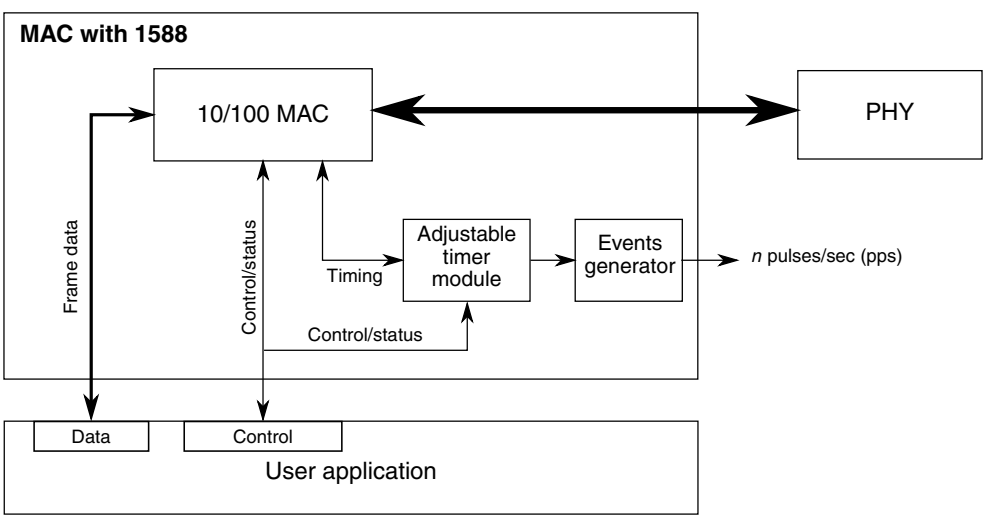


Figure 11-78. IEEE 1588 functions overview

11.5.6.10.1 Adjustable timer module

The adjustable timer module (TSM) implements the free-running counter (FRC), which generates the timestamps. The FRC operates with the time-stamping clock, which can be set to any value depending on your system requirements.

Through dedicated correction logic, the timer can be adjusted to allow synchronization to a remote master and provide a synchronized timing reference to the local system. The timer can be configured to cause an interrupt after a fixed time period, to allow synchronization of software timers or perform other synchronized system functions.

The timer is typically used to implement a period of one second; hence, its value ranges from 0 to (1×10^9)-1. The period event can trigger an interrupt, and software can maintain the seconds and hours time values as necessary.

11.5.6.10.1.1 Adjustable timer implementation

The adjustable timer consists of a programmable counter/accumulator and a correction counter. The periods of both counters and their increment rates are freely configurable, allowing very fine tuning of the timer.

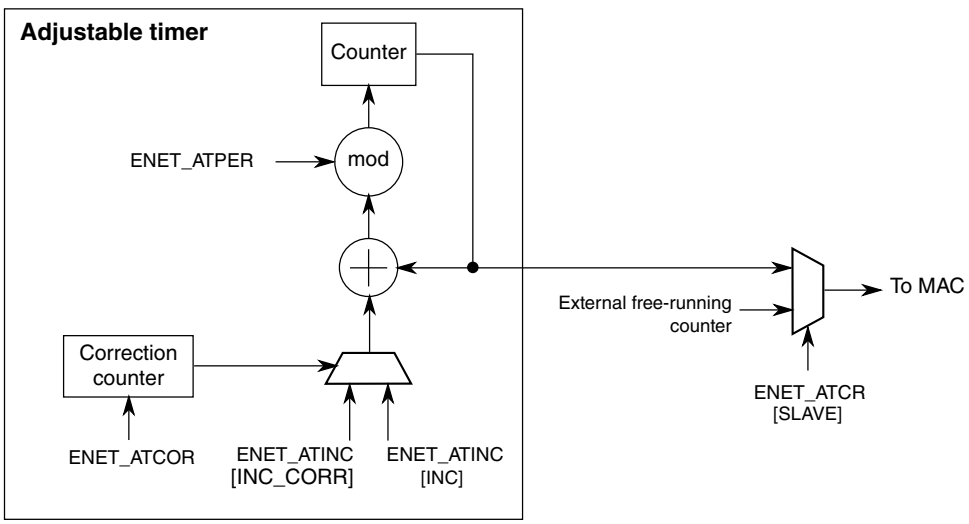


Figure 11-79. Adjustable timer implementation detail

The counter produces the current time. During each time-stamping clock cycle, a constant value is added to the current time as programmed in ENET*n*_ATINC. The value depends on the chosen time-stamping clock frequency. For example, if it operates at 125 MHz, setting the increment to eight represents 8 ns.

The period, configured in ENET*n*_ATPER, defines the modulo when the counter wraps. In a typical implementation, the period is set to 1×10^9 so that the counter wraps every second, and hence all timestamps represent the absolute nanoseconds within the one second period. When the period is reached, the counter wraps to start again respecting the period modulo. This means it does not necessarily start from zero, but instead the counter is loaded with the value (Current + Inc $-(1 \times 10^9)$), assuming the period is set to 1×10^9.

The correction counter operates fully independently, and increments by one with each time-stamping clock cycle. When it reaches the value configured in ENET*n*_ATCOR, it restarts and instructs the timer once to increment by the correction value, instead of the normal value.

The normal and correction increments are configured in ENET*n*_ATINC. To speed up the timer, set the correction increment more than the normal increment value. To slow down the timer, set the correction increment less than the normal increment value.

The correction counter only defines the distance of the corrective actions, not the amount. This allows very fine corrections and low jitter (in the range of 1 ns) independent of the chosen clock frequency.

By enabling slave mode (ENET*n*_ATCR[SLAVE] = 1), the timer is ignored and the current time is externally provided from one of the external modules. See the Chip Configuration details for which clock source is used. This is useful if multiple modules within the system must operate from a single timer. When slave mode is enabled, you still must set ENET*n*_ATINC[INC] to the value of the master, since it is used for internal comparisons.

11.5.6.10.2 Transmit timestamping

Only 1588 event frames need to be time-stamped on transmit. The client application (for example, the MAC driver) should detect 1588 event frames and set TxBD[TS] together with the frame.

If TxBD[TS] is set, the MAC records the timestamp for the frame in ENET*n*_ATSTMP. ENET*n*_EIR[TS_AVAIL] is set to indicate that a new timestamp is available.

Software implements a handshaking procedure by setting TxBD[TS] when it transmits the frame for which a timestamp is needed, and then waits for ENET*n*_EIR[TS_AVAIL] to determine when the timestamp is available. The timestamp is then read from ENET*n*_ATSTMP. This is done for all event frames. Other frames do not use TxBD[TS] and, therefore, do not interfere with the timestamp capture.

11.5.6.10.3 Receive timestamping

When a frame is received, the MAC latches the value of the timer when the frame's start of frame delimiter (SFD) field is detected, and provides the captured timestamp on RxBD[1588 timestamp]. This is done for all received frames.

11.5.6.10.4 Time synchronization

The adjustable timer module is available to synchronize the local clock of a node to a remote master. It implements a free running 32-bit counter, and also contains an additional correction counter.

The correction counter increases or decreases the rate of the free running counter, enabling very fine granular changes of the timer for synchronization, yet adding only very low jitter when performing corrections.

The application software implements, in a slave scenario, the required control algorithm, setting the correction to compensate for local oscillator drifts and locking the timer to the remote master clock on the network.

The timer and all timestamp-related information should be configured to show the true nanoseconds value of a second (in other words, the timer is configured to have a period of one second). Hence, the values range from 0 to (1×10^9)–1. In this application, the seconds counter is implemented in software using an interrupt function that is executed when the nanoseconds counter wraps at 1×10^9.

11.5.6.10.5 Input Capture and Output Compare

The Input Capture Output Compare block can be used to provide precise hardware timing for input and output events.

11.5.6.10.5.1 Input capture

The TCCR*n* capture registers latch the time value when the corresponding external event occurs. An event can be a rising-, falling-, or either-edge of one of the 1588_TMR*n* signals. An event will cause the corresponding TCSR*n*[TF] timer flag to be set, indicating that an input capture has occurred. If the corresponding interrupt is enabled with the TCSR*n*[TIE] field, an interrupt can be generated.

11.5.6.10.5.2 Output compare

The TCCR*n* compare registers are loaded with the time at which the corresponding event should occur. When the ENET free-running counter value matches the output compare reference value in the TCCR*n* register, the corresponding flag, TCSR*n*[TF], is set, indicating that an output compare has occurred. The corresponding interrupt, if enabled by TCSR*n*[TIE], will be generated.The corresponding 1588_TMR*n* output signal will be asserted according to TCSR*n*[TMODE].

NOTE

If TSCR*n*[TMODE] is set to 10X1b or 1010b then the timer output pin toggle-on-overflow will occur only when PINPER, PEREN, and EN bits of the ATCR register are one.

11.5.6.10.5.3 DMA requests

A DMA request can be enabled by setting TCSR*n*[TDRE]. The corresponding DMA request is generated when the TCSR*n*[TF] timer flag is set. When the DMA has completed, the corresponding TCSRn[TF] flag is cleared.

11.5.6.11 FIFO thresholds

The core FIFO thresholds are fully programmable to dynamically change the FIFO operation.

For example, store and forward transfer can be enabled by a simple change in the FIFO threshold registers.

The thresholds are defined in 64-bit words.

The receive and transmit FIFOs both have a depth of 1024 words.

11.5.6.11.1 Receive FIFO

Four programmable thresholds are available, which can be set to any value to control the core operation as follows.

Table 11-128. Receive FIFO thresholds definition

Register	Description
ENET*n*_RSFL [RX_SECTION_FULL]	When the FIFO level reaches the ENET*n*_RSFL value, the MAC status signal is asserted to indicate that data is available in the receive FIFO (cut-through operation). Once asserted, if the FIFO empties below the threshold set with ENET*n*_RAEM and if the end-of-frame is not yet stored in the FIFO, the status signal is deasserted again. If a frame has a size smaller than the threshold (in other words, an end-of-frame is available for the frame), the status is also asserted. To enable store and forward on the receive path, clear ENET*n*_RSFL. The MAC status signal is asserted only when a complete frame is stored in the receive FIFO. When programming a non-zero value to ENET*n*_RSFL (cut-through operation) it should be greater than ENET*n*_RAEM.
ENET*n*_RAEM [RX_ALMOST_EMPTY]	When the FIFO level reaches the ENET*n*_RAEM value, and the end-of-frame has not been received, the core receive read control stops the FIFO read (and subsequently stops transferring data to the MAC client application).

Table continues on the next page...

Table 11-128. Receive FIFO thresholds definition (continued)

Register	Description
	It continues to deliver the frame, if again more data than the threshold or the end-of-frame is available in the FIFO. Set ENET*n*_RAEM to a minimum of six.
ENET*n*_RAFL [RX_ALMOST_FULL]	When the FIFO level approaches the maximum and there is no more space remaining for at least ENETn_RAFL number of words, the MAC control logic stops writing data in the FIFO and truncates the receive frame to avoid FIFO overflow. The corresponding error status is set when the frame is delivered to the application. Set ENET*n*_RAFL to a minimum of 4.
ENET*n*_RSEM [RX_SECTION_EMPTY]	When the FIFO level reaches the ENET*n*_RSEM value, an indication is sent to the MAC transmit logic, which generates an XOFF pause frame. This indicates FIFO congestion to the remote Ethernet client. When the FIFO level goes below the value programmed in ENET*n*_RSEM, an indication is sent to the MAC transmit logic, which generates an XON pause frame. This indicates the FIFO congestion is cleared to the remote Ethernet client. Clearing ENET*n*_RSEM disables any pause frame generation.

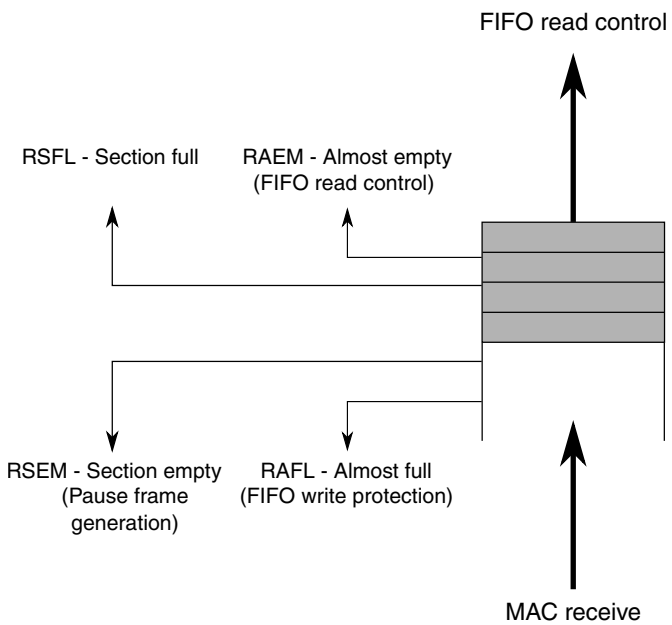


Figure 11-80. Receive FIFO overview

11.5.6.11.2 Transmit FIFO

Four programmable thresholds are available which control the core operation as described below.

Table 11-129. Transmit FIFO thresholds definition

Register	Description
ENET*n*_TAEM [TX_ALMOST _EMPTY]	When the FIFO level reaches the ENET*n*_TAEM value and no end-of-frame is available for the frame, the MAC transmit logic avoids a FIFO underflow by stopping FIFO reads and transmitting the Ethernet frame with an MII error indication. Set ENET*n*_TAEM to a minimum of 4.
ENET*n*_TAFL [TX_ALMOST _FULL]	When the FIFO level approaches the maximum, so that there is no more space for at least ENET*n*_TAFL number of words, the MAC deasserts its control signal to the application. If the application does not react on this signal, the FIFO write control logic avoids FIFO overflow by truncating the current frame and setting the error status. As a result, the frame is transmitted with an GMII/MII error indication. Set ENET*n*_TAFL to a minimum of 4. Larger values allow more latency for the application to react on the MAC control signal deassertion, before the frame is truncated. A typical setting is 8, which offers 3–4 clock cycles of latency to the application to react on the MAC control signal deassertion.
ENET*n*_TSEM [TX_SECTION _EMPTY]	When the FIFO level reaches the ENET*n*_TSEM value, a MAC status signal is deasserted to indicate that the transmit FIFO is getting full. This gives the ENET module an indication to slow or stop its write transaction to avoid a buffer overflow. This is a pure indication function to the application. It has no effect within the MAC. When ENET*n*_TSEM is 0, the signal is never deasserted.
ENET*n*_TFWR	When the FIFO level reaches the ENET*n*_TFWR value and when STRFWD is cleared, the MAC transmit control logic starts frame transmission before the end-of-frame is available in the FIFO (cut-through operation). If a complete frame has a size smaller than the ENET*n*_TFWR threshold, the MAC also transmits the frame to the line. To enable store and forward on the transmit path, set STRFWD. In this case, the MAC starts to transmit data only when a complete frame is stored in the transmit FIFO.

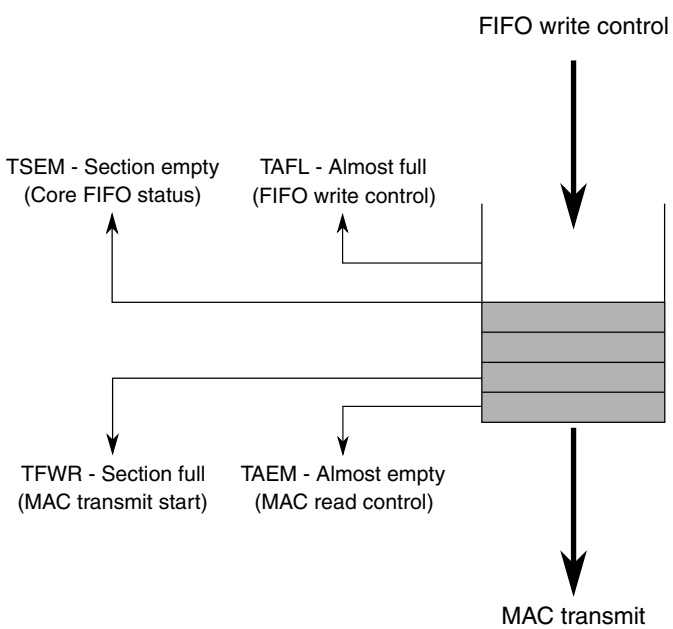


Figure 11-81. Transmit FIFO overview

11.5.6.12 Loopback options

The core implements external and internal loopback options, which are controlled by the ENET*n*_RCR register fields found here.

The core implements external and internal loopback options, which are controlled by the following ENET*n*_RCR register fields:

Table 11-130. Loopback options

Register field	Description
LOOP	Internal MII loopback. The MAC transmit is returned to the MAC receive. No data is transmitted to the external interfaces. In MII internal loopback, MII_TXCLK and MII_RXCLK must be provided with a clock signal (2.5 MHz for 10 Mbit/s, and 25 MHz for 100 Mbit/s))

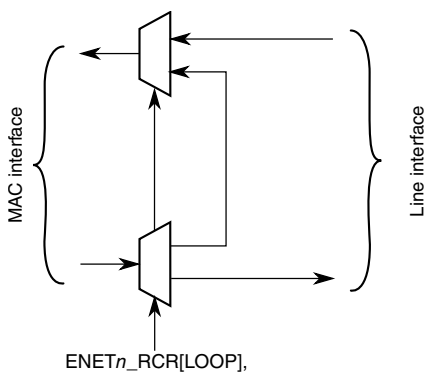


Figure 11-82. Loopback options

11.5.6.13 Legacy buffer descriptors

To support the Ethernet controller on previous chips, legacy FEC buffer descriptors are available. To enable legacy support, write 0 to ENET*n*_ECR[1588EN].

NOTE

- Legacy buffer descriptors are used in single-ring mode, that is, when DMA*n*CFG[DMA_CLASS_EN] are zero. In multi-ring mode, legacy TxBDs are used only with the round-robin scheme.
- The legacy buffer descriptor tables show the byte order for big-endian chips. DBSWP must be set to 0 after reset to enable big-endian mode.

11.5.6.13.1 Legacy receive buffer descriptor

The following table shows the legacy FEC receive buffer descriptor. Table 11-134 contains the descriptions for each field.

Table 11-131. Legacy FEC receive buffer descriptor (RxBD)

	Byte 0								Byte 1							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Offset + 0	E	RO1	W	RO2	L	—	—	M	BC	MC	LG	NO	—	CR	OV	TR
Offset + 2	Data length															
Offset + 4	Rx data buffer pointer — high halfword															
Offset + 6	Rx data buffer pointer — low halfword															

11.5.6.13.2 Legacy transmit buffer descriptor

The following table shows the legacy FEC transmit buffer descriptor. Table 11-136 contains the descriptions for each field.

Table 11-132. Legacy FEC transmit buffer descriptor (TxBD)

	Byte 0								Byte 1							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Offset + 0	R	TO1	W	TO2	L	TC	ABC[1]	—	—	—	—	—	—	—	—	—
Offset + 2	Data Length															
Offset + 4	Tx Data Buffer Pointer — high halfword															
Offset + 6	Tx Data Buffer Pointer — low halfword															

1. This field is not supported by the uDMA.

11.5.6.14 Enhanced buffer descriptors

This section provides a description of the enhanced operation of the driver/uDMA via the buffer descriptors.

It is followed by a detailed description of the receive and transmit descriptor fields. To enable the enhanced features, set ENET*n*_ECR[1588EN].

NOTE

The enhanced buffer descriptor tables show the byte order for big-endian chips. DBSWP must be set to 0 after reset to enable big-endian mode.

11.5.6.14.1 Enhanced receive buffer descriptor

The following table shows the enhanced uDMA receive buffer descriptor. Table 11-134 contains the descriptions for each field.

Table 11-133. Enhanced uDMA receive buffer descriptor (RxBD)

	Byte 0								Byte 1							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Offset + 0	E	RO1	W	RO2	L	—	—	M	BC	MC	LG	NO	—	CR	OV	TR
Offset + 2	Data length															
Offset + 4	Rx data buffer pointer – high halfword															
Offset + 6	Rx data buffer pointer – low halfword															
Offset + 8	ME	—	—	—	—	PE	CE	UC	INT	—	—	—	—	—	—	—
Offset + A	VPCP			—	—	—	—	—	—	—	ICE	PCR	—	VLAN	IPV6	FRAG
Offset + C	Header length					—	—	—	Protocol type							
Offset + E	Payload checksum															
Offset + 10	BDU	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 12	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 14	1588 timestamp – high halfword															
Offset + 16	1588 timestamp – low halfword															
Offset + 18	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 1A	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 1C	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 1E	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

Table 11-134. Receive buffer descriptor field definitions

Word	Field	Description
Offset + 0	15 E	Empty. Written by the MAC (= 0) and user (= 1). 0 The data buffer associated with this BD is filled with received data, or data reception has aborted due to an error condition. The status and length fields have been updated as required. 1 The data buffer associated with this BD is empty, or reception is currently in progress.
Offset + 0	14 RO1	Receive software ownership. This field is reserved for use by software. This read/write field is not modified by hardware, nor does its value affect hardware.
Offset + 0	13 W	Wrap. Written by user. 0 The next buffer descriptor is found in the consecutive location. 1 The next buffer descriptor is found at the location defined in ENET*n*_RDSR.
Offset + 0	12 RO2	Receive software ownership. This field is reserved for use by software. This read/write field is not modified by hardware, nor does its value affect hardware.
Offset + 0	11 L	Last in frame. Written by the uDMA. 0 The buffer is not the last in a frame.

Table continues on the next page...

Table 11-134. Receive buffer descriptor field definitions (continued)

Word	Field	Description
		1 The buffer is the last in a frame.
Offset + 0	10–9	Reserved, must be cleared.
Offset + 0	8 M	Miss. Written by the MAC. This field is set by the MAC for frames accepted in promiscuous mode, but flagged as a miss by the internal address recognition. Therefore, while in promiscuous mode, you can use this field to quickly determine whether the frame was destined to this station. This field is valid only if the L and PROM fields are set to 1. 0 The frame was received because of an address recognition hit. 1 The frame was received because of promiscuous mode. The information needed for this field comes from the promiscuous_miss(ff_rx_err_stat[26]) sideband signal.
Offset + 0	7 BC	Set if the DA is broadcast (FFFF_FFFF_FFFF).
Offset + 0	6 MC	Set if the DA is multicast and not BC.
Offset + 0	5 LG	Receive frame length violation. Written by the MAC. A frame length greater than RCR[MAX_FL] was recognized. This field is valid only if the L field is set. The receive data is not altered in any way unless the length exceeds TRUNC_FL bytes.
Offset + 0	4 NO	Receive non-octet aligned frame. Written by the MAC. A frame that contained a number of bits not divisible by 8 was received, and the CRC check that occurred at the preceding byte boundary generated an error or a PHY error occurred. This field is valid only if the L field is set. If this field is set, the CR field is not set.
Offset + 0	3	Reserved, must be cleared.
Offset + 0	2 CR	Receive CRC or frame error. Written by the MAC. This frame contains a PHY or CRC error and is an integral number of octets in length. This field is valid only if the L field is set.
Offset + 0	1 OV	Overrun. Written by the MAC. A receive FIFO overrun occurred during frame reception. If this field is set, the other status fields, M, LG, NO, and CR, lose their normal meaning and are zero. This field is valid only if the L field is set.
Offset + 0	0 TR	Set if the receive frame is truncated (frame length >TRUNC_FL). If the TR field is set, the frame must be discarded and the other error fields must be ignored because they may be incorrect.
Offset + 2	15–0 Data Length	Data length. Written by the MAC. Data length is the number of octets written by the MAC into this BD's data buffer if L is cleared (the value is equal to EMRBR), or the length of the frame including CRC if L is set. It is written by the MAC once as the BD is closed.
0ffset + 4	15–0 Data buffer pointer high	Receive data buffer pointer, high halfword[1]
Offset + 6	15–0 Data buffer pointer low	Receive data buffer pointer, low halfword
Offset + 8	15 ME	MAC error. This field is written by the uDMA. This field means that the frame stored in the system memory was received with an error (typically, a receive FIFO overflow). This field is only valid when the L field is set.
Offset + 8	14–11	Reserved, must be cleared.

Table continues on the next page...

Table 11-134. Receive buffer descriptor field definitions (continued)

Word	Field	Description
Offset + 8	10 PE	PHY Error. This field is written by the uDMA. Set to "1"when the frame was received with an Error character on the PHY interface. The frame is invalid. This field is valid only when the L field is set.
Offset + 8	9 CE	Collision. This field is written by the uDMA. Set when the frame was received with a collision detected during reception. The frame is invalid and sent to the user application. This field is valid only when the L field is set.
Offset + 8	8 UC	Unicast. This field is written by the uDMA, and means that the frame is unicast. This field is valid regardless of whether the L field is set.
Offset + 8	7 INT	Generate RXB/RXF interrupt. This field is set by the user to indicate that the uDMA is to generate an interrupt on the *dma_int_rxb* / *dma_int_rxfevent*.
Offset + 8	6–0	Reserved, must be cleared.
Offset + A	15–13 VPCP	VLAN priority code point. This field is written by the uDMA to indicate the frame priority level. Valid values are from 0 (best effort) to 7 (highest). This value can be used to prioritize different classes of traffic (e.g., voice, video, data). This field is only valid if the L field is set.
Offset + A	12–6	Reserved, must be cleared.
Offset + A	5 ICE	IP header checksum error. This is an accelerator option. This field is written by the uDMA. Set when either a non-IP frame is received or the IP header checksum was invalid. An IP frame with less than 3 bytes of payload is considered to be an invalid IP frame. This field is only valid if the L field is set.
Offset + A	4 PCR	Protocol checksum error. This is an accelerator option. This field is written by the uDMA. Set when the checksum of the protocol is invalid or an unknown protocol is found and checksumming could not be performed. This field is only valid if the L field is set.
Offset + A	3	Reserved, must be cleared.
Offset + A	2 VLAN	VLAN. This is an accelerator option. This field is written by the uDMA. It means that the frame has a VLAN tag. This field is valid only if the L field is set.
Offset + A	1 IPV6	IPV6 Frame. This field is written by the uDMA. This field indicates that the frame has an IPv6 frame type. If this field is not set it means that an IPv4 or other protocol frame was received. This field is valid only if the L field is set.
Offset + A	0 FRAG	IPv4 Fragment.This is an accelerator option.This field is written by the uDMA. It indicates that the frame is an IPv4 fragment frame. This field is only valid when the L field is set.
Offset + C	15–11 Header length	Header length. This is an accelerator option. This field is written by the uDMA. This field is the sum of 32-bit words found within the IP and its following protocol headers. If an IP datagram with an unknown protocol is found, then the value is the length of the IP header. If no IP frame or an erroneous IP header is found, the value is 0. The following values are minimum values if no header options exist in the respective headers: • ICMP/IP: 6 (5 IP header, 1 ICMP header) • UDP/IP: 7 (5 IP header, 2 UDP header) • TCP/IP: 10 (5 IP header, 5 TCP header) This field is only valid if the L field is set.
Offset + C	10–8	Reserved, must be cleared.
Offset + C	7–0 Protocol type	Protocol type. This is an accelerator option. The 8-bit protocol field found within the IP header of the frame. It is valid only when ICE is cleared. This field is valid only when the L field is set.

Table continues on the next page...

Table 11-134. Receive buffer descriptor field definitions (continued)

Word	Field	Description
Offset + E	15–0 Payload checksum	Internet payload checksum. This is an accelerator option. It is the one's complement sum of the payload section of the IP frame. The sum is calculated over all data following the IP header until the end of the IP payload. This field is valid only when the L field is set.
Offset + 10	15 BDU	Last buffer descriptor update done. Indicates that the last BD data has been updated by uDMA. This field is written by the user (=0) and uDMA (=1).
Offset + 10	14–0	Reserved, must be cleared.
Offset + 12	15–0	Reserved, must be cleared.
Offset + 14	15–0 1588 timestamp	This value is written by the uDMA. It is only valid if the L field is set.
Offset + 16		
Offset + 18 – Offset + 1E	15–0	Reserved, must be cleared.

1. The receive buffer pointer, containing the address of the associated data buffer, must always be evenly divisible by 64. The buffer must reside in memory external to the MAC. The Ethernet controller never modifies this value.

11.5.6.14.2 Enhanced transmit buffer descriptor

Table 11-135. Enhanced transmit buffer descriptor (TxBD)

	Byte 0								Byte 1							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Offset + 0	R	TO1	W	TO2	L	TC	—	—	—	—	—	—	—	—	—	—
Offset + 2	Data length															
Offset + 4	Tx Data Buffer Pointer – high halfword															
Offset + 6	Tx Data Buffer Pointer – low halfword															
Offset + 8	—	INT	TS	PINS	IINS	—	—	UTLT	FTYPE				—	—	—	—
Offset + A	TXE	—	UE	EE	FE	LCE	OE	TSE	—	—	—	—	—	—	—	—
Offset + C	TLT – high halfword															
Offset + E	TLT – low halfword															
Offset + 10	BDU	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 12	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 14	1588 timestamp – high halfword															
Offset + 16	1588 timestamp – low halfword															
Offset + 18	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 1A	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 1C	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 1E	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

Table 11-136. Enhanced transmit buffer descriptor field definitions

Word	Field	Description
Offset + 0	15 R	Ready. Written by the MAC and user. 0 The data buffer associated with this BD is not ready for transmission. You are free to manipulate this BD or its associated data buffer. The MAC clears this field after the buffer has been transmitted or after an error condition is encountered. 1 The data buffer, prepared for transmission by you, has not been transmitted or currently transmits. You may write no fields of this BD after this field is set.
Offset + 0	14 TO1	Transmit software ownership. This field is reserved for software use. This read/write field is not modified by hardware and its value does not affect hardware.
Offset + 0	13 W	Wrap. Written by user. 0 The next buffer descriptor is found in the consecutive location 1 The next buffer descriptor is found at the location defined in ETDSR.
Offset + 0	12 TO2	Transmit software ownership. This field is reserved for use by software. This read/write field is not modified by hardware and its value does not affect hardware.
Offset + 0	11 L	Last in frame. Written by user. 0 The buffer is not the last in the transmit frame 1 The buffer is the last in the transmit frame
Offset + 0	10 TC	Transmit CRC. Written by user, and valid only when L is set. 0 End transmission immediately after the last data byte 1 Transmit the CRC sequence after the last data byte This field is valid only when the L field is set.
Offset + 0	9 ABC	Append bad CRC. **Note:** This field is not supported by the uDMA and is ignored.
Offset + 0	8–0	Reserved, must be cleared.
Offset + 2	15–0 Data Length	Data length, written by user. Data length is the number of octets the MAC should transmit from this BD's data buffer. It is never modified by the MAC.
Offset + 4	15–0 Data buffer pointer high	Tx data buffer pointer, high halfword. The buffer must reside in memory external to the MAC. This value is never modified by the Ethernet controller. **NOTE:** For optimal performance, make the transmit buffer pointer evenly divisible by 64.
Offset + 6	15–0 Data buffer pointer low	Tx data buffer pointer, low halfword
Offset + 8	15	Reserved, must be cleared.
Offset + 8	14 INT	Generate interrupt flags. This field is written by the user. This field is valid regardless of the L field and must be the same for all EBD for a given frame. The uDMA does not update this value.
Offset + 8	13 TS	Timestamp. This field is written by the user. This indicates that the uDMA is to generate a timestamp frame to the MAC. This field is valid regardless of the L field and must be the same for all EBD for the given frame. The uDMA does not update this value.

Table continues on the next page...

Table 11-136. Enhanced transmit buffer descriptor field definitions (continued)

Word	Field	Description
Offset + 8	12 PINS	Insert protocol specific checksum. This field is written by the user. If set, the MAC's IP accelerator calculates the protocol checksum and overwrites the corresponding checksum field with the calculated value. The checksum field must be cleared by the application generating the frame. The uDMA does not update this value. This field is valid regardless of the L field and must be the same for all EBD for a given frame.
Offset + 8	11 IINS	Insert IP header checksum. This field is written by the user. If set, the MAC's IP accelerator calculates the IP header checksum and overwrites the corresponding header field with the calculated value. The checksum field must be cleared by the application generating the frame. The uDMA does not update this value. This field is valid regardless of the L field and must be the same for all EBD for a given frame.
Offset + 8	10-9	Reserved, must be cleared.
Offset + 8	8 UTLT	Use transmit launch time. If set to 1, TLT high and low values are used to determine if the frame can be transmitted. This field must only be set in the *first* BD of a frame. TLT is always used in combination with either the round-robin scheme or the credit-based shaper. TLT can be used with single a BD queue. However, at least one DMA class, DMA*n*CFG[DMA_CLASS_ EN] must be enabled. For example, to use TLT with only queue 0, enable one of the DMA*n*CFG[DMA_CLASS_EN] fields and clear both TDAR1 and TDAR2. Although this enables multi-queue, only the single queue 0 is used.
Offset + 8	7–4 FTYPE	Type of frame to be transmitted. If a credit-based scheme is used, this field must match the BD ring queue. 0x0 – Non-AVB. Corresponds to ENET_TDSR. 0x1 – AVB Class A. Corresponds to ENET_TDSR1. 0x2 – AVB Class B. Corresponds to ENET_TDSR2. All other values are reserved.
Offset + 8	3–0	Reserved, must be cleared.
Offset + A	15 TXE	Transmit error occurred. This field is written by the uDMA. This field indicates that there was a transmit error of some sort reported with the frame. Effectively this field is an OR of the other error fields including UE, EE, FE, LCE, OE, and TSE. This field is valid only when the L field is set.
Offset + A	14	Reserved, must be cleared.
Offset + A	13 UE	Underflow error. This field is written by the uDMA. This field indicates that the MAC reported an underflow error on transmit. This field is valid only when the L field is set.
Offset + A	12 EE	Excess Collision error. This field is written by the uDMA. This field indicates that the MAC reported an excess collision error on transmit. This field is valid only when the L field is set.
Offset + A	11 FE	Frame with error. This field is written by the uDMA. This field indicates that the MAC reported that the uDMA reported an error when providing the packet. This field is valid only when the L field is set.
Offset + A	10 LCE	Late collision error. This field is written by the uDMA. This field indicates that the MAC reported that there was a Late Collision on transmit. This field is valid only when the L field is set.

Table continues on the next page...

Table 11-136. Enhanced transmit buffer descriptor field definitions (continued)

Word	Field	Description
Offset + A	9 OE	Overflow error. This field is written by the uDMA. This field indicates that the MAC reported that there was a FIFO overflow condition on transmit. This field is only valid when the L field is set.
Offset + A	8 TSE	Timestamp error. This field is written by the uDMA. This field indicates that the MAC reported a different frame type then a timestamp frame. This field is valid only when the L field is set.
Offset + A	7–0	Reserved, must be cleared.
Offset + C	15–0 TLT high	Transmit launch time high. Specifies when frame can by transmitted.
Offset + E	15–0 TLT low	Transmit launch time low. Specifies when frame can by transmitted.
Offset + 10	15 BDU	Last buffer descriptor update done. Indicates that the last BD data has been updated by uDMA. This field is written by the user (=0) and uDMA (=1).
Offset + 10	14–0	Reserved, must be cleared.
Offset + 12	15–0	Reserved, must be cleared.
Offset + 14	15–0 1588 timestamp	This value is written by the uDMA . It is valid only when the L field is set.
Offset + 16		
Offset + 18–Offset + 1E	15–0	Reserved, must be cleared.

11.5.6.15 Client FIFO application interface

The FIFO interface is completely asynchronous from the Ethernet line, and the transmit and receive interface can operate at a different clock rate.

All transfers to/from the user application are handled independently of the core operation, and the core provides a simple interface to user applications based on a two-signal handshake.

11.5.6.15.1 Data structure description

The data structure defined in the following tables for the FIFO interface must be respected to ensure proper data transmission on the Ethernet line. Byte 0 is sent to and received from the line first.

Table 11-137. FIFO interface data structure

	63 56	55 48	47 40	39 32	31 24	23 16	15 8	7 0
Word 0	Byte 7	Byte 6	Byte 5	Byte 4	Byte 3	Byte 2	Byte 1	Byte 0
Word 1	Byte 15	Byte 14	Byte 13	Byte 12	Byte 11	Byte 10	Byte 9	Byte 8
...	...							

The size of a frame on the FIFO interface may not be a modulo of 64-bit.

The user application may not care about the Ethernet frame formats in full detail. It needs to provide and receive an Ethernet frame with the following structure:

- Ethernet MAC destination address
- Ethernet MAC source address
- Optional 802.1q VLAN tag (VLAN type and info field)
- Ethernet length/type field
- Payload

Frames on the FIFO interface do not contain preamble and SFD fields, which are inserted and discarded by the MAC on transmit and receive, respectively.

- On receive, CRC and frame padding can be stripped or passed through transparently.
- On transmit, padding and CRC can be provided by the user application, or appended automatically by the MAC independently for each frame. No size restrictions apply.

Note

On transmit, if ENET*n*_TCR[ADDINS] is set, bytes 6–11 of each frame can be set to any value, since the MAC overwrites the bytes with the MAC address programmed in the ENET*n*_PAUR and ENET*n*_PALR registers.

Table 11-138. FIFO interface frame format

Byte number	Field
0–5	Destination MAC address
6–11	Source MAC address
12–13	Length/type field
14–N	Payload data

VLAN-tagged frames are supported on both transmit and receive, and implement additional information (VLAN type and info).

Table 11-139. FIFO interface VLAN frame format

Byte number	Field
0–5	Destination MAC address
6–11	Source MAC address

Table continues on the next page...

Table 11-139. FIFO interface VLAN frame format (continued)

Byte number	Field
12–15	VLAN tag and info
16–17	Length/type field
18–N	Payload data

Note

The standard defines that the LSB of the MAC address is sent/received first, while for all the other header fields — in other words, length/type, VLAN tag, VLAN info, and pause quanta — the MSB is sent/received first.

11.5.6.15.2 Data structure examples

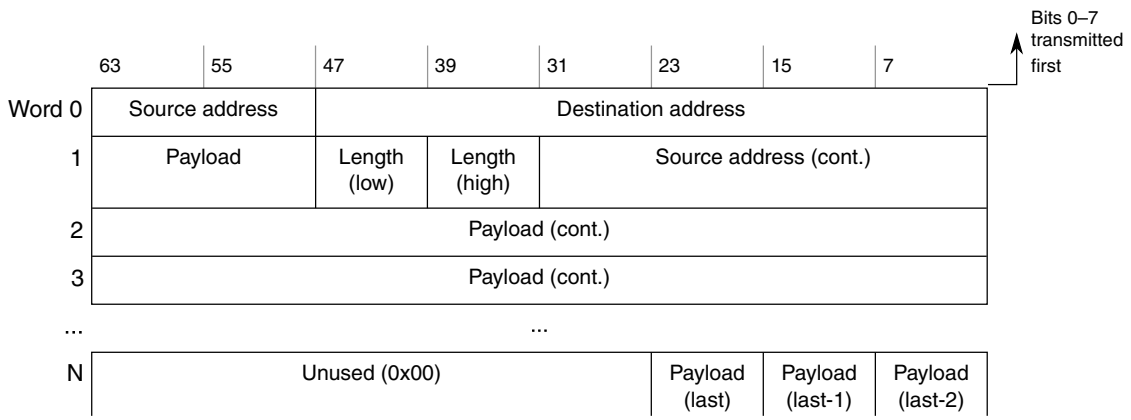


Figure 11-83. Normal Ethernet frame 64-bit mapping example

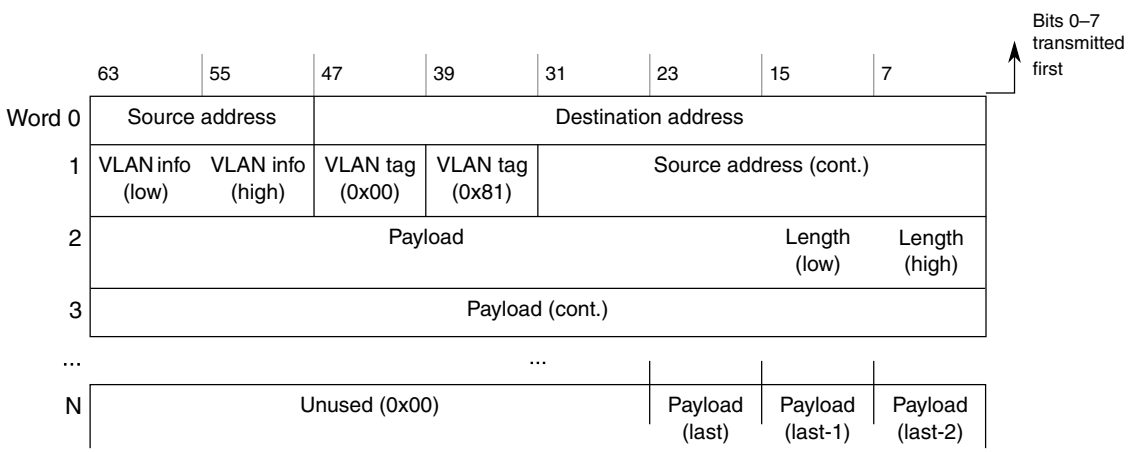


Figure 11-84. VLAN-tagged frame 64-bit mapping example

If CRC forwarding is enabled (CRCFWD = 0), the last four valid octets of the frame contain the FCS field. The non-significant bytes of the last word can have any value.

11.5.6.15.3 Frame status

A MAC layer status word and an accelerator status word is available in the receive buffer descriptor.

See Enhanced buffer descriptors for details.

The status is available with each frame with the last data of the frame.

If the frame status contains a MAC layer error (for example, CRC or length error), RxBD[ME] is also set with the last data of the frame.

11.5.6.16 FIFO protection

The following sections describe the FIFO protection mechanisms.

11.5.6.16.1 Transmit FIFO underflow

During a frame transfer, when the transmit FIFO reaches the almost empty threshold with no end-of-frame indication stored in the FIFO, the MAC logic:

- Stops reading data from the FIFO
- Asserts the MII error signal (MII_TXER) (1 in Figure 11-85) to indicate that the fragment already transferred is not valid
- Deasserts the MII transmit enable signal (MII_TXEN) to terminate the frame transfer (2)

After an underflow, when the application completes the frame transfer (3), the MAC transmit logic discards any new data available in the FIFO until the end of packet is reached (4) and sets the enhanced TxBD[UE] field.

The MAC starts to transfer data on the MII interface when the application sends a new frame with a start of frame indication (5).

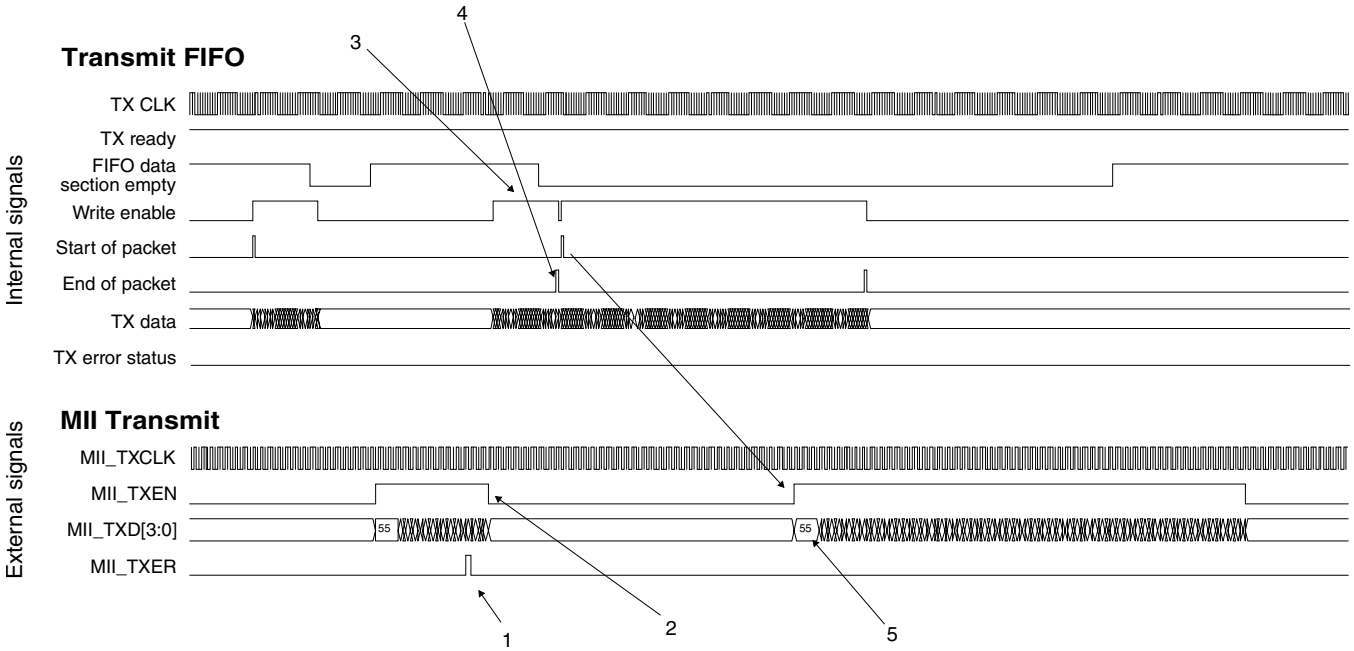


Figure 11-85. Transmit FIFO underflow protection

11.5.6.16.2 Transmit FIFO overflow

On the transmit path, when the FIFO reaches the programmable almost full threshold, the internal MAC ready signal is deasserted. The application should stop sending new data.

However, if the application keeps sending data, the transmit FIFO overflows, corrupting contents that were previously stored. The core logic sets the enhanced TxBD[OE] field for the next frame transmitted to indicate this overflow occurence.

Note

Overflow is a fatal error and must be addressed by resetting the core or clearing ENET*n*_ECR[ETHER_EN], to clear the FIFOs and prepare for normal operation again.

11.5.6.16.3 Receive FIFO overflow

During a frame reception, if the client application is not able to receive data (1), the MAC receive control truncates the incoming frame when the FIFO reaches the programmable almost-full threshold to avoid an overflow.

The frame is subsequently received on the FIFO interface with an error indication (enhanced RxBD[ME] field set together with receive end-of-packet) (2) with the truncation error status field set (3).

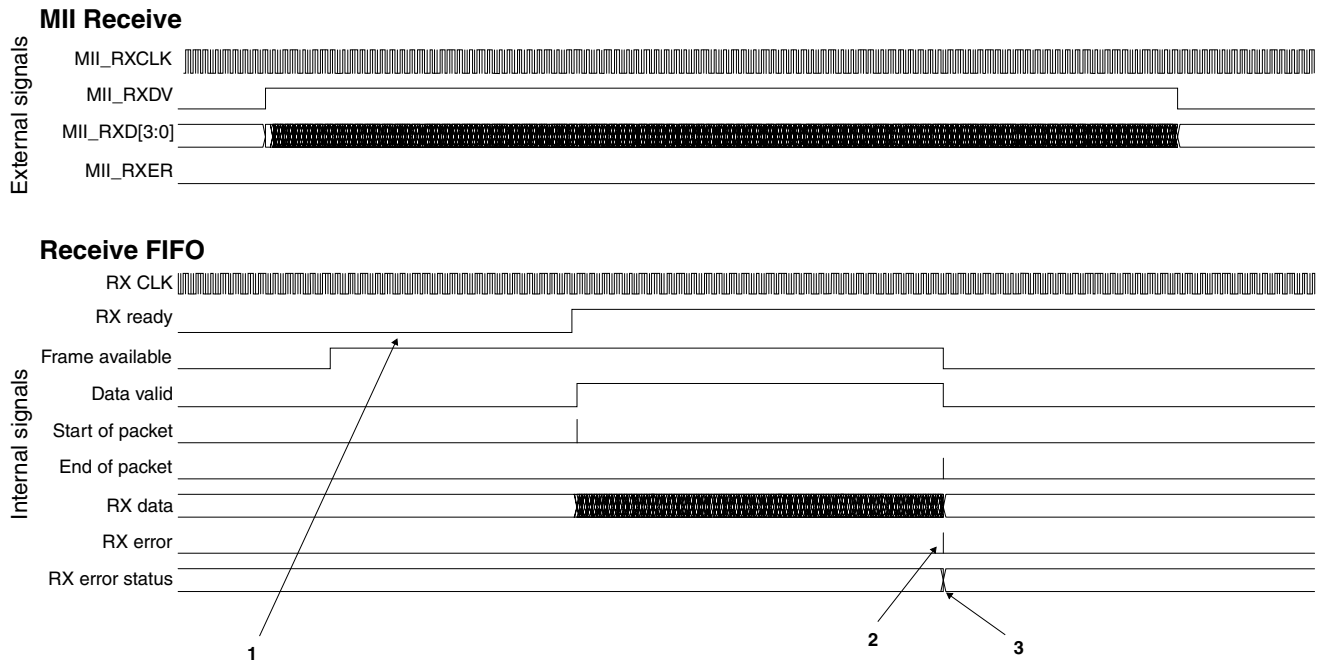


Figure 11-86. Receive FIFO overflow protection

11.5.6.17 PHY management interface

The MDIO interface is a two-wire management interface. The MDIO management interface implements a standardized method to access the PHY device management registers.

The core implements a master MDIO interface, which can be connected to up to 32 PHY devices.

11.5.6.17.1 MDIO clause 22 frame format

The core MDIO master controller communicates with the slave (PHY device) using frames that are defined in the following table.

A complete frame has a length of 64 bits made up of an optional 32-bit preamble, 14-bit command, 2-bit bus direction change, and 16-bit data. Each bit is transferred on the rising edge of the MDIO clock (MDC signal). The MDIO data signal is tri-stated between frames.

The core PHY management interface supports the standard MDIO specification (IEEE 802.3 Clause 22).

Table 11-140. MDIO clause 22 frame structure

ST	OP	PHYADR	REGADR	TA	DATA

Table 11-141. MDIO frame field descriptions

Field	Description
ST (2 bits)	Start indication field, programmed with ENET*n*_MMFR[ST] and equal to 01 for Standard MDIO (Clause 22).
OP (2 bits)	Opcode defines type of operation. Programmed with ENET*n*_MMFR[OP]. 01 Write operation 10 Read operation
PHYADR (5 bits)	Five-bit PHY device address, programmed with ENET*n*_MMFR[PA]. Up to 32 devices can be addressed.
REGADR (5 bits)	Five-bit register address, programmed with ENET*n*_MMFR[RA]. Each PHY can implement up to 32 registers.
TA (2 bits)	Turnaround time, programmed with ENET*n*_MMFR[TA]. Two bit-times are reserved for read operations to switch the data bus from write to read. The PHY device presents its register contents in the data phase and drives the bus from the second bit of the turnaround phase.
Data (16 bits)	Data, set by ENET*n*_MMFR[DATA]. Written to or read from the PHY

11.5.6.17.2 MDIO clause 45 frame format

The extended MDIO frame structure defined in IEEE 802.3 Clause 45 introduces indirect addressing. First, a write transaction to an address register is done, followed by a write or read transaction which will put the 16-bit data in the register or retrieve the register contents respectively. A preamble of 32 bits of logical ones is sent prior to every transaction. The MDIO data signal is tri-stated between frames.

The extended MDIO defines four transactions, which are determined by the two-bit opcode field.

Table 11-142. MDIO clause 45 frame structure

ST	OP	PRTAD	DEVAD	TA	ADDR/DATA

All bits are transmitted from left to right (Preamble bits first) and all fields have their Most-Significant bit sent first (leftmost in above table). The complete frame has a length of 64 bits (32-bit preamble, 14-bit command, 2-bit bus direction change, 16-bit data). Each bit is transferred with the rising edge of the MDIO clock (MDC).

The fields and transactions are summarized in the following tables.

Table 11-143. MDIO clause 45 frame field descriptions

Field	Description
ST	Start indication. Indicates the end of the preamble and start of the frame. This value is 00 for extended MDIO (Clause 45) frames.
OP	Opcode defines if a read or write operation is performed and is programmed with ENET*n*_MMFR[OP]. See Table 11-144 for more information. 00 Address write 01 Write operation 10 Read inc. operation 11 Read operation
PRTAD	The port address specifies a MDIO port. Each Port can have up to 32 devices which each can have a separate set of registers.
DEVAD	Device address. Up to 32 devices can be addressed (within a port).
TA	Turnaround time, programmed with ENET*n*_MMFR[TA]. Two bit-times are reserved for read operations to switch the data bus from write to read. The PHY device presents its register contents in the data phase and drives the bus from the second bit of the turnaround phase.
ADDR/DATA	16-bit address (for address write) or data, set by ENET*n*_MMFR[DATA], written to or read from the PHY.

Table 11-144. MDIO Clause 45 Transactions

Transaction Type	Description
Address	A write transaction to the internal address register of the device/port. The data section of the frame contains the value to be stored in the device's internal address "pointer" register for further transactions.
Write	Data write to a register. The 16 bit data will be written to the register identified by the device-internal address.
Read	Data is read from the register identified by the device-internal address.
Read inc.	Read with address postincrement. The register identified by the device-internal address is read. After this, the device-internal address is incremented. If the address register is all '1' (0xFFFF) no increment is done (i.e. increment does not wrap around).

11.5.6.17.3 MDIO clock generation

The MDC clock is generated from the internal bus clock (i.e., IPS bus clock) divided by the value programmed in ENET*n*_MSCR[MII_SPEED].

11.5.6.17.4 MDIO operation

To perform an MDIO access, set the MDIO command register (ENET*n*_MMFR) according to the description provided in MII Management Frame Register (ENETn_MMFR).

To check when the programmed access completes, read the ENET*n*_EIR[MII] field.

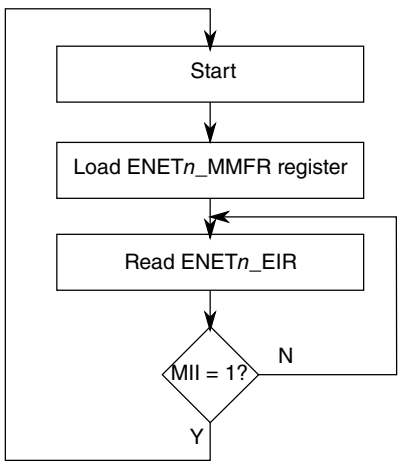


Figure 11-87. MDIO access overview

11.5.6.18 Ethernet interfaces

The following Ethernet interfaces are implemented:

- Fast Ethernet MII (Media Independent Interface)
- RMII 10/100 using interface converters/gaskets
- RGMII 10/100/1000 by way of interface converters/gaskets

The following table shows how to configure ENET registers to select each interface.

Mode	ECR[SPEED]	RCR[RMII_10T]	RCR[RMII_MODE]	RCR[RGMII_EN]
MII - 10 Mbit/s	0	—	0	0
MII - 100 Mbit/s	0	—	0	0
RMII - 10 Mbit/s	0	1	1	0
RMII - 100 Mbit/s	0	0	1	0
RGMII - 10 Mbit/s	0	1	0	1
RGMII - 100 Mbit/s	0	0	0	1
RGMII - 1000 Mbit/s	1	—	0	1

11.5.6.18.1 RMII interface

In RMII receive mode, for normal reception following assertion of CRS_DV, RXD[1:0] is 00 until the receiver determines that the receive event has a proper start-of-stream delimiter (SSD).

The preamble appears (RXD[1:0]=01) and the MACs begin capturing data following detection of SFD.

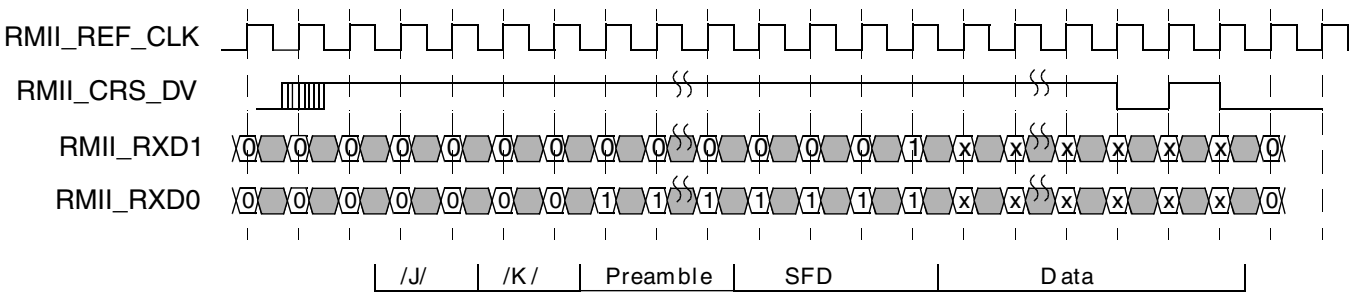


Figure 11-88. RMII receive operation

If a false carrier is detected (bad SSD), then RXD[1:0] is 10 until the end of the receive event. This is a unique pattern since a false carrier can only occur at the beginning of a packet where the preamble is decoded (RXD[1:0] = 01).

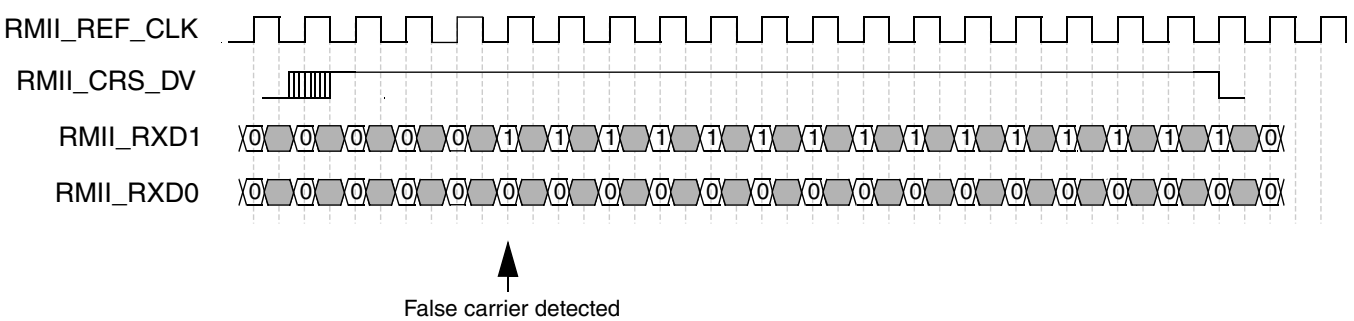


Figure 11-89. RMII receive operation with false carrier

In RMII transmit mode, TXD[1:0] provides valid data for each REF_CLK period while TXEN is asserted.

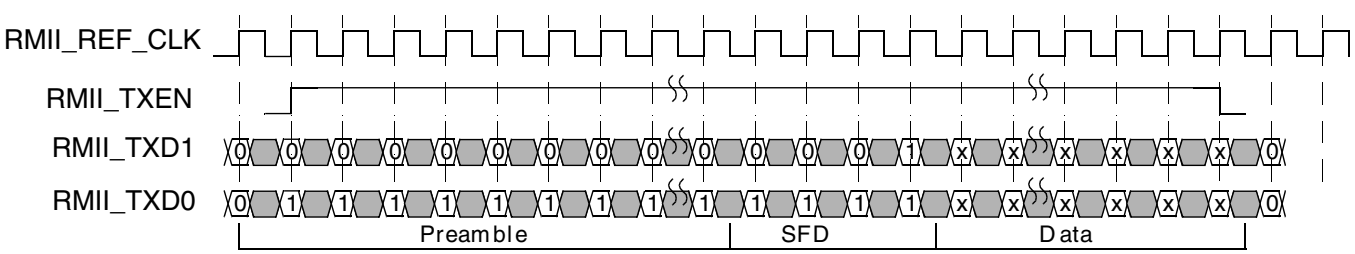


Figure 11-90. RMII transmit operation

11.5.6.18.2 RGMII interface

In RGMII modes, the data and control information is multiplexed by taking advantage of both edges of the reference clocks.

The data signals contain the lower four data bits on the rising edge and the upper four bits on the falling edge. The control signals are multiplexed into a single clock cycle using the same technique.

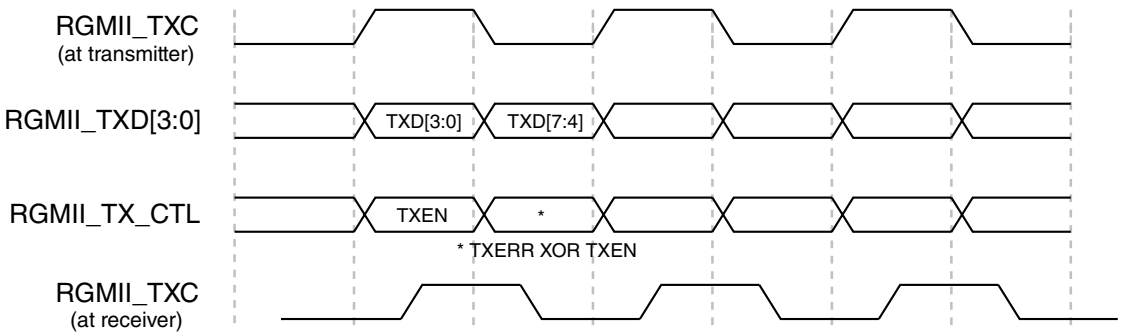


Figure 11-91. RGMII transmit operation

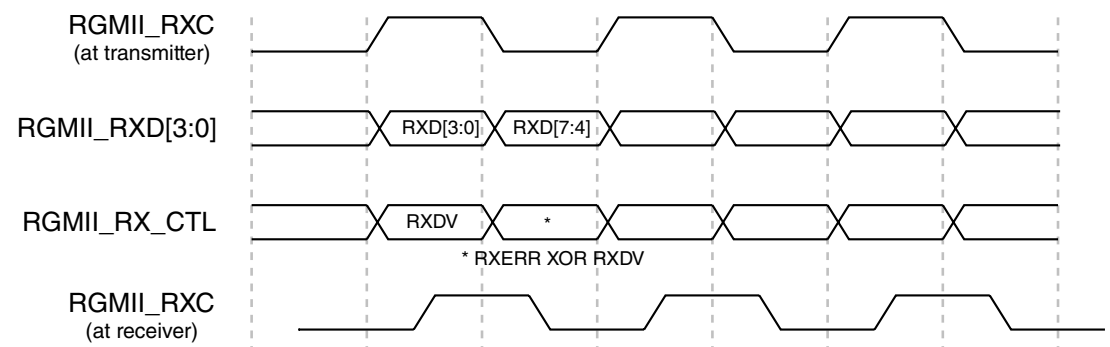


Figure 11-92. RGMII receive operation

11.5.6.18.3 MII Interface — transmit

On transmit, all data transfers are synchronous to MII_TXCLK rising edge. The MII data enable signal MII_TXEN is asserted to indicate the start of a new frame, and remains asserted until the last byte of the frame is present on the MII_TXD[3:0] bus.

Between frames, MII_TXEN remains deasserted.

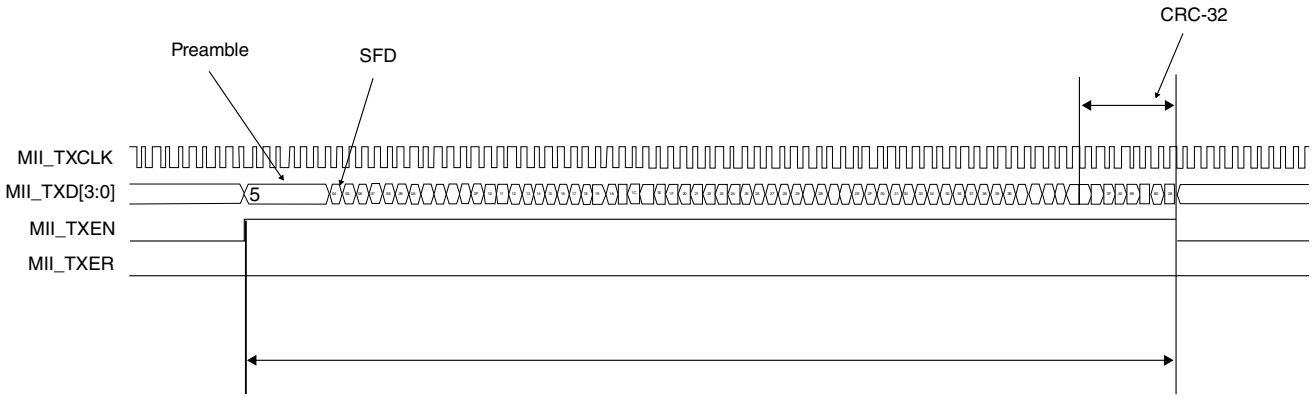


Figure 11-93. MII transmit operation

If a frame is received on the FIFO interface with an error (for example, RxBD[ME] set) the frame is subsequently transmitted with the MII_TXER error signal for one clock cycle at any time during the packet transfer.

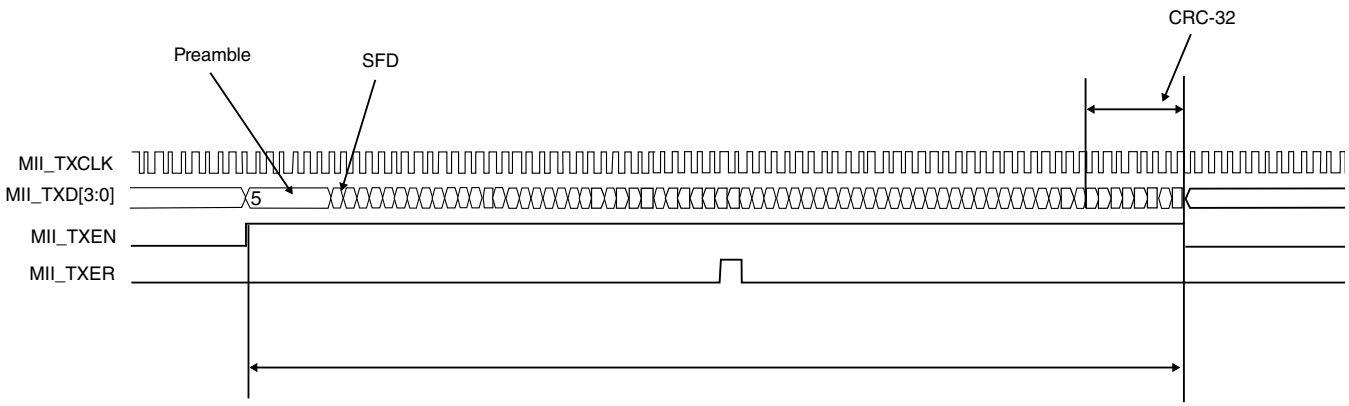


Figure 11-94. MII transmit operation — errored frame

11.5.6.18.3.1 Transmit with collision — half-duplex

When a collision is detected during a frame transmission (MII_COL asserted), the MAC stops the current transmission, sends a 32-bit jam pattern, and re-transmits the current frame.

(See Collision detection in half-duplex mode for details)

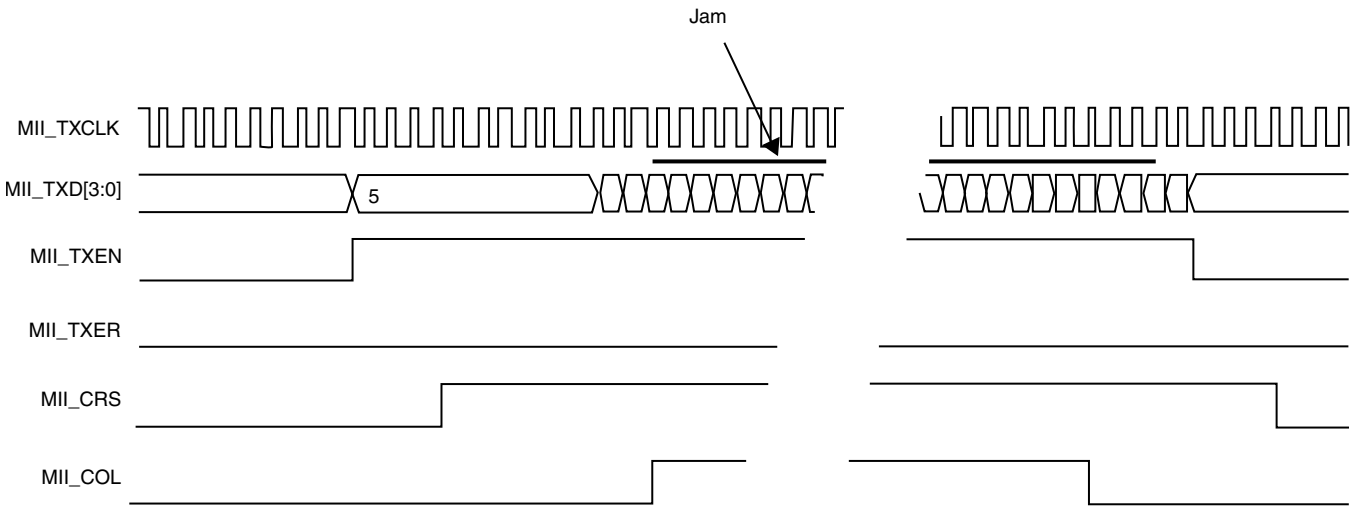


Figure 11-95. MII transmit operation — transmission with collision

11.5.6.18.4 MII interface — receive

On receive, all signals are sampled on the MII_RXCLK rising edge. The MII data enable signal, MII_RXDV, is asserted by the PHY to indicate the start of a new frame and remains asserted until the last byte of the frame is present on MII_RXD[3:0] bus.

Between frames, MII_RXDV remains deasserted.

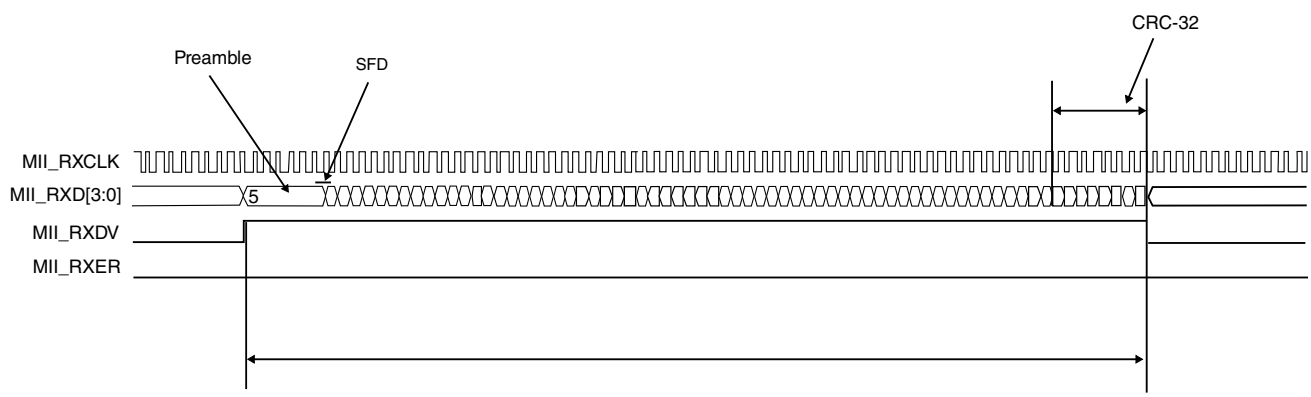


Figure 11-96. MII receive operation

If the PHY detects an error on the frame received from the line, the PHY asserts the MII error signal, MII_RXER, for at least one clock cycle at any time during the packet transfer.

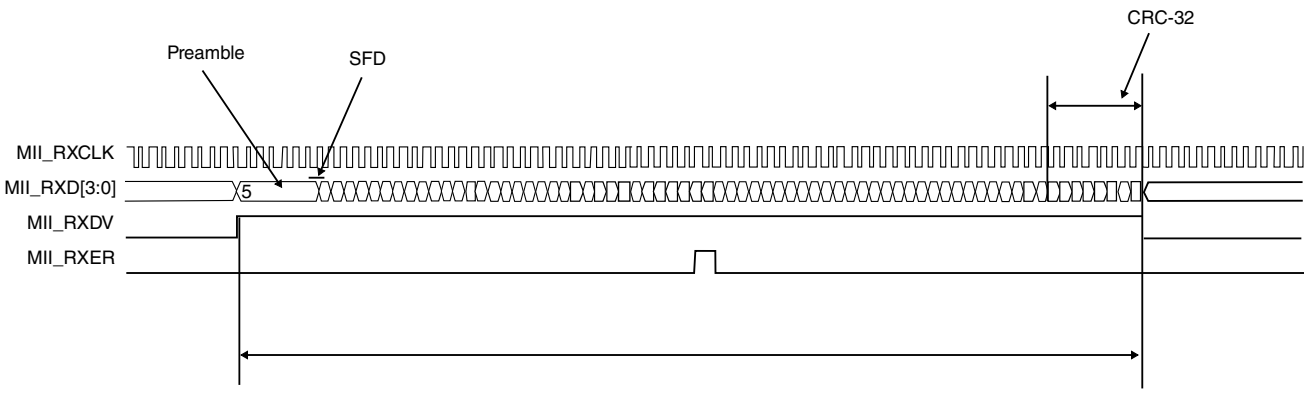


Figure 11-97. MII receive operation — errored frame

A frame received on the MII interface with a PHY error indication is subsequently transferred on the FIFO interface with RxBD[ME] set.

11.5.6.19 AVB configuration

The following steps give an example of how to initialize the ENET module for AVB.

1. Set up ENET_QOS:
   - Set TX_SCHEME to 000b, credit-based scheme.
   - Set RX_FLUSH0 to 1, enable RX flush for ring 0.
2. Set up TX BD ring and RX BD ring for each queue, 0-2.
   - Program ENET_MRBR, ENET_TDSR, and ENET_RDSR.
   - Program ENET_MRBR1, ENET_TDSR1, and ENET_RDSR1.
   - Program ENET_MRBR2, ENET_TDSR2, and ENET_RDSR2.
   - Program each TX and RX BD ring queue for all classes used in memory.

NOTE

If using credit-based scheme, ensure that enhanced transmit buffer descriptor FTYPE field matches BD ring queue, for example:
- FTYPE = 0h corresponds to ENET_TDSR
- FTYPE = 1h corresponds to ENET_TDSR1
- FTYPE = 2h corresponds to ENET_TDSR2

3. Program ENET_DMA1CFG and ENET_DMA2CFG for class 1 and class 2 corresponding to BD ring queue 1 and 2. DMA_CLASS_EN must be set to one for

that class to be enabled. See DMA Class Based Configuration (ENET_DMA*n*CFG) for information on how to program IDLE_SLOPE.

4. Program ENET_RCMR1 and ENET_RCMR2 for class 1 and class 2 matching for receive.

NOTE

Even if a match occurs, if ENET_DMA*n*CFG[DMA_CLASS_EN] is zero for the corresponding class, the RX frame will be automatically forwarded to BD ring queue 0.

5. Program the other ENET registers according to application requirements.
6. Program ENET_RDAR, ENET_RDAR1, ENET_RDAR2, ENET_TDAR, ENET_TDAR1, and ENET_TDAR2 to 0100_0000h according to the classes used.
7. Set ENET_ECR[ETHEREN] to one.

11.5.6.20 Interrupt coalescence

The purpose of the interrupt coalescing is to reduce the number of interrupts generated by the MAC so as to reduce the CPU loading.

To facilitate this interrupt coalescing for each queue, these registers are available with the same control and configuration fields.

- Transmit Interrupt Coalescing Register (ENET_TXIC*n*) where n=0,1,2 for queue/class 0,1,2.
- Receive Interrupt Coalescing Register (ENET_RXIC*n*) where n=0,1,2 for queue/class 0,1,2.

When coalescing is enabled by asserting the corresponding ICEN field and such interrupt is also enabled by the corresponding interrupt mask of the EIMR register, the MAC generates an interrupt when the threshold number of frames is reached (defined by ICFT) or when the threshold timer expires (defined by ICTT).

When coalescing is disabled by de-asserting ICEN, but interrupt is enabled by the corresponding interrupt mask of the EIMR register, the MAC generates an interrupt as they are received without using coalescing. Interrupt coalescing is done for each transmit and receive queue/class independently.

11.5.6.20.1 Interrupt coalescence setup

Interrupt coalescence supports both legacy and enhanced BDs. The following guidelines are recommended when setting up interrupt coalescence.

- When the MAC is configured for enhanced (IEEE 1588) mode, that is, enhanced BDs:
  - Set the INT bit in the enhanced received buffer descriptor to one.
  - Set the INT bit in the enhanced transmit buffer descriptor(s) to one.
- Clear the RXB, RXB1, and RXB2 fields in the EIMR register.
- Clear the TXB, TXB1, and TXB2 fields in the EIMR register.

11.5.6.20.2 Updating the frame count threshold on-the-fly

To update the ICFT field in the RXIC*n* and TXIC*n* registers:

1. Disable interrupt coalescence by clearing the appropriate ICEN field. This will allow the internal interrupt coalescence counter to reset to zero.

NOTE

When disabling interrupt coalescence, if an interrupt event is pending, that is, the interrupt counter is not zero, then an interrupt will occur.

2. Write the new threshold value to the ICFT field.
3. Set ICEN to one.

NOTE

The ICFT field can be updated on-the-fly without disabling the ICEN field. The hardware interrupt will continue and there is a possibility that an interrupt will occur depending on the state of the hardware counter and the previous ICFT value.

11.5.6.20.3 Updating the timer threshold on-the-fly

To update the ICTT field in the RXIC*n* and TXIC*n* registers:

1. Disable interrupt coalescence by clearing the appropriate ICEN field. This will allow the internal interrupt coalescence counter to reset to zero.

NOTE

When disabling interrupt coalescence, if an interrupt event is pending, that is, the interrupt counter is not zero, then an interrupt will occur.

2. Write the new timer value to the ICTT field.
3. Set ICEN to one.

Chapter 12
Timers

12.1 General Purpose Timer (GPT)

12.1.1 Overview

This chapter describes the General Purpose Timer (GPT) module interface. It is also a reference for software driver programming. The GPT has a 32-bit up-counter. The timer counter value can be captured in a register using an event on an external pin. The capture trigger can be programmed to be a rising or/and falling edge. The GPT can also generate an event on the output compare pins and an interrupt when the timer reaches a programmed value. The GPT has a 12-bit prescaler, which provides a programmable clock frequency derived from multiple clock sources.

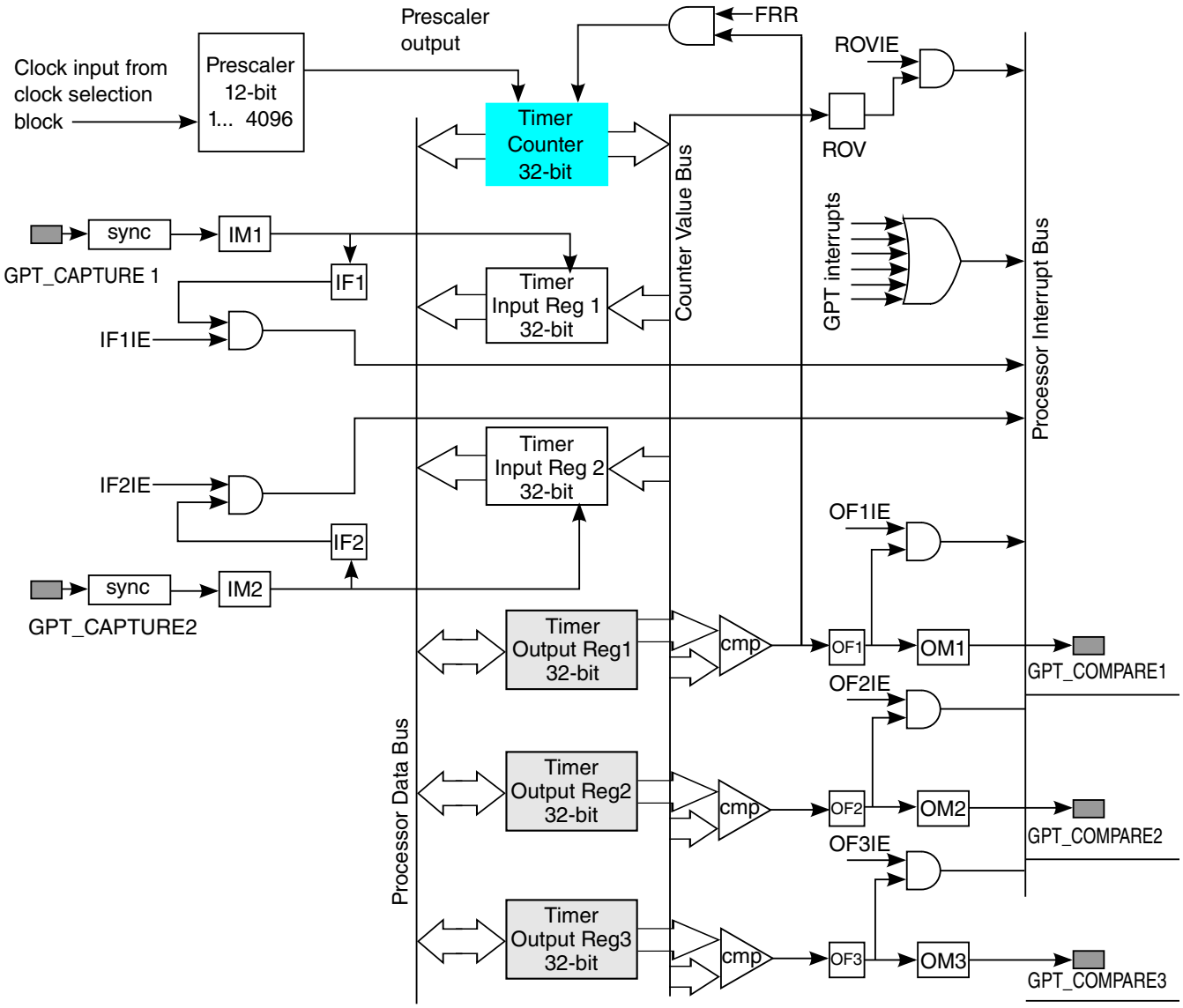


Figure 12-1. GPT Block Diagram

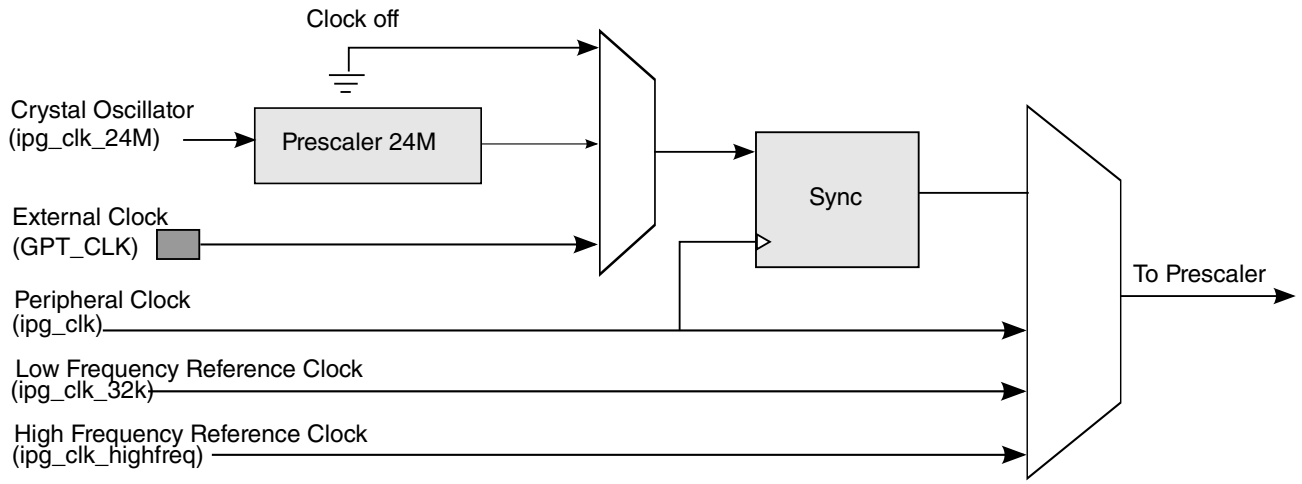


Figure 12-2. GPT Counter Clocks Diagram

12.1.1.1 Features

- One 32-bit up-counter with clock source selection, including external clock.
- Two input capture channels with a programmable trigger edge.
- Three output compare channels with a programmable output mode. A "forced compare" feature is also available.
- Can be programmed to be *active* in low power and debug modes.
- Interrupt generation at capture, compare, and rollover events.
- Restart or free-run modes for counter operations.

12.1.1.2 Modes and Operation

The GPT supports the modes described in the indicated sections:

- Operating Modes
  - Restart Mode
  - Free-Run Mode

12.1.2 Functional Description

This section provides a complete functional description of the GPT.

12.1.2.1 Operating Modes

The GPT counter can be programmed to work in either of two modes: Restart mode or Free-Run mode.

12.1.2.1.1 Restart Mode

In Restart mode (selectable through the GPT Control Register GPT_CR), when the counter reaches the compared value, the counter resets and starts again from 0x00000000. The Restart feature is associated only with Compare Channel 1.

Any write access to the Compare register of Channel 1 will reset the GPT counter. This is done to avoid possibly missing a compare event when compare value is changed from a higher value to lower value while counting is proceeding.

For the other two compare channels, when the compare event occurs the counter is *not reset*.

12.1.2.1.2 Free-Run Mode

In Free-Run mode, when compare events occur for all 3 channels, the counter is *not reset*; instead the counter continues to count until 0xffffffff, and then rolls over (to 0x00000000).

12.1.2.2 Operation

The General Purpose Timer (GPT) has a single counter (GPT_CNT) that is a 32-bit free-running *up-counter,* which starts counting *after it is enabled by software* (EN=1). The counter's clock source is the output of the prescaler labelled "Prescaler output" in Figure 12-1.

- If the GPT timer is disabled (EN=0), then the Main Counter *and* Prescaler Counter freeze their current count values. The ENMOD bit determines the value of the GPT counter when the EN bit is set and the Counter is enabled again.
  - If the ENMOD bit is set (=1), then the Main Counter and Prescaler Counter values are reset to 0, when GPT is enabled (EN=1).
  - If ENMOD bit is programmed to 0, then the Main Counter and Prescaler Counter restart counting from their frozen values, when GPT is enabled again (EN=1).
- If GPT is programmed to be disabled in a low power mode (STOP/WAIT), then the Main Counter and Prescaler Counter freeze at their current count values *when* GPT enters low power mode. When GPT exits a low power mode, the Main Counter and Prescaler Counter start counting from their frozen values *regardless* of the ENMOD

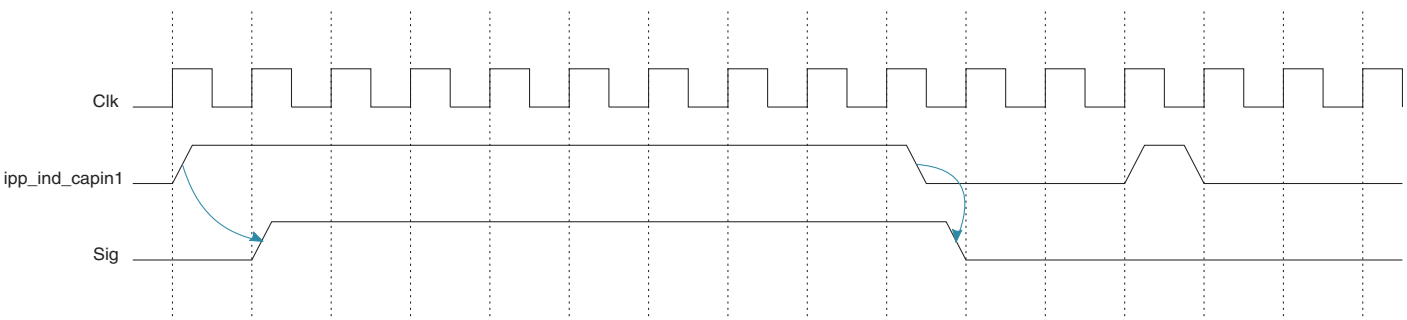
bit value. Note that the GPT_CNT can be read *at any time* by the processor, and that *both* Input Capture Channels use the *same* counter (GPT_CNT).

- A hardware reset resets all the GPT registers to their respective reset values. All registers except the Output Compare Registers (OCR1, OCR2, OCR3) obtain a value of 0x0. The Compare registers are reset to 0xFFFF_FFFF.
- The software reset (SWR bit in the GPT_CR control register) resets *all* of the register bits *except* the EN, ENMOD, STOPEN, WAITEN, and DBGEN bits. The state of these bits is not affected by a software reset. Note that a software reset can be given *while the GPT is disabled*.

12.1.2.2.1 Input Capture

There are two Input Capture Channels, and each Input Capture Channel has a dedicated capture pin, capture register and input edge detection/selection logic. Each input capture function has an associated status flag, and can cause the processor to make an interrupt service request.

When a selected edge transition occurs on an Input Capture pin, the contents of the GPT_CNT is captured on the corresponding capture register and the appropriate interrupt status flag is set. An interrupt request can be generated when the transition is detected *if* its corresponding enable bit is set (in the Interrupt Register). The capture can be programmed to occur on the input pin's rising edge, falling edge, on both rising and falling edges, or the capture can be disabled. The events are synchronized with the clock that was selected to run the counter. Only those transitions that occur at least one clock cycle (clock selected to run the counter) *after* the previous recorded transition will be guaranteed to trigger a capture event. There can be up to one clock cycle of uncertainty in the latching of the input transition. The Input Capture registers can be read *at any time* without affecting their values.



Clk = The clock selected from CLKSRC bit field setting

ipp_ind_capin1 = Pad signal for capture channel 1

Sig = Capture signal as sensed by the module

Figure 12-3. Input Capture Event Timing

12.1.2.2.2 Output Compare

The three Output Compare Channels *use the same counter* (GPT_CNT) as the Input Capture Channels. When the programmed content of an Output Compare register matches the value in GPT_CNT, an output compare status flag is set and an interrupt is generated (if the corresponding bit is set in the interrupt register). Consequently, the Output Compare timer pin will be set, cleared, toggled, not affected at all or provide an active-low pulse for one input clock period (subject to the restriction on the maximum frequency allowed on the pad) according to the mode bits (that were programmed).

There is also a "forced-compare" feature that allows the software to generate a compare event when required, *without the condition of the counter value that is equal to the compare value*. The action taken as a result of a forced compare is the same as when an output compare match occurs, *except that the status flags are not set and no interrupt can be generated*. Forced channels take programmed action immediately after the write to the force-compare bits. These bits are self-negating and always read as zeros.

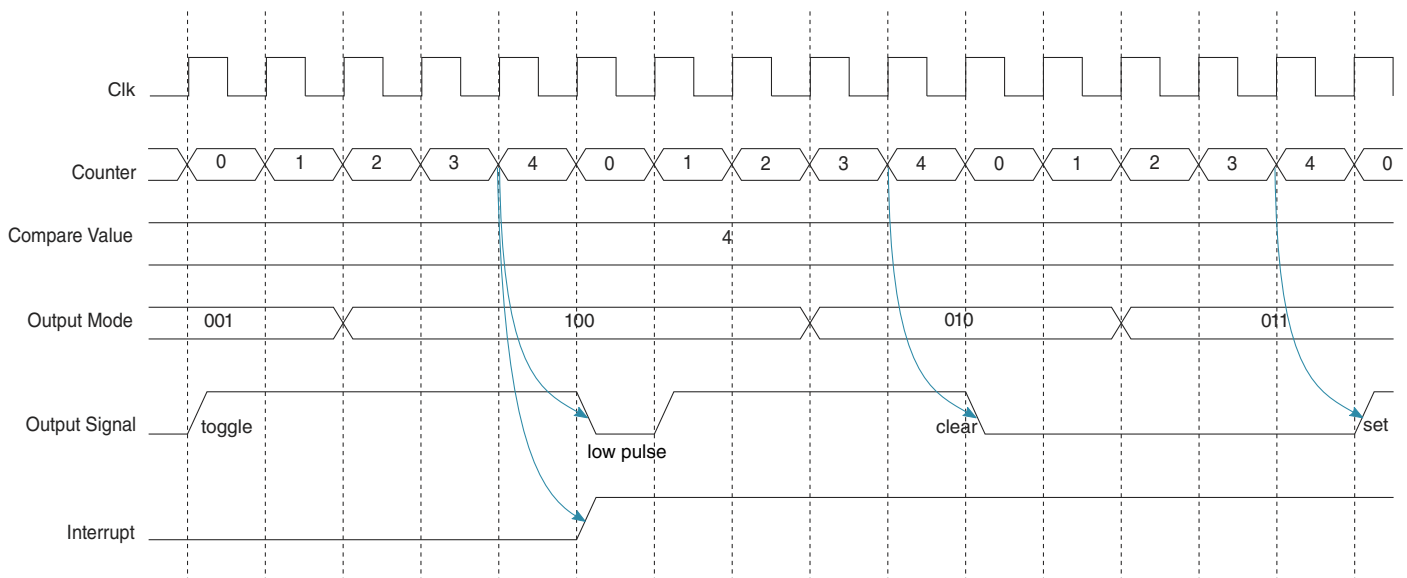


Figure 12-4. Output Compare and Interrupt Timing

12.1.2.2.3 Interrupts

There are 6 different interrupts that are generated by the GPT. If the selected clock for running the counter is available, then *all interrupts can be generated in Low Power and Debug modes*.

- Rollover Interrupt

The Rollover Interrupt is generated when the GPT counter reaches 0xffffffff, then resets to 0x00000000 and continues counting. The Rollover Interrupt is enabled by the ROVIE bit in the GPT_IR register; the associated status bit is the ROV bit in the GPT_SR register.

- Input Capture Interrupt 1, 2

  After a capture event occurs, the associated Input Capture Channel generates an interrupt. The "capture event" interrupts are enabled by the IF2IE and IF1IE bits (in the GPT_IR register); the associated status bits are IF2 and IF1 (in the GPT_SR register). The capture of the counter value because of a capture event is *not affected by a pending capture interrupt*. The Capture register is updated with a new counter value when a capture event occurs, regardless of whether that Capture Channels' interrupt has been serviced or not.

- Output Compare Interrupt 1, 2, 3

  After a compare event occurs, the associated Output Compare Channel generates an interrupt. The "compare event" interrupts are enabled by the OF3IE, OF2IE, and OF1IE bits (in the GPT_IR register); the associated status bits are OF3, OF2, and OF1 (in the GPT_SR register). A "forced compare" does not generate an interrupt.

A *cumulative* interrupt line is also present, which is asserted whenever any of the above interrupts are posted. The cumulative interrupt line has *no* associated enables or status bits.

12.1.2.2.4 Low Power Mode Behavior

In Low Power modes, if the clock from the selected clock source is available (except for the External Clock (GPT_CLK), which can be used *only if* the Peripheral Clock (ipg_clk) is available), the counter will continue to run depending on whether the control bit for that mode is set. If the clock is not present or if the corresponding low power bit in the GPT_CR control register is 0, the Main Counter and the Prescaler Counter freeze at their current values and resume counting (from their frozen values) when the Low Power mode is exited.

12.1.2.2.5 Debug Mode Behavior

In Debug mode, the modules in the device have the option of continuing to run or be halted.

- If the DBGEN bit is set, then the GPT timer will continue to run in Debug mode.
- If the DBGEN bit is not set (in the GPT_CR control register), then the GPT timer is halted.

12.1.3 Initialization/ Application Information

12.1.3.1 Selecting the Clock Source

The CLKSRC field in the GPT_CR register selects the clock source. The CLKSRC field value should be changed only after disabling the GPT (EN=0).

The software sequence to be followed while changing clock source is:

1. Disable GPT by setting EN=0 in GPT_CR register.
2. Disable GPT interrupt register (GPT_IR).
3. Configure Output Mode to unconnected/ disconnected—Write zeros in OM3, OM2, and OM1 in GPT_CR
4. Disable Input Capture Modes—Write zeros in IM1 and IM2 in GPT_CR
5. Change clock source CLKSRC to the desired value in GPT_CR register.
6. Assert the SWR bit in GPT_CR register.
7. Clear GPT status register (GPT_SR) (i.e., w1c).
8. Set ENMOD=1 in GPT_CR register, to bring GPT counter to 0x00000000.
9. Enable GPT (EN=1) in GPT_CR register.
10. Enable GPT interrupt register (GPT_IR).

12.1.4 GPT Memory Map/Register Definition

The GPT has 10 user-accessible 32-bit registers, which are used to configure, operate, and monitor the state of the GPT.

An IP bus write access to the GPT Control Register (GPT_CR) and the GPT Output Compare Register1 (GPT_OCR1) results in *one cycle of wait state*, while other valid IP bus accesses incur 0 wait states.

Irrespective of the Response Select signal value, a Write access to the GPT Status Registers (Read-only registers GPT_ICR1, GPT_ICR2, GPT_CNT) will generate a bus exception.

- If the Response Select signal is driven Low, then the Read/Write access to the *unimplemented* address space of GPT (*ips_addr* is greater than or equal to $BASE + $028) will generate a bus exception.
- If the Response Select is driven High, then the Read/Write access to the unimplemented address space of GPT will *not* generate any error response (like a bus exception).

GPT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302D_0000	GPT Control Register (GPT1_CR)	32	R/W	0000_0000h	12.1.4.1/ 3858
302D_0004	GPT Prescaler Register (GPT1_PR)	32	R/W	0000_0000h	12.1.4.2/ 3862
302D_0008	GPT Status Register (GPT1_SR)	32	R/W	0000_0000h	12.1.4.3/ 3863
302D_000C	GPT Interrupt Register (GPT1_IR)	32	R/W	0000_0000h	12.1.4.4/ 3864
302D_0010	GPT Output Compare Register 1 (GPT1_OCR1)	32	R/W	FFFF_FFFFh	12.1.4.5/ 3865
302D_0014	GPT Output Compare Register 2 (GPT1_OCR2)	32	R/W	FFFF_FFFFh	12.1.4.6/ 3866
302D_0018	GPT Output Compare Register 3 (GPT1_OCR3)	32	R/W	FFFF_FFFFh	12.1.4.7/ 3866
302D_001C	GPT Input Capture Register 1 (GPT1_ICR1)	32	R	0000_0000h	12.1.4.8/ 3867
302D_0020	GPT Input Capture Register 2 (GPT1_ICR2)	32	R	0000_0000h	12.1.4.9/ 3867
302D_0024	GPT Counter Register (GPT1_CNT)	32	R	0000_0000h	12.1.4.10/ 3868
302E_0000	GPT Control Register (GPT2_CR)	32	R/W	0000_0000h	12.1.4.1/ 3858
302E_0004	GPT Prescaler Register (GPT2_PR)	32	R/W	0000_0000h	12.1.4.2/ 3862
302E_0008	GPT Status Register (GPT2_SR)	32	R/W	0000_0000h	12.1.4.3/ 3863
302E_000C	GPT Interrupt Register (GPT2_IR)	32	R/W	0000_0000h	12.1.4.4/ 3864
302E_0010	GPT Output Compare Register 1 (GPT2_OCR1)	32	R/W	FFFF_FFFFh	12.1.4.5/ 3865
302E_0014	GPT Output Compare Register 2 (GPT2_OCR2)	32	R/W	FFFF_FFFFh	12.1.4.6/ 3866
302E_0018	GPT Output Compare Register 3 (GPT2_OCR3)	32	R/W	FFFF_FFFFh	12.1.4.7/ 3866
302E_001C	GPT Input Capture Register 1 (GPT2_ICR1)	32	R	0000_0000h	12.1.4.8/ 3867

Table continues on the next page...

GPT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
302E_0020	GPT Input Capture Register 2 (GPT2_ICR2)	32	R	0000_0000h	12.1.4.9/ 3867
302E_0024	GPT Counter Register (GPT2_CNT)	32	R	0000_0000h	12.1.4.10/ 3868
302F_0000	GPT Control Register (GPT3_CR)	32	R/W	0000_0000h	12.1.4.1/ 3858
302F_0004	GPT Prescaler Register (GPT3_PR)	32	R/W	0000_0000h	12.1.4.2/ 3862
302F_0008	GPT Status Register (GPT3_SR)	32	R/W	0000_0000h	12.1.4.3/ 3863
302F_000C	GPT Interrupt Register (GPT3_IR)	32	R/W	0000_0000h	12.1.4.4/ 3864
302F_0010	GPT Output Compare Register 1 (GPT3_OCR1)	32	R/W	FFFF_FFFFh	12.1.4.5/ 3865
302F_0014	GPT Output Compare Register 2 (GPT3_OCR2)	32	R/W	FFFF_FFFFh	12.1.4.6/ 3866
302F_0018	GPT Output Compare Register 3 (GPT3_OCR3)	32	R/W	FFFF_FFFFh	12.1.4.7/ 3866
302F_001C	GPT Input Capture Register 1 (GPT3_ICR1)	32	R	0000_0000h	12.1.4.8/ 3867
302F_0020	GPT Input Capture Register 2 (GPT3_ICR2)	32	R	0000_0000h	12.1.4.9/ 3867
302F_0024	GPT Counter Register (GPT3_CNT)	32	R	0000_0000h	12.1.4.10/ 3868
306E_0000	GPT Control Register (GPT6_CR)	32	R/W	0000_0000h	12.1.4.1/ 3858
306E_0004	GPT Prescaler Register (GPT6_PR)	32	R/W	0000_0000h	12.1.4.2/ 3862
306E_0008	GPT Status Register (GPT6_SR)	32	R/W	0000_0000h	12.1.4.3/ 3863
306E_000C	GPT Interrupt Register (GPT6_IR)	32	R/W	0000_0000h	12.1.4.4/ 3864
306E_0010	GPT Output Compare Register 1 (GPT6_OCR1)	32	R/W	FFFF_FFFFh	12.1.4.5/ 3865
306E_0014	GPT Output Compare Register 2 (GPT6_OCR2)	32	R/W	FFFF_FFFFh	12.1.4.6/ 3866
306E_0018	GPT Output Compare Register 3 (GPT6_OCR3)	32	R/W	FFFF_FFFFh	12.1.4.7/ 3866
306E_001C	GPT Input Capture Register 1 (GPT6_ICR1)	32	R	0000_0000h	12.1.4.8/ 3867
306E_0020	GPT Input Capture Register 2 (GPT6_ICR2)	32	R	0000_0000h	12.1.4.9/ 3867
306E_0024	GPT Counter Register (GPT6_CNT)	32	R	0000_0000h	12.1.4.10/ 3868

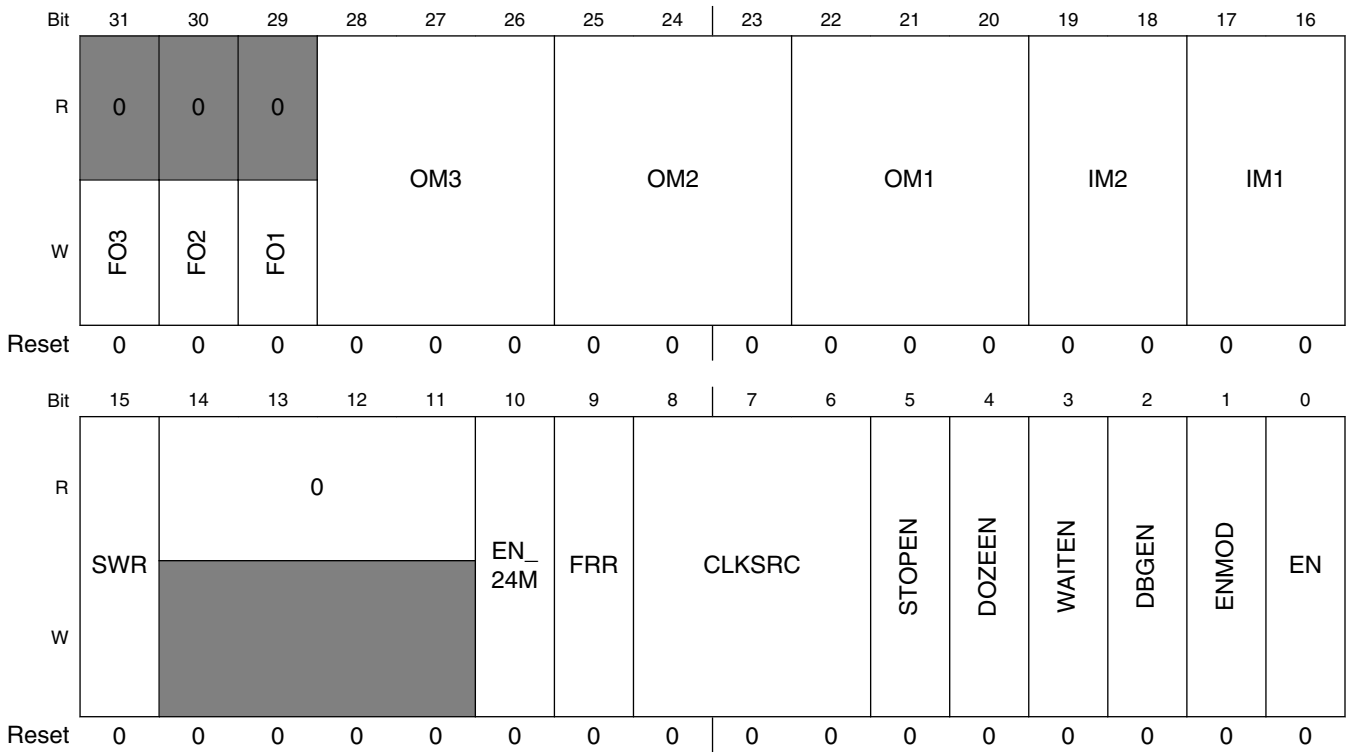
GPT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
306F_0000	GPT Control Register (GPT5_CR)	32	R/W	0000_0000h	12.1.4.1/ 3858
306F_0004	GPT Prescaler Register (GPT5_PR)	32	R/W	0000_0000h	12.1.4.2/ 3862
306F_0008	GPT Status Register (GPT5_SR)	32	R/W	0000_0000h	12.1.4.3/ 3863
306F_000C	GPT Interrupt Register (GPT5_IR)	32	R/W	0000_0000h	12.1.4.4/ 3864
306F_0010	GPT Output Compare Register 1 (GPT5_OCR1)	32	R/W	FFFF_FFFFh	12.1.4.5/ 3865
306F_0014	GPT Output Compare Register 2 (GPT5_OCR2)	32	R/W	FFFF_FFFFh	12.1.4.6/ 3866
306F_0018	GPT Output Compare Register 3 (GPT5_OCR3)	32	R/W	FFFF_FFFFh	12.1.4.7/ 3866
306F_001C	GPT Input Capture Register 1 (GPT5_ICR1)	32	R	0000_0000h	12.1.4.8/ 3867
306F_0020	GPT Input Capture Register 2 (GPT5_ICR2)	32	R	0000_0000h	12.1.4.9/ 3867
306F_0024	GPT Counter Register (GPT5_CNT)	32	R	0000_0000h	12.1.4.10/ 3868
3070_0000	GPT Control Register (GPT4_CR)	32	R/W	0000_0000h	12.1.4.1/ 3858
3070_0004	GPT Prescaler Register (GPT4_PR)	32	R/W	0000_0000h	12.1.4.2/ 3862
3070_0008	GPT Status Register (GPT4_SR)	32	R/W	0000_0000h	12.1.4.3/ 3863
3070_000C	GPT Interrupt Register (GPT4_IR)	32	R/W	0000_0000h	12.1.4.4/ 3864
3070_0010	GPT Output Compare Register 1 (GPT4_OCR1)	32	R/W	FFFF_FFFFh	12.1.4.5/ 3865
3070_0014	GPT Output Compare Register 2 (GPT4_OCR2)	32	R/W	FFFF_FFFFh	12.1.4.6/ 3866
3070_0018	GPT Output Compare Register 3 (GPT4_OCR3)	32	R/W	FFFF_FFFFh	12.1.4.7/ 3866
3070_001C	GPT Input Capture Register 1 (GPT4_ICR1)	32	R	0000_0000h	12.1.4.8/ 3867
3070_0020	GPT Input Capture Register 2 (GPT4_ICR2)	32	R	0000_0000h	12.1.4.9/ 3867
3070_0024	GPT Counter Register (GPT4_CNT)	32	R	0000_0000h	12.1.4.10/ 3868

12.1.4.1 GPT Control Register (GPTx_CR)

The GPT Control Register (GPT_CR) is used to program and configure GPT operations. An IP Bus Write to the GPT Control Register occurs after one cycle of wait state, while an IP Bus Read occurs after 0 wait states.

Address: Base address + 0h offset



GPTx_CR field descriptions

Field	Description
31 FO3	FO3 Force Output Compare Channel 3 FO2 Force Output Compare Channel 2 FO1 Force Output Compare Channel 1 The FO*n* bit causes the pin action *programmed* for the timer Output Compare *n* pin (according to the OM*n* bits in this register). • The OF*n* flag (OF3, OF2, OF1) in the status register is **not affected**. • This bit is self-negating and always read as zero. 0 Writing a 0 has no effect. 1 Causes the programmed pin action on the timer Output Compare *n* pin; the OF*n* flag is not set.
30 FO2	See F03

Table continues on the next page...

GPTx_CR field descriptions (continued)

Field	Description
29 FO1	See F03
28–26 OM3	OM3 (bits 28-26) controls the Output Compare Channel 3 operating mode. OM2 (bits 25-23) controls the Output Compare Channel 2 operating mode. OM1 (bits 22-20) controls the Output Compare Channel 1 operating mode. The OM*n* bits specify the response that a compare event will generate on the output pin of Output Compare Channel *n*. • The toggle, clear, and set options cause a change on the output pin *only* if a compare event occurs. • When OM*n* is programmed as 1xx (active low pulse), the output pin is set to one immediately on the next input clock; a low pulse (that is an input clock in width) occurs when there is a compare event. Note that here, "input clock" refers to the clock selected by the CLKSRC bits of the GPT Control Register. 000 Output disconnected. No response on pin. 001 Toggle output pin 010 Clear output pin 011 Set output pin 1xx Generate an active low pulse (that is one input clock wide) on the output pin.
25–23 OM2	See OM3
22–20 OM1	See OM3
19–18 IM2	IM2 (bits 19-18, Input Capture Channel 2 operating mode) IM1 (bits 17-16, Input Capture Channel 1 operating mode) The IM*n* bit field determines the transition on the input pin (for Input capture channel *n*), which will trigger a capture event. 00 capture disabled 01 capture on rising edge only 10 capture on falling edge only 11 capture on both edges
17–16 IM1	See IM2
15 SWR	Software reset. This is the software reset of the GPT module. It is a self-clearing bit. • The SWR bit is set when the module is in reset state. • The SWR bit is cleared when the reset procedure finishes. • Setting the SWR bit resets **all of the registers** to their default reset values, except for the EN, ENMOD, STOPEN, DOZEEN, WAITEN, and DBGEN bits in the GPT Control Register (this control register). 0 GPT is not in reset state 1 GPT is in reset state
14–11 Reserved	This read-only field is reserved and always has the value 0.
10 EN_24M	Enable 24 MHz clock input from crystal.

Table continues on the next page...

GPTx_CR field descriptions (continued)

Field	Description
	• A hardware reset resets the EN_24M bit. • A software reset *does not affect* the EN_24M bit. 0 24M clock disabled 1 24M clock enabled
9 FRR	Free-Run or Restart mode. The FFR bit determines the behavior of the GPT when a compare event in channel 1 occurs. • In Restart mode, after a compare event, the counter resets to 0x00000000 and resumes counting (after the occurrence of a compare event). • In Free-Run mode, after a compare event, the counter continues counting until 0xFFFFFFFF and then rolls over to 0. 0 Restart mode 1 Free-Run mode
8–6 CLKSRC	Clock Source select. The CLKSRC bits select which clock will go to the prescaler (and subsequently be used to run the GPT counter). • The CLKSRC bit field value should only be changed after disabling the GPT by clearing the EN bit in this register (GPT_CR). • A software reset does not affect the CLKSRC bit. 000 No clock 001 Peripheral Clock (ipg_clk) 010 High Frequency Reference Clock (ipg_clk_highfreq) 011 External Clock 100 Low Frequency Reference Clock (ipg_clk_32k) 101 Crystal oscillator as Reference Clock (ipg_clk_24M) others Reserved
5 STOPEN	GPT Stop Mode enable. The STOPEN read/write control bit enables GPT operation *during Stop mode*. • A hardware reset resets the STOPEN bit. • A software reset *does not affect* the STOPEN bit. 0 GPT is disabled in Stop mode. 1 GPT is enabled in Stop mode.
4 DOZEEN	GPT Doze Mode Enable. • A hardware reset resets the DOZEEN bit. • A software reset *does not affect* the DOZEEN bit. 0 GPT is disabled in doze mode. 1 GPT is enabled in doze mode.
3 WAITEN	GPT Wait Mode enable. The WAITEN read/write control bit enables GPT operation *during Wait mode*. • A hardware reset resets the WAITEN bit. • A software reset *does not affect* the WAITEN bit. 0 GPT is disabled in wait mode. 1 GPT is enabled in wait mode.

Table continues on the next page...

GPTx_CR field descriptions (continued)

Field	Description
2 DBGEN	GPT debug mode enable. The DBGEN read/write control bit enables GPT operation *during Debug mode*. • A hardware reset resets the DBGEN bit. • A software reset *does not affect* the DBGEN bit. 0 GPT is disabled in debug mode. 1 GPT is enabled in debug mode.
1 ENMOD	GPT Enable mode. When the GPT is disabled (EN=0), then both the Main Counter and Prescaler Counter *freeze their current count values*. The ENMOD bit determines the value of the GPT counter when Counter is enabled again (if the EN bit is set). • If the ENMOD bit is 1, then the Main Counter and Prescaler Counter values are reset to 0 after GPT is enabled (EN=1). • If the ENMOD bit is 0, then the Main Counter and Prescaler Counter restart counting *from their frozen values* after GPT is enabled (EN=1). • If GPT is programmed to be disabled in a low power mode (STOP/WAIT), then the Main Counter and Prescaler Counter *freeze at their current count values* when the GPT enters low power mode. • When GPT exits low power mode, the Main Counter and Prescaler Counter start counting from their frozen values, regardless of the ENMOD bit value. • Setting the SWR bit will clear the Main Counter and Prescaler Counter values, regardless of the value of EN or ENMOD bits. • A hardware reset resets the ENMOD bit. • A software reset *does not affect* the ENMOD bit. 0 GPT counter will retain its value when it is disabled. 1 GPT counter value is reset to 0 when it is disabled.
0 EN	GPT Enable. The EN bit is the GPT module enable bit. **Before setting the EN bit**, we recommend that *all registers be properly programmed*. • A hardware reset resets the EN bit. • A software reset *does not affect* the EN bit. 0 GPT is disabled. 1 GPT is enabled.

12.1.4.2 GPT Prescaler Register (GPTx_PR)

The GPT Prescaler Register (GPT_PR) contains bits that determine the *divide value* of the clock that runs the counter.

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PRESCALER24M				PRESCALER											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPTx_PR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–12 PRESCALER24M	Prescaler bits. 24M crystal clock is divided by [PRESCALER24M + 1] before selected by the CLKSRC field. If 24M crystal clock is not selected, this feild takes no effect. 0x0 Divide by 1 0x1 Divide by 2 ... ... 0xF Divide by 16
PRESCALER	Prescaler bits. The clock selected by the CLKSRC field is divided by [PRESCALER + 1], and then used to run the counter. • A change in the value of the PRESCALER bits cause the Prescaler counter to reset and a new count period to start immediately. • See Figure 1 for the timing diagram. 0x000 Divide by 1 0x001 Divide by 2 ... ... 0xFFF Divide by 4096

12.1.4.3 GPT Status Register (GPTx_SR)

The GPT Status Register (GPT_SR) contains bits that indicate that a counter has rolled over, and if any event has occurred on the Input Capture and Output Compare channels. The bits are cleared by writing a 1 to them.

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										ROV	IF2	IF1	OF3	OF2	OF1
W											w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPTx_SR field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.
5 ROV	Rollover Flag. The ROV bit indicates that the counter has reached its *maximum possible value* and *rolled over* to 0 (from which the counter continues counting). The ROV bit is only set if the counter has reached 0xFFFFFFFF in both Restart and Free-Run modes. 0 Rollover has not occurred. 1 Rollover has occurred.
4 IF2	IF2 Input capture 2 Flag IF1 Input capture 1 Flag The IF*n* bit indicates that a capture event has occurred on Input Capture channel *n*. 0 Capture event has not occurred. 1 Capture event has occurred.
3 IF1	See IF2
2 OF3	OF3 Output Compare 3 Flag OF2 Output Compare 2 Flag OF1 Output Compare 1 Flag The OF*n* bit indicates that a compare event has occurred on Output Compare channel *n*. 0 Compare event has not occurred. 1 Compare event has occurred.

Table continues on the next page...

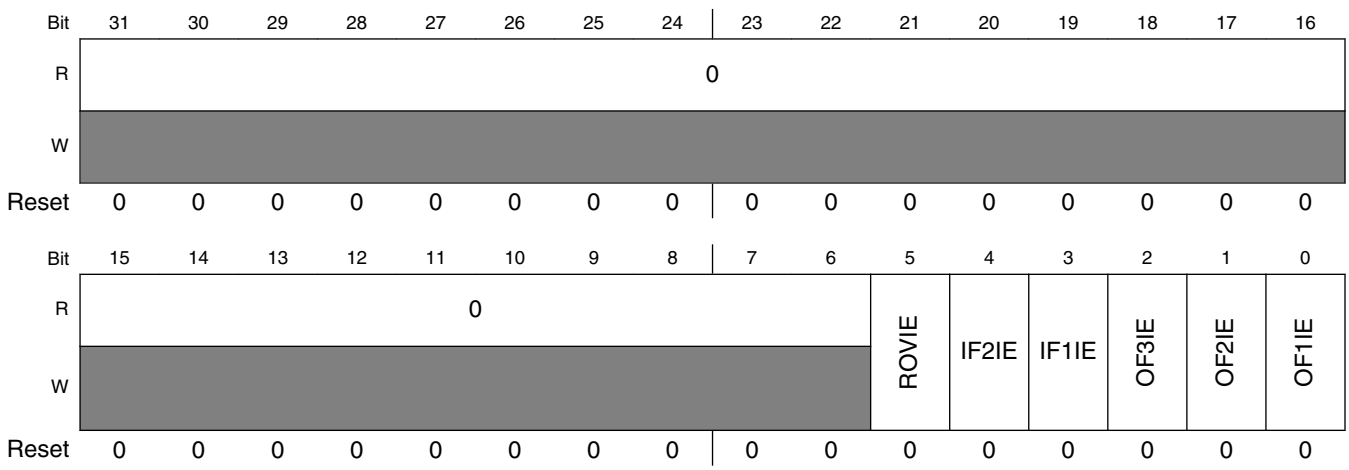
GPTx_SR field descriptions (continued)

Field	Description
1 OF2	See OF3
0 OF1	See OF3

12.1.4.4 GPT Interrupt Register (GPTx_IR)

The GPT Interrupt Register (GPT_IR) contains bits that control whether interrupts are generated after rollover, input capture and output compare events.

Address: Base address + Ch offset



GPTx_IR field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.
5 ROVIE	Rollover Interrupt Enable. The ROVIE bit controls the Rollover interrupt. 0 Rollover interrupt is disabled. 1 Rollover interrupt enabled.
4 IF2IE	IF2IE Input capture 2 Interrupt Enable IF1IE Input capture 1 Interrupt Enable The IF*n*IE bit controls the IF*n*IE Input Capture *n* Interrupt Enable. 0 IF2IE Input Capture *n* Interrupt Enable is disabled. 1 IF2IE Input Capture *n* Interrupt Enable is enabled.

Table continues on the next page...

GPTx_IR field descriptions (continued)

Field	Description
3 IF1IE	See IF2IE
2 OF3IE	OF3IE Output Compare 3 Interrupt Enable OF2IE Output Compare 2 Interrupt Enable OF1IE Output Compare 1 Interrupt Enable The OF*n*IE bit controls the Output Compare Channel *n* interrupt. 0 Output Compare Channel *n* interrupt is disabled. 1 Output Compare Channel *n* interrupt is enabled.
1 OF2IE	See OF3IE
0 OF1IE	See OF3IE

12.1.4.5 GPT Output Compare Register 1 (GPTx_OCR1)

The GPT Compare Register 1 (GPT_OCR1) holds the value that determines when a compare event will be generated on Output Compare Channel 1. Any write access to the Compare register of Channel 1 while in Restart mode (FRR=0) will reset the GPT counter.

An IP Bus Write access to the GPT Output Compare Register1 (GPT_OCR1) occurs *after* one cycle of wait state; an IP Bus Read access occurs *immediately* (0 wait states).

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COMP																															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

GPTx_OCR1 field descriptions

Field	Description
COMP	Compare Value. When the counter value equals the COMP bit field value, a compare event is generated on Output Compare Channel 1.

12.1.4.6 GPT Output Compare Register 2 (GPTx_OCR2)

The GPT Compare Register 2 (GPT_OCR2) holds the value that determines when a compare event will be generated on Output Compare Channel 2.

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COMP																															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

GPTx_OCR2 field descriptions

Field	Description
COMP	Compare Value. When the counter value equals the COMP bit field value, a compare event is generated on Output Compare Channel 2.

12.1.4.7 GPT Output Compare Register 3 (GPTx_OCR3)

The GPT Compare Register 3 (GPT_OCR3) holds the value that determines when a compare event will be generated on Output Compare Channel 3.

Address: Base address + 18h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COMP																															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

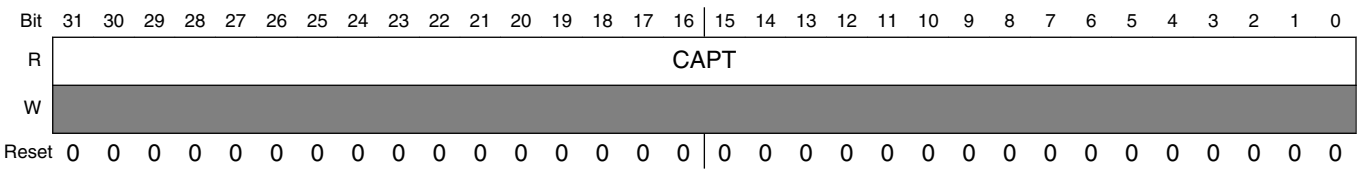
GPTx_OCR3 field descriptions

Field	Description
COMP	Compare Value. When the counter value equals the COMP bit field value, a compare event is generated on Output Compare Channel 3.

12.1.4.8 GPT Input Capture Register 1 (GPTx_ICR1)

The GPT Input Capture Register 1 (GPT_ICR1) is a read-only register that holds the value *that was in the counter during the last capture event* on Input Capture Channel 1.

Address: Base address + 1Ch offset



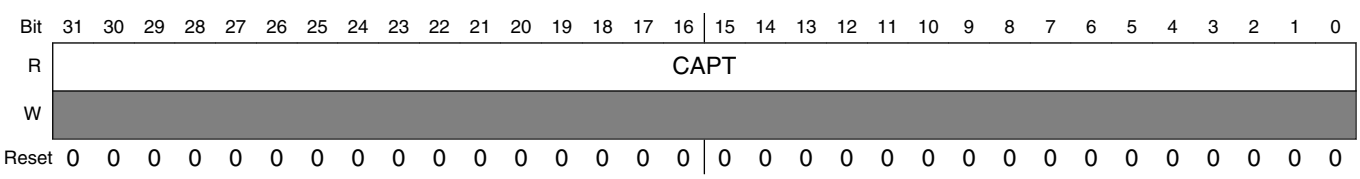
GPTx_ICR1 field descriptions

Field	Description
CAPT	Capture Value. After a capture event on Input Capture Channel 1 occurs, the current value of the counter is loaded into GPT Input Capture Register 1.

12.1.4.9 GPT Input Capture Register 2 (GPTx_ICR2)

The GPT Input capture Register 2 (GPT_ICR2) is a read-only register which holds the value that was in the counter during the last capture event on input capture channel 2.

Address: Base address + 20h offset



GPTx_ICR2 field descriptions

Field	Description
CAPT	Capture Value. After a capture event on Input Capture Channel 2 occurs, the current value of the counter is loaded into GPT Input Capture Register 2.

12.1.4.10 GPT Counter Register (GPTx_CNT)

The GPT Counter Register (GPT_CNT) is the main counter's register. GPT_CNT is a read-only register and can be read *without affecting the counting process* of the GPT.

Address: Base address + 24h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNT																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPTx_CNT field descriptions

Field	Description
COUNT	Counter Value. The COUNT bits show the current count value of the GPT counter.

12.2 Pulse Width Modulation (PWM)

12.2.1 Overview

The Pulse Width Modulation (PWM) has a 16-bit counter, and is optimized to generate sound from stored sample audio images and it can also generate tones. It uses 16-bit resolution and a 4 x 16 data FIFO.

This section presents an overview of the PWM. A block diagram of the PWM module is shown in the figure below.

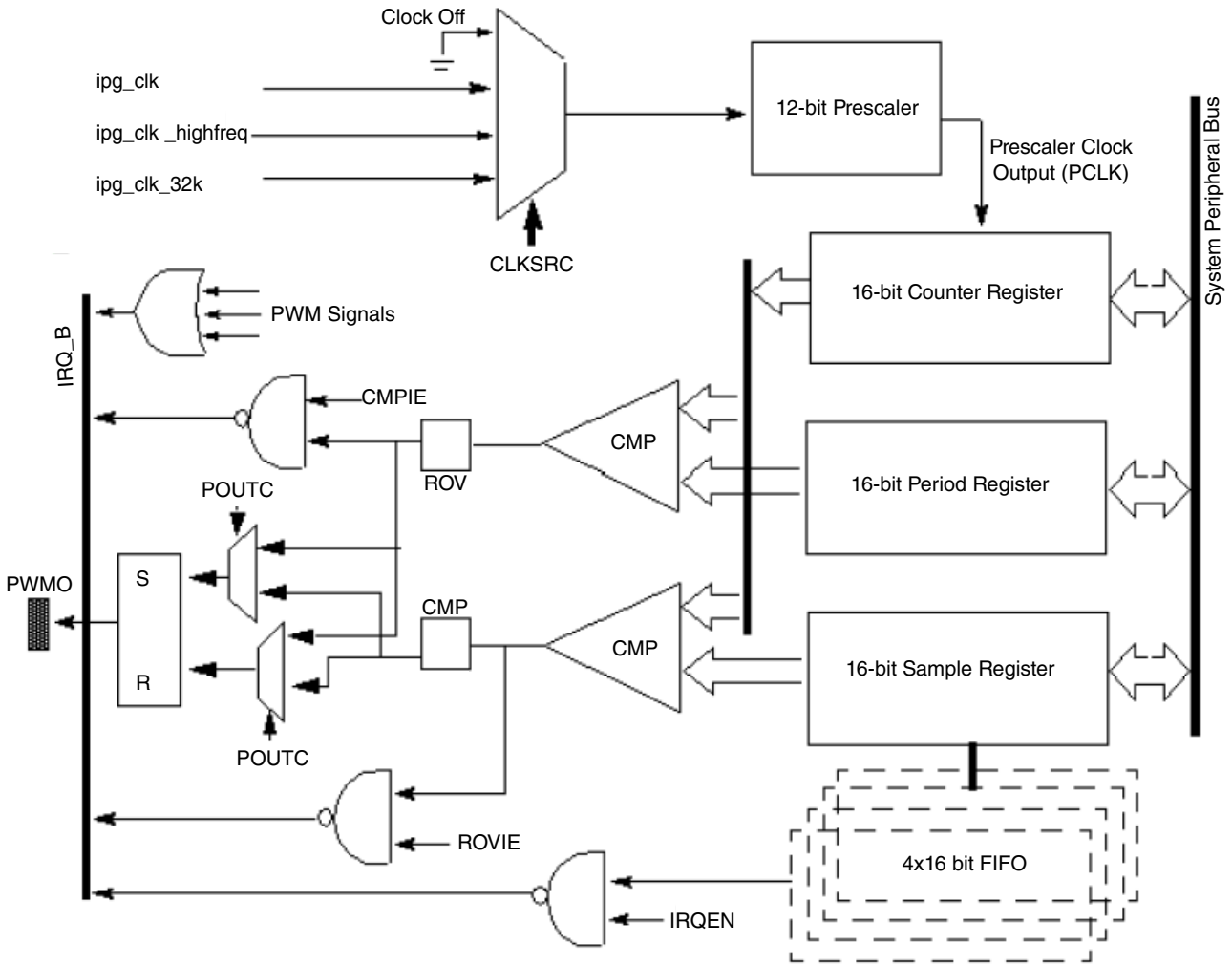


Figure 12-5. Pulse-Width Modulator Block Diagram

The following features characterize the PWM:

- 16-bit up-counter with clock source selection
- 4 x 16 FIFO to minimize interrupt overhead
- 12-bit prescaler for division of clock
- Sound and melody generation
- Active high or active low configured output
- Can be programmed to be active in low-power mode
- Can be programmed to be active in debug mode
- Interrupts at compare and rollover

12.2.2 Functional Description

The following sections detail the PWM operation and function.

12.2.2.1 Operation

The output of the PWM is a toggling signal whose frequency and duty cycle can be modulated by programming the appropriate registers. It has a 16-bit up counter which counts from 0x0000 until the counter value equals the PWM_PR + 1. After this match occurs the counter is reset to 0x0000.

At the beginning of a count period cycle, the PWMO pin is set to one (default) and the counter begins counting up from 0x0000. The sample value in the sample FIFO is compared on each count of prescaler clock. When the sample and count values match, the PWMO signal is cleared to zero (default). The counter continues counting until the period match occurs and subsequently another period cycle begins.

When the PWM is enabled, the counter starts running and generates an output with the reset values in the period and sample registers. It is recommended that the programming of these registers be done before PWM is enabled.

A hardware reset results in all the PWM count and sample registers being cleared and the FIFO being flushed. The control register shows that FIFO is empty and it can be written into, and the PWM is disabled. A software reset has the same results, however the state of the DBGEN, STOPEN, DOZEN, and WAITEN bits in the control register are not affected. Software reset can be asserted even when the PWM is in disabled state.

12.2.2.1.1 FIFO

Digital sample values can be loaded into the pulse-width modulator as 16-bit words. The endianess can be changed using the BCTR and HCTR bits of the control register. A 4-word (16-bit) FIFO minimizes interrupt overhead. A maskable interrupt is generated when the number of data words fall below the water level set by the FWM field in the control register.

A write to the PWM_SAR sample register results in the value being stored into the FIFO if it is not full. A write when the FIFO is full sets FWE (FIFO write error) bit in the status register and the FIFO contents remain unchanged. The FIFO can be written at any time, but can be read only when the PWM is enabled. The PWM_SR[FIFOAV] field shows how many data words are currently contained in the FIFO and whether or not it can be written into.

A read on the sample register yields the current FIFO value that is being used, or will be used, by the PWM for generation on the output signal. Therefore, a write and a subsequent read on the sample register may result in different values being obtained.

12.2.2.1.2 Rollover and Compare Event

The counter is reset to 0x0000 after its value equals the PWM_PR[PERIOD] + 1 and resumes counting thereafter. This event is referred to as a rollover. For example, if PWM_PR[PERIOD] = 0x0000, the counter is reset when it equals 0x0001. When PWM_PR[PERIOD] = 0xFFFF or 0xFFFE, the counter is reset when it equals 0xFFFF. For more information, see the PWM Period Register (PWM_PR) description.

During a rollover event the output is either set (default), reset or has no effect according to the programming of the POUTC field in the control register. This event can also generate an interrupt if the respective interrupt enable bit is set in the control register.

When the counter value reaches the sample value, the output of the PWM is reset (default), set or has no effect according to the programming of the POUTC field of control register. This event is referred to as a compare event. This event can also generate an interrupt if the respective interrupt enable bit is set in the control register.

If the rollover event sets the PWM output signal, the compare event will reset it and vice versa for a particular programming configuration of POUTC field.

12.2.2.1.3 Low Power Mode Behavior

In low power mode, if the clock from the selected clock source is available, the PWM counter continues to run and an output is produced, depending on whether the control bit for that mode is set or not. In the absence of the clock itself, or if the corresponding low power bit in the control register is 0, the counter is reset and resumes counting when it exits the low power mode.

12.2.2.1.4 Debug Mode Behavior

In debug mode, PWM has the option of continuing to run or be halted. If the DBGEN bit is not set in the PWM_PWMCR, the PWM is halted. If the DBGEN bit is set, then the PWM will continue to run in the debug mode.

12.2.3 Enable Sequence for the PWM

The sequence found here should be used to enable the PWM.

1. Configure the desired settings for the PWM Control Register (PWMx_PWMCR) while keeping the PWM disabled (PWMx_PWMCR[0]=0).
2. Enable the desired interrupts in the PWM Interrupt Register (PWMx_PWMIR).
3. One to three initial samples may be written to the PWM Sample Register (PWMx_PWMSAR). The initial sample values will be loaded into the PWM FIFO even if the PWM is not yet enabled. Do not write a 4th sample because the FIFO will become full and trigger a FIFO Write Error (FWE). This error will prevent the PWM from starting once it is enabled.
4. Check the FIFO Write Error status bit (FWE), the Compare status bit (CMP) and the Roll-over status bit (ROV) in the PWM Status Register (PWMx_PWMSR) to make sure they are all zero. Any non-zero status bits should be cleared by writing a 1 to them.
5. Write the desired period to the PWM Period Register (PWMx_PWMPR).
6. Enable the PWM by writing a 1 to the PWM Enable bit, PWMx_PWMCR[0], while maintaining the other register bits in their previously configured state.

12.2.4 Disable Sequence for the PWM

The PWM can be disabled at any time by clearing the PWM enable bit, PWMx_PWMCR[0] to 0.

Any data remaining in the FIFO will not be produced at the PWM output after the PWM has been disabled and will remain in the FIFO until the PWM is enabled again. A software reset (setting PWMx_PWMCR[3] to 1) or a hardware reset will clear the FIFO and any remaining data will be lost.

12.2.5 PWM Memory Map/Register Definition

The PWM includes six user-accessible 32-bit registers.

PWM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3066_0000	PWM Control Register (PWM1_PWMCR)	32	R/W	0000_0000h	12.2.5.1/ 3874
3066_0004	PWM Status Register (PWM1_PWMSR)	32	w1c	0000_0008h	12.2.5.2/ 3876

Table continues on the next page...

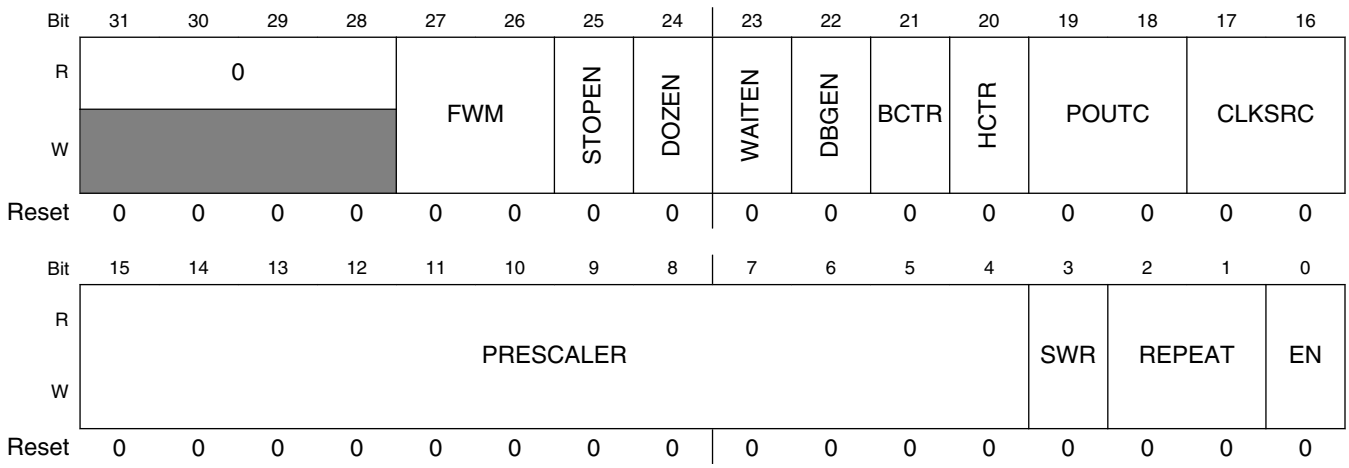
PWM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3066_0008	PWM Interrupt Register (PWM1_PWMIR)	32	R/W	0000_0000h	12.2.5.3/ 3877
3066_000C	PWM Sample Register (PWM1_PWMSAR)	32	R/W	0000_0000h	12.2.5.4/ 3878
3066_0010	PWM Period Register (PWM1_PWMPR)	32	R/W	0000_FFFEh	12.2.5.5/ 3879
3066_0014	PWM Counter Register (PWM1_PWMCNR)	32	R	0000_0000h	12.2.5.6/ 3879
3067_0000	PWM Control Register (PWM2_PWMCR)	32	R/W	0000_0000h	12.2.5.1/ 3874
3067_0004	PWM Status Register (PWM2_PWMSR)	32	w1c	0000_0008h	12.2.5.2/ 3876
3067_0008	PWM Interrupt Register (PWM2_PWMIR)	32	R/W	0000_0000h	12.2.5.3/ 3877
3067_000C	PWM Sample Register (PWM2_PWMSAR)	32	R/W	0000_0000h	12.2.5.4/ 3878
3067_0010	PWM Period Register (PWM2_PWMPR)	32	R/W	0000_FFFEh	12.2.5.5/ 3879
3067_0014	PWM Counter Register (PWM2_PWMCNR)	32	R	0000_0000h	12.2.5.6/ 3879
3068_0000	PWM Control Register (PWM3_PWMCR)	32	R/W	0000_0000h	12.2.5.1/ 3874
3068_0004	PWM Status Register (PWM3_PWMSR)	32	w1c	0000_0008h	12.2.5.2/ 3876
3068_0008	PWM Interrupt Register (PWM3_PWMIR)	32	R/W	0000_0000h	12.2.5.3/ 3877
3068_000C	PWM Sample Register (PWM3_PWMSAR)	32	R/W	0000_0000h	12.2.5.4/ 3878
3068_0010	PWM Period Register (PWM3_PWMPR)	32	R/W	0000_FFFEh	12.2.5.5/ 3879
3068_0014	PWM Counter Register (PWM3_PWMCNR)	32	R	0000_0000h	12.2.5.6/ 3879
3069_0000	PWM Control Register (PWM4_PWMCR)	32	R/W	0000_0000h	12.2.5.1/ 3874
3069_0004	PWM Status Register (PWM4_PWMSR)	32	w1c	0000_0008h	12.2.5.2/ 3876
3069_0008	PWM Interrupt Register (PWM4_PWMIR)	32	R/W	0000_0000h	12.2.5.3/ 3877
3069_000C	PWM Sample Register (PWM4_PWMSAR)	32	R/W	0000_0000h	12.2.5.4/ 3878
3069_0010	PWM Period Register (PWM4_PWMPR)	32	R/W	0000_FFFEh	12.2.5.5/ 3879
3069_0014	PWM Counter Register (PWM4_PWMCNR)	32	R	0000_0000h	12.2.5.6/ 3879

12.2.5.1 PWM Control Register (PWMx_PWMCR)

The PWM control register (PWM_PWMCR) is used to configure the operating settings of the PWM. It contains the prescaler for the clock division.

Address: Base address + 0h offset



PWMx_PWMCR field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–26 FWM	FIFO Water Mark. These bits are used to set the data level at which the FIFO empty flag will be set and the corresponding interrupt generated 00 FIFO empty flag is set when there are more than or equal to 1 empty slots in FIFO 01 FIFO empty flag is set when there are more than or equal to 2 empty slots in FIFO 10 FIFO empty flag is set when there are more than or equal to 3 empty slots in FIFO 11 FIFO empty flag is set when there are more than or equal to 4 empty slots in FIFO
25 STOPEN	Stop Mode Enable. This bit keeps the PWM functional while in stop mode. When this bit is cleared, the input clock is gated off in stop mode. This bit is not affected by software reset. It is cleared by hardware reset. 0 Inactive in stop mode 1 Active in stop mode
24 DOZEN	Doze Mode Enable. This bit keeps the PWM functional in doze mode. When this bit is cleared, the input clock is gated off in doze mode. This bit is not affected by software reset. It is cleared by hardware reset. 0 Inactive in doze mode 1 Active in doze mode
23 WAITEN	Wait Mode Enable. This bit keeps the PWM functional in wait mode. When this bit is cleared, the input clock is gated off in wait mode. This bit is not affected by software reset. It is cleared by hardware reset.

Table continues on the next page...

PWMx_PWMCR field descriptions (continued)

Field	Description
	0 Inactive in wait mode 1 Active in wait mode
22 DBGEN	Debug Mode Enable. This bit keeps the PWM functional in debug mode. When this bit is cleared, the input clock is gated off in debug mode. This bit is not affected by software reset. It is cleared by hardware reset. 0 Inactive in debug mode 1 Active in debug mode
21 BCTR	Byte Data Swap Control. This bit determines the byte ordering of the 16-bit data when it goes into the FIFO from the sample register. 0 byte ordering remains the same 1 byte ordering is reversed
20 HCTR	Half-word Data Swap Control. This bit determines which half word data from the 32-bit IP Bus interface is written into the lower 16 bits of the sample register. 0 Half word swapping does not take place 1 Half words from write data bus are swapped
19–18 POUTC	PWM Output Configuration. This bit field determines the mode of PWM output on the output pin. 00 Output pin is set at rollover and cleared at comparison 01 Output pin is cleared at rollover and set at comparison 10 PWM output is disconnected 11 PWM output is disconnected
17–16 CLKSRC	Select Clock Source. These bits determine which clock input will be selected for running the counter. After reset the system functional clock is selected. The input clock can also be turned off if these bits are set to 00. This field value should only be changed when the PWM is disabled 00 Clock is off 01 ipg_clk 10 ipg_clk_highfreq 11 ipg_clk_32k
15–4 PRESCALER	Counter Clock Prescaler Value. This bit field determines the value by which the clock will be divided before it goes to the counter. 0x000 Divide by 1 0x001 Divide by 2 0xfff Divide by 4096
3 SWR	Software Reset. PWM is reset when this bit is set to 1. It is a self clearing bit. A write 1 to this bit is a single wait state write cycle. When the block is in reset state this bit is set and is cleared when the reset procedure is over. Setting this bit resets all the registers to their reset values except for the DBGEN, STOPEN, DOZEN, and WAITEN bits in this control register. 0 PWM is out of reset 1 PWM is undergoing reset
2–1 REPEAT	Sample Repeat. This bit field determines the number of times each sample from the FIFO is to be used. 00 Use each sample once 01 Use each sample twice 10 Use each sample four times 11 Use each sample eight times

Table continues on the next page...

PWMx_PWMCR field descriptions (continued)

Field	Description
0 EN	PWM Enable. This bit enables the PWM. If this bit is not enabled, the clock prescaler and the counter is reset. When the PWM is enabled, it begins a new period, the output pin is set to start a new period while the prescaler and counter are released and counting begins. To make the PWM work with softreset and disable/enable, users can do software reset by seting the SWR bit, wait software reset done, configure the registers, and then enable the PWM by setting this bit to "1" Users can also disable/enable the PWM if PWM would like to be stopped and resumed with same registers configurations . 0 PWM disabled 1 PWM enabled

12.2.5.2 PWM Status Register (PWMx_PWMSR)

The PWM status register (PWM_PWMSR) contains seven bits which display the state of the FIFO and the occurrence of rollover and compare events. The FIFOAV bit is read-only but the other four bits can be cleared by writing 1 to them. The FE, ROV, and CMP bits are associated with FIFO-Empty, Roll-over, and Compare interrupts, respectively.

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0									FWE	CMP	ROV	FE	FIFOAV		
W										w1c	w1c	w1c	w1c			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

PWMx_PWMSR field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 FWE	FIFO Write Error Status. This bit shows that an attempt has been made to write FIFO when it is full. 0 FIFO write error not occurred 1 FIFO write error occurred
5 CMP	Compare Status. This bit shows that a compare event has occurred. 0 Compare event not occurred 1 Compare event occurred

Table continues on the next page...

PWMx_PWMSR field descriptions (continued)

Field	Description
4 ROV	Roll-over Status. This bit shows that a roll-over event has occurred. 0 Roll-over event not occurred 1 Roll-over event occurred
3 FE	FIFO Empty Status Bit. This bit indicates the FIFO data level in comparison to the water level set by FWM field in the control register. 0 Data level is above water mark 1 When the data level falls below the mark set by FWM field
FIFOAV	FIFO Available. These read-only bits indicate the data level remaining in the FIFO. An attempted write to these bits will not affect their value and no transfer error is generated. 000 No data available 001 1 word of data in FIFO 010 2 words of data in FIFO 011 3 words of data in FIFO 100 4 words of data in FIFO 101 unused 110 unused 111 unused

12.2.5.3 PWM Interrupt Register (PWMx_PWMIR)

The PWM Interrupt register (PWM_PWMIR) contains three bits which control the generation of the compare, rollover and FIFO empty interrupts.

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													CIE	RIE	FIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMx_PWMIR field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 CIE	Compare Interrupt Enable. This bit controls the generation of the Compare interrupt.

Table continues on the next page...

PWMx_PWMIR field descriptions (continued)

Field	Description
	0 Compare Interrupt not enabled 1 Compare Interrupt enabled
1 RIE	Roll-over Interrupt Enable. This bit controls the generation of the Rollover interrupt. 0 Roll-over interrupt not enabled 1 Roll-over Interrupt enabled
0 FIE	FIFO Empty Interrupt Enable. This bit controls the generation of the FIFO Empty interrupt. 0 FIFO Empty interrupt disabled 1 FIFO Empty interrupt enabled

12.2.5.4 PWM Sample Register (PWMx_PWMSAR)

The PWM sample register (PWM_PWMSAR) is the input to the FIFO. 16-bit words are loaded into the FIFO. The FIFO can be written at any time, but can be read only when the PWM is enabled. The PWM will run at the last set duty-cycle setting if all the values of the FIFO has been utilized, until the FIFO is reloaded or the PWM is disabled. When a new value is written, the duty cycle changes after the current period is over.

A value of zero in the sample register will result in the PWMO output signal always being low/high (POUTC = 00 it will be low and POUTC = 01 it will be high), and no output waveform will be produced. If the value in this register is higher than the PERIOD + 1, the output will never be set/reset depending on POUTC value.

Address: Base address + Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																SAMPLE															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMx_PWMSAR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SAMPLE	Sample Value. This is the input to the 4x16 FIFO. The value in this register denotes the value of the sample being currently used.

12.2.5.5 PWM Period Register (PWMx_PWMPR)

The PWM period register (PWM_PWMPR) determines the period of the PWM output signal. After the counter value matches PERIOD + 1, the counter is reset to start another period.

PWMO (Hz) = PCLK(Hz) / (period +2)

A value of zero in the PWM_PWMPR will result in a period of two clock cycles for the output signal. Writing 0xFFFF to this register will achieve the same result as writing 0xFFFE.

A change in the period value due to a write in PWM_PWMPR results in the counter being reset to zero and the start of a new count period.

NOTE

Settings PWM_PWMPR to 0xFFFF when PWMx_PWMCR REPEAT bits are set to non-zero values is not allowed.

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																PERIOD															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

PWMx_PWMPR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PERIOD	Period Value. These bits determine the Period of the count cycle. The counter counts up to [Period Value] +1 and is then reset to 0x0000.

12.2.5.6 PWM Counter Register (PWMx_PWMCNR)

The read-only pulse-width modulator counter register (PWM_PWMCNR) contains the current count value and can be read at any time without disturbing the counter.

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMx_PWMCNR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Counter Value. These bits are the counter register value and denotes the current count state the counter register is in.

Chapter 13
Multimedia

13.1 Multimedia Overview

13.1.1 Multimedia Overview

13.1.1.1 Display Interface

The display and camera subsystem consists of four major blocks: LCDIF, CSI, MIPI CSI and MIPI DSI.

- The LCDIF is a simple display controller which has been widely used on i.MX chips. The key features of the display controller include:
  - Support 8-bit / 16-bit / 18-bit / 24-bit / 32-bit pixel depth
  - Support DOTCLK mode for MIPI-DPI interface
  - Support resolution up to 1920x1080p60 and 1800x1200p60
  - Support one base layer and one graphics overlay with alpha blending
- This chip supports one 4-lane MIPI DSI display with pixels from the LCDIF. The key features of the MIPI DSI (controller and PHY) include:
  - Compliant to MIPI-DSI standard v1.1
  - Support up to 4 data lanes
  - Support 80Mbps - 1.5Gbps data rate in high speed operation
  - Support 10Mbps data rate in low power operation
- The CSI is a simple camera interface. It captures the MIPI CSI input and saves the pixels into memory. The key features of the CSI include:
  - Configurable interface logic to support most commonly available CMOS sensors
  - 8-bit / 16-bit / 24-bit data port for YCbCr, YUV, or RGB data input
  - Full control of 8-bit/pixel, 10-bit/pixel or 16-bit / pixel data format to 64-bit receive
  - Support up to 1080p@60fps video capture
- The chip supports one 4-lane MIPI CSI2 camera input. The key features of the MIPI CSI2 (controller and PHY) include:

- Compliant to MIPI-CSI2 standard
- Support up to 4 data lanes
- Support 80Mbps - 1.5Gbps data rate in high speed operation
- Support 10Mbps data rate in low power operation
- Support 5M pixel at 15 fps, 1080p60, 720p60, VGA at 60 fps

13.1.1.2 Graphics Processing Unit (GPU)

The GPU consists of a 3D graphics core and a 2D graphics core.

3D graphics core features include:

- Run up to 800MHz (CLK2X) at nominal voltage (0.8V +/- 10%)
- Run up to 1000MHz (CLK2X) at overdrive voltage (0.9V +5%/-10%)
- Support 40M triangles/sec at 800MHz, 50M triangles/sec at 1000MHz
- Support 400M pixel/sec fill rate at 800MHz, 500M pixel/sec fill rate at 1000MHz
- Support 6.4 GFLOPs at 800MHz, 8 GFLOPs at 1000MHz
- Support OpenGL ES 1.1, 2.0
- Support OpenVG 1.1
- TrustZone support using a local MMU to manage secure regions
- AXI clock 800MHz at both nominal and overdrive voltage

2D graphics core features include:

- Run up to 800MHz (CLK2X) at nominal voltage (0.8V +/- 10%)
- Run up to 1000MHz (CLK2X) at overdrive voltage (0.9V +5%/-10%)
- Support multi-source composition
- Support one-pass filter
- Support tile format from GCNanoUltra
- AXI clock 800MHz at both nominal and overdrive voltage

The GPU is powered by a dedicated power VDD_GPU. In order to support DVFS for both high-performance and low power use cases, it should support following operating range:

- 0.8V (0.8V +/-10%), up to 800MHz
- 0.9V (0.9V +5%/-10%), up to 1000MHz

13.1.1.3 Video Processing Unit (VPU)

The chip supports video decoding for various video format using the Hantro G2 + G1. Features include:

- 1080p60 VP9 Profile 0, 2 (10 bit) decoder (Hantro G2), input video stream can be 10-bit, the output decoded video is always 8-bit after post-processing in G2 core

- 1080p60 HEVC/H.265 decoder (Hantro G2)
- 1080p60 AVC/H.264 Baseline, Main, High decoder (Hantro G1)
- 1080p60 VP8 decoder (Hantro G1)
- 1080p60 AVC/H.264 Encoder (Hantro H1)
- 1080p60 VP8 Encoder (Hantro H1)
- TrustZone support

The VPU are powered by a dedicated power VDD_VPU. In order to support DVFS for both high-performance and low power use cases, it should support following operating range:

- 0.8V (0.8V +/-10%), G1/G2/H1 target frequency 600MHz
- 0.9V (0.9V +5%/-10%), G1/H1 target frequency 800MHz, G2 target frequency 700MHz
- AXI clock target frequency 800MHz at both 0.8V and 0.9V case
- Support 1080p60 decoding and 1080p60 encoding at 0.8V, with overdrive to 0.9V, higher frame rate is expected.

13.1.1.4 Audio Interface

The device supports multiple audio interfaces as listed below:

Table 13-1. Audio Interface Summary

Interface	Function	RX Data Line	TX Data Line	Note
SAI-1	External audio	8	8	
SAI-2	External audio	2	2	
SAI-3	External audio	2	2	
SAI-5	External audio	4	4	
SAI-6	External audio	1	1	
SPDIF-1	External audio	1	1	
PDM	External audio	4		

Besides the general audio input/output function, the audio interfaces will supports following features:

- SAI-1 supports up to 16-channels TX (8 lanes) and 16-channels RX (8 lanes) at 384KHz/32-bit
- SAI-5 supports up to 8-channels TX (4 lanes) and 8-channels RX (4 lanes) at 384KHz/32-bit
- SAI-2/3 support up to 4-channels TX (2 lanes) and 4-channels RX (2 lanes) at **768KHz**/32-bit
- SAI-6 support to up to 2-channels TX (1 lanes) and 2-channels RX (1 lanes) at 384KHz/32-bit

- SAI-1/2/3 supports glue-less switching between PCM and DSD operation for popular audio DACs
- SPDIF-1 supports raw capture mode that can save all incoming bits into an audio buffer
- PDM supports up to 8-channels (4 lanes)

The SAI-1/2/3/5/6, SPDIF-1 and PDM share GPIO pads on the chip through IOMUX. Common use cases supported by the audio interfaces are listed in the table below (many other configurations are possible). The number is the data lanes supported

Table 13-2. Audio Interface Use Cases

		UC 1	UC 2	UC 3	UC 4	UC 5	UC 6	UC 7	UC 8
SAI-1	TX	8 + DSD	8 + DSD	8 + DSD	8	8	8	4	4
	RX	8	8	8	8	8	8	4	4
SAI-2	TX	1		1		1	1		1
	RX	1		1		1	1		1
SAI-3	TX	1			1	1	1	1	1
	RX	1			1	1	1	1	1
SAI-5	TX		4		4	1		4	1
	RX		4	4	4	1	4	4	1
SAI-6	TX							1	1
	RX							1	1
SPDIF-1	TX	1	1	1	1	1	1	1	1
	RX	1	1	1	1	1	1	1	1

13.1.1.4.1 SAI Master Clock Inputs/Outputs

The MCLK pin on each SAI module can be configured as either an input or an output. When configured as an output, the SAIn_CLK_ROOT from the CCM is routed to the pad output. When configured as an input, the external input to the pad is routed to SAIn_MCLK, which can be used as master clock for SAI. Below is the diagram showing the both input/output options, by using SAI1 as the example.

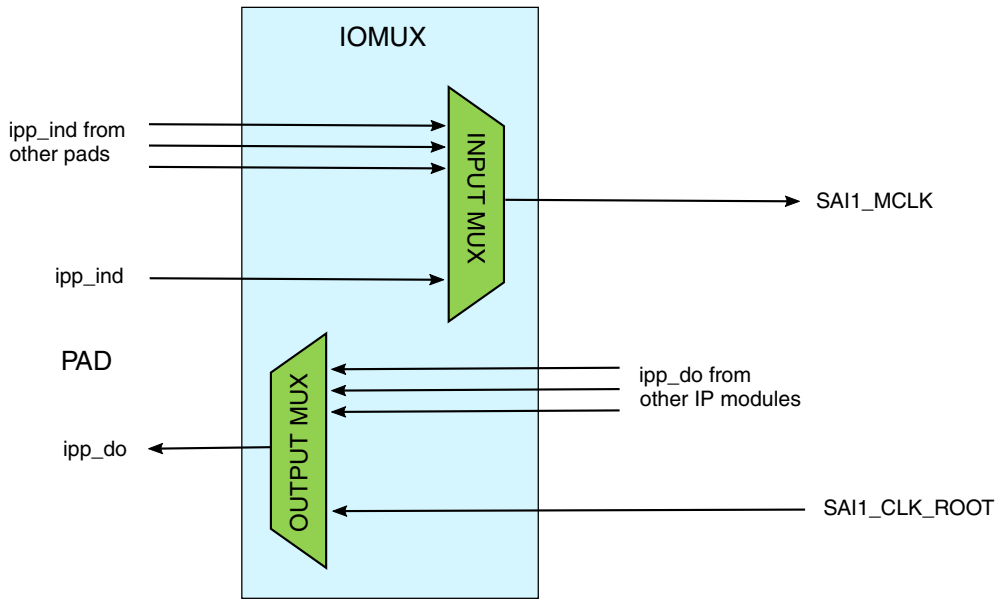


Figure 13-1. SAI MCLK Input and Output

Each SAI module supports up to 3 master clock inputs. The TX and RX sub-module inside each SAI can independently select one of the clock inputs as its master clock. This allows TX and RX of one SAI to run from different clock sources. The master clock inputs have the following options:

- SAIn_MCLK[1] can be selected from SAIn_CLK_ROOT from CCM or SAIn_MCLK from IOMUX. This is the most straight-forward clock routing in which SAIn only uses its own clock source from CCM or IO pad
- SAIn_MCLK[2] can be selected from the following clock sources:
  - Any of the SAIn_CLK_ROOT from CCM
  - Any of the SAIn_MCLK from IOMUX
  - Other clock sources from SPIDF
- SAIn_MCLK[3] has the same clock source options as SAIn_MCLK[2]. This allows both TX and RX can have access to all the options without any dependency between each other

The clock options for master clock on SAI are shown in the diagram blow, by using SAI-1 as an example.

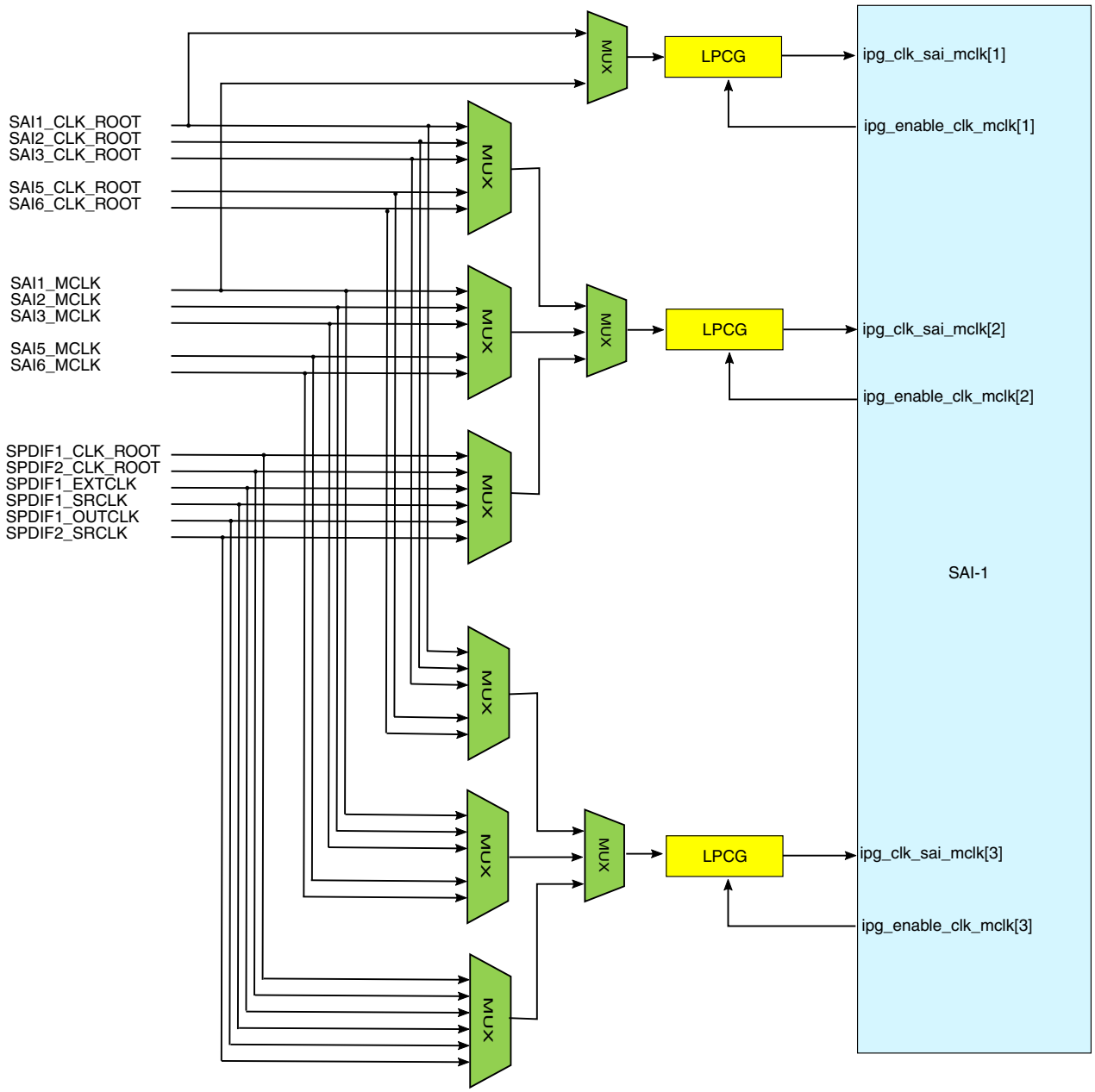


Figure 13-2. SAI Master Clock Options

The options on MCLK[1] are also available on MCLK[2] and MCLK[3]. This provides a similar SAI clock structure as i.MX6/i.MX7 processors.

The configuration of the MUX for master clock is controlled by IOMUXC_GPR registers. They should be configured before the SAI clock is enabled to avoid glitches on the clock.

13.1.1.4.2 SDMA Support

All the SAI and SPDIF instances have SDMA support. In order to meet the audio data rate, two dedicated SDMA modules are used. The DMA request from all audio interface modules will be routed to both SMDA modules, and both SDMA modules' IPG master port are connected to the SPBA. (SPBA has 3 master ports, one for chip-level IPG bus, two for the two SDMAs). With this implementation, the two SDMA modules can be flexibly allocated to all the audio interfaces, based on the needs of audio use cases.

The SDMA-2/3 target frequency is 400MHz IPG and 400MHz AHB, always 1:1 mode, to make sure there is enough throughput for all the audio use cases.

Table 13-3. SDMA Assignment for Audio Interface

SDMA	Audio Interface	Note
SDMA-2/3	SAI-2	2 TX + 2 RX
	SAI-3	2 TX + 2 RX
	SPDIF-1	TX + RX
	PDM	8 RX
	SAI-1	8 TX + 8 RX
	SAI-5	4 TX + 4 RX
	SAI-6	1 TX + 1 RX

13.2 Enhanced LCD Interface (eLCDIF)

13.2.1 Overview

The enhanced Liquid Crystal Display Interface (LCDIF) is a general purpose display controller used to drive a wide range of display devices varying in size and capability.

The LCDIF block supports the following:

- Displays with an asynchronous parallel MPU interface for command and data transfer to an integrated frame buffer.
- Displays that support moving pictures and require the RGB interface mode (DOTCLK interface).
- VSYNC mode for high-speed data transfers.
- Digital video encoders that accept ITU-R BT.656 format 4:2:2 YCbCr digital component video and convert it to analog TV signals.

The LCDIF provides fully programmable functionality to supported interfaces:

- Bus master interface to source frame buffer data for display refresh. This interface can also be used to drive data for "Smart" displays.

- PIO interface to manage data transfers between "Smart" displays and SoC.
- 8/16/18/24/32 bit LCD data bus support available.
- Programmable timing and parameters for MPU, VSYNC, and DOTCLK LCD interfaces to support a wide variety of displays.
- ITU-R BT.656 mode (called Digital Video Interface or DVI mode here) including progressive-to-interlace feature and RGB to YCbCr 4:2:2 color space conversion to support 525/60 and 625/50 operation.

13.2.2 Functional Description

Bus Interface Mechanisms through Initializing the LCDIF, describe the internal pipeline for the LCDIF interfaces. Differences for each mode are then described in separate sections, as follows:

- MPU Interface
- VSYNC Interface
- DOTCLK Interface
- ITU-R BT.656 Digital Video Interface (DVI)

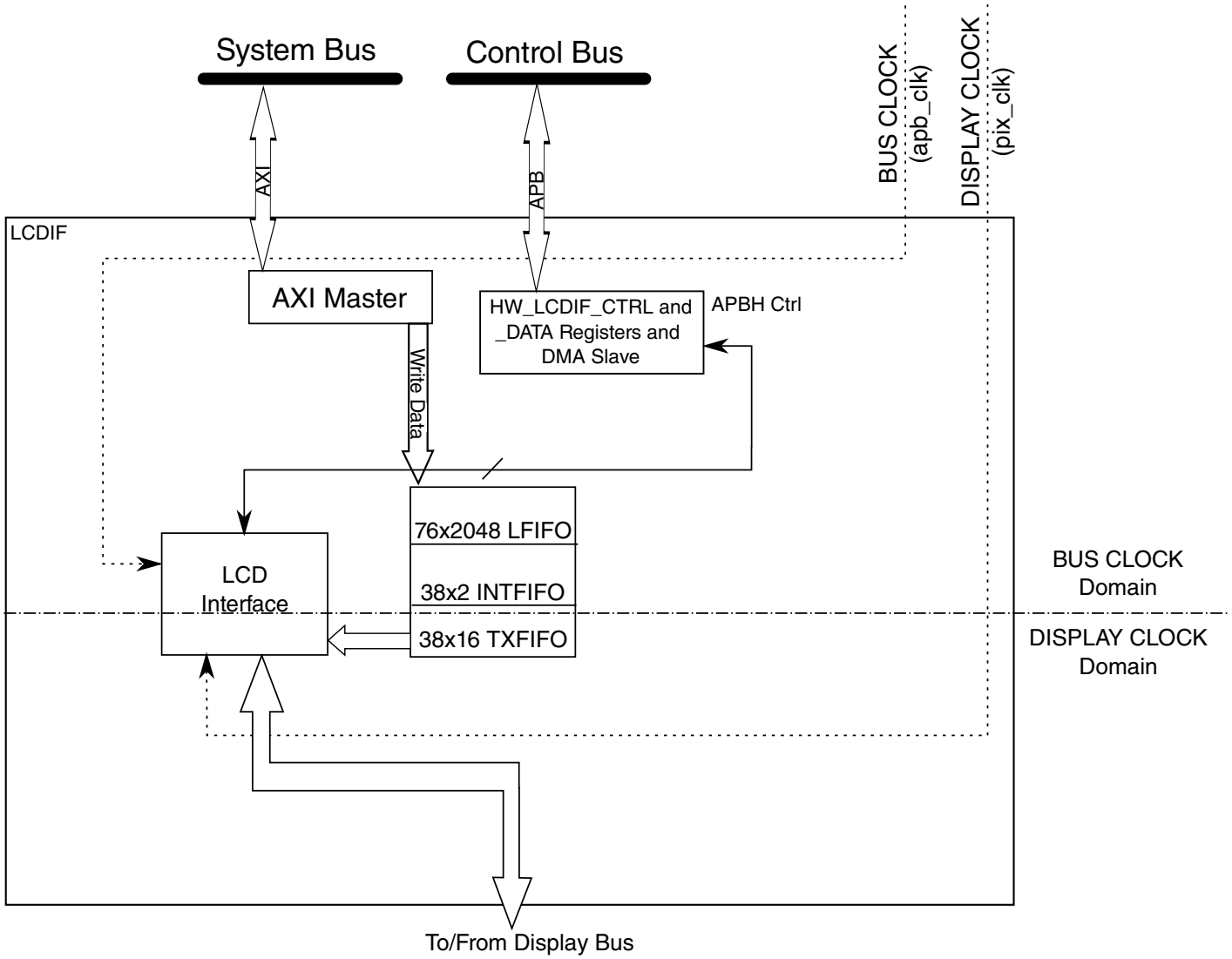


Figure 13-3. Top-Level Block Diagram of LCDIF subsystem

13.2.2.1 Bus Interface Mechanisms

The LCDIF module has memory-mapped control, data and status registers. It provides several interfaces to transfer data between the display and SoC.

The bus master interface is used to initiate the requests to transfer data from external memory to the display. It is completely autonomous, or no CPU intervention is required, to manage the cyclical nature of refreshing standard display types. Bus mastering can also be used for MPU mode data writes.

The PIO interface is used to interface to "Smart" displays to transfer frame buffer data and control information to/from the external display. The host CPU executes display drivers to manage the display solution. The following sections describe the system bus interface mechanisms.

13.2.2.1.1 Bus Master Operation in Write/Display Modes

The LCDIF block has a bus master interface that initiates requests for data to drive the display. The LCDIF_MASTER bit must be set to 1 to enable the bus master interface. Software should program all control registers required to transfer the frame sequence.

In the MPU and VSYNC mode, single frames are transferred. When a complete frame is transferred, LCDIF enters idle and clears the RUN bit in the CTRL register. For subsequent frame transmission, the LCDIF setup sequence should be repeated.

The DOTCLK and DVI mode is used to refresh the display at the desired refresh rate and resolution, and drive displays that don't integrate a display buffer memory. When the display is refreshed, the LCDIF will automatically update the LCDIF_CUR_BUF_ADDR register with the value in LCDIF_NEXT_BUF_ADDR at the end of current frame and start fetching the next frame from the new address. If the LCDIF_NEXT_BUF_ADDR register was not updated within a frame refresh cycle, LCDIF will keep transmitting the last frame until a new value is programmed into that register.

LCDIF also provides the capability of interlacing a progressive frame by fetching odd lines in the first field and then fetching even lines in the second field. This feature can be used in the DVI mode and can be turned on by setting the INTERLACE_FIELDS bit in the LCDIF_CTRL1 register.

13.2.2.1.2 System Bus Master Performance

The performance of the LCDIF block can be controlled by changing the burst length and the outstanding cycle issuing capability depending on the memory bandwidth requirements. Two fields in the LCDIF_CTRL2 register will throttle system memory requests. The LCDIF_CTRL2_OUTSTANDING_REQS field will control how many requests the LCDIF can have in flight on any given clock cycle. This should be programmed based on the expected system bus latency for returned read data. Also, the LCDIF_CTRL2_BURST_LEN_8 bit will set the number of 64 bit words requested for each LCDIF system bus request to either 8 or 16 QWORDS. Generally, 4 outstanding requests of length 16 will provide enough performance to drive any standard display resolution. These configuration bits are intended to change the access pattern of the LCDIF to optimize system bus throughput when other system masters will contend for system memory resources.

The LCDIF_THRES register can also be used to optimize bus throughput and power consumption.

The LCDIF_THRES_PANIC value can be used to raise the priority of requests initiated by the LCDIF to alter how the LCDIF requests are arbitrated by the system bus infrastructure. The panic output control signal is raised when the number of 32bpp pixel equivalents in the LFIFO is less than this programmed value. Since the LFIFO is arranged as a 8 * 256x64bit quadword FIFO, it contains two 32bpp pixels per quadword, or 512 32bpp pixels total. To set the panic output when 3/4s of the LFIFO is empty, set the LCDIF_THRES_PANIC value to 3/4 * 512, or 128. The panic signal output is used to assess higher priority to LCDIF system requests to avoid LCDIF under run errors during periods of high system bandwidth utilization.

The features available with the LCDIF_THRES register require support from system clocking and dynamic priority control. Refer to the appropriate block documentation to assess the system support for these features.

13.2.2.2 Write Data Path

LCDIF supports raster based frame buffers and there is no support for tiled buffers.

There are several options to accommodate endianness of display buffers in memory before the data is processed for the external display. The LCDIF_CTRL[INPUT_DATA_SWIZZLE] field provides the following options for data word multiplexing:

```
00 (0): No swizzle (little-endian)
01 (1): Swap bytes 0 and 3, swap bytes 1 and 2 (big-endian)
10 (2): Swap half-words
11 (3): Swap bytes within each half-word
```

The LCDIF_CTRL[WORD_LENGTH] field indicates the input data/pixel format. LCDIF_TRANSFER_COUNT register denotes how much data is contained in each frame. The LCDIF_TRANSFER_COUNT[H_COUNT] field indicates the number of pixels per line and LCDIF_TRANSFER_COUNT[V_COUNT] indicates the total number of lines per frame. The LCDIF_CTRL1[BYTE_PACKING_FORMAT] field can be used to specify which bytes within the 32-bit word are going to be valid. For example, if the entire 32-bit word is valid, LCDIF_CTRL1[BYTE_PACKING_FORMAT] should be set to 0xF, if only lower 3 bytes of each word in the frame buffer are valid, then LCDIF_CTRL1[BYTE_PACKING_FORMAT] should be set to 0x7.

The LCDIF_CTRL[LCD_DATABUS_WIDTH] field suggests the width of the bus going to the display controller. There is an option to source all 32 bits of the input word and transfer it to the output display interface. If the LCDIF_CTRL[LCD_DATABUS_WIDTH] is not the same as

LCDIF_CTRL[WORD_LENGTH], LCDIF will perform RGB to RGB color space conversion. For example, if the input frame has fewer bits per pixel than the display, as in a 16 bpp input frame going to 24 bpp LCD, LCDIF will pad the MSBs of each color to the LSBs of the same color for each pixel. If the input frame has more bits per pixel than the display, for example, 24 bpp input frame going to 16 bpp LCD, LCDIF will drop the LSBs of each color channel to convert to the lower color depth. LCDIF also has the capability to support delta pixel displays by swizzling the R, G and B colors of each pixel in the odd and even lines of the frame separately by programming the LCDIF_CTRL2[ODD_LINE_PATTERN] and the LCDIF_CTRL2[EVEN_LINE_PATTERN] bit fields. This operation occurs after the RGB-to-RGB color space conversion operation.

LCDIF also supports RGB to YCbCr 4:2:2 color space conversion. This is useful in the DVI mode since the TV encoder requires input in YCbCr 4:2:2 format. The LCDIF_CSC* registers have complete programmability over the CSC coefficients and offsets. The values must be written into these registers in the signed two's complement format.

The following list shows how the different input/output combinations can be obtained:

- LCDIF_CTRL[WORD_LENGTH]=1 indicates that the input is 8-bit data. This is most likely going to be used for sending commands in MPU interface, or maybe a gray scale image. Any combination of LCDIF_CTRL1[BYTE_PACKING_FORMAT] is permissible.

  Limitation: LCDIF_TRANSFER_COUNT[H_COUNT] must be a multiple of the sum of BYTE_PACKING_FORMAT [3], BYTE_PACKING_FORMAT [2], BYTE_PACKING_FORMAT [1] and BYTE_PACKING_FORMAT [0]. LCDIF_CTRL[LCD_DATABUS_WIDTH] must be 1, indicating an 8-bit data bus.

- LCDIF_CTRL[WORD_LENGTH]=0 implies the input frame buffer is RGB 16 bits per pixel. LCDIF_CTRL[DATA_FORMAT_16_BIT] field determines the pixels are RGB 555 or RGB 565.

  Limitation: LCDIF_CTRL1[BYTE_PACKING_FORMAT] should be 0x3 or 0xC if there is only one pixel per word. If there are two pixels per word, it should be 0xF and LCDIF_TRANSFER_COUNT[H_COUNT] will be restricted to be a multiple of 2 pixels.

- LCDIF_CTRL[WORD_LENGTH]=2 indicates that input frame buffer is RGB 18 bits per pixel, that is, RGB 666. The valid RGB values can be left-aligned or right-aligned within a 32-bit word. The alignment of the valid 18 bits within a word is indicated by the LCDIF_CTRL[DATA_FORMAT_18_BIT] bit.

Limitation: LCDIF_CTRL1[BYTE_PACKING_FORMAT] can be 0x7, 0xE or 0xF. Packed pixels are not supported in this case.
LCDIF_TRANSFER_COUNT[H_COUNT] can be any number.

- LCDIF_CTRL[WORD_LENGTH]=3 indicates that the input frame-buffer is RGB 24 bits per pixel (RGB 888). If LCDIF_CTRL1[BYTE_PACKING_FORMAT] is 0x7, it indicates that there is only one pixel per 32-bit word and there is no restriction on LCDIF_TRANSFER_COUNT[H_COUNT].

  Limitation: If LCDIF_CTRL1[BYTE_PACKING_FORMAT] is 0xF, it indicates that the pixels are packed, that is, there are 4 pixels in 3 words or 12 bytes and LCDIF_TRANSFER_COUNT[H_COUNT] must be a multiple of 4 pixels.

- LCDIF_CTRL1[YCBCR422_INPUT]=1 implies that the input frame is in YCbCr 4:2:2 format. LCDIF_CTRL1[BYTE_PACKING_FORMAT] must be 0xF.

  Limitation: LCDIF_CTRL[LCD_DATABUS_WIDTH] must be 8-bit and LCDIF_TRANSFER_COUNT[H_COUNT] must be a multiple of 2 pixels.

  LCDIF_CTRL2[ODD_LINE_PATTERN] and
  LCDIF_CTRL2[EVEN_LINE_PATTERN] must be 0 when any of
  LCDIF_CTRL[RGB_TO_YCBCR422_CSC] or
  LCDIF_CTRL1[INTERLACE_FIELDS] or LCDIF_CTRL[YCBCR422_INPUT] bits is 1.

After the RGB to RGB or RGB to YCbCr 4:2:2 color space conversions, there is one more opportunity to swizzle the data before sending it out to the display or the encoder. This can be done with the LCDIF_CTRL[CSC_DATA_SWIZZLE] field, and it provides the same options as the LCDIF_CTRL[INPUT_DATA_SWIZZLE] register.

Finally, there is an option to shift the output data before sending it out to the display. This is done based on the LCDIF_CTRL[SHIFT_DIR] and
LCDIF_CTRL[SHIFT_NUM_BITS] fields.

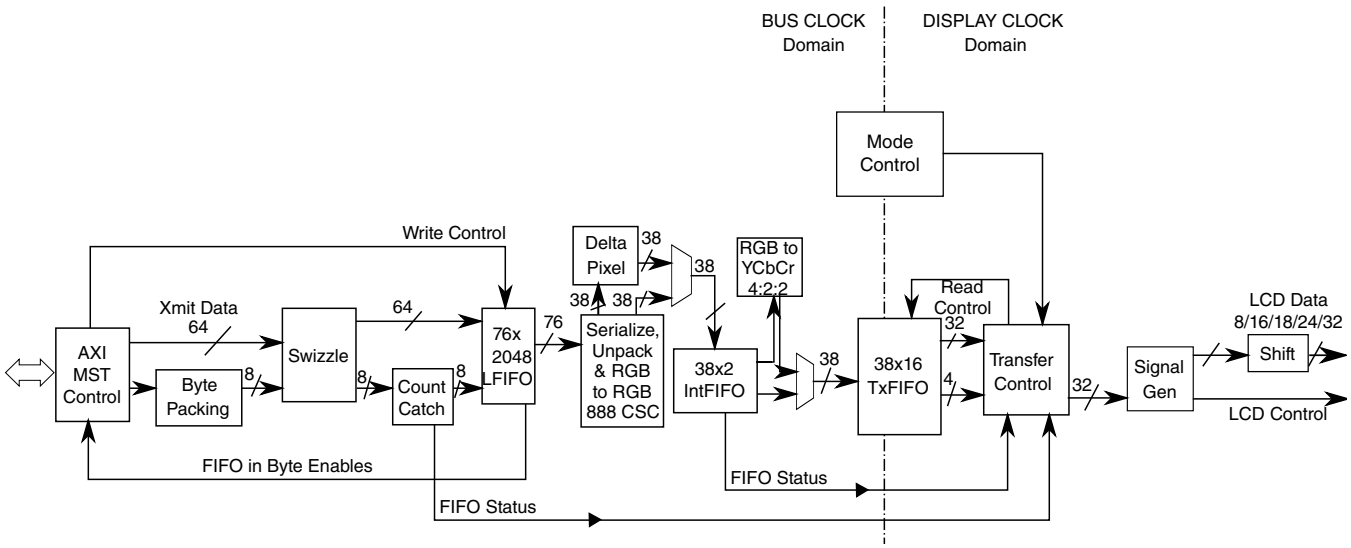


Figure 13-4. General Operations in Write Data Path

The examples in the following figures illustrate some different combinations of register programming for write mode. Assume that the data transferred over the system bus within a 32 bit word is organized as {A7-A0, B7-B0, C7-C0, D7-D0} in 8-bit mode and {A15-A0, B15-B0} in 16-bit mode.

In this example, all 32 bits of the input word are transferred out over an 8 bit display bus. Each byte within the 32 bit word is shifted to the right with zeros appended to bits D[7:6]. The input data bits [7:2] are shifted to the right by 2 bits and presented on the D[5:0].

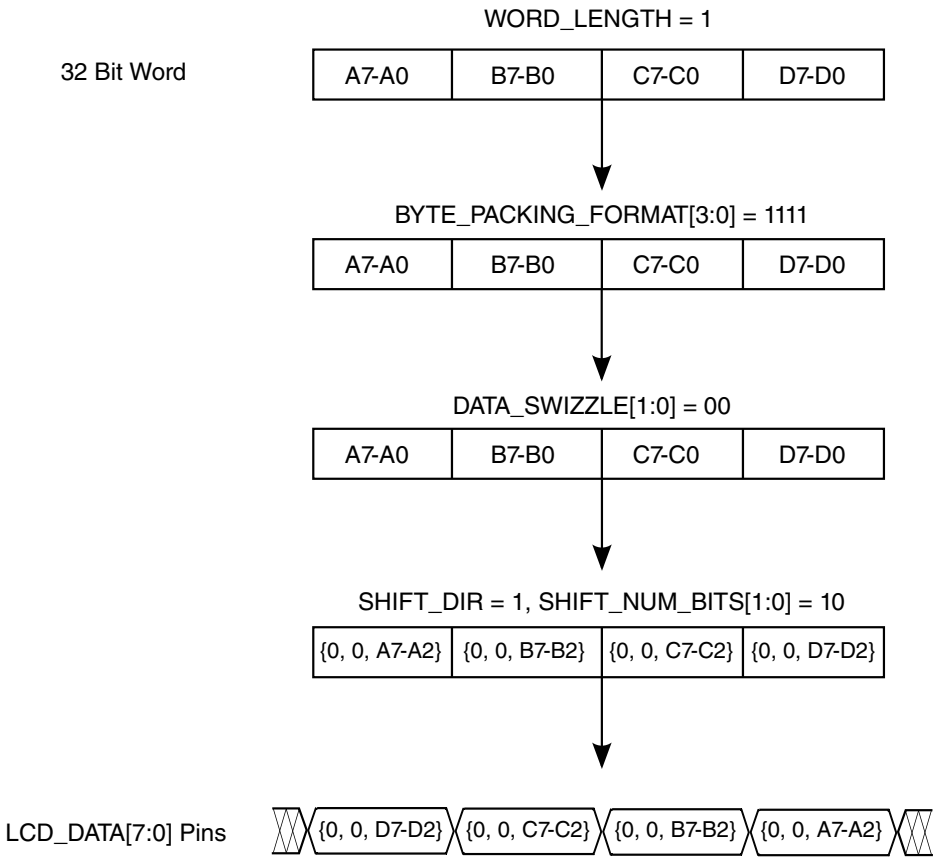


Figure 13-5. Register programming for write mode

In this 8 bit display interface example, one byte of the input word is deleted and not transferred over the external 8 bit display interface. This mode could be used to transfer 24bpp pixels over the 8 bit interface. In this case, the 4th unused byte is not transferred.

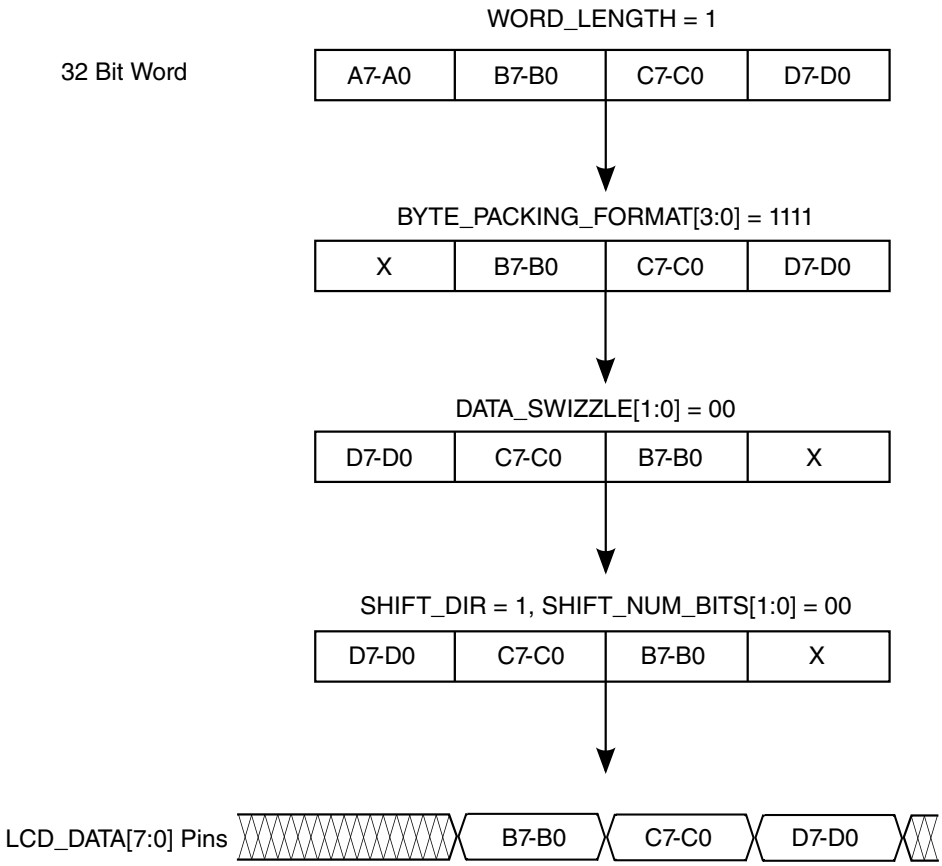


Figure 13-6. Register programming for write mode

The following example uses a 16 bit display interface. Each 16 bit half word is shifted to the right by two bits with zeros appended to the most significant two bits.

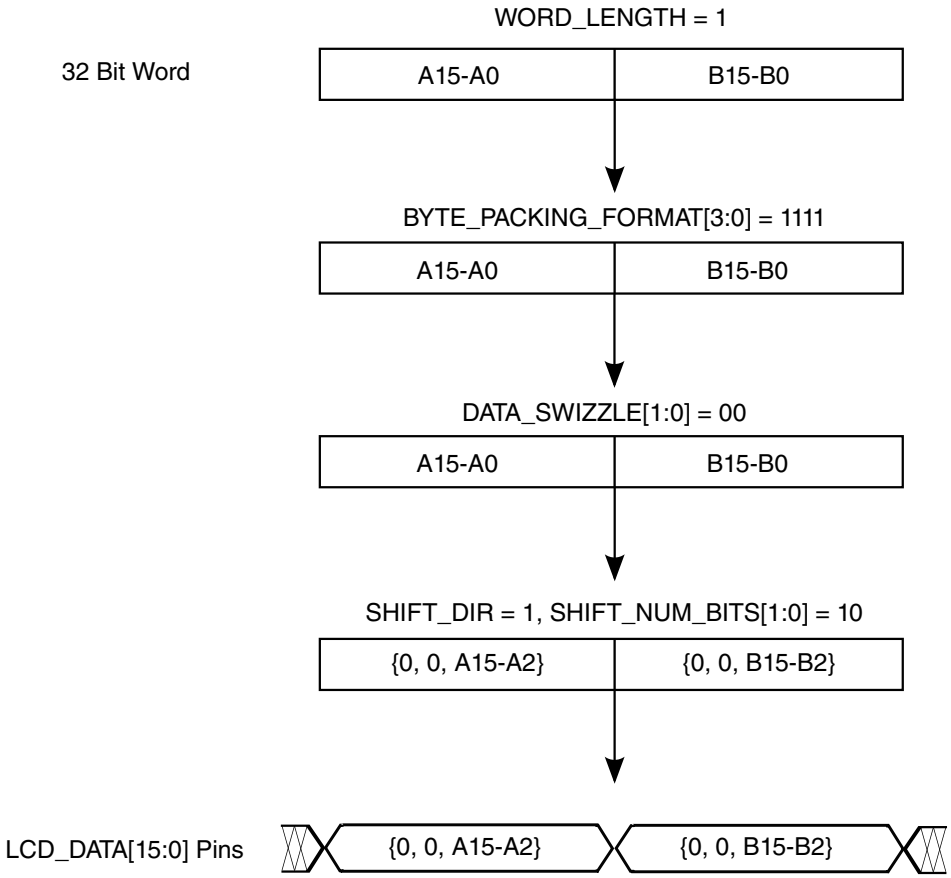


Figure 13-7. Register programming for write mode

This example indicates how an unpacked frame buffer can sourced for display. Only a single 16 bit half word within the 32 bit word is tranferred out via the 16 display bus.

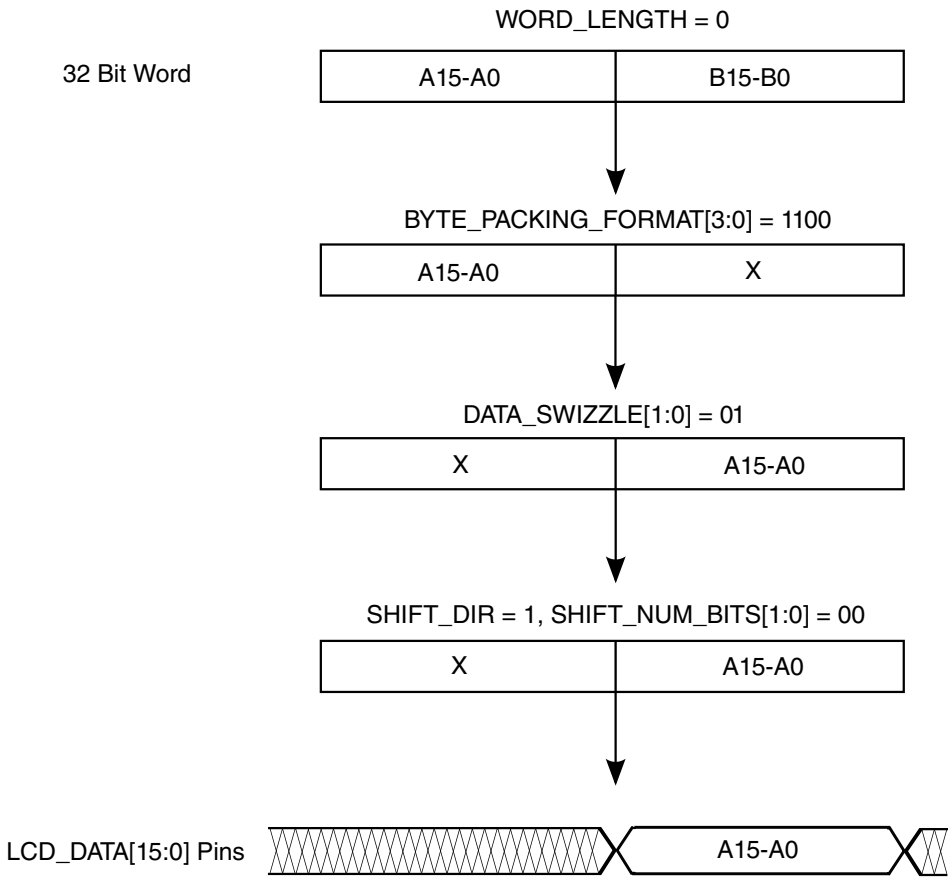


Figure 13-8. Register programming for write mode

13.2.2.3 Read Data Path

Figure 13-9 shows the MPU read data path in detail.

LCDIF can read from an external display that follows the 6800/8080 MPU protocol.

The display bus width is determined by the LCD_DATABUS_WIDTH bit field. The data sampled at every read strobe is called a subword and the number of subwords that can be packed in a 32-bit word is given by the READ_MODE_NUM_PACKED_SUBWORDS bit field. The INITIAL_DUMMY_READ bit field directs the LCDIF to skip the number of programmed subwords before starting to process read data. This feature is useful in the case of an LCD controller that returns the last written data the first time a read is issued, and then sends the correct data thereafter. SHIFT_DIR and SHIFT_NUM_BITS bit fields indicate whether the data needs to be shifted before getting stored in the internal registers. For example, a value of 2 in READ_MODE_NUM_PACKED_SUBWORDS if lcd databus width is 8 bits indicates two bytes should be packed in a 32-bit word, while if the lcd databus width is 16 bits, it indicates that two half words (or 4 bytes) should be packed.

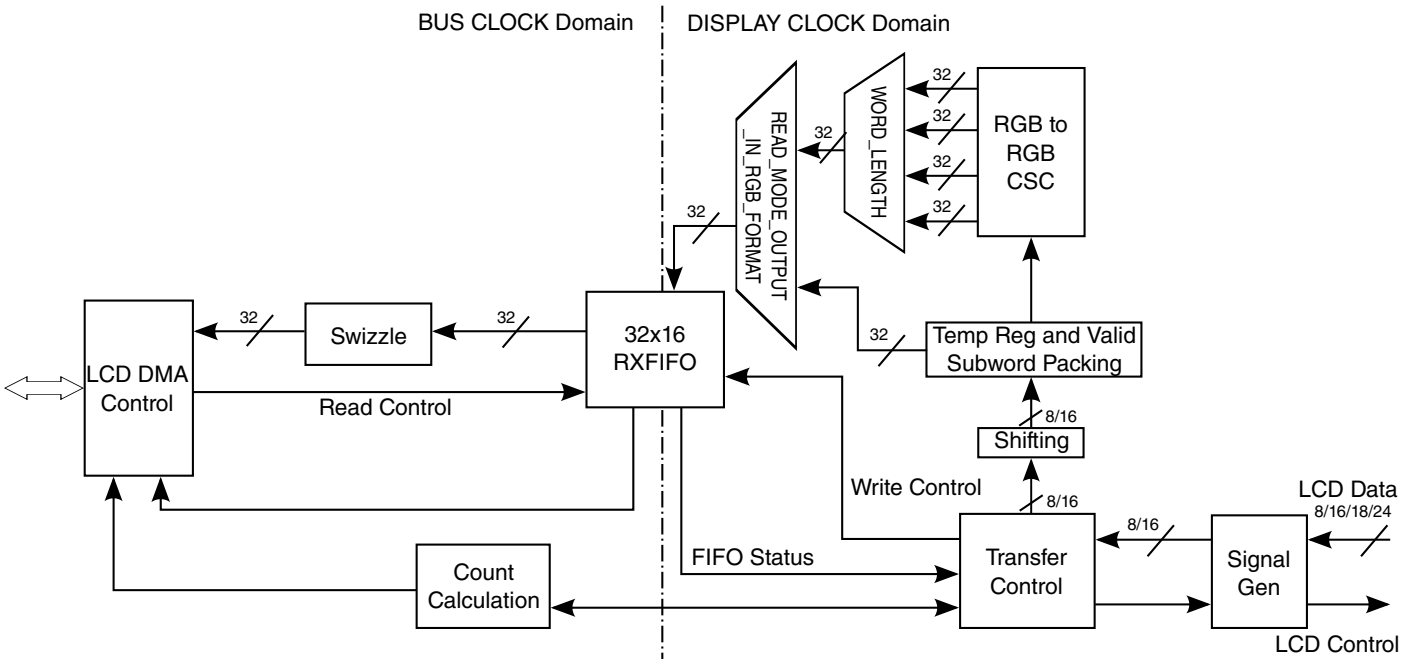


Figure 13-9. MPU Read Data Path

After the last subword within a word is reached, the block looks at the READ_PACK_DIR in the HW_LCDIF_CTRL2 register. If this bit is set, the block will swizzle the data, but only within the valid bytes, unlike in the write mode, where swizzle occurs across all 4 bytes. If the READ_MODE_OUTPUT_IN_RGB_FORMAT bit is set, LCDIF will convert the data obtained from the READ_PACK_DIR operation into 24-bit unpacked RGB and then re-convert it into 16/18/24 bpp RGB depending on the WORD_LENGTH field. The DATA_FORMAT_16/18/24_BIT bit fields are also considered while converting to 24-bit unpacked RGB format. For example, if DATA_FORMAT_18_BIT is 1, the RGB666 data will be packed in the upper bits [31:4]of a 32-bit word, and that bit is 0, the data will be packed in the lower bits [17:0]. After all these operations, the data gets written into the RXFIFO.

The following figures show some examples of how data is handled in different MPU read modes.

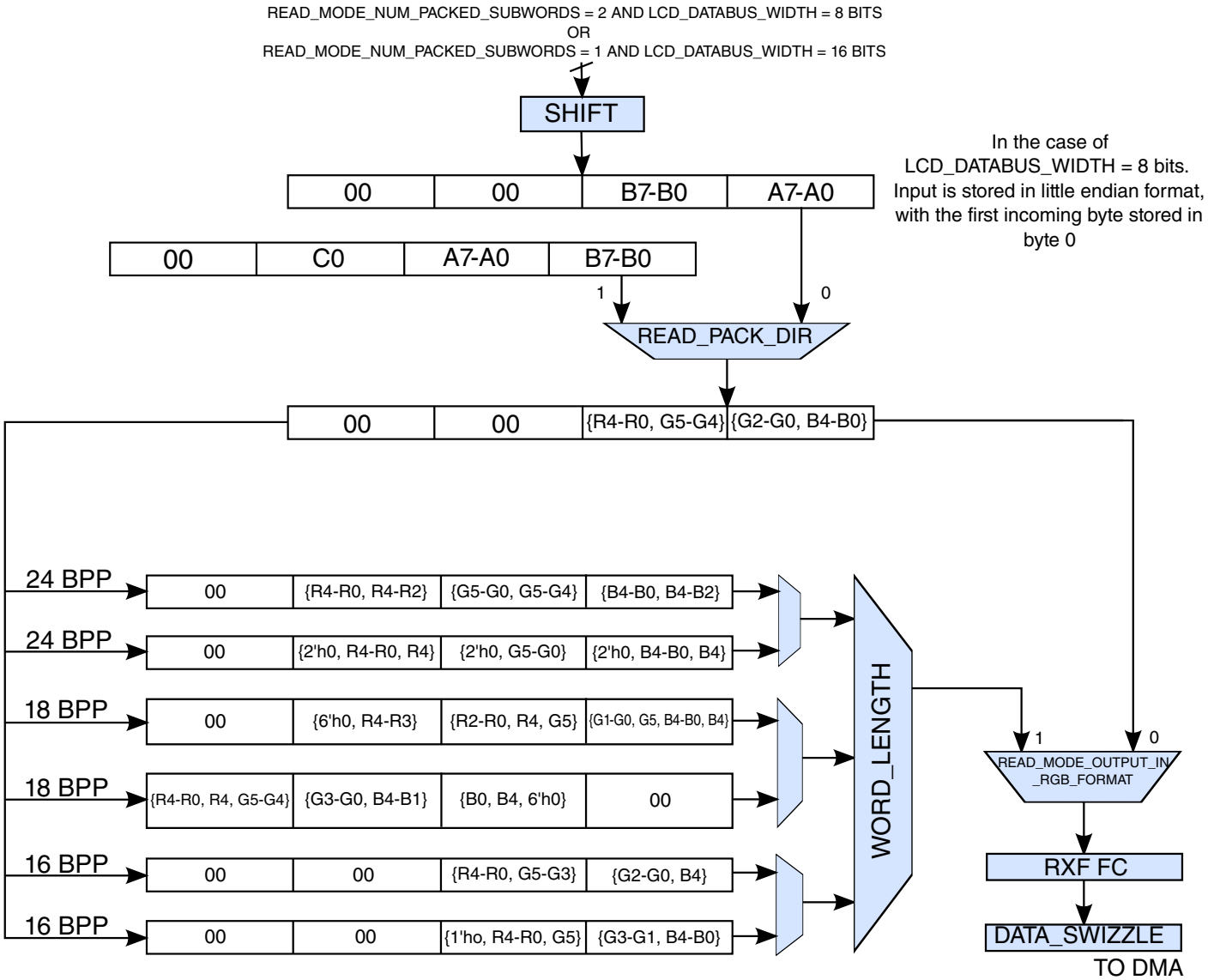


Figure 13-10. Data in MPU read mode

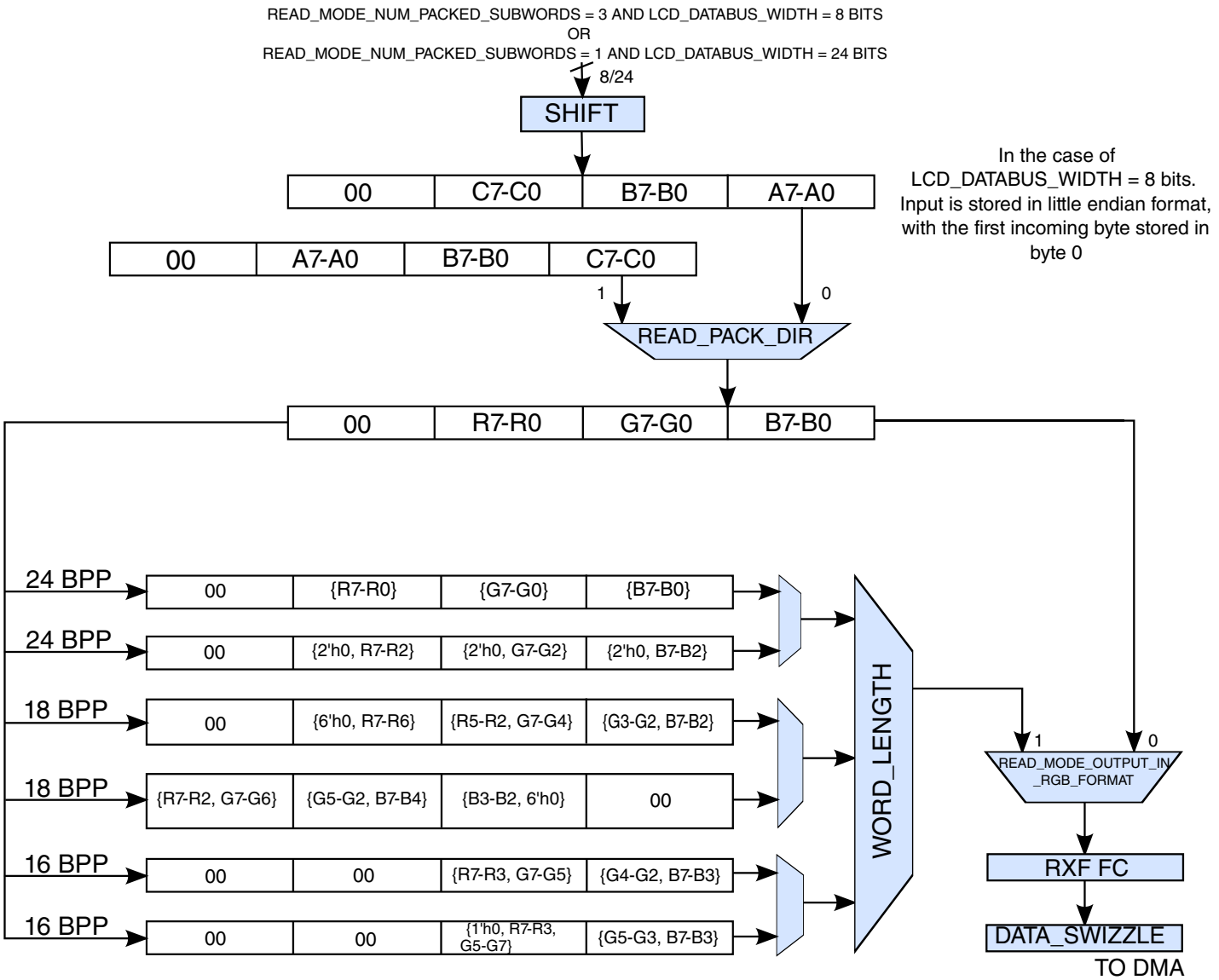


Figure 13-11. Data in MPU read mode

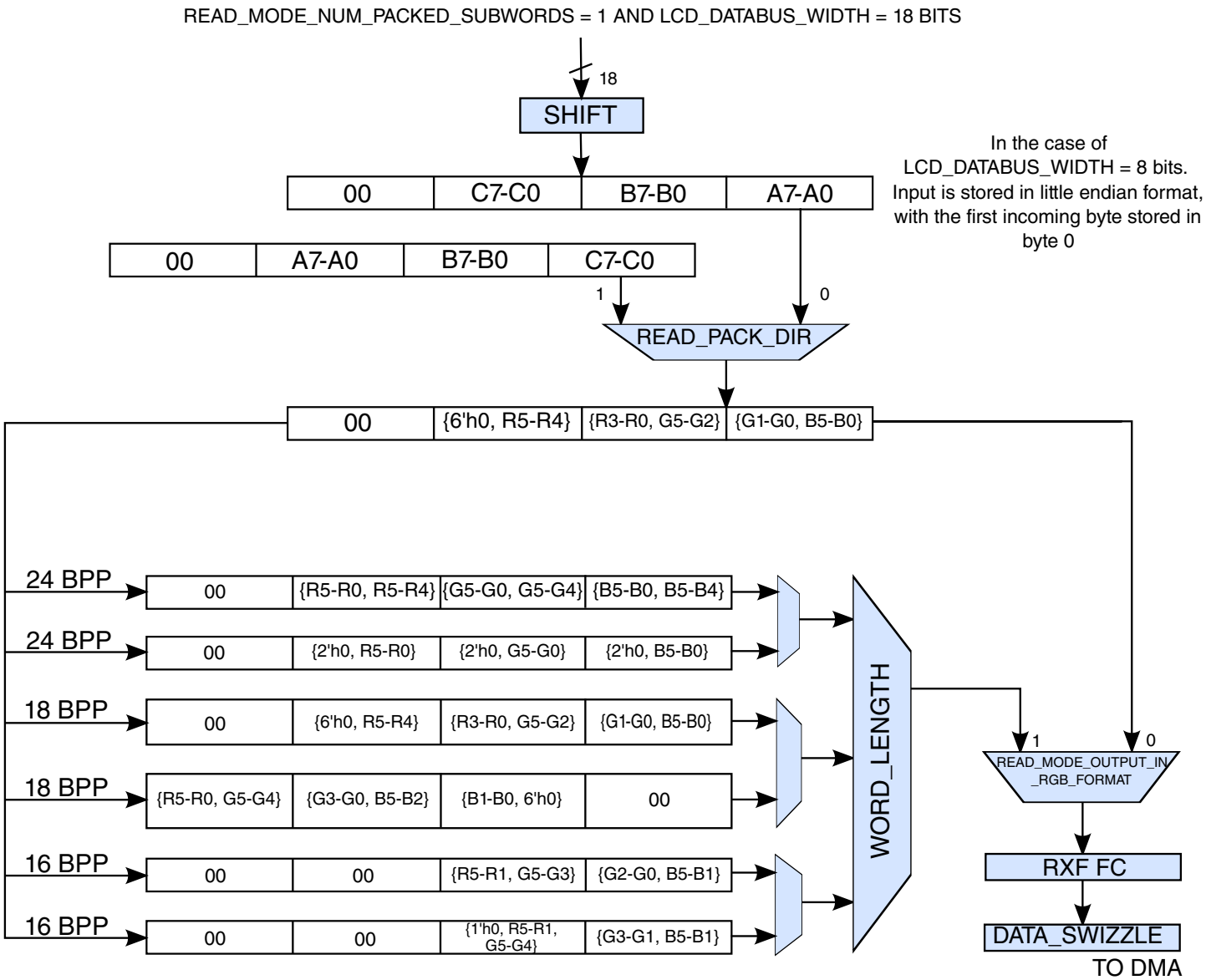


Figure 13-12. Data in MPU read mode

Restrictions:

READ_PACK_DIR should only be used if it is required to swizzle the subwords before doing RGB to RGB CSC, otherwise the DATA_SWIZZLE field should be used to swizzle across bytes.

READ_PACK_DIR must be 0 if LCD_DATABUS_WIDTH is 8 bits and READ_MODE_NUM_PACKED_SUBWORDS =1

If READ_MODE_OUTPUT_IN_RGB_FORMAT bit is set, the following restrictions should be followed:

- If LCD_DATABUS_WIDTH = 8 bits, then READ_MODE_NUM_PACKED_SUBWORDS <= 3.
- If LCD_DATABUS_WIDTH = 16/18/24 bits, then READ_MODE_NUM_PACKED_SUBWORDS = 1.

13.2.2.4 LCDIF Interrupts

LCDIF supports a number of interrupts to aid controlling and status reporting of the block.

All the interrupts have individual mask bits for enabling or disabling each of them. They all get funneled through a single interrupt line connected to the interrupt collector (ICOLL).

The following list describes the different interrupts supported by LCDIF:

- Underflow interrupt is asserted when the clock domain crossing FIFO (TXFIFO) becomes empty but the block is in active display portion during that time. Software should take corrective action to make sure that this does not happen.
- In the bus master mode, the overflow interrupt will be asserted if the block has requested more data than it's FIFOs could hold. In the read mode, it will be asserted if the RxFIFO becomes full and the block reads more data.
- VSYNC edge interrupt will be asserted every time a leading VSYNC edge occurs.
- Cur_frame_done interrupt occurs at the end of every frame in all modes except DVI. In DVI mode, if IRQ_ON_ALTERNATE_FIELDS bit is set, it will occur at the end of every frame, otherwise it will occur at the end of every field.

13.2.2.5 Initializing the LCDIF

This section describes write modes and MPU read mode.

13.2.2.5.1 Write Modes

The following initialization steps are common to all LCDIF write modes of operation before entering any particular mode.

Initialization steps:

1. Configure the external I/Os to correctly interface the external display, when required.
2. Start the DISPLAY CLOCK (pix_clk) clock and set the appropriate frequency by programming the registers in CCM.

3. Start the BUS CLOCK (apb_clk) and set the appropriate frequency by programming the registers in CCM.
4. Bring the LCDIF out of soft reset and disable the clock gate bit.
5. Reset the LCD controller by setting LCDIF_CTRL1[RESET] bit appropriately, being careful to observe the reset requirements of the controller. See Behavior During Reset for more information on Reset requirements.
6. Make sure LCDIF_CTRL[READ_WRITEB] bit is 0.
7. Set the transfer mode of operation to bus master. The LCDIF_CTRL[MASTER] bit determines the transfer mode selected. Bus master (LCDIF_CTRL[MASTER] =1), or PIO (LCDIF_CTRL[MASTER] =0) mode is the transfer mode to select.
8. Set the LCDIF_CTRL[INPUT_DATA_SWIZZLE] according to the endianness of the LCD controller. Also, set the LCDIF_CTRL[DATA_SHIFT_DIR] and LCDIF_CTRL[SHIFT_NUM_BITS] if it is required to shift the data left or right before it is output.
9. Set the LCDIF_CTRL[WORD_LENGTH] field appropriately: 0 = 16-bit input, 1 = 8-bit input, 2 = 18-bit input, 3 = 24/32-bit input. Also, select the correct 16/18/24 bit data format with the corresponding fields in LCDIF_CTRL register.
10. Set the LCDIF_CTRL1[BYTE_PACKING_FORMAT] field according to the input frame.
11. Set the LCDIF_CTRL[LCD_DATABUS_WIDTH] appropriately: 0 = 16-bit output, 1 = 8-bit output, 2 = 18-bit output, 3 = 24/32-bit output.
12. Enable the necessary IRQs.

13.2.2.5.2 MPU Read Mode

The following initialization steps should be done to enter the MPU read mode of operation:

Initialization steps:

1. Configure the external I/Os to correctly interface the external display.
2. Start the DISPLAY CLOCK (pix_clk) and set the appropriate frequency by programming the registers in CCM.
3. Start the BUS CLOCK (apb_clk) and set the appropriate frequency by programming the registers in CCM.
4. Bring the LCDIF out of soft reset and clock gate.
5. Reset the LCD controller by setting LCDIF_CTRL1_RESET bit appropriately, being careful to observe the reset requirements of the controller.
6. Set the READ_WRITEB bit in LCDIF_CTRL register to 1.
7. Set the LCDIF_MASTER bit in LCDIF_CTRL register to 0. Bus master mode is not supported for reading data from the display.

8. Also, set the DATA_SHIFT_DIR and SHIFT_NUM_BITS if it is required to shift the data left or right before it is output.
9. Indicate if the read data needs to color-space-converted and stored in a different RGB format by setting the READ_MODE_OUTPUT_IN_RGB_FORMAT field accordingly.
10. Set the WORD_LENGTH field appropriately: 0 = 16-bit input, 1 = 8-bit input, 2 = 18-bit input, 3 = 24-bit input if READ_MODE_OUTPUT_IN_RGB_FORMAT is required. Also, select the correct 16/18/24 bit data format with the corresponding fields in LCDIF_CTRL register.
11. Set the READ_MODE_NUM_PACKED_SUBWORDS field in LCDIF_CTRL2 according to the number of subwords per word required to be packed.
12. Set the READ_PACK_DIR to 1 if it is required to store the data in big-endian format.
13. Set the LCD_DATABUS_WIDTH appropriately: 0 = 16-bit output, 1 = 8-bit output, 2 = 18-bit output, 3 = 24-bit output.
14. Enable the necessary IRQs.

13.2.2.6 MPU Interface

The MPU interface is used to transfer data and commands between the SoC via the LCDIF and the external display at modest data rates.

Bus master or PIO transactions using the LCDIF_DATA register can be used for MPU mode write operations. For MPU mode read operations, only PIO can be used. LCDIF can support the 6800 as well as the 8080 MPU protocol. If DOTCLK_MODE, DVI_MODE and VSYNC_MODE bits in LCDIF_CTRL registers are 0, it implies that the block is in MPU interface mode of operation. The LCDIF MPU mode has four basic timing parameters: Setup and Hold for the Command/Data register selection (TCS, TCH) and Setup and Hold for the Data bus (TDS, TDH). These parameters are expressed in DISPLAY CLOCK (pix_clk) cycles. The LCD_WR signal is used as the write strobe while LCD_RS signal is typically used to switch between command and data modes.

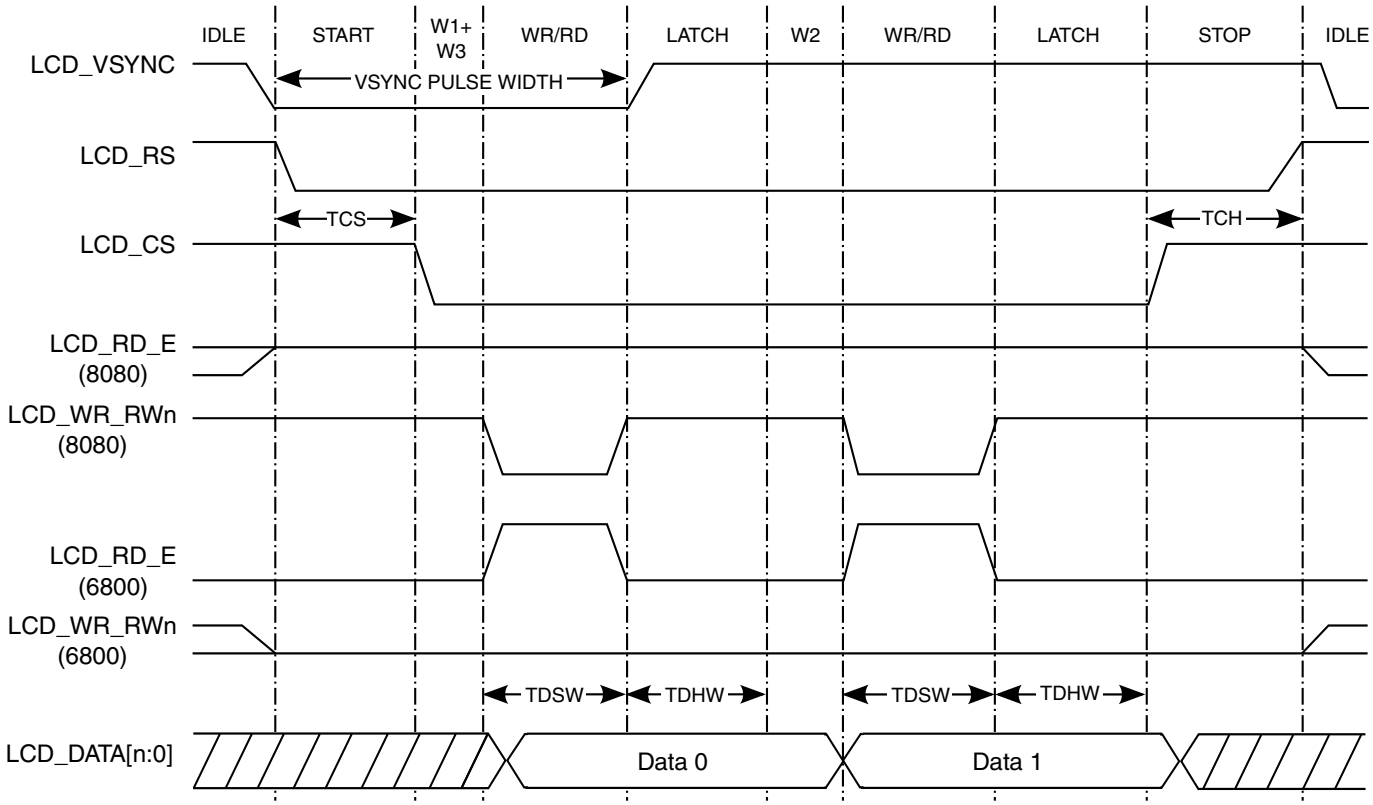


Figure 13-13. Timing in write mode of 6800 and 8080 protocols

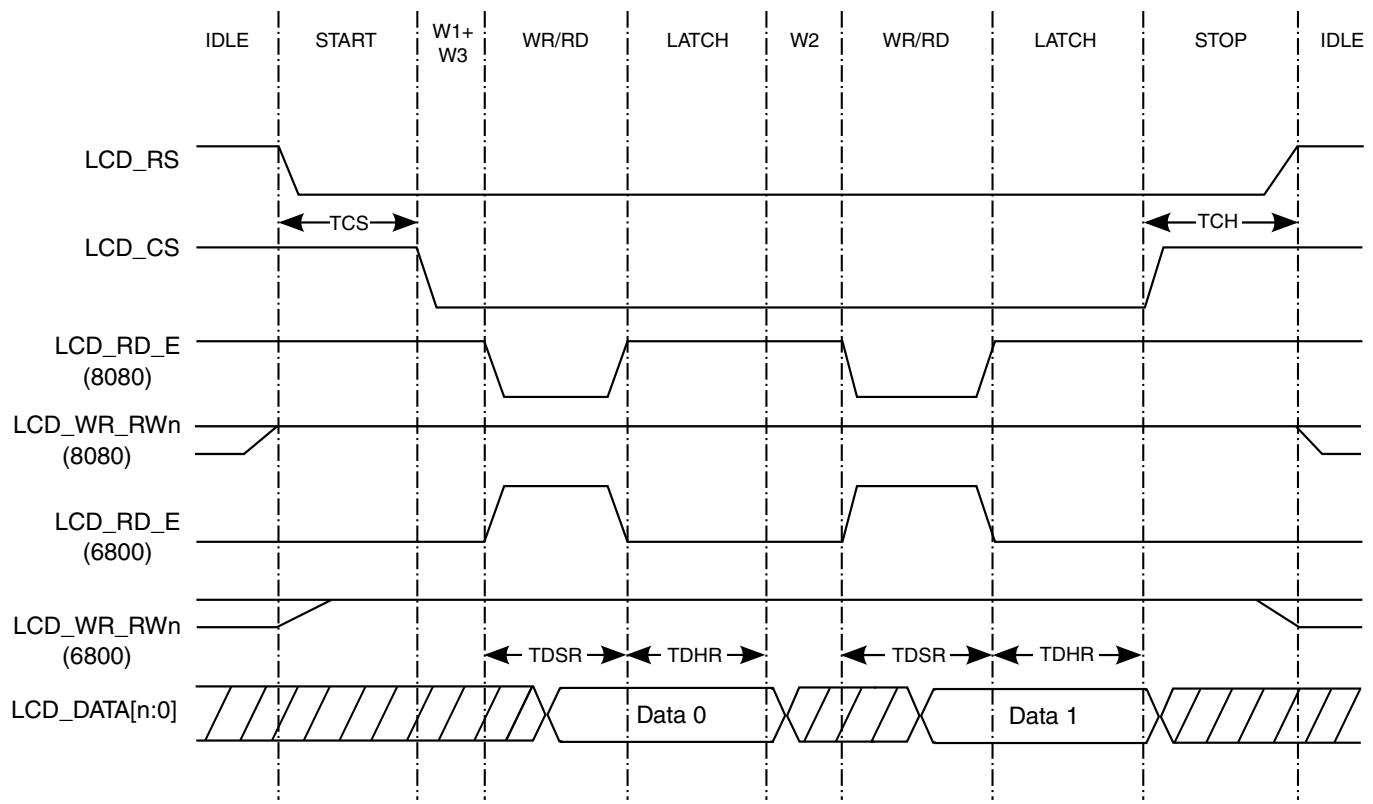


Figure 13-14. Read timing interface in 6800 and 8080 protocols

The LCDIF has flexible pin and strobe timings which enable it to optimally support a wide range of LCDs. The minimum cycle time is two DISPLAY CLOCK (pix_clk) cycles (TDS=TDH=1). For example, this results in a maximum LCD data rate of 12 MB/s when DISPLAY CLOCK (pix_clk) is 24 MHz. TDS and TDH are 8-bit values, so the minimum LCDIF period is 510 DISPLAY CLOCK (pix_clk) cycles (47 KHz with a 24 MHz DISPLAY CLOCK (pix_clk)). The timings are not automatically adjusted if the DISPLAY CLOCK (pix_clk) frequency changes, so it may be necessary to adjust the timings if DISPLAY CLOCK (pix_clk) changes.

In the MPU interface mode, the LCDIF_CTRL_ BYPASS_COUNT bit must be 0. The RUN bit is cleared automatically once the LCDIF has received/transmitted all the data as per the LCDIF_TRANSFER_COUNT register and has completed the transfer to the panel. The current transfer can be cancelled/aborted if the RUN bit is manually made 0.

13.2.2.6.1 Code Example to Initialize the LCDIF in MPU Write Mode

```
// Note: Common initialization steps in Initializing the LCDIF must also be
// executed along with the following code
BF_CS1(LCDIF_CTRL, DATA_SELECT, 1); // 0 if sending command, 1 if sending data. Note that the
                                    // idle state for LCD_RS signal is high, regardless of the
                                     // programming of the DATA_SELECT register.
BF_CS1 (LCDIF_CTRL, MODE86, 8080_MODE);
BF_CS1 (LCDIF_CTRL, READ_WRITEB, 0);
BF_CS1 (LCDIF_CTRL, BYPASS_COUNT, 0); //Must be 0 in MPU mode
BF_CS1 (LCDIF_CTRL1, BUSY_ENABLE, 1);//Only if LCD controller implements a busy line
BF_CS4 (LCDIF_TIMING, CMD_HOLD, 2, CMD_SETUP, 2, DATA_HOLD, 2, DATA_SETUP, 2);  //Values
based
                               // on DISPLAY CLOCK (pix_clk) frequency and timing requirements
of controller.
                              // Note that these register must be non-zero for correct
operation.
BF_CS2 (LCDIF_TRANSFER_COUNT, H_COUNT, 320, V_COUNT, 240); //For a 320 RGB x 240 display
BF_CS1 (LCDIF_CTRL, RUN, 1);
```

The LCDIF is now ready to receive data via bus master PIO write transactions using the LCDIF_DATA register. Note that when using the PIO write operations to the LCDIF_DATA register, the software will need to poll the FIFO STATUS bits to ensure that it does not overflow the LCDIF data buffers. When LCDIF is done transmitting H_COUNT x V_COUNT pixels, it will stop, turn off the RUN bit and assert the cur_frame_done interrupt.

13.2.2.7 VSYNC Interface

The VSYNC interface uses the same protocol as the MPU interface, with an additional signal VSYNC at the frame rate of the display, as shown in the figure given in MPU Interface section.

It is used in the moving picture display mode where data has to be written to the internal LCD buffer at a speed higher than the display rate and displayed in synchronization with the VSYNC signal. This mode is selected by setting the VSYNC_MODE bit in LCDIF_CTRL register. The VSYNC signal is programmable for period, polarity and direction. Many other programmable parameters are shared with the MPU interface. The VSYNC_OEB bit in LCDIF_VDCTRL0 register indicates whether the display controller will send the VSYNC signal, or whether it should be generated by LCDIF. The timing of the VSYNC signal is based on the DISPLAY CLOCK (pix_clk) (make sure VSYNC_PULSE_WIDTH_UNIT = VSYNC_PERIOD_UNIT = 0 and VSYNC_ONLY = 1) and it is determined by the VSYNC_PERIOD, VSYNC_PULSE_WIDTH and VSYNC_POL fields in LCDIF_VDCTRL0-4 registers. The SYNC_SIGNALS_ON bit in LCDIF_VDCTRL4 register must be set if the target requires the VSYNC signal to be generated by LCDIF. If the WAIT_FOR_VSYNC_EDGE bit in LCDIF_CTRL register is set, it indicates that the hardware should wait until it sees the leading VSYNC edge before starting the data transfer. The VERITCAL_WAIT_CNT indicates the number of DISPLAY CLOCK (pix_clk) cycles from the leading VSYNC edge after which data transfer will be started on the interface.

In the VSYNC interface mode, the LCDIF_CTRL_ BYPASS_COUNT bit must be 0. The RUN bit is cleared automatically once the LCDIF has received/transmitted all the data as per the LCDIF_TRANSFER_COUNT register and has completed the transfer to the panel. The current transfer can be cancelled/aborted if the RUN bit is manually made 0.

13.2.2.7.1 Code Example to Initialize LCDIF in VSYNC Mode

```
// Note: Common initialization steps in Initializing the LCDIF must also be
// executed along with the following code
BF_CS1 (LCDIF_CTRL, DATA_SELECT, 1); // 0 if sending command, 1 if sending data. Note that
//the idle state for LCD_RS signal is high, regardless of the programming of the DATA_SELECT
//register.

BF_CS1 (LCDIF_CTRL, MODE86, 8080_MODE);
BF_CS1 (LCDIF_CTRL, BYPASS_COUNT, 0); //Must be 0 in MPU mode
BF_CS1 (LCDIF_CTRL1, BUSY_ENABLE, 0);
BF_CS4 (LCDIF_TIMING, CMD_HOLD, 2, CMD_SETUP, 2, DATA_HOLD, 2, DATA_SETUP, 2);  //Values
//based on DISPLAY CLOCK (pix_clk) frequency and timing requirements of controller. Note
that these
//register must be non-zero for the MPU and VSYNC modes.
BF_CS2 (LCDIF_TRANSFER_COUNT, H_COUNT, 320, V_COUNT, 240);//For a 320 RGB x 240 display
//The following section indicates setting up the VSYNC signal timing when VSYNC is an output
BF_CS1 (LCDIF_VDCTRL0, VSYNC_OEB, 0); //Making VSYNC signal an output
BF_CS1 (LCDIF_VDCTRL4, VSYNC_ONLY, 1); //Only need to generate VSYNC signal
BF_CS1 (VDCTRL0, VSYNC_POL, 0); //Setting the polarity of VSYNC signal to be low during
//VSYNC_PULSE_WIDTH time
BF_CS2 (LCDIF_VDCTRL0, VSYNC_PERIOD_UNIT, 0, VSYNC_PULSE_WIDTH_UNIT, 0);
BF_CS2 (LCDIF_VDCTRL1, VSYNC_PERIOD, 400000, VSYNC_PULSE_WIDTH, 100);//Frame display rate in
//terms of number of DISPLAY CLOCKS (pix_clk).
BF_CS2 (LCDIF_VDCTRL2, HSYNC_PULSE_WIDTH, 0, HSYNC_PERIOD, 0);
BF_CS1 (LCDIF_VDCTRL3, VERTICAL_WAIT_CNT, 50);
BF_CS1 (LCDIF_VDCTRL4, SYNC_SIGNALS_ON, 1);
BF_CS2 (LCDIF_CTRL, VSYNC_MODE, 1, WAIT_FOR_VSYNC_EDGE, 1); //set WAIT_FOR_VSYNC_EDGE if
```

```
//software wishes to transfer the next frame after the VSYNC edge occurs.
BF_CS1 (LCDIF_CTRL, RUN, 1);
```

The LCDIF is now ready to receive data via bus master requests or PIO writes to the LCDIF_DATA register. When LCDIF is done transmitting H_COUNT x V_COUNT pixels, it will stop, turn off the RUN bit and assert the cur_frame_done interrupt.

13.2.2.8 DOTCLK Interface

The DOTCLK interface is another mode used in moving picture displays.

It includes the VSYNC, HSYNC, DOTCLK and (optional) ENABLE signals. The interface is popularly called the RGB interface if the ENABLE signal is present.

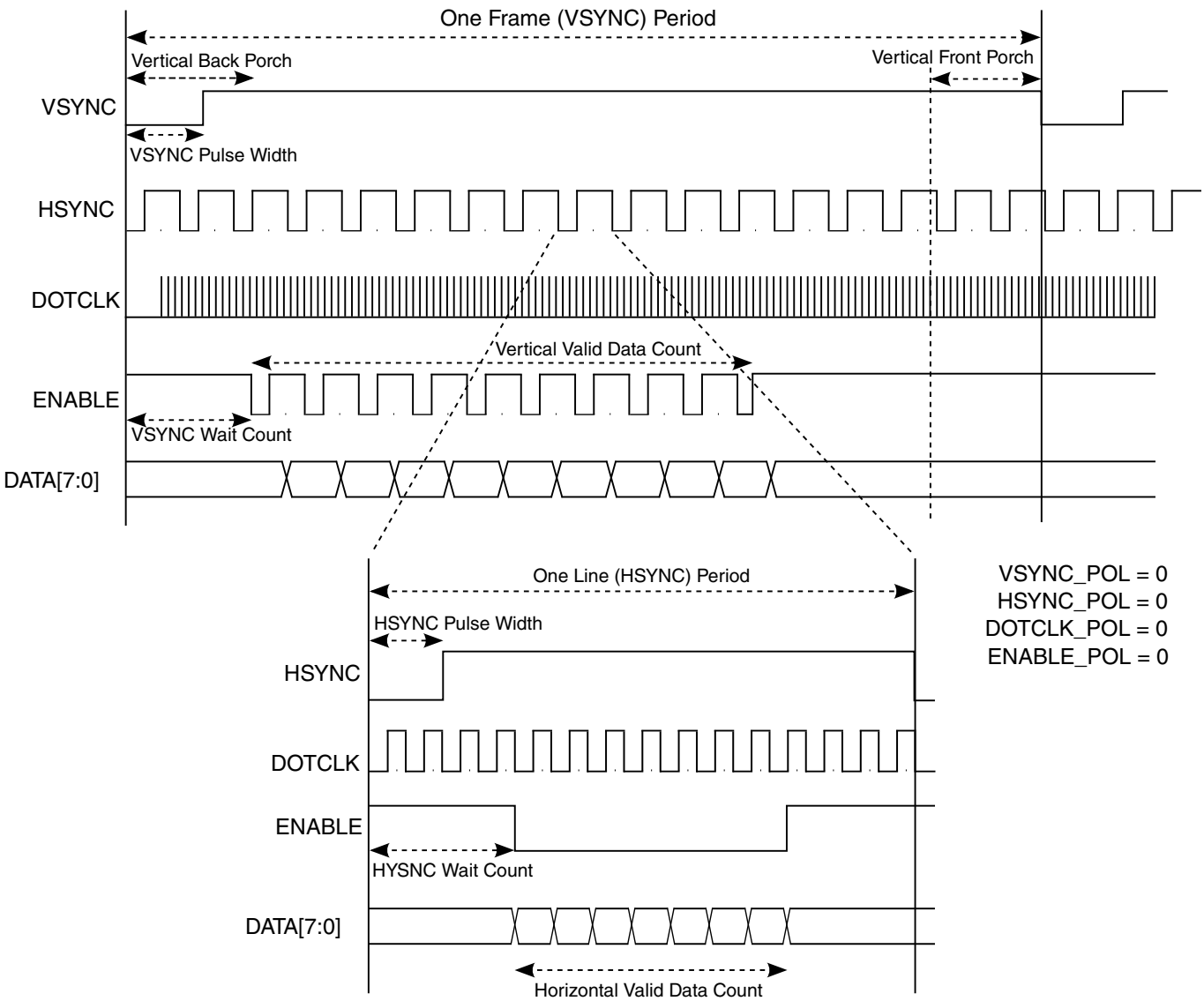


Figure 13-15. DOTCLK protocol with programmable parameters

The DOTCLK mode writes data at high speed to the LCD, and the display operation is synchronized with the VSYNC, HSYNC, ENABLE and DOTCLK signals. The polarities, periods and pulse-widths of the sync signals are programmable using the LCDIF_VDCTRL0-4 registers. The units for the VSYNC signal must be number of horizontal lines and can be selected using the VSYNC_PULSE_WIDTH_UNIT and VSYNC_PERIOD_UNIT bit fields. The VERTICAL_WAIT_CNT is by default given the same unit as the VSYNC_PERIOD. The DISPLAY CLOCK (pix_clk) frequency is managed by the CCM.

In DOTCLK mode, LCDIF_CTRL_BYPASS_COUNT bit must be set to 1. To end the current transfer, the software should make the DOTCLK_MODE bit 0, so that all data that is currently in the LCDIF LFIFO and TXFIFO is transmitted. Once that transfer is complete, the block will automatically clear the RUN bit and issue the cur_frame_done interrupt.

13.2.2.8.1 Code Example

The following code shows an example for programming a 320x240 display.

NOTE

Setting up the display must be done through the MPU mode or via SPI.

```
// Note: Common initialization steps in Initializing the LCDIF must also be
// executed along with the following code
BF_CS1 (LCDIF_CTRL, DOTCLK_MODE, 1);
BF_CS1 (LCDIF_CTRL, BYPASS_COUNT, 1); //Always for DOTCLK mode
BF_CS1 (LCDIF_VDCTRL0, VSYNC_OEB, 0); //Vsync is always an output in the DOTCLK mode
BF_CS4 (LCDIF_VDCTRL0, VSYNC_POL, 0, HSYNC_POL, 0, DOTCLK_POL, 0, ENABLE_POL, 0);
BF_CS1 (LCDIF_VDCTRL0, ENABLE_PRESENT, 1);
BF_CS2 (LCDIF_VDCTRL0, VSYNC_PERIOD_UNIT, 1, VSYNC_PULSE_WIDTH_UNIT, 1);
BF_CS1 (LCDIF_VDCTRL0, VSYNC_PULSE_WIDTH, 2);
BF_CS1 (LCDIF_VDCTRL1, VSYNC_PERIOD, 280);
BF_CS2 (LCDIF_VDCTRL2, HSYNC_PULSE_WIDTH, 10, HSYNC_PERIOD, 360); //Assuming
                                                    //  LCD_DATABUS_WIDTH is 24bit
BF_CS2 (LCDIF_VDCTRL3, VSYNC_ONLY, 0);
BF_CS2 (LCDIF_VDCTRL3, HORIZONTAL_WAIT_CNT, 20, VERTICAL_WAIT_CNT, 20);
BF_CS1 (LCDIF_VDCTRL4, DOTCLK_H_VALID_DATA_CNT, 320);//Note that DOTCLK_V_VALID_DATA_CNT is
                            //implicitly assumed to be HW_LCDIF_TRANSFER_COUNT_V_COUNT
BF_CS1 (LCDIF_VDCTRL4, SYNC_SIGNALS_ON, 1);
BF_CS1 (LCDIF_CTRL, RUN, 1);
```

To stop the transfer completely, the ideal way is to make DOTCLK_MODE = 0. In that case, the block will transmit the contents in the FIFO and reset the RUN bit.

13.2.2.9 Alpha Blending Interface

The LCDIF have the capability to add an extra overlay on the normal display buffer,LCDIF can fetch data from two buffers and combine them before display,one buffer data can have the alpha value with the RGB pixels. With LCDIF_AS_CTRL[AS_ENABLE] is set, the LCDIF will start fetching alpha surface buffer data in bus master mode and combine it with another buffer.

The LCDIF_AS_CTRL[ALPHA_CTRL] bits determines how the alpha value is constructed for the alpha surface and alpha blending process is as same as in PXP block, for the alpha blend and color key process refer to the PXP block descriptions.

13.2.2.10 ITU-R BT.656 Digital Video Interface (DVI)

ITU-R BT.656 Digital Video Interface shown below transmits 4:2:2 YCbCr digital component video to a digital video encoder that can translate it into 525/60 or 625/50 analog TV signal.

Unique timing codes (timing reference signals) are embedded within the video stream to indicate the different timing events that would have been otherwise indicated by VSYNC, HSYNC and BLANK signals. The hardware supports 8-bit data transfers; the pins are shared with the lower 8 bits of LCD data bus. The LCD_RS pin is shared with the clock signal of the interface (called CCIRCLK here for uniqueness). CCIRCLK also can be obtained on the LCD_DOTCLK pin. The mode shares the write FIFO with the LCD interface and the associated pipeline. The programmable parameters in registers LCDIF_DVICTRL0-3 allow setting the total number of horizontal lines per frame, vertical and horizontal blanking interval, odd and even field start and end positions, and so on. In short, these parameters are provided to ensure that the hardware has enough flexibility to generate the right 525/60 or 625/50 data streams. Most of the initialization steps in Initializing the LCDIF such as data shifting, swizzle, and so on, are applicable to DVI mode also. The register descriptions in the programmable registers section at the end of this chapter include example code for programming the DVICTRL0-3 registers.

In DVI mode, LCDIF_CTRL_BYPASS_COUNT bit must be set to 1. To end the current transfer, the software should make the DVI_MODE bit the value 0, so that all data that is currently in the LCDIF LFIFO and TXFIFO is transmitted. Once that transfer is complete, the block will automatically clear the RUN bit and assert the cur_frame_done interrupt.

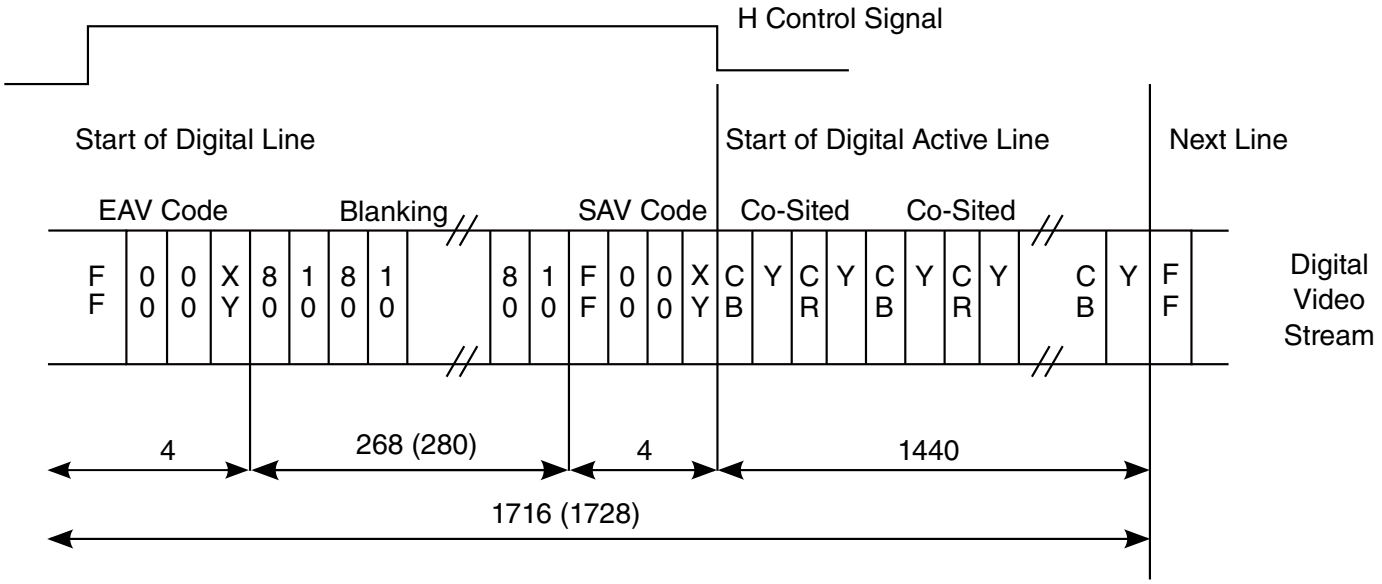


Figure 13-16. Digital Video Interface

13.2.2.11 Panel Interface Generator (Pigeon Mode)

There are several panel interface signal outputs (CLK signal not included), each of them with dedicated timing purpose as default. This is called "Legacy Mode". Pigeon Mode is a timing mode which can be independently enabled on any of the timing signals, with a unified flexible configuration. Signals within pigeon mode can be programmed into any supported signals, and are interchangeable. The following are the legacy timing signals which support pigeon mode:

- PIGEON[00] – LCD_DATA16
- PIGEON[01] – LCD_DATA17
- PIGEON[02] – LCD_DATA18
- PIGEON[03] – LCD_DATA19
- PIGEON[04] – LCD_DATA20
- PIGEON[05] – LCD_DATA21
- PIGEON[06] – LCD_DATA22
- PIGEON[07] – LCD_DATA23
- PIGEON[08] – LCD_ENABLE
- PIGEON[09] – LCD_HSYNC
- PIGEON[10] – LCD_VSYNC
- PIGEON[11] – LCD_CLK

Each pigeon signal has one local counter with a configurable start point and incremental condition. It will be compared to configuration register value for signal assertion/de-assertion control, plus delta offset for data alignment and other options like polarity/logic operation. A detailed running scenario is as follows:

1. Start local counter on the MASK rising edge (reference point/start point)
2. Increment on event selected through INC_SEL
3. Count and match SET_CNT: assert signal, reset counter (SET_CNT==0 means assert immediately on MASK's rising edge)
4. Count and match CLR_CNT: de-assert signal, stop counter (CLR_CNT==0 means de-assert on MASK's falling edge)

NOTE

When local counter is running, further changes to MASK are not cared unless CLR_CNT==0

MASK is the start point for local counter created using a combination any of the options (ANDed) from below:

1. STATE_MASK= (FS|FB|FD|FE) AND (LS|LB|LD|LE)

   In this case you have 8 bits to select in which vertical/horizontal state your counter starts ticking. This is the most common use-case because timing signals generally relate to scan states.

   For example, for a line timing signal start on Line Begin phase during Frame Begin/ Frame Data lines, use the configuration below:

Table 13-4. State Mask Combinations

STATE_MASK	LS	LB=1	LD	LE
FS				
FB=1		X		
FD=1		X		
FE				

2. MASK_CNT/MASK_CNT_SEL (global counter)

   Sometimes a more accurate reference point is required, such as "line 20 in a frame", or "line 12 in Frame Begin state" or "pixel 23 in LD phase". For such use-cases, several Global Counters shared by all pigeons are provided. Global counter type (line counter/frame counter/state counter, etc.) is selected using MASK_CNT_SEL, when global counter matches the MASK_CNT value. The pigeon local counter will start ticking.

3. Use another pigeon signal as mask (SIG_LOGIC=MASK)

   For some tightly coupled signals it is possible to use one as a reference to generate another.

   INC_SEL- select local counter tick event
   a. pclk - pixel clk
   b. line - line start pulse
   c. frame - frame start pulse
   d. another - another pigeon signal

   OFFSET- offset on pclk basis: Some signals need to come out slightly earlier or later than programmed. For example, the CE type signal usually aligns with the Line Data phase, but some panels need it as one cycle pulse before Line Data. The user can set OFFSET to a negative value to achieve this.

Global counters (selectable through MASK_CNT_SEL):

- HCNT / VCNT : normal pclk counter / line counter
- HSTATE_CNT / VSTATE_CNT: similar to above, but reset when state changes
- HSTATE_CYCLE / VSTATE_CYCLE: (see figure below for definition of CYCLE/PERIOD/CNT)

  Some panels have multiple Gate Drivers/Source Drivers, so Frame Data / Line Data state may be further split to match each driver, and signals such as CE[n] are only valid during part n of Line Data. For such signals, use LCDIF_PIGEON_CTRL[*_PERIOD] to specify PERIOD where CNT is reset and CYCLE is included. Then the user can select CYCLE as MASK_CNT to generate mask for CE[n].

- FRAME_CNT / FRAME_CYCLE (only for frame-crossing signals): frame cycle counter doesn't have a reset condition; use LCDIF_PIGEON_CTRL1[FRAME_CNT_CYCLES] to reset it.

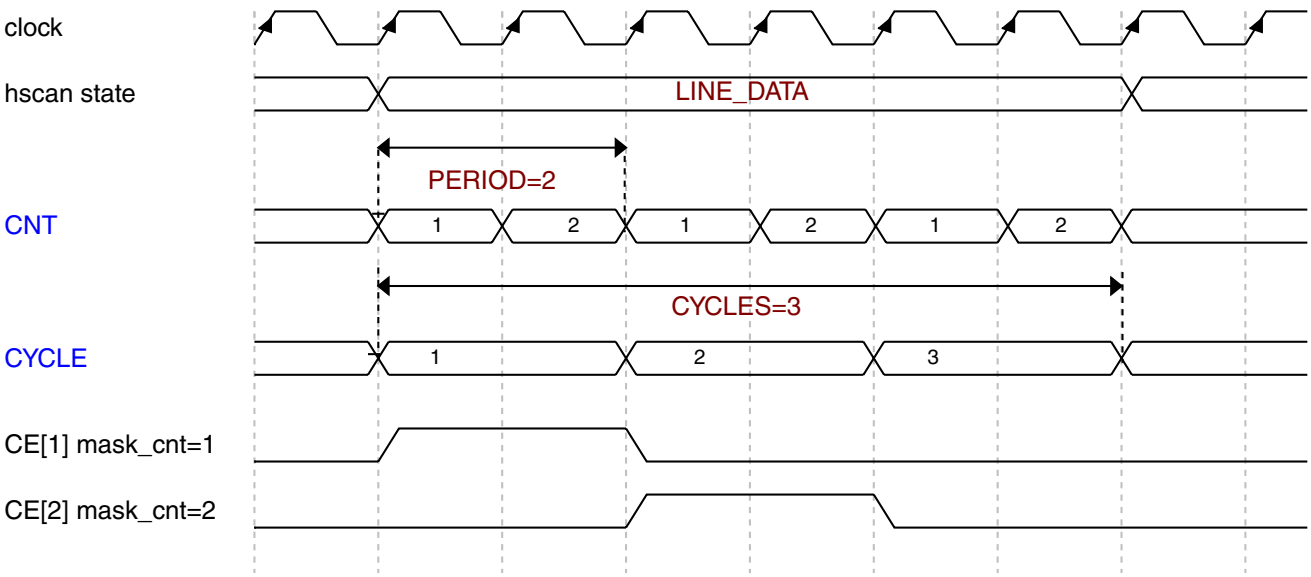


Figure 13-17. Definition of CNT, CYCLE, PERIOD

The following are register settings for the figure above:

- LCDIF_PIGEON_CTRL0[LD_PERIOD]=2
- MASK_CNT_SEL = 1 // HSTATE_CYCLE
- MASK_CNT = 1 // CE[1]
- MASK_CNT = 2 // CE[2]

13.2.3 Behavior During Reset

BUS CLOCK (apb_clk) and DISPLAY CLOCK (pix_clk) must be running before making any changes to SFTRST or CLKGATE bits.

A soft reset (SFTRST) can take multiple clock periods to complete, so do not set CLKGATE when setting SFTRST.

The reset process gates the clocks automatically.

13.2.4 LCDIF Memory Map/Register Definition

Some of the LCDIF registers (XXX_SET, XXX_CLR, and XXX_TOG) allow direct bit field masking and access.

- When writing 1 to XXX_SET bit fields, these registers allow setting the masked 1 bit fields, while keeping unchanged all bit fields which remain on 0 logic state.

- When writing 1 to XXX_CLR bit fields, these registers allow clearing the masked 1 bit fields, while keeping unchanged all other bit fields which remained on 0 logic state.
- When writing 1 to XXX_TOG bit fields, these registers allow inverting the logic state of all masked 1 bit fields, while they keep unchanged the remaining bit fields which were kept on 0 logic state.

LCDIF Hardware Register Format Summary

LCDIF memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E0_0000	LCDIF General Control Register (LCDIF_CTRL)	32	R/W	C000_0000h	13.2.4.1/ 3921
32E0_0004	LCDIF General Control Register (LCDIF_CTRL_SET)	32	R/W	C000_0000h	13.2.4.1/ 3921
32E0_0008	LCDIF General Control Register (LCDIF_CTRL_CLR)	32	R/W	C000_0000h	13.2.4.1/ 3921
32E0_000C	LCDIF General Control Register (LCDIF_CTRL_TOG)	32	R/W	C000_0000h	13.2.4.1/ 3921
32E0_0010	LCDIF General Control1 Register (LCDIF_CTRL1)	32	R/W	000F_0000h	13.2.4.2/ 3924
32E0_0014	LCDIF General Control1 Register (LCDIF_CTRL1_SET)	32	R/W	000F_0000h	13.2.4.2/ 3924
32E0_0018	LCDIF General Control1 Register (LCDIF_CTRL1_CLR)	32	R/W	000F_0000h	13.2.4.2/ 3924
32E0_001C	LCDIF General Control1 Register (LCDIF_CTRL1_TOG)	32	R/W	000F_0000h	13.2.4.2/ 3924
32E0_0020	LCDIF General Control2 Register (LCDIF_CTRL2)	32	R/W	0020_0000h	13.2.4.3/ 3926
32E0_0024	LCDIF General Control2 Register (LCDIF_CTRL2_SET)	32	R/W	0020_0000h	13.2.4.3/ 3926
32E0_0028	LCDIF General Control2 Register (LCDIF_CTRL2_CLR)	32	R/W	0020_0000h	13.2.4.3/ 3926
32E0_002C	LCDIF General Control2 Register (LCDIF_CTRL2_TOG)	32	R/W	0020_0000h	13.2.4.3/ 3926
32E0_0030	LCDIF Horizontal and Vertical Valid Data Count Register (LCDIF_TRANSFER_COUNT)	32	R/W	0001_0000h	13.2.4.4/ 3929
32E0_0040	LCD Interface Current Buffer Address Register (LCDIF_CUR_BUF)	32	R/W	0000_0000h	13.2.4.5/ 3929
32E0_0050	LCD Interface Next Buffer Address Register (LCDIF_NEXT_BUF)	32	R/W	0000_0000h	13.2.4.6/ 3930
32E0_0060	LCD Interface Timing Register (LCDIF_TIMING)	32	R/W	0000_0000h	13.2.4.7/ 3930

Table continues on the next page...

LCDIF memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E0_0070	LCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF_VDCTRL0)	32	R/W	0000_0000h	13.2.4.8/ 3931
32E0_0074	LCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF_VDCTRL0_SET)	32	R/W	0000_0000h	13.2.4.8/ 3931
32E0_0078	LCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF_VDCTRL0_CLR)	32	R/W	0000_0000h	13.2.4.8/ 3931
32E0_007C	LCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF_VDCTRL0_TOG)	32	R/W	0000_0000h	13.2.4.8/ 3931
32E0_0080	LCDIF VSYNC Mode and Dotclk Mode Control Register1 (LCDIF_VDCTRL1)	32	R/W	0000_0000h	13.2.4.9/ 3932
32E0_0090	LCDIF VSYNC Mode and Dotclk Mode Control Register2 (LCDIF_VDCTRL2)	32	R/W	0000_0000h	13.2.4.10/ 3933
32E0_00A0	LCDIF VSYNC Mode and Dotclk Mode Control Register3 (LCDIF_VDCTRL3)	32	R/W	0000_0000h	13.2.4.11/ 3933
32E0_00B0	LCDIF VSYNC Mode and Dotclk Mode Control Register4 (LCDIF_VDCTRL4)	32	R/W	0000_0000h	13.2.4.12/ 3934
32E0_00C0	Digital Video Interface Control0 Register (LCDIF_DVICTRL0)	32	R/W	0000_0000h	13.2.4.13/ 3935
32E0_00D0	Digital Video Interface Control1 Register (LCDIF_DVICTRL1)	32	R/W	0000_0000h	13.2.4.14/ 3936
32E0_00E0	Digital Video Interface Control2 Register (LCDIF_DVICTRL2)	32	R/W	0000_0000h	13.2.4.15/ 3937
32E0_00F0	Digital Video Interface Control3 Register (LCDIF_DVICTRL3)	32	R/W	0000_0000h	13.2.4.16/ 3938
32E0_0100	Digital Video Interface Control4 Register (LCDIF_DVICTRL4)	32	R/W	0000_0000h	13.2.4.17/ 3939
32E0_0110	RGB to YCbCr 4:2:2 CSC Coefficient0 Register (LCDIF_CSC_COEFF0)	32	R/W	0000_0000h	13.2.4.18/ 3940
32E0_0120	RGB to YCbCr 4:2:2 CSC Coefficient1 Register (LCDIF_CSC_COEFF1)	32	R/W	0000_0000h	13.2.4.19/ 3941
32E0_0130	RGB to YCbCr 4:2:2 CSC Coefficent2 Register (LCDIF_CSC_COEFF2)	32	R/W	0000_0000h	13.2.4.20/ 3941
32E0_0140	RGB to YCbCr 4:2:2 CSC Coefficient3 Register (LCDIF_CSC_COEFF3)	32	R/W	0000_0000h	13.2.4.21/ 3942
32E0_0150	RGB to YCbCr 4:2:2 CSC Coefficient4 Register (LCDIF_CSC_COEFF4)	32	R/W	0000_0000h	13.2.4.22/ 3943
32E0_0160	RGB to YCbCr 4:2:2 CSC Offset Register (LCDIF_CSC_OFFSET)	32	R/W	0080_0010h	13.2.4.23/ 3944
32E0_0170	RGB to YCbCr 4:2:2 CSC Limit Register (LCDIF_CSC_LIMIT)	32	R/W	00FF_00FFh	13.2.4.24/ 3944
32E0_0180	LCD Interface Data Register (LCDIF_DATA)	32	R/W	0000_0000h	13.2.4.25/ 3945
32E0_0190	Bus Master Error Status Register (LCDIF_BM_ERROR_STAT)	32	R/W	0000_0000h	13.2.4.26/ 3946

Table continues on the next page...

LCDIF memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E0_01A0	CRC Status Register (LCDIF_CRC_STAT)	32	R/W	0000_0000h	13.2.4.27/ 3946
32E0_01B0	LCD Interface Status Register (LCDIF_STAT)	32	R	9500_0000h	13.2.4.28/ 3947
32E0_0200	LCDIF Threshold Register (LCDIF_THRES)	32	R/W	0100_000Fh	13.2.4.29/ 3949
32E0_0210	LCDIF AS Buffer Control Register (LCDIF_AS_CTRL)	32	R/W	0000_0000h	13.2.4.30/ 3950
32E0_0220	Alpha Surface Buffer Pointer (LCDIF_AS_BUF)	32	R/W	0000_0000h	13.2.4.31/ 3952
32E0_0230	LCDIF_AS_NEXT_BUF	32	R/W	0000_0000h	13.2.4.32/ 3953
32E0_0240	LCDIF Overlay Color Key Low (LCDIF_AS_CLRKEYLOW)	32	R/W	00FF_FFFFh	13.2.4.33/ 3953
32E0_0250	LCDIF Overlay Color Key High (LCDIF_AS_CLRKEYHIGH)	32	R/W	0000_0000h	13.2.4.34/ 3954
32E0_0260	LCD working insync mode with CSI for VSYNC delay (LCDIF_SYNC_DELAY)	32	R/W	0000_0000h	13.2.4.35/ 3954
32E0_0380	LCDIF Pigeon Mode Control0 Register (LCDIF_PIGEONCTRL0)	32	R/W	0000_0000h	13.2.4.36/ 3955
32E0_0384	LCDIF Pigeon Mode Control0 Register (LCDIF_PIGEONCTRL0_SET)	32	R/W	0000_0000h	13.2.4.36/ 3955
32E0_0388	LCDIF Pigeon Mode Control0 Register (LCDIF_PIGEONCTRL0_CLR)	32	R/W	0000_0000h	13.2.4.36/ 3955
32E0_038C	LCDIF Pigeon Mode Control0 Register (LCDIF_PIGEONCTRL0_TOG)	32	R/W	0000_0000h	13.2.4.36/ 3955
32E0_0390	LCDIF Pigeon Mode Control1 Register (LCDIF_PIGEONCTRL1)	32	R/W	0000_0000h	13.2.4.37/ 3955
32E0_0394	LCDIF Pigeon Mode Control1 Register (LCDIF_PIGEONCTRL1_SET)	32	R/W	0000_0000h	13.2.4.37/ 3955
32E0_0398	LCDIF Pigeon Mode Control1 Register (LCDIF_PIGEONCTRL1_CLR)	32	R/W	0000_0000h	13.2.4.37/ 3955
32E0_039C	LCDIF Pigeon Mode Control1 Register (LCDIF_PIGEONCTRL1_TOG)	32	R/W	0000_0000h	13.2.4.37/ 3955
32E0_03A0	LCDIF Pigeon Mode Control2 Register (LCDIF_PIGEONCTRL2)	32	R/W	0000_0000h	13.2.4.38/ 3956
32E0_03A4	LCDIF Pigeon Mode Control2 Register (LCDIF_PIGEONCTRL2_SET)	32	R/W	0000_0000h	13.2.4.38/ 3956
32E0_03A8	LCDIF Pigeon Mode Control2 Register (LCDIF_PIGEONCTRL2_CLR)	32	R/W	0000_0000h	13.2.4.38/ 3956
32E0_03AC	LCDIF Pigeon Mode Control2 Register (LCDIF_PIGEONCTRL2_TOG)	32	R/W	0000_0000h	13.2.4.38/ 3956
32E0_0800	Panel Interface Signal Generator Register (LCDIF_PIGEON_0_0)	32	R/W	0000_0000h	13.2.4.39/ 3957

Table continues on the next page...

LCDIF memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E0_0810	Panel Interface Signal Generator Register (LCDIF_PIGEON_0_1)	32	R/W	0000_0000h	13.2.4.40/ 3958
32E0_0820	Panel Interface Signal Generator Register (LCDIF_PIGEON_0_2)	32	R/W	0000_0000h	13.2.4.41/ 3958
32E0_0840	Panel Interface Signal Generator Register (LCDIF_PIGEON_1_0)	32	R/W	0000_0000h	13.2.4.39/ 3957
32E0_0850	Panel Interface Signal Generator Register (LCDIF_PIGEON_1_1)	32	R/W	0000_0000h	13.2.4.40/ 3958
32E0_0860	Panel Interface Signal Generator Register (LCDIF_PIGEON_1_2)	32	R/W	0000_0000h	13.2.4.41/ 3958
32E0_0880	Panel Interface Signal Generator Register (LCDIF_PIGEON_2_0)	32	R/W	0000_0000h	13.2.4.39/ 3957
32E0_0890	Panel Interface Signal Generator Register (LCDIF_PIGEON_2_1)	32	R/W	0000_0000h	13.2.4.40/ 3958
32E0_08A0	Panel Interface Signal Generator Register (LCDIF_PIGEON_2_2)	32	R/W	0000_0000h	13.2.4.41/ 3958
32E0_08C0	Panel Interface Signal Generator Register (LCDIF_PIGEON_3_0)	32	R/W	0000_0000h	13.2.4.39/ 3957
32E0_08D0	Panel Interface Signal Generator Register (LCDIF_PIGEON_3_1)	32	R/W	0000_0000h	13.2.4.40/ 3958
32E0_08E0	Panel Interface Signal Generator Register (LCDIF_PIGEON_3_2)	32	R/W	0000_0000h	13.2.4.41/ 3958
32E0_0900	Panel Interface Signal Generator Register (LCDIF_PIGEON_4_0)	32	R/W	0000_0000h	13.2.4.39/ 3957
32E0_0910	Panel Interface Signal Generator Register (LCDIF_PIGEON_4_1)	32	R/W	0000_0000h	13.2.4.40/ 3958
32E0_0920	Panel Interface Signal Generator Register (LCDIF_PIGEON_4_2)	32	R/W	0000_0000h	13.2.4.41/ 3958
32E0_0940	Panel Interface Signal Generator Register (LCDIF_PIGEON_5_0)	32	R/W	0000_0000h	13.2.4.39/ 3957
32E0_0950	Panel Interface Signal Generator Register (LCDIF_PIGEON_5_1)	32	R/W	0000_0000h	13.2.4.40/ 3958
32E0_0960	Panel Interface Signal Generator Register (LCDIF_PIGEON_5_2)	32	R/W	0000_0000h	13.2.4.41/ 3958
32E0_0980	Panel Interface Signal Generator Register (LCDIF_PIGEON_6_0)	32	R/W	0000_0000h	13.2.4.39/ 3957
32E0_0990	Panel Interface Signal Generator Register (LCDIF_PIGEON_6_1)	32	R/W	0000_0000h	13.2.4.40/ 3958
32E0_09A0	Panel Interface Signal Generator Register (LCDIF_PIGEON_6_2)	32	R/W	0000_0000h	13.2.4.41/ 3958
32E0_09C0	Panel Interface Signal Generator Register (LCDIF_PIGEON_7_0)	32	R/W	0000_0000h	13.2.4.39/ 3957
32E0_09D0	Panel Interface Signal Generator Register (LCDIF_PIGEON_7_1)	32	R/W	0000_0000h	13.2.4.40/ 3958

Table continues on the next page...

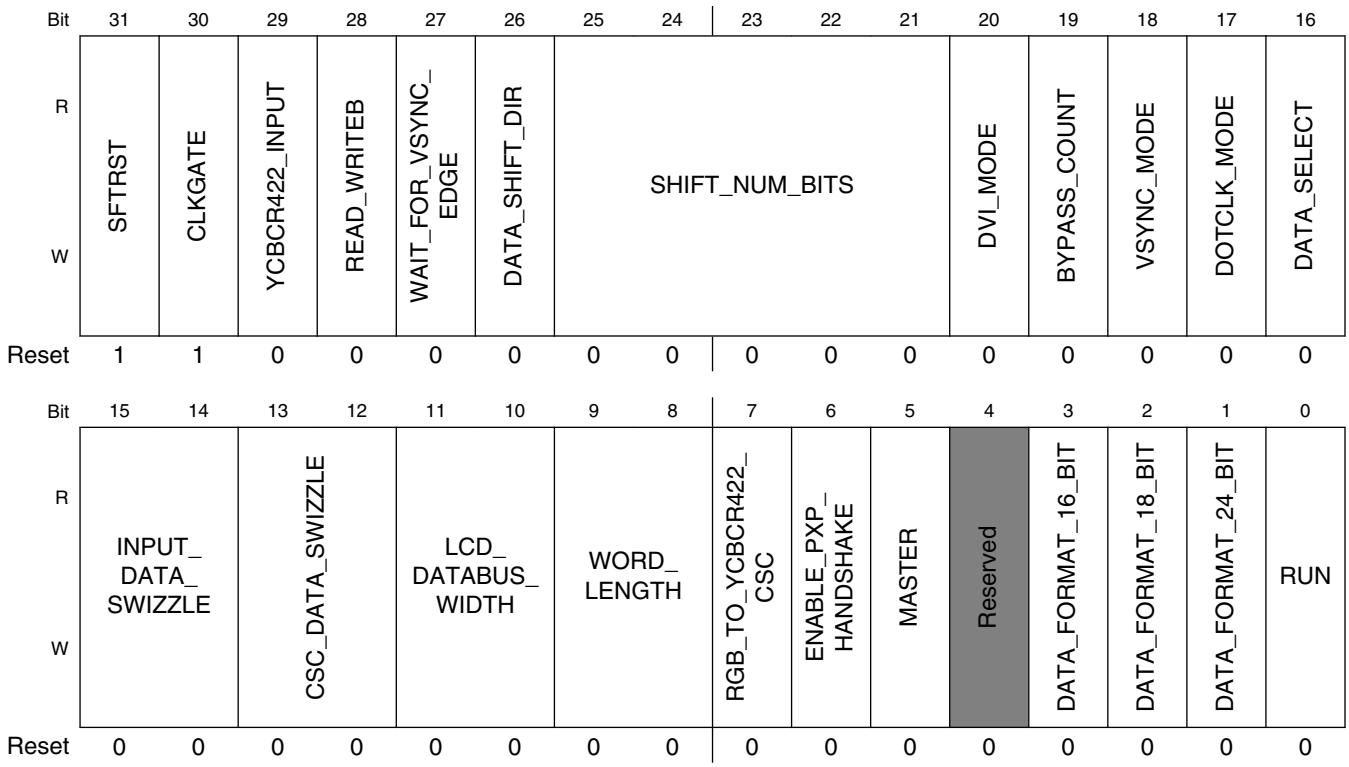
LCDIF memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E0_09E0	Panel Interface Signal Generator Register (LCDIF_PIGEON_7_2)	32	R/W	0000_0000h	13.2.4.41/ 3958
32E0_0A00	Panel Interface Signal Generator Register (LCDIF_PIGEON_8_0)	32	R/W	0000_0000h	13.2.4.39/ 3957
32E0_0A10	Panel Interface Signal Generator Register (LCDIF_PIGEON_8_1)	32	R/W	0000_0000h	13.2.4.40/ 3958
32E0_0A20	Panel Interface Signal Generator Register (LCDIF_PIGEON_8_2)	32	R/W	0000_0000h	13.2.4.41/ 3958
32E0_0A40	Panel Interface Signal Generator Register (LCDIF_PIGEON_9_0)	32	R/W	0000_0000h	13.2.4.39/ 3957
32E0_0A50	Panel Interface Signal Generator Register (LCDIF_PIGEON_9_1)	32	R/W	0000_0000h	13.2.4.40/ 3958
32E0_0A60	Panel Interface Signal Generator Register (LCDIF_PIGEON_9_2)	32	R/W	0000_0000h	13.2.4.41/ 3958
32E0_0A80	Panel Interface Signal Generator Register (LCDIF_PIGEON_10_0)	32	R/W	0000_0000h	13.2.4.39/ 3957
32E0_0A90	Panel Interface Signal Generator Register (LCDIF_PIGEON_10_1)	32	R/W	0000_0000h	13.2.4.40/ 3958
32E0_0AA0	Panel Interface Signal Generator Register (LCDIF_PIGEON_10_2)	32	R/W	0000_0000h	13.2.4.41/ 3958
32E0_0AC0	Panel Interface Signal Generator Register (LCDIF_PIGEON_11_0)	32	R/W	0000_0000h	13.2.4.39/ 3957
32E0_0AD0	Panel Interface Signal Generator Register (LCDIF_PIGEON_11_1)	32	R/W	0000_0000h	13.2.4.40/ 3958
32E0_0AE0	Panel Interface Signal Generator Register (LCDIF_PIGEON_11_2)	32	R/W	0000_0000h	13.2.4.41/ 3958

13.2.4.1 LCDIF General Control Register (LCDIF_CTRL*n*)

The LCD Interface Control Register provides overall control of the LCDIF block. The LCDIF Control Register provides a variety of control functions to the programmer. These functions allow the interface to be very flexible to work with a variety of LCD controllers, and to minimize overhead and increase performance of LCD programming. The register has been organized such that switching between the different LCD modes can be done with minimum PIO writes.

Address: 32E0_0000h base + 0h offset + (4d × i), where i=0d to 3d



LCDIF_CTRL*n* field descriptions

Field	Description
31 SFTRST	This bit must be set to zero to enable normal operation of the LCDIF. When set to one, it forces a block level reset.
30 CLKGATE	This bit must be set to zero for normal operation. When set to one it gates off the clocks to the block.
29 YCBCR422_INPUT	Zero implies input data is in RGB color space. One implies input data is in YCbCr 4:2:2 format, such that YCbYCr are packed in a 32-bit word. It also means that there are 2 pixels in 4 bytes. If this bit is set, software should program the H_COUNT field in the TRANSFER_COUNT register to the total number of pixels that will have to be fetched by the LCDIF block per line and the BYTE_PACKING_FORMAT should be 0xF. The WORD_LENGTH does not matter in this case.

Table continues on the next page...

LCDIF_CTRL*n* field descriptions (continued)

Field	Description
28 READ_WRITEB	By default, LCDIF is in the write mode. Setting this bit to 1 will make the hardware go into 6800/8080 MPU read mode. The LCDIF_MASTER bit must be 0, since bus master mode can only be used for writing the display.
27 WAIT_FOR_ VSYNC_EDGE	Setting this bit to 1 will make the hardware wait for the triggering VSYNC edge before starting write transfers to the LCD. Used only in the VSYNC mode of operation.
26 DATA_SHIFT_ DIR	Use this bit to determine the direction of shift of transmit data. In the DVI mode, it works only on the active data, not on the timing codes and ancillary data. 0x0 **TXDATA_SHIFT_LEFT** — Data to be transmitted is shifted LEFT by SHIFT_NUM_BITS bits. 0x1 **TXDATA_SHIFT_RIGHT** — Data to be transmitted is shifted RIGHT by SHIFT_NUM_BITS bits.
25–21 SHIFT_NUM_ BITS	The data to be transmitted is shifted left or right by this number of bits.
20 DVI_MODE	Set this bit to 1 to get into the ITU-R BT.656 digital video interface mode. Toggle this bit from 1 to 0 to make the hardware go out of DVI mode after completing all data transfer and after the RUN bit has been deasserted.
19 BYPASS_ COUNT	When this bit is 0, it means that LCDIF will stop the block operation and turn off the RUN bit after the amount of data indicated by the LCDIF_TRANSFER_COUNT register has been transferred out. When this bit is set to 1, the block will continue normal operation indefinitely until it is told to stop. This bit must be 0 in MPU and VSYNC modes, and must be 1 in DOTCLK and DVI modes of operation.
18 VSYNC_MODE	Setting this bit to 1 will make the LCDIF hardware go into VSYNC mode. WAIT_FOR_VSYNC_EDGE can be used only if this bit is set. If VSYNC signal is required to be an output from the block, SYNC_SIGNALS_ON bit in LCDIF_VDCTRL4 register must be set.
17 DOTCLK_MODE	Set this bit to 1 to make the hardware go into the DOTCLK mode, i.e. VSYNC/HSYNC/DOTCLK/ENABLE interface mode. ENABLE is optional, selected by the ENABLE_PRESENT bit. Toggle this bit from 1 to 0 to make the hardware go out of DOTCLK mode after completing all data transfer and deasserting the RUN bit.
16 DATA_SELECT	Command Mode polarity bit. This bit should only be changed when RUN is 0. 0x0 **CMD_MODE** — Command Mode. LCD_RS signal is Low. 0x1 **DATA_MODE** — Data Mode. LCD_RS signal is High.
15–14 INPUT_DATA_ SWIZZLE	This field specifies how to swap the bytes fetched by the bus master interface. The swizzle function is independent of the WORD_LENGTH bit. The supported swizzle configurations are: 0x0 **NO_SWAP** — No byte swapping.(Little endian) 0x0 **LITTLE_ENDIAN** — Little Endian byte ordering (same as NO_SWAP). 0x1 **BIG_ENDIAN_SWAP** — Big Endian swap (swap bytes 0,3 and 1,2). 0x1 **SWAP_ALL_BYTES** — Swizzle all bytes, swap bytes 0,3 and 1,2 (aka Big Endian). 0x2 **HWD_SWAP** — Swap half-words. 0x3 **HWD_BYTE_SWAP** — Swap bytes within each half-word.
13–12 CSC_DATA_ SWIZZLE	This field specifies how to swap the bytes after the data has been converted into an internal representation of 24 bits per pixel and before it is transmitted over the LCD interface bus. The data is always transmitted with the least significant byte/hword (half word) first after the swizzle takes place. So, INPUT_DATA_SWIZZLE takes place first on the incoming data, and then CSC_DATA_SWIZZLE is applied. The swizzle function is independent of the WORD_LENGTH or the LCD_DATABUS_WIDTH fields. If RGB_TO_YCRCB422_CSC bit is set, the swizzle occurs on the Y, Cb, Cr values. The supported swizzle configurations are: 0x0 **NO_SWAP** — No byte swapping.(Little endian)

Table continues on the next page...

LCDIF_CTRL*n* field descriptions (continued)

Field	Description
	0x0 **LITTLE_ENDIAN** — Little Endian byte ordering (same as NO_SWAP). 0x1 **BIG_ENDIAN_SWAP** — Big Endian swap (swap bytes 0,3 and 1,2). 0x1 **SWAP_ALL_BYTES** — Swizzle all bytes, swap bytes 0,3 and 1,2 (aka Big Endian). 0x2 **HWD_SWAP** — Swap half-words. 0x3 **HWD_BYTE_SWAP** — Swap bytes within each half-word.
11–10 LCD_DATABUS_WIDTH	LCD Data bus transfer width. 0x0 **16_BIT** — 16-bit data bus mode. 0x1 **8_BIT** — 8-bit data bus mode. 0x2 **18_BIT** — 18-bit data bus mode. 0x3 **24_BIT** — 24-bit data bus mode.
9–8 WORD_LENGTH	Input data format. 0x0 **16_BIT** — Input data is 16 bits per pixel. 0x1 **8_BIT** — Input data is 8 bits wide. 0x2 **18_BIT** — Input data is 18 bits per pixel. 0x3 **24_BIT** — Input data is 24 bits per pixel.
7 RGB_TO_YCBCR422_CSC	Set this bit to 1 to enable conversion from RGB to YCbCr colorspace. See the LCDIF_CSC_ registers for further details.
6 ENABLE_PXP_HANDSHAKE	If this bit is set and LCDIF_MASTER bit is set, the LCDIF will act as bus master and the handshake mechanism between LCDIF and PXP will be turned on. If LCDIF_MASTER bit is not set, this bit becomes a don't care.
5 MASTER	Set this bit to make the LCDIF act as a bus master.
4 RSRVD0	This field is reserved. Reserved bits. Write as 0.
3 DATA_FORMAT_16_BIT	When this bit is 1 and WORD_LENGTH = 0, it implies that the 16-bit data is in ARGB555 format. When this bit is 0 and WORD_LENGTH = 0, it implies that the 16-bit data is in RGB565 format. When WORD_LENGTH is not 0, this bit does not care.
2 DATA_FORMAT_18_BIT	Used only when WORD_LENGTH = 2, i.e. 18-bit. 0x0 **LOWER_18_BITS_VALID** — Data input to the block is in 18 bpp format, such that lower 18 bits contain RGB 666 and upper 14 bits do not contain any useful data. 0x1 **UPPER_18_BITS_VALID** — Data input to the block is in 18 bpp format, such that upper 18 bits contain RGB 666 and lower 14 bits do not contain any useful data.
1 DATA_FORMAT_24_BIT	Used only when WORD_LENGTH = 3, i.e. 24-bit. Note that this applies to both packed and unpacked 24-bit data. 0x0 **ALL_24_BITS_VALID** — Data input to the block is in 24 bpp format, such that all RGB 888 data is contained in 24 bits. 0x1 **DROP_UPPER_2_BITS_PER_BYTE** — Data input to the block is actually RGB 18 bpp, but there is 1 color per byte, hence the upper 2 bits in each byte do not contain any useful data, and should be dropped.
0 RUN	When this bit is set by software, the LCDIF will begin transferring data between the SoC and the display. This bit must remain set until the operation is complete.

13.2.4.2 LCDIF General Control1 Register (LCDIF_CTRL1*n*)

The LCDIF Control Register provides overall control of the LCDIF block.

The LCDIF Control1 Register provides additional programming to the LCDIF. It implements some bits which are unlikely to change often in a particular application. It also carries interrupt-related bits which are common across more than one mode of operation.

Address: 32E0_0000h base + 10h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved		COMBINE_MPU_WR_STRB	BM_ERROR_IRQ_EN	BM_ERROR_IRQ	RECOVER_ON_UNDERFLOW	INTERLACE_FIELDS	START_INTERLACE_FROM_SECOND_FIELD	FIFO_CLEAR	IRQ_ON_ALTERNATE_FIELDS	BYTE_PACKING_FORMAT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OVERFLOW_IRQ_EN	UNDERFLOW_IRQ_EN	CUR_FRAME_DONE_IRQ_EN	VSYNC_EDGE_IRQ_EN	OVERFLOW_IRQ	UNDERFLOW_IRQ	CUR_FRAME_DONE_IRQ	VSYNC_EDGE_IRQ	Reserved					BUSY_ENABLE	MODE86	RESET
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_CTRL1*n* field descriptions

Field	Description
31 -	This field is reserved. Reserved bits. Write as 0.
30 -	This field is reserved. Reserved bits. Write as 0.
29–28 -	This field is reserved. Reserved bits. Write as 0.
27 COMBINE_MPU_WR_STRB	If this bit is not set, the write strobe will be driven on LCD_WR_RWn pin in the 8080 mode and on the LCD_RD_E pin in the 6800 mode. If it is set, the write strobe of both the 6800 and 8080 modes will be driven only on the LCD_WR_RWn pin. Note that this does not work for read strobe.
26 BM_ERROR_IRQ_EN	This bit is set to enable bus master error interrupt in the LCDIF master mode.

Table continues on the next page...

LCDIF_CTRL1*n* field descriptions (continued)

Field	Description
25 BM_ERROR_ IRQ	This bit is set to indicate that an interrupt is requested by the LCDIF block. This bit is cleared by software by writing a one to its SCT clear address. This bit will be set when the LCDIF is in master mode and an error response was returned by the slave. 0x0 **NO_REQUEST** — No Interrupt Request Pending. 0x1 **REQUEST** — Interrupt Request Pending.
24 RECOVER_ON_ UNDERFLOW	Set this bit to enable the LCDIF block to recover in the next field/frame if there was an underflow in the current field/frame.
23 INTERLACE_ FIELDS	Set this bit if it is required that the LCDIF block fetches odd lines in one field and even lines in the other field. It will work only in LCDIF_MASTER is set to 1.
22 START_ INTERLACE_ FROM_ SECOND_FIELD	The default is to grab the odd lines first and then the even lines. Set this bit if it is required to grab the even lines first and then the odd lines. (Line numbers start from 1, so odd lines are 1,3,5,etc. and even lines are 2,4,6, etc.)
21 FIFO_CLEAR	Set this bit to clear all the data in the latency FIFO (LFIFO), TXFIFO and the RXFIFO.
20 IRQ_ON_ ALTERNATE_ FIELDS	If this bit is set, the LCDIF block will assert the cur_frame_done interrupt only on alternate fields, otherwise it will issue the interrupt on both odd and even field. This bit is mostly relevant if INTERLACE_FIELDS is set.
19–16 BYTE_ PACKING_ FORMAT	This bitfield is used to show which data bytes in a 32-bit word are valid. Default value 0xf indicates that all bytes are valid. For 8-bit transfers, any combination in this bitfield will mean valid data is present in the corresponding bytes. In the 16-bit mode, a 16-bit half-word is valid only if adjacent bits [1:0] or [3:2] or both are 1. A value of 0x0 will mean that none of the bytes are valid and should not be used. For example, set the bit field value to 0x7 if the display data is arranged in the 24-bit unpacked format (A-R-G-B where A value does not have be transmitted). When input data is in YCbCr 4:2:2 format (YCBCR422_INPUT is 1), H_COUNT should be the number of pixels that should be fetched by the block and the BYTE_PACKING_FORMAT should be 0xF.(Note - YCBCR422_INPUT = 1 implies 2 pixels per 32 bits).
15 OVERFLOW_ IRQ_EN	This bit is set to enable an overflow interrupt in the TXFIFO in the write mode.
14 UNDERFLOW_ IRQ_EN	This bit is set to enable an underflow interrupt in the TXFIFO in the write mode.
13 CUR_FRAME_ DONE_IRQ_EN	This bit is set to 1 enable an interrupt every time the hardware enters in the vertical blanking state.
12 VSYNC_EDGE_ IRQ_EN	This bit is set to enable an interrupt every time the hardware encounters the leading VSYNC edge in the VSYNC and DOTCLK modes, or the beginning of every field in DVI mode.
11 OVERFLOW_ IRQ	This bit is set to indicate that an interrupt is requested by the LCDIF block. This bit is cleared by software by writing a one to its SCT clear address. A latency FIFO (LFIFO) overflow in the write mode (MPU/VSYNC/DOTCLK/DVI mode) was detected, data samples have been lost. 0x0 **NO_REQUEST** — No Interrupt Request Pending. 0x1 **REQUEST** — Interrupt Request Pending.

Table continues on the next page...

LCDIF_CTRL1*n* field descriptions (continued)

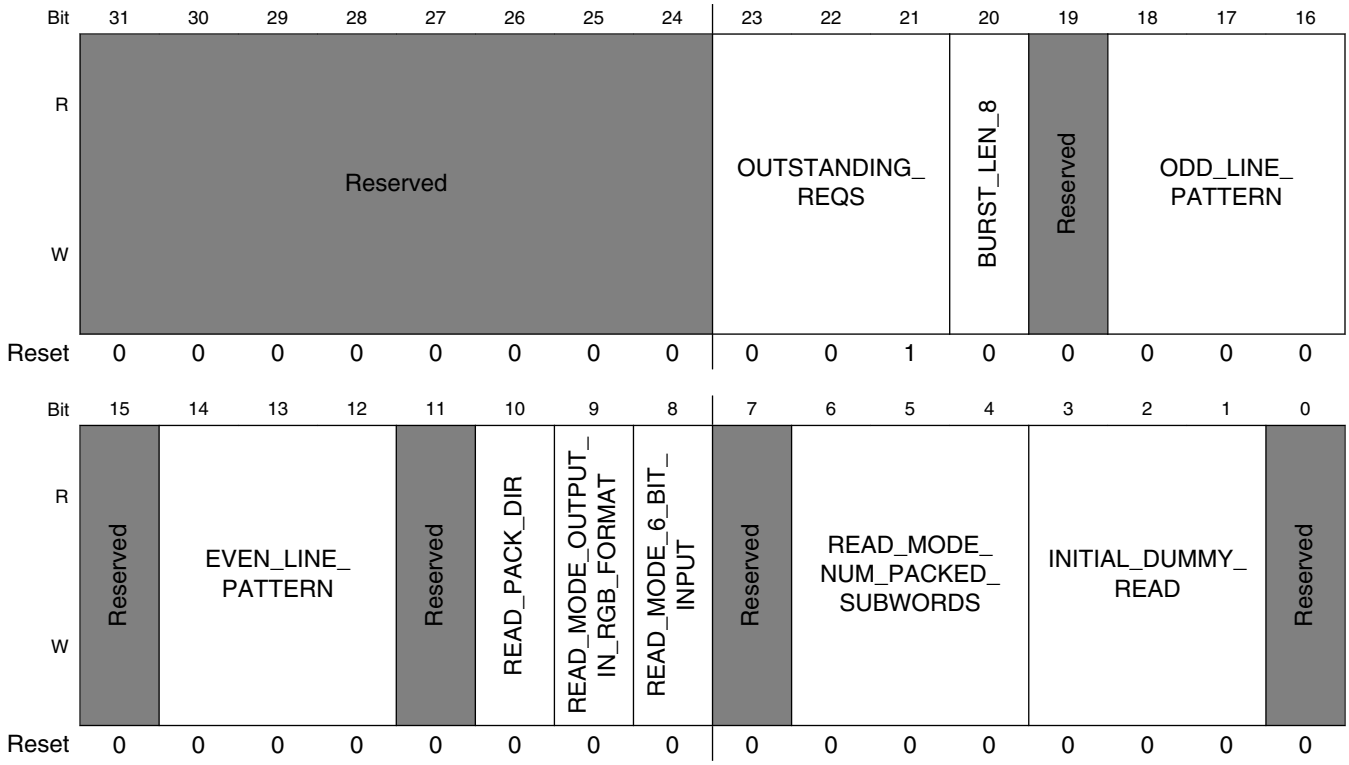
Field	Description
10 UNDERFLOW_IRQ	This bit is set to indicate that an interrupt is requested by the LCDIF block. This bit is cleared by software by writing a one to its SCT clear address. A TXFIFO underflow in the write mode (MPU/VSYNC/ DOTCLK/DVI mode) was detected. Could produce an error in the DOTCLK / DVI modes. 0x0 **NO_REQUEST** — No Interrupt Request Pending. 0x1 **REQUEST** — Interrupt Request Pending.
9 CUR_FRAME_DONE_IRQ	This bit is set to indicate that an interrupt is requested by the LCDIF block. This bit is cleared by software by writing a one to its SCT clear address. It indicates that the hardware has completed transmitting the current frame and is in the vertical blanking period in the DOTCLK/DVI modes. In the MPU and VSYNC modes, this IRQ is asserted at the end of the data transfer indicated by LCDIF_TRANSFER_COUNT register. 0x0 **NO_REQUEST** — No Interrupt Request Pending. 0x1 **REQUEST** — Interrupt Request Pending.
8 VSYNC_EDGE_IRQ	This bit is set to indicate that an interrupt is requested by the LCDIF block. This bit is cleared by software by writing a one to its SCT clear address. It is set whenever the leading VSYNC edge is detected in the VSYNC and DOTCLK modes. In the DVI mode, it is asserted every time the block enters a new field. 0x0 **NO_REQUEST** — No Interrupt Request Pending. 0x1 **REQUEST** — Interrupt Request Pending.
7–3 RSRVD0	This field is reserved. Reserved bits. Write as 0.
2 BUSY_ENABLE	This bit enables the use of the interface's busy signal input. This should be enabled for LCD controllers that implement a busy line (to stall the LCDIF from sending more data until ready). Otherwise this bit should be cleared. 0x0 **BUSY_DISABLED** — The busy signal from the LCD controller will be ignored. 0x1 **BUSY_ENABLED** — Enable the use of the busy signal from the LCD controller.
1 MODE86	This bit is used to select between the 8080 and 6800 series of microprocessor modes. This bit should only be changed when RUN is 0. 0x0 **8080_MODE** — Pins LCD_WR_RWn and LCD_RD_E function as active low WR and active low RD signals respectively. 0x1 **6800_MODE** — Pins LCD_WR_RWn and LCD_RD_E function as Read/Write and active high Enable signals respectively.
0 RESET	Reset bit for the external LCD controller. This bit can be changed at any time. It CANNOT be reset by SFTRST. 0x0 **LCDRESET_LOW** — LCD_RESET output signal is low. 0x1 **LCDRESET_HIGH** — LCD_RESET output signal is high.

13.2.4.3 LCDIF General Control2 Register (LCDIF_CTRL2*n*)

The LCDIF Control Register provides overall control of the LCDIF block.

The LCDIF Control2 Register provides additional programming to the LCDIF. It implements some bits which are unlikely to change often in a particular application.

Address: 32E0_0000h base + 20h offset + (4d × i), where i=0d to 3d



LCDIF_CTRL2*n* field descriptions

Field	Description
31–24 RSRVD5	This field is reserved. Reserved bits. Write as 0.
23–21 OUTSTANDING_ REQS	This bitfield indicates the maximum number of outstanding transactions that LCDIF should request when it is acting as a bus master. Default is 2 outstanding transactions. 0x0 **REQ_1** — 0x1 **REQ_2** — 0x2 **REQ_4** — 0x3 **REQ_8** — 0x4 **REQ_16** —
20 BURST_LEN_8	By default, when the LCDIF is in the bus master mode, it will issue AXI bursts of length 16 (except when in packed 24 bpp mode, it will issue bursts of length 15). When this bit is set to 1, the block will issue bursts of length 8 (except when in packed 24 bpp mode, it will issue bursts of length 9). Note that this bitfield is only applicable when LCDIF_MASTER is set to 1.
19 RSRVD4	This field is reserved. Reserved bits. Write as 0.
18–16 ODD_LINE_ PATTERN	This field determines the order of the RGB components of each pixel in ODD lines (line numbers 1,3,5,..). This bitfield must be 0 in DVI mode. 0x0 **RGB** — 0x1 **RBG** — 0x2 **GBR** — 0x3 **GRB** —

Table continues on the next page...

LCDIF_CTRL2*n* field descriptions (continued)

Field	Description
	0x4 **BRG** — 0x5 **BGR** —
15 RSRVD3	This field is reserved. Reserved bits. Write as 0.
14–12 EVEN_LINE_ PATTERN	This field determines the order of the RGB components of each pixel in EVEN lines (line numbers 2,4,6,..). This bitfield must be 0 in DVI mode. 0x0 **RGB** — 0x1 **RBG** — 0x2 **GBR** — 0x3 **GRB** — 0x4 **BRG** — 0x5 **BGR** —
11 RSRVD2	This field is reserved. Reserved bits. Write as 0.
10 READ_PACK_DIR	The default value of 0 indicates data is stored in the little endian format. When LCD_DATABUS_WIDTH is 8-bit, this bit provides the option of rearranging the data byte-wise in the big endian format. For example, if READ_MODE_NUM_PACKED_SUBWORDS = 3 and the order of incoming data is 0x11, 0x22 and 0x33, then setting this bit to 1 will cause the data to be stored as 0x00112233 as opposed to the default 0x00332211. This operation occurs after the shifting operation done by SHIFT_NUM_BITS bitfield.
9 READ_MODE_ OUTPUT_IN_ RGB_FORMAT	Setting this bit will enable the LCDIF to convert the incoming data to the RGB format given by WORD_LENGTH bitfield. This feature is not available when WORD_LENGTH is set to 8 bits. LCDIF performs this operation of converting to RGB format after the endianness has been determined by the READ_PACK_DIR bitfield.
8 READ_MODE_6_ BIT_INPUT	Setting this bit to 1 indicates to LCDIF that even though LCD_DATABUS_WIDTH is set to 8 bits, the input data is actually only 6 bits wide and exists on D5-D0.
7 RSRVD1	This field is reserved. Reserved bits. Write as 0.
6–4 READ_MODE_ NUM_PACKED_ SUBWORDS	Indicates the number of valid 8/16/18/24-bit subwords that will be packed into the 32-bit word in read mode. The subword size (8,16, 18 or 24 bits) is determined by the LCD_DATABUS_WIDTH field. The swizzle operation is performed after READ_MODE_NUM_PACKED_SUBWORDS number of subwords has been received and stored in little-endian format. For example, if LCD_DATABUS_WIDTH is set to 8-bit and data to be read back has to be stored in memory in 24-bit unpacked RGB format, set READ_MODE_NUM_PACKED_SUBWORDS to 0x3 so that each 32-bit word will contain only 3 valid bytes (RGB). Maximum value of READ_MODE_NUM_PACKED_SUBWORDS is 4 for 8-bit databus, 2 for 16-bit databus and 1 for 18/24-bit databus.
3–1 INITIAL_DUMMY_ READ	The value in this field determines the number of dummy 8/16/18/24-bit subwords that have to be read back from the LCD panel/controller. They will then not be stored in the read FIFO.
0 RSRVD0	This field is reserved. Reserved bits. Write as 0.

13.2.4.4 LCDIF Horizontal and Vertical Valid Data Count Register (LCDIF_TRANSFER_COUNT)

This register tells the LCDIF how much data will be sent for this frame, or transaction. The total number of words is a product of the V_COUNT and H_COUNT fields. The word size is specified by the WORD_LENGTH field.

This register gives the dimensions of the input frame. For normal operation, but V_COUNT and H_COUNT should be non-zero.

Address: 32E0_0000h base + 30h offset = 32E0_0030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	V_COUNT																H_COUNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_TRANSFER_COUNT field descriptions

Field	Description
31–16 V_COUNT	Number of horizontal lines per frame which contain valid data. In DOTCLK mode, V_COUNT should be the same as the number of active horizontal lines in a progressive frame. In DVI mode, V_COUNT should be the number of active horizontal lines per frame, and not per field.
H_COUNT	Total valid data (pixels) in each horizontal line. The data size is given by the WORD_LENGTH. When input data is in YCbCr 4:2:2 format (YCBCR422_INPUT is 1), H_COUNT should be the number of 32-bit words that should be fetched by the block and the BYTE_PACKING_FORMAT should be 0xF. In 24-bit packed format (WORD_LENGTH=0x3, BYTE_PACKING_FORMAT=0xF), the H_COUNT must be a multiple of 4 pixels. In 16-bit packed format (WORD_LENGTH=0x0, BYTE_PACKING_FORMAT=0xF), the H_COUNT must be a multiple of 2 pixels.

13.2.4.5 LCD Interface Current Buffer Address Register (LCDIF_CUR_BUF)

This register indicates the address of the current frame being transmitted by LCDIF.

When the LCDIF is behaving as a master, this address points to the address of the current frame of data being sent out via the LCDIF. When the current frame is done, the LCDIF block will assert the cur_frame_done interrupt for software to take action. The block will also copy the LCDIF_NEXT_BUF_ADDR into this bitfield so that the software can program the next frame address into the LCDIF_NEXT_BUF_ADDR bitfield. This address must always be double-word aligned.

Address: 32E0_0000h base + 40h offset = 32E0_0040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_CUR_BUF field descriptions

Field	Description
ADDR	Address of the current frame being transmitted by LCDIF.

13.2.4.6 LCD Interface Next Buffer Address Register (LCDIF_NEXT_BUF)

This register indicates the address of next frame that will be transmitted by LCDIF.

When the LCDIF is behaving as a master, this address points to the address of the next frame of data that will be sent out via the LCDIF. It is up to the software to make sure that this register is programmed before the end of the current frame, otherwise it might result in old data going out the LCDIF. This address must always be double-word aligned.

Address: 32E0_0000h base + 50h offset = 32E0_0050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_NEXT_BUF field descriptions

Field	Description
ADDR	Address of the next frame that will be transmitted by LCDIF.

13.2.4.7 LCD Interface Timing Register (LCDIF_TIMING)

The LCD interface timing register controls the various setup and hold times enforced by the LCD interface in the 6800/8080 MPU and VSYNC modes of operation.

The values used in this register are dependent on the particular LCD controller used, consult the users manual for the particular controller for required timings. Each field of the register must be non-zero, therefore the minimum value is: 0x01010101. NOTE: the timings are not automatically adjusted if the DISPLAY CLOCK (pix_clk) frequency changes—it may be necessary to adjust the timings if DISPLAY CLOCK (pix_clk) changes. NOTE: Each field in this register must be non-zero for the MPU and VSYNC modes to function. The settings in this register do not affect the DOTCLK and DVI modes.

Address: 32E0_0000h base + 60h offset = 32E0_0060h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CMD_HOLD								CMD_SETUP								DATA_HOLD								DATA_SETUP							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_TIMING field descriptions

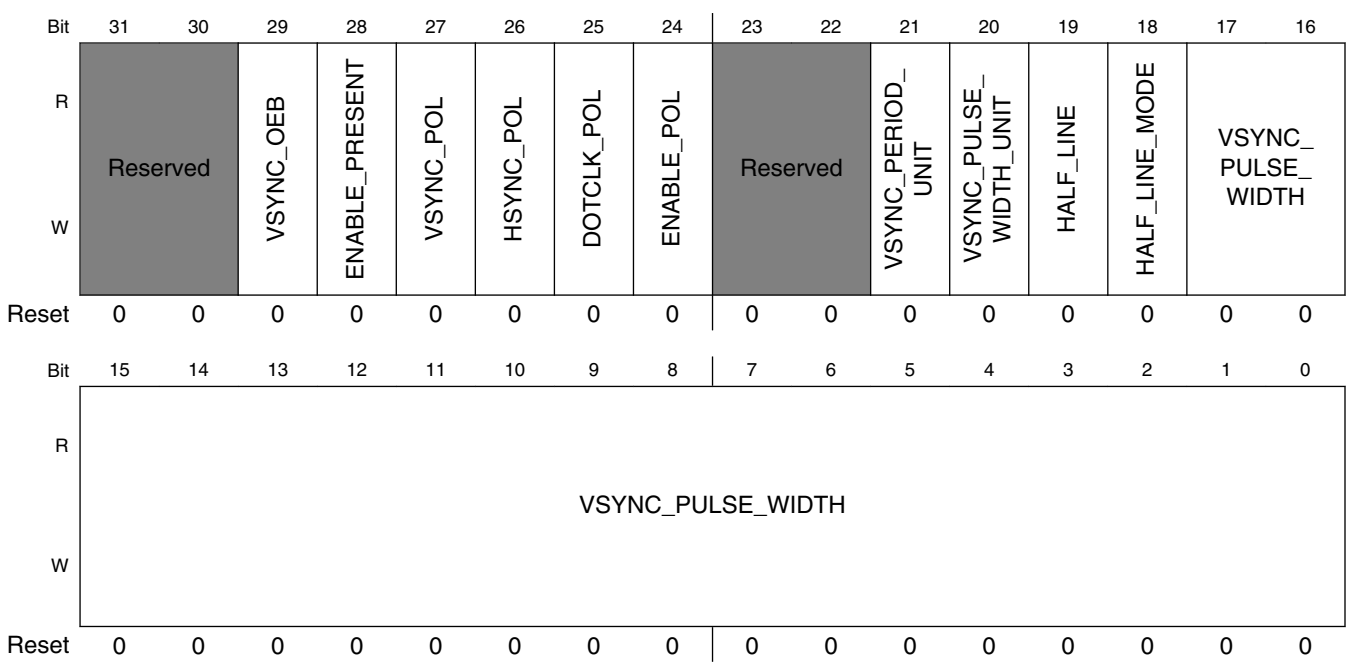
Field	Description
31–24 CMD_HOLD	Number of DISPLAY CLOCK (pix_clk) cycles that the LCD_RS signal is active after LCD_CS is deasserted.
23–16 CMD_SETUP	Number of DISPLAY CLOCK (pix_clk) cycles that the LCD_RS signal is active before LCD_CS is asserted.
15–8 DATA_HOLD	Data bus hold time in DISPLAY CLOCK (pix_clk) cycles. Also the time that the data strobe is deasserted in a cycle
DATA_SETUP	Data bus setup time in DISPLAY CLOCK (pix_clk) cycles. Also the time that the data strobe is asserted in a cycle.

13.2.4.8 LCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF_VDCTRL0*n*)

This register is used to control the VSYNC and DOTCLK modes of the LCDIF so as to work with different types of LCDs like moving picture displays and delta pixel displays.

This register gives general programmability to the VSYNC signal including polarity, direction, pulse width, etc.

Address: 32E0_0000h base + 70h offset + (4d × i), where i=0d to 3d



LCDIF_VDCTRL0*n* field descriptions

Field	Description
31–30 RSRVD2	This field is reserved. Reserved bits. Write as 0.
29 VSYNC_OEB	0 means the VSYNC signal is an output, 1 means it is an input. Should be set to 0 in the DOTCLK mode. 0x0 **VSYNC_OUTPUT** — The VSYNC pin is in the output mode and the VSYNC signal has to be generated by the LCDIF block. 0x1 **VSYNC_INPUT** — The VSYNC pin is in the input mode and the LCD controller sends the VSYNC signal to the block.
28 ENABLE_ PRESENT	Setting this bit to 1 will make the hardware generate the ENABLE signal in the DOTCLK mode, thereby making it the true RGB interface along with the remaining three signals VSYNC, HSYNC and DOTCLK.
27 VSYNC_POL	Default 0 active low during VSYNC_PULSE_WIDTH time and will be high during the rest of the VSYNC period. Set it to 1 to invert the polarity.
26 HSYNC_POL	Default 0 active low during HSYNC_PULSE_WIDTH time and will be high during the rest of the HSYNC period. Set it to 1 to invert the polarity.
25 DOTCLK_POL	Default is data launched at negative edge of DOTCLK and captured at positive edge. Set it to 1 to invert the polarity. Set it to 0 in DVI mode.
24 ENABLE_POL	Default 0 active low during valid data transfer on each horizontal line.
23–22 RSRVD1	This field is reserved. Reserved bits. Write as 0.
21 VSYNC_ PERIOD_UNIT	Default 0 for counting VSYNC_PERIOD in terms of DISPLAY CLOCK (pix_clk) cycles. Set it to 1 to count in terms of complete horizontal lines. DISPLAY CLOCK (pix_clk) cycles should be used in the VSYNC mode, while horizontal line should be used in the DOTCLK mode.
20 VSYNC_PULSE_ WIDTH_UNIT	Default 0 for counting VSYNC_PULSE_WIDTH in terms of DISPLAY CLOCK (pix_clk) cycles. Set it to 1 to count in terms of complete horizontal lines.
19 HALF_LINE	Setting this bit to 1 will make the total VSYNC period equal to the VSYNC_PERIOD field plus half the HORIZONTAL_PERIOD field (i.e. VSYNC_PERIOD field plus half horizontal line), otherwise it is just VSYNC_PERIOD. Should be only used in the DOTCLK mode, not in the VSYNC interface mode.
18 HALF_LINE_ MODE	When this bit is 0, the first field (VSYNC period) will end in half a horizontal line and the second field will begin with half a horizontal line. When this bit is 1, all fields will end with half a horizontal line, and none will begin with half a horizontal line.
VSYNC_PULSE_ WIDTH	Number of units for which VSYNC signal is active. For the DOTCLK mode, the unit is determined by the VSYNC_PULSE_WIDTH_UNIT. If the VSYNC_PULSE_WIDTH_UNIT is 0 for DOTCLK mode, VSYNC_PULSE_WIDTH must be less than HSYNC_PERIOD. For the VSYNC interface mode, it should be in terms of number of DISPLAY CLOCK (pix_clk) cycles only.

13.2.4.9 LCDIF VSYNC Mode and Dotclk Mode Control Register1 (LCDIF_VDCTRL1)

This register is used to control the VSYNC signal in the VSYNC and DOTCLK modes of the block.

This register determines the period and duty cycle of the VSYNC signal when it is generated in the block.

Address: 32E0_0000h base + 80h offset = 32E0_0080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VSYNC_PERIOD																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_VDCTRL1 field descriptions

Field	Description
VSYNC_PERIOD	Total number of units between two positive or two negative edges of the VSYNC signal. If HALF_LINE is set, it is implicitly calculated to be VSYNC_PERIOD plus half HSYNC_PERIOD.

13.2.4.10 LCDIF VSYNC Mode and Dotclk Mode Control Register2 (LCDIF_VDCTRL2)

This register is used to control the HSYNC signal in the DOTCLK mode of the block.

This register determines the period and duty cycle of the HSYNC signal when it is generated in the block.

Address: 32E0_0000h base + 90h offset = 32E0_0090h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HSYNC_PULSE_WIDTH														HSYNC_PERIOD																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_VDCTRL2 field descriptions

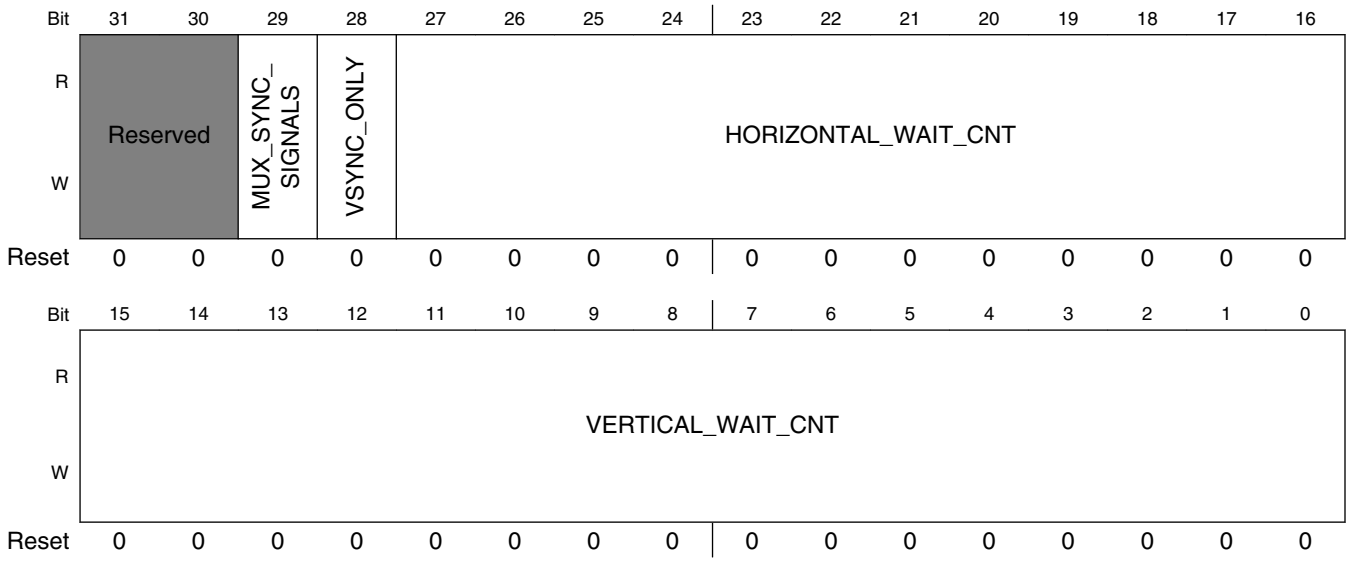
Field	Description
31–18 HSYNC_PULSE_ WIDTH	Number of DISPLAY CLOCK (pix_clk) cycles for which HSYNC signal is active.
HSYNC_PERIOD	Total number of DISPLAY CLOCK (pix_clk) cycles between two positive or two negative edges of the HSYNC signal.

13.2.4.11 LCDIF VSYNC Mode and Dotclk Mode Control Register3 (LCDIF_VDCTRL3)

This register is used to determine the vertical and horizontal wait counts.

This register determines the back porches of HSYNC and VSYNC signals when they are generated by the block.

Address: 32E0_0000h base + A0h offset = 32E0_00A0h



LCDIF_VDCTRL3 field descriptions

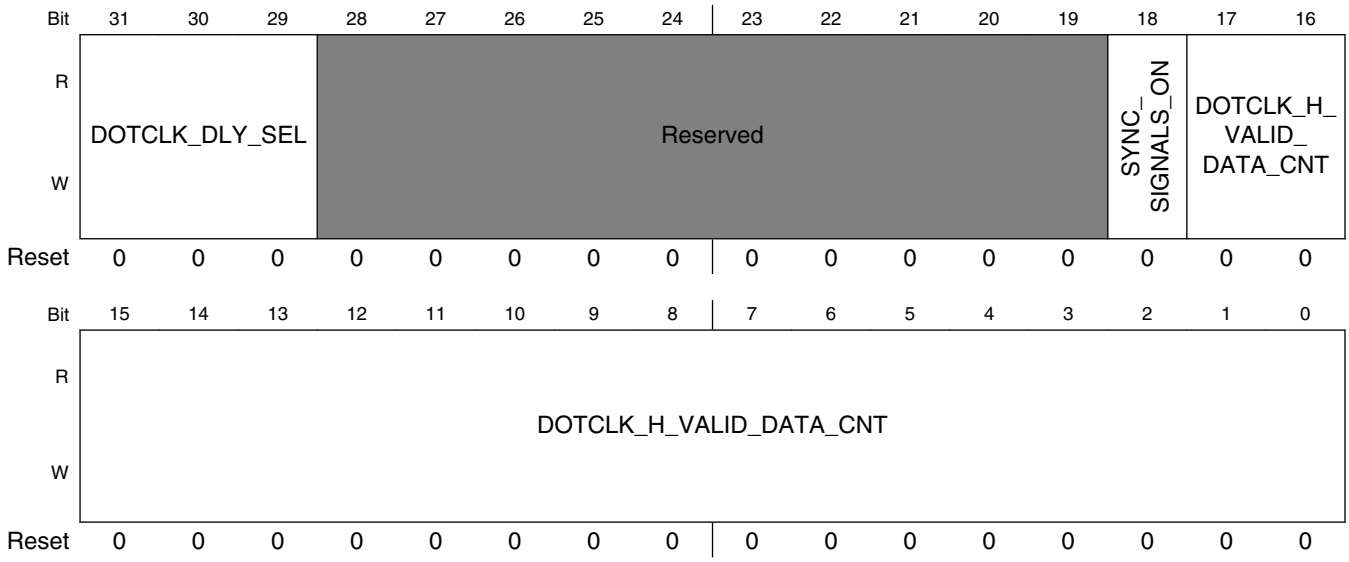
Field	Description
31–30 RSRVD0	This field is reserved. Reserved bits, write as 0.
29 MUX_SYNC_SIGNALS	When this bit is set, the LCDIF block will internally mux HSYNC with LCD_D14, DOTCLK with LCD_D13 and ENABLE with LCD_D12, otherwise these signals will go out on separate pins. This feature can be used to maintain backward compatible with 37xx.
28 VSYNC_ONLY	This bit must be set to 1 in the VSYNC mode of operation, and 0 in the DOTCLK mode of operation.
27–16 HORIZONTAL_WAIT_CNT	In the DOTCLK mode, wait for this number of clocks from falling edge (or rising if HSYNC_POL is 1) of HSYNC signal to account for horizontal back porch plus the number of DOTCLKs before the moving picture information begins.
VERTICAL_WAIT_CNT	In the VSYNC interface mode, wait for this number of DISPLAY CLOCK (pix_clk) cycles from the falling VSYNC edge (or rising if VSYNC_POL is 1) before starting LCD transactions and is applicable only if WAIT_FOR_VSYNC_EDGE is set. Minimum is CMD_SETUP+5. In the DOTCLK mode, it accounts for the vertical back porch lines plus the number of horizontal lines before the moving picture begins. The unit for this parameter is inherently the same as the VSYNC_PERIOD_UNIT.

13.2.4.12 LCDIF VSYNC Mode and Dotclk Mode Control Register4 (LCDIF_VDCTRL4)

This register is used to control the DOTCLK mode of the block.

This register determines the active data in each horizontal line in the DOTCLK mode. Note that the total number of active horizontal lines in the DOTCLK mode is the same as the V_COUNT bitfield in the LCDIF_TRANSFER_COUNT register.

Address: 32E0_0000h base + B0h offset = 32E0_00B0h



LCDIF_VDCTRL4 field descriptions

Field	Description
31–29 DOTCLK_DLY_ SEL	This bitfield selects the amount of time by which the DOTCLK signal should be delayed before coming out of the LCD_DOTCK pin. 0 = 2ns; 1=4ns;2=6ns;3=8ns. Remaining values are reserved.
28–19 RSRVD0	This field is reserved. Reserved bits, write as 0.
18 SYNC_ SIGNALS_ON	Set this field to 1 if the LCD controller requires that the VSYNC or VSYNC/HSYNC/DOTCLK control signals should be active at least one frame before the data transfers actually start and remain active at least one frame after the data transfers end. The hardware does not count the number of frames automatically. Rather, the VSYNC edge interrupt can be monitored by software to count the number of frames that have occurred after this bit is set and then the RUN bit can be set to start the data transactions. This bit must always be set in the DOTCLK mode of operation, and it must be set in the VSYNC mode of operation when VSYNC signal is an output.
DOTCLK_H_ VALID_DATA_ CNT	Total number of DISPLAY CLOCK (pix_clk) cycles on each horizontal line that carry valid data in DOTCLK mode.

13.2.4.13 Digital Video Interface Control0 Register (LCDIF_DVICTRL0)

The Digital Video interface Control0 register provides the overall control of the Digital Video interface.

This register gives information about the horizontal active, horizontal blanking and total number of lines in the ITU-R BT.656 interface.

EXAMPLE

```
//525/60 video system
```

```
HW_LCDIF_DVICTRL0_H_ACTIVE_CNT_WR(0x5A0);//1440
HW_LCDIF_DVICTRL0_H_BLANKING_CNT_WR(0x106);//262
//625/50 video system
HW_LCDIF_DVICTRL0_H_ACTIVE_CNT_WR(0x5A0);//1440
HW_LCDIF_DVICTRL0_H_BLANKING_CNT_WR(0x112);//274
```

Address: 32E0_0000h base + C0h offset = 32E0_00C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				H_ACTIVE_CNT												Reserved				H_BLANKING_CNT											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_DVICTRL0 field descriptions

Field	Description
31–28 RSRVD1	This field is reserved. Reserved bits, write as 0.
27–16 H_ACTIVE_CNT	Number of active video samples to be transmitted. (Mostly will be 1440 for both PAL and NTSC). Must always be a multiple of 4.
15–12 RSRVD0	This field is reserved. Reserved bits, write as 0.
H_BLANKING_ CNT	Number of blanking samples to be inserted between EAV and SAV during horizontal blanking interval.

13.2.4.14 Digital Video Interface Control1 Register (LCDIF_DVICTRL1)

The Digital Video interface Control1 register provides the overall control of the Digital Video interface.

This register contains information about the Field1 start and end, and the Field2 start in the ITU-R BT.656 interface.

EXAMPLE

```
//525/60 video system
HW_LCDIF_DVICTRL1_F1_START_LINE_WR(0x4);//4
HW_LCDIF_DVICTRL1_F1_END_LINE_WR(0x109);//265
HW_LCDIF_DVICTRL1_F2_START_LINE_WR(0x10A);//266
//625/50 video system
HW_LCDIF_DVICTRL1_F1_START_LINE_WR(0x1);//1
HW_LCDIF_DVICTRL1_F1_END_LINE_WR(0x138);//312
HW_LCDIF_DVICTRL1_F2_START_LINE_WR(0x139);//313
```

Address: 32E0_0000h base + D0h offset = 32E0_00D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		F1_START_LINE										F1_END_LINE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	F1_END_LINE								F2_START_LINE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_DVICTRL1 field descriptions

Field	Description
31–30 RSRVD0	This field is reserved. Reserved bits, write as 0.
29–20 F1_START_LINE	Vertical line number from which Field 1 begins.
19–10 F1_END_LINE	Vertical line number at which Field1 ends.
F2_START_LINE	Vertical line number from which Field 2 begins.

13.2.4.15 Digital Video Interface Control2 Register (LCDIF_DVICTRL2)

The Digital Video interface Control2 register provides the overall control of the Digital Video interface.

This register contains information about the Field2 end, and the Vertical Blanking1 interval in the ITU-R BT.656 interface.

EXAMPLE

```
//525/60 video system
HW_LCDIF_DVICTRL2_F2_END_LINE_WR(0x3);//3
HW_LCDIF_DVICTRL2_V1_BLANK_START_LINE_WR(0x108);//264
HW_LCDIF_DVICTRL2_V1_BLANK_END_LINE_WR(0x11A);//282
//625/50 video system
HW_LCDIF_DVICTRL2_F2_END_LINE_WR(0x271);//625
HW_LCDIF_DVICTRL2_V1_BLANK_START_LINE_WR(0x137);//311
HW_LCDIF_DVICTRL2_V1_BLANK_END_LINE_WR(0x14F);//335
```

Address: 32E0_0000h base + E0h offset = 32E0_00E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		F2_END_LINE										V1_BLANK_START_LINE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	V1_BLANK_START_LINE								V1_BLANK_END_LINE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_DVICTRL2 field descriptions

Field	Description
31–30 RSRVD0	This field is reserved. Reserved bits, write as 0.
29–20 F2_END_LINE	Vertical line number at which Field 2 ends.
19–10 V1_BLANK_ START_LINE	Vertical line number towards the end of Field1 where first Vertical Blanking interval starts.
V1_BLANK_ END_LINE	Vertical line number in the beginning part of Field2 where first Vertical Blanking interval ends.

13.2.4.16 Digital Video Interface Control3 Register (LCDIF_DVICTRL3)

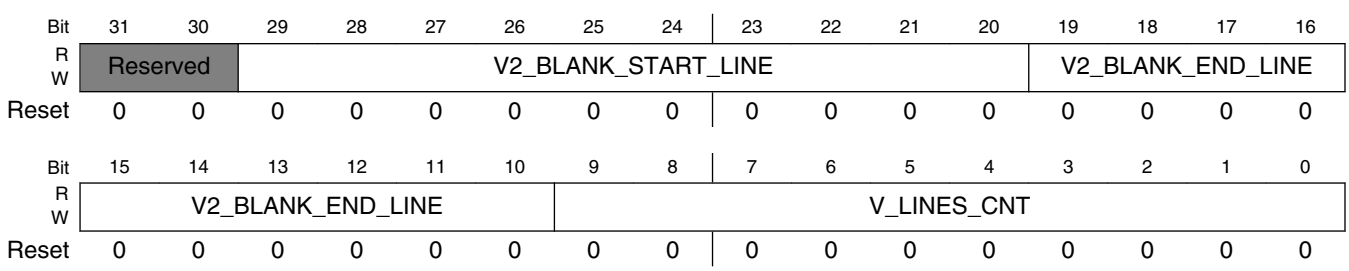
The Digital Video interface Control3 register provides the overall control of the Digital Video interface.

This register contains information about the Vertical Blanking2 interval in the ITU-R BT. 656 interface.

EXAMPLE

```
//525/60 video system
HW_LCDIF_DVICTRL3_V2_BLANK_START_LINE_WR(0x1);//1
HW_LCDIF_DVICTRL3_V2_BLANK_END_LINE_WR(0x13);//19
HW_LCDIF_DVICTRL0_V_LINES_CNT_WR(0x20D);//525
//625/50 video system
HW_LCDIF_DVICTRL3_V2_BLANK_START_LINE_WR(0x270);//624
HW_LCDIF_DVICTRL3_V2_BLANK_END_LINE_WR(0x16);//22
HW_LCDIF_DVICTRL0_V_LINES_CNT_WR(0x271);//625
```

Address: 32E0_0000h base + F0h offset = 32E0_00F0h



LCDIF_DVICTRL3 field descriptions

Field	Description
31–30 RSRVD0	This field is reserved. Reserved bits, write as 0.

Table continues on the next page...

LCDIF_DVICTRL3 field descriptions (continued)

Field	Description
29–20 V2_BLANK_ START_LINE	Vertical line number towards the end of Field2 where second Vertical Blanking interval starts.
19–10 V2_BLANK_ END_LINE	Vertical line number in the beginning part of Field1 where second Vertical Blanking interval ends.
V_LINES_CNT	Total number of vertical lines per frame (generally 525 or 625)

13.2.4.17 Digital Video Interface Control4 Register (LCDIF_DVICTRL4)

The Digital Video interface Control4 register provides the overall control of the Digital Video interface.

This register is used to add side borders to the output if the input frame width is less than 720 pixels.

EXAMPLE

```
                //If input frame has only 640 pixels per line, but output is supposed to have
720 pixels per line.
                HW_LCDIF_DVICTRL4_H_FILL_CNT_WR(0x50);//80
                HW_LCDIF_DVICTRL4_Y_FILL_VALUE_WR(0x10);//16
                HW_LCDIF_DVICTRL4_CB_FILL_VALUE_WR(0x80);//128
                HW_LCDIF_DVICTRL4_CR_FILL_VALUE_WR(0x80);//128
```

Address: 32E0_0000h base + 100h offset = 32E0_0100h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Y_FILL_VALUE								CB_FILL_VALUE								CR_FILL_VALUE								H_FILL_CNT							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_DVICTRL4 field descriptions

Field	Description
31–24 Y_FILL_VALUE	Value of Y component of filler data
23–16 CB_FILL_VALUE	Value of CB component of filler data
15–8 CR_FILL_VALUE	Value of CR component of filler data.
H_FILL_CNT	Number of active video samples that have to be filled with the filler data in the front and back portions of the active horizontal interval. Must be a multiple of 4. This field will have to be programmed if the input frame has less than 720 pixels per line.

13.2.4.18 RGB to YCbCr 4:2:2 CSC Coefficient0 Register (LCDIF_CSC_COEFF0)

LCDIF_CSC_COEFF0 register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by: Y = C0*R + C1*G + C2*B + Y_offset Cb= C3*R + C4*G + C5*B + CbCr_offset Cr= C6*R + C7*G + C8*B + CbCr_offset

This register carries programming information about RGB to YCbCr 4:2:2 CSC.

EXAMPLE

```
HW_LCDIF_CSC_COEFF0_C0_WR(0x41);//0.257x256=65
HW_LCDIF_CSC_COEFF0_CSC_SUBSAMPLE_FILTER_WR(0x3);
```

Address: 32E0_0000h base + 110h offset = 32E0_0110h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								C0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														CSC_ SUBSAMPL E_FILTER	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_CSC_COEFF0 field descriptions

Field	Description
31–26 RSRVD1	This field is reserved. Reserved bits, write as 0.
25–16 C0	Two's complement red multiplier coefficient for Y
15–2 RSRVD0	This field is reserved. Reserved bits, write as 0.
CSC_ SUBSAMPLE_ FILTER	This register describes the filtering and subsampling scheme to be performed on the chroma components in order to convert from YCbCr 4:4:4 to YCbCr 4:2:2 space. Note that the following descriptions apply individually to Cb and Cr. 0x0 **SAMPLE_AND_HOLD** — No filtering, simply keep every chroma value for samples numbered 2n and discard chroma values associated with all samples numbered 2n+1. 0x1 **RSRVD** — Reserved 0x2 **INTERSTITIAL** — Chroma samples numbered 2n and 2n+1 are averaged (weights 1/2, 1/2) and that chroma value replaces the two chroma values at 2n and 2n+1. This chroma now exists horizontally halfway between the two luma samples. 0x3 **COSITED** — Chroma samples numbered 2n-1, 2n, and 2n+1 are averaged (weights 1/4, 1/2, 1/4) and that chroma value exists at the same site as the luma sample numbered 2n and the chroma samples at 2n+1 are discarded.

13.2.4.19 RGB to YCbCr 4:2:2 CSC Coefficient1 Register (LCDIF_CSC_COEFF1)

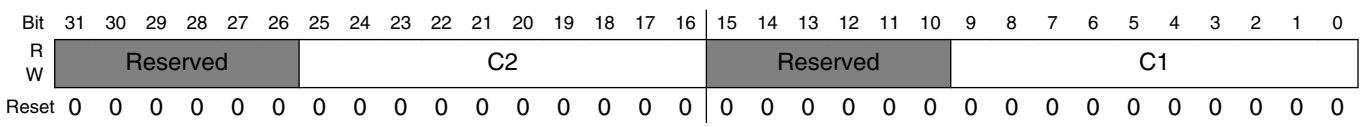
LCDIF_CSC_COEFF1 register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by: Y = C0*R + C1*G + C2*B + Y_offset Cb= C3*R + C4*G + C5*B + CbCr_offset Cr= C6*R + C7*G + C8*B + CbCr_offset

This register carries programming information about RGB to YCbCr 4:2:2 CSC.

EXAMPLE

```
HW_LCDIF_CSC_COEFF1_C1_WR(0x81);//0.504x256=129
HW_LCDIF_CSC_COEFF1_C2_WR(0x19);//0.098x256=25
```

Address: 32E0_0000h base + 120h offset = 32E0_0120h



LCDIF_CSC_COEFF1 field descriptions

Field	Description
31–26 RSRVD1	This field is reserved. Reserved bits, write as 0.
25–16 C2	Two's complement blue multiplier coefficient for Y
15–10 RSRVD0	This field is reserved. Reserved bits, write as 0.
C1	Two's complement green multiplier coefficient for Y

13.2.4.20 RGB to YCbCr 4:2:2 CSC Coefficent2 Register (LCDIF_CSC_COEFF2)

LCDIF_CSC_COEFF2 register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by: Y = C0*R + C1*G + C2*B + Y_offset Cb= C3*R + C4*G + C5*B + CbCr_offset Cr= C6*R + C7*G + C8*B + CbCr_offset

This register carries programming information about RGB to YCbCr 4:2:2 CSC.

EXAMPLE

```
HW_LCDIF_CSC_COEFF2_C3_WR(0x3DB);//-0.148x256=-37
HW_LCDIF_CSC_COEFF2_C4_WR(0x3B6);//-0.291x256=-74
```

Address: 32E0_0000h base + 130h offset = 32E0_0130h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						C4										Reserved						C3									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_CSC_COEFF2 field descriptions

Field	Description
31–26 RSRVD1	This field is reserved. Reserved bits, write as 0.
25–16 C4	Two's complement green multiplier coefficient for Cb
15–10 RSRVD0	This field is reserved. Reserved bits, write as 0.
C3	Two's complement red multiplier coefficient for Cb

13.2.4.21 RGB to YCbCr 4:2:2 CSC Coefficient3 Register (LCDIF_CSC_COEFF3)

LCDIF_CSC_COEFF3 register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by: Y = C0*R + C1*G + C2*B + Y_offset Cb= C3*R + C4*G + C5*B + CbCr_offset Cr= C6*R + C7*G + C8*B + CbCr_offset

This register carries programming information about RGB to YCbCr 4:2:2 CSC.

EXAMPLE

```
HW_LCDIF_CSC_COEFF3_C5_WR(0x70);//0.439x256=112
HW_LCDIF_CSC_COEFF3_C6_WR(0x70);//0.439x256=112
```

Address: 32E0_0000h base + 140h offset = 32E0_0140h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						C6										Reserved						C5									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_CSC_COEFF3 field descriptions

Field	Description
31–26 RSRVD1	This field is reserved. Reserved bits, write as 0.

Table continues on the next page...

LCDIF_CSC_COEFF3 field descriptions (continued)

Field	Description
25–16 C6	Two's complement red multiplier coefficient for Cr
15–10 RSRVD0	This field is reserved. Reserved bits, write as 0.
C5	Two's complement blue multiplier coefficient for Cb

13.2.4.22 RGB to YCbCr 4:2:2 CSC Coefficient4 Register (LCDIF_CSC_COEFF4)

LCDIF_CSC_COEFF4 register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by: Y = C0*R + C1*G + C2*B + Y_offset Cb= C3*R + C4*G + C5*B + CbCr_offset Cr= C6*R + C7*G + C8*B + CbCr_offset

This register carries programming information about RGB to YCbCr 4:2:2 CSC.

EXAMPLE

```
HW_LCDIF_CSC_COEFF4_C7_WR(0x3A2);//-0.368x256=-94
HW_LCDIF_CSC_COEFF4_C8_WR(0x3EE);//-0.071x256=-18
```

Address: 32E0_0000h base + 150h offset = 32E0_0150h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						C8										Reserved						C7									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_CSC_COEFF4 field descriptions

Field	Description
31–26 RSRVD1	This field is reserved. Reserved bits, write as 0.
25–16 C8	Two's complement blue multiplier coefficient for Cr
15–10 RSRVD0	This field is reserved. Reserved bits, write as 0.
C7	Two's complement green multiplier coefficient for Cr

13.2.4.23 RGB to YCbCr 4:2:2 CSC Offset Register (LCDIF_CSC_OFFSET)

LCDIF_CSC_ register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by: Y = C0*R + C1*G + C2*B + Y_offset Cb= C3*R + C4*G + C5*B + CbCr_offset Cr= C6*R + C7*G + C8*B + CbCr_offset

This register carries programming information about RGB to YCbCr 4:2:2 CSC.

Address: 32E0_0000h base + 160h offset = 32E0_0160h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							CBCR_OFFSET									Reserved							Y_OFFSET								
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

LCDIF_CSC_OFFSET field descriptions

Field	Description
31–25 RSRVD1	This field is reserved. Reserved bits, write as 0.
24–16 CBCR_OFFSET	Two's complement offset for the Cb and Cr components
15–9 RSRVD0	This field is reserved. Reserved bits, write as 0.
Y_OFFSET	Two's complement offset for the Y component

13.2.4.24 RGB to YCbCr 4:2:2 CSC Limit Register (LCDIF_CSC_LIMIT)

LCDIF_CSC_CTRL0 register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by: Y = C0*R + C1*G + C2*B + Y_offset Cb= C3*R + C4*G + C5*B + CbCr_offset Cr= C6*R + C7*G + C8*B + CbCr_offset

This register carries programming information about RGB to YCbCr 4:2:2 CSC. Note that the values in this register are unsigned.

EXAMPLE

```
HW_LCDIF_CSC_LIMIT_CBCR_MIN_WR(0x10);//16
HW_LCDIF_CSC_LIMIT_CBCR_MAX_WR(0xF0);//240
HW_LCDIF_CSC_LIMIT_Y_MIN_WR(0x10);//16
HW_LCDIF_CSC_LIMIT_Y_MAX_WR(0xEB);//235
```

Address: 32E0_0000h base + 170h offset = 32E0_0170h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CBCR_MIN								CBCR_MAX								Y_MIN								Y_MAX							
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

LCDIF_CSC_LIMIT field descriptions

Field	Description
31–24 CBCR_MIN	Lower limit of Cb and Cr after RGB to 4:2:2 YCbCr conversion
23–16 CBCR_MAX	Upper limit of Cb and Cr after RGB to 4:2:2 YCbCr conversion
15–8 Y_MIN	Lower limit of Y after RGB to 4:2:2 YCbCr conversion
Y_MAX	Upper limit of Y after RGB to 4:2:2 YCbCr conversion

13.2.4.25 LCD Interface Data Register (LCDIF_DATA)

This register is used to transfer data using the PIO interface mode of operation. In MPU mode, data written to this register will be transferred out to the display device. When receiving data from the display, data is read from this register using PIO operations. During write operations, data can be written to this register (from the processor's perspective) as bytes, half-words (16 bits), or words (32 bits) as desired.

This register holds the 32-bit word written by the Arm platform into LCDIF. This data then gets sent out by the block across the interface.

Address: 32E0_0000h base + 180h offset = 32E0_0180h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA_THREE								DATA_TWO								DATA_ONE								DATA_ZERO							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_DATA field descriptions

Field	Description
31–24 DATA_THREE	Byte 3 (most significant byte) of data written to LCDIF.
23–16 DATA_TWO	Byte 2 of data written to LCDIF.
15–8 DATA_ONE	Byte 1 of data written to LCDIF.
DATA_ZERO	Byte 0 (least significant byte) of data written to LCDIF.

13.2.4.26 Bus Master Error Status Register (LCDIF_BM_ERROR_STAT)

This register reflects the virtual address at which the AXI master received an error response from the slave.

When the BM_ERROR_IRQ is asserted, the address of the bus error is updated in the register.

Address: 32E0_0000h base + 190h offset = 32E0_0190h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_BM_ERROR_STAT field descriptions

Field	Description
ADDR	Virtual address at which bus master error occurred.

13.2.4.27 CRC Status Register (LCDIF_CRC_STAT)

This register reflects the CRC value of each frame sent out by LCDIF. The CRC is done on the final output bus, so the value will be dependent on the LCD_DATABUS_WIDTH bitfield even if the input data is the same.

This register will be updated when the CUR_FRAME_DONE_IRQ is asserted. In the case of DVI mode, the CRC is calculated for the entire frame, not separately for each field in the frame.

Address: 32E0_0000h base + 1A0h offset = 32E0_01A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CRC_VALUE																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_CRC_STAT field descriptions

Field	Description
CRC_VALUE	Calculated CRC value.

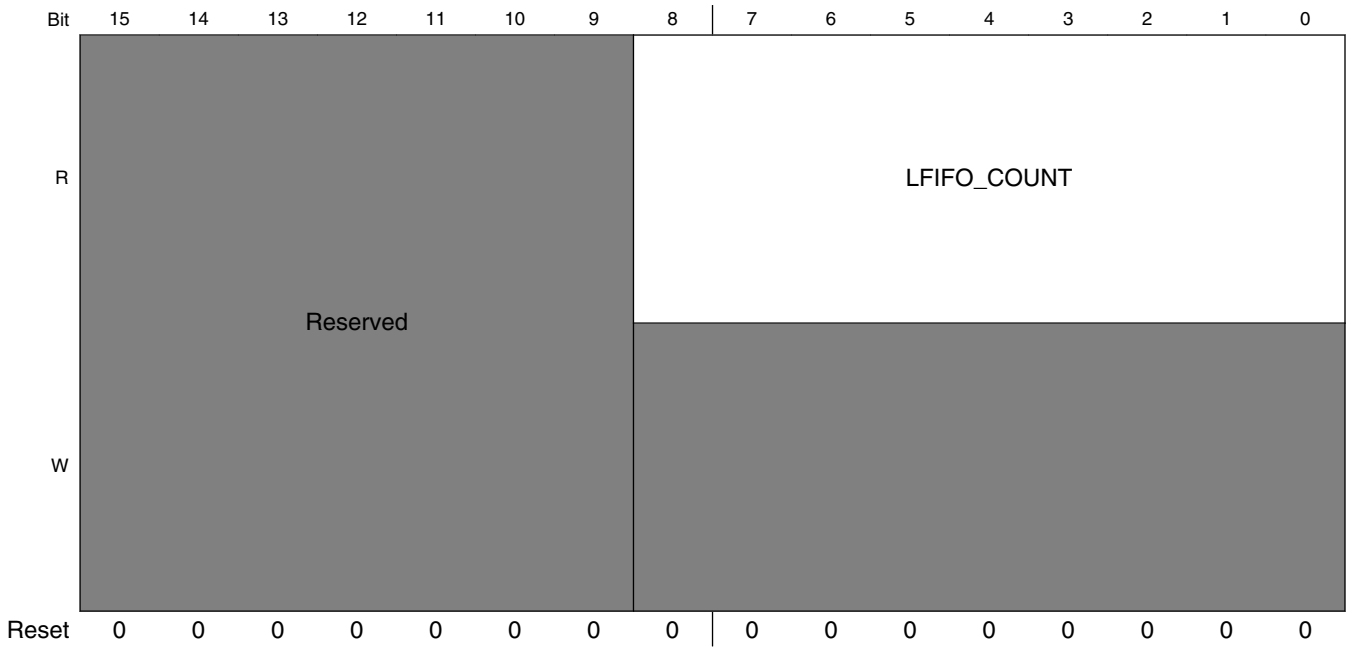
13.2.4.28 LCD Interface Status Register (LCDIF_STAT)

The LCD interface status register can be used to check the current status of the LCDIF block.

The LCD interface status register that contains read only views of some parameters or current state of the block.

Address: 32E0_0000h base + 1B0h offset = 32E0_01B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRESENT	Reserved	LFIFO_FULL	LFIFO_EMPTY	TXFIFO_FULL	TXFIFO_EMPTY	BUSY	DVI_CURRENT_FIELD	Reserved							
W																
Reset	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0



LCDIF_STAT field descriptions

Field	Description
31 PRESENT	0: LCDIF not present on this product 1: LCDIF is present.
30 -	This field is reserved. Reserved.
29 LFIFO_FULL	Read only view of the signals that indicates LCD LFIFO is full.
28 LFIFO_EMPTY	Read only view of the signals that indicates LCD LFIFO is empty.
27 TXFIFO_FULL	Read only view of the signals that indicates LCD TXFIFO is full.
26 TXFIFO_EMPTY	Read only view of the signals that indicates LCD TXFIFO is empty.
25 BUSY	Read only view of the input busy signal from the external LCD controller.
24 DVI_CURRENT_FIELD	Read only view of the current field being transmitted. DVI_CURRENT_FIELD = 0 means field 1. DVI_CURRENT_FIELD = 1 means field 2.
23–9 RSRVD0	This field is reserved. Reserved bits. Write as 0.
LFIFO_COUNT	Read only view of the current count in Latency buffer (LFIFO).

13.2.4.29 LCDIF Threshold Register (LCDIF_THRES)

This register is used to activate control signals when the number of pixels reaches the programmed threshold. These control signals, in turn, can be used to manipulate access priority or dynamically change the input clock frequency to meet the required pixel throughput.

Memory request priority threshold register.

Address: 32E0_0000h base + 200h offset = 32E0_0200h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							FASTCLOCK									Reserved							PANIC								
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

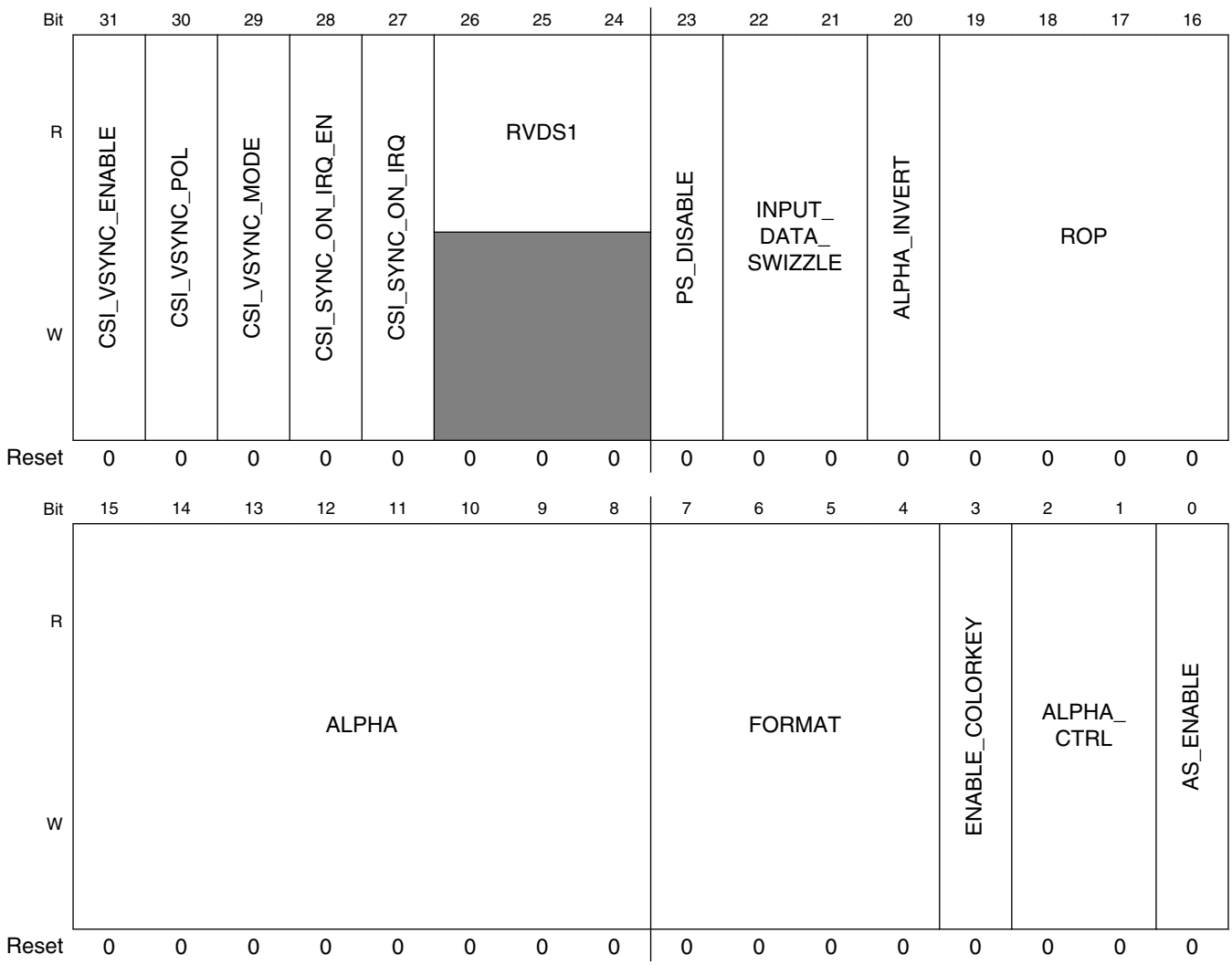
LCDIF_THRES field descriptions

Field	Description
31–25 RSRVD2	This field is reserved. Reserved bits. Write as 0.
24–16 FASTCLOCK	This value should be set to a value of pixels, from 0 to 511. When the number of pixels in the input pixel FIFO is LESS than this value, the fast clock control output will be raised. This signal can be used to reduce the system bus clock frequency to save power during horizontal or vertical blanking intervals. This value should also be programmed to a value that is greater than the "PANIC" threshold value. This will allow a faster clock to recover the number of pixels in the FIFO before a "panic" level is encountered.
15–9 RSRVD1	This field is reserved. Reserved bits. Write as 0.
PANIC	This value should be set to a value of pixels from 0 to 511. When the number of pixels in the input pixel FIFO is less than this value, the internal panic control output will be raised. This signal can be used to raise the access LCDIF's access priority.

13.2.4.30 LCDIF AS Buffer Control Register (LCDIF_AS_CTRL)

The Alpha Surface Parameter register provides additional controls for AS.

Address: 32E0_0000h base + 210h offset = 32E0_0210h



LCDIF_AS_CTRL field descriptions

Field	Description
31 CSI_VSYNC_ ENABLE	When this bit is set by software, the LCDIF work as sync mode with CSI input.
30 CSI_VSYNC_ POL	Default 0 active low during VSYNC_PULSE_WIDTH time and will be high during the rest of the VSYNC period. Set it to 1 to invert the polarity.

Table continues on the next page...

LCDIF_AS_CTRL field descriptions (continued)

Field	Description
29 CSI_VSYNC_MODE	this bit is set by software to decide which vsync generate mode. LCDIF vsync generate by internal counter when set to 0, LCDIF vsync delayed by each csi_vsync_in when set to 1; INT_SYNC_MODE = 0x0 LCDIF vsync generate by internal counter. EXT_SYNC_MODE = 0x1 LCDIF vsync delayed by each csi_vsync_in.
28 CSI_SYNC_ON_IRQ_EN	This bit is set to enable an interrupt when LCDIF lock with CSI vsync input.
27 CSI_SYNC_ON_IRQ	this bit is set by software to decide which vsync generate mode. LCDIF vsync generate by internal counter when set to 0, LCDIF vsync delayed by each csi_vsync_in when set to 1; INT_SYNC_MODE = 0x0 LCDIF vsync generate by internal counter. EXT_SYNC_MODE = 0x1 LCDIF vsync delayed by each csi_vsync_in.
26–24 RVDS1	Reserved, always set to zero.
23 PS_DISABLE	When this bit is set by software, the LCDIF will disable PS buffer data.
22–21 INPUT_DATA_SWIZZLE	This field specifies how to swap the bytes either in the HW_LCDIF_DATA register or those fetched by the AXI master part of LCDIF. The swizzle function is independent of the WORD_LENGTH bit. See the explanation of the HW_LCDIF_DATA below for names and definitions of data register fields. The supported swizzle configurations are: NO_SWAP = 0x0 No byte swapping.(Little endian) LITTLE_ENDIAN = 0x0 Little Endian byte ordering (same as NO_SWAP). BIG_ENDIAN_SWAP = 0x1 Big Endian swap (swap bytes 0, 3 and 1, 2). SWAP_ALL_BYTES = 0x1 Swizzle all bytes, swap bytes 0, 3 and 1, 2 (aka Big Endian). HWD_SWAP = 0x2 Swap half-words. HWD_BYTE_SWAP = 0x3 Swap bytes within each half-word.
20 ALPHA_INVERT	Setting this bit to logic 0 will not alter the alpha value. A logic 1 will invert the alpha value and apply (1-alpha) for image composition.
19–16 ROP	Indicates a raster operation to perform when enabled. Raster operations are enabled through the ALPHA_CTRL field. MASKAS = 0x0 AS AND PS MASKNOTAS = 0x1 nAS AND PS MASKASNOT = 0x2 AS AND nPS MERGEAS = 0x3 AS OR PS MERGENOTAS = 0x4 nAS OR PS MERGEASNOT = 0x5 AS OR nPS NOTCOPYAS = 0x6 nAS NOT = 0x7 nPS NOTMASKAS = 0x8 AS NAND PS NOTMERGEAS = 0x9 AS NOR PS XORAS = 0xA AS XOR PS NOTXORAS = 0xB AS XNOR PS

Table continues on the next page...

LCDIF_AS_CTRL field descriptions (continued)

Field	Description
15–8 ALPHA	Alpha modifier used when the ALPHA_MULTIPLY or ALPHA_OVERRIDE values are programmed in REG_AS_CTRL[ALPHA_CTRL]. The output alpha value will either be replaced (ALPHA_OVERRIDE) or scaled (ALPHA_MULTIPLY) when selected.
7–4 FORMAT	Indicates the input buffer format for AS. ARGB8888 = 0x0 32-bit pixels with alpha RGB888 = 0x4 32-bit pixels without alpha (unpacked 24-bit format) ARGB1555 = 0x8 16-bit pixels with alpha ARGB4444 = 0x9 16-bit pixels with alpha RGB555 = 0xC 16-bit pixels without alpha RGB444 = 0xD 16-bit pixels without alpha RGB565 = 0xE 16-bit pixels without alpha
3 ENABLE_ COLORKEY	Indicates that colorkey functionality is enabled for this alpha surface. Pixels found in the alpha surface colorkey range will be displayed as transparent (the PS pixel will be used).
2–1 ALPHA_CTRL	Determines how the alpha value is constructed for this alpha surface. Indicates that the value in the ALPHA field should be used instead of the alpha values present in the input pixels. Embedded = 0x0 Indicates that the AS pixel alpha value will be used to blend the AS with PS. The ALPHA field is ignored. Override = 0x1 Indicates that the value in the ALPHA field should be used instead of the alpha values present in the input pixels. Multiply = 0x2 Indicates that the value in the ALPHA field should be used to scale all pixel alpha values. Each pixel alpha is multiplied by the value in the ALPHA field. ROPs = 0x3 Enable ROPs. The ROP field indicates an operation to be performed on the alpha surface and PS pixels.
0 AS_ENABLE	When this bit is set by software, the LCDIF will start fetching AS buffer data in bus master mode and combine it with another buffer.

13.2.4.31 Alpha Surface Buffer Pointer (LCDIF_AS_BUF)

This register is used to indicate the base address of the AS buffer.

Address: 32E0_0000h base + 220h offset = 32E0_0220h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_AS_BUF field descriptions

Field	Description
ADDR	Address pointer for the alpha surface 0 buffer.

13.2.4.32 LCDIF_AS_NEXT_BUF

When the LCDIF is behaving as a master, this address points to the address of the next frame of data that will be sent out via the LCDIF. It is upto the software to make sure that this register is programmed before the end of the current frame, otherwise it might result in old data going out the LCDIF. This address must always be double-word aligned.

Address: 32E0_0000h base + 230h offset = 32E0_0230h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_AS_NEXT_BUF field descriptions

Field	Description
ADDR	Address of the next frame that will be transmitted by LCDIF.

13.2.4.33 LCDIF Overlay Color Key Low (LCDIF_AS_CLRKEYLOW)

If a pixel in the current overlay image with a color that falls in the range from the ASCOLORKEYLOW to ASCOLORKEYHIGH range, it will use the PS pixel value for that location. Colorkey operations are higher priority than alpha or ROP operations.

Address: 32E0_0000h base + 240h offset = 32E0_0240h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RSVD1								PIXEL																							
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

LCDIF_AS_CLRKEYLOW field descriptions

Field	Description
31–24 RSVD1	Reserved, always set to zero.
PIXEL	Low range of RGB color key applied to AS buffer

13.2.4.34 LCDIF Overlay Color Key High (LCDIF_AS_CLRKEYHIGH)

If a pixel in the current overlay image with a color that falls in the range from the ASCOLORKEYLOW to ASCOLORKEYHIGH range, it will use the PS pixel value for that location. Colorkey operations are higher priority than alpha or ROP operations.

Address: 32E0_0000h base + 250h offset = 32E0_0250h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RSVD1								PIXEL																							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_AS_CLRKEYHIGH field descriptions

Field	Description
31–24 RSVD1	Reserved, always set to zero.
PIXEL	High range of RGB color key applied to AS buffer

13.2.4.35 LCD working insync mode with CSI for VSYNC delay (LCDIF_SYNC_DELAY)

The LCDIF DOTCLK mode VSYNC will delay from CSI_VSYNC as
(V_COUNT_DELAY * HSYNC_PERIOD + H_COUNT_DELAY) PIXCLK cycles

Address: 32E0_0000h base + 260h offset = 32E0_0260h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	V_COUNT_DELAY																H_COUNT_DELAY															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_SYNC_DELAY field descriptions

Field	Description
31–16 V_COUNT_ DELAY	LCDIF VSYNC delayed counter for CSI_VSYNC.
H_COUNT_ DELAY	LCDIF VSYNC delayed counter for CSI_VSYNC.

13.2.4.36 LCDIF Pigeon Mode Control0 Register (LCDIF_PIGEONCTRL0*n*)

This register contains global counter settings for Pigeon Mode also houses general purpose timing adjustment registers

Address: 32E0_0000h base + 380h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				LD_PERIOD												Reserved				FD_PERIOD											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_PIGEONCTRL0*n* field descriptions

Field	Description
31–28 -	This field is reserved. Reserved
27–16 LD_PERIOD	Period of pclk counter during LD phase
15–12 -	This field is reserved. Reserved
FD_PERIOD	Period of line counter during FD phase

13.2.4.37 LCDIF Pigeon Mode Control1 Register (LCDIF_PIGEONCTRL1*n*)

This register contains global counter settings for Pigeon Mode also houses general purpose timing adjustment registers

Address: 32E0_0000h base + 390h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				FRAME_CNT_CYCLES												Reserved				FRAME_CNT_PERIOD											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_PIGEONCTRL1*n* field descriptions

Field	Description
31–28 -	This field is reserved. Reserved

Table continues on the next page...

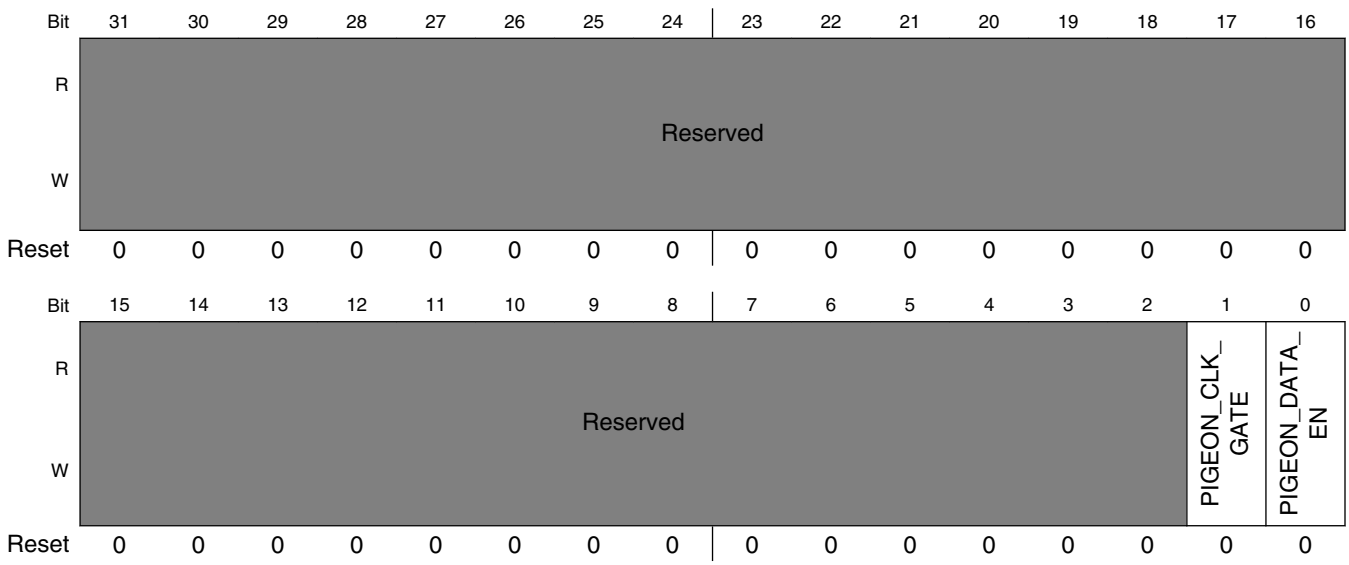
LCDIF_PIGEONCTRL1*n* field descriptions (continued)

Field	Description
27–16 FRAME_CNT_CYCLES	Max cycles of frame counter
15–12 -	This field is reserved. Reserved
FRAME_CNT_PERIOD	Period of frame counter

13.2.4.38 LCDIF Pigeon Mode Control2 Register (LCDIF_PIGEONCTRL2*n*)

This register contains global counter settings for Pigeon Mode also houses clock gating and data enable registers

Address: 32E0_0000h base + 3A0h offset + (4d × i), where i=0d to 3d



LCDIF_PIGEONCTRL2*n* field descriptions

Field	Description
31–2 -	This field is reserved. Reserved
1 PIGEON_CLK_GATE	Pigeon mode dot clock gate enable
0 PIGEON_DATA_EN	Pigeon mode data enable

13.2.4.39 Panel Interface Signal Generator Register (LCDIF_PIGEON*n*)

This register contains parameters for timing signal generation

Address: 32E0_0000h base + 800h offset + (64d × i), where i=0d to 11d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STATE_MASK								MASK_CNT							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_PIGEON*n* field descriptions

Field	Description
31–24 STATE_MASK	state_mask = (FS|FB|FD|FE) and (LS|LB|LD|LE) , select any combination of scan states as reference point for local counter to start ticking 0x1 **FS** — FRAME SYNC 0x2 **FB** — FRAME BEGIN 0x4 **FD** — FRAME DATA 0x8 **FE** — FRAME END 0x10 **LS** — LINE SYNC 0x20 **LB** — LINE BEGIN 0x40 **LD** — LINE DATA 0x80 **LE** — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT 0x0 **HSTATE_CNT** — pclk counter within one hscan state 0x1 **HSTATE_CYCLE** — pclk cycle within one hscan state 0x2 **VSTATE_CNT** — line counter within one vscan state 0x3 **VSTATE_CYCLE** — line cycle within one vscan state 0x4 **FRAME_CNT** — frame counter 0x5 **FRAME_CYCLE** — frame cycle 0x6 **HCNT** — horizontal counter (pclk counter within one line) 0x7 **VCNT** — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	Event to incrment local counter 0x0 **PCLK** — pclk 0x1 **LINE** — Line start pulse

Table continues on the next page...

LCDIF_PIGEON*n* field descriptions (continued)

Field	Description
	0x2 **FRAME** — Frame start pulse 0x3 **SIG_ANOTHER** — Use another signal as tick event
1 POL	Polarity of signal output 0x0 **ACTIVE_HIGH** — Normal Signal (Active high) 0x1 **ACTIVE_LOW** — Inverted signal (Active low)
0 EN	Enable pigeon Mode on this signal

13.2.4.40 Panel Interface Signal Generator Register (LCDIF_PIGEON*n*)

This register contains parameters for timing signal generation

Address: 32E0_0000h base + 810h offset + (64d × i), where i=0d to 11d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CLR_CNT																SET_CNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_PIGEON*n* field descriptions

Field	Description
31–16 CLR_CNT	Deassert signal output when counter match this value 0x0 **CLEAR_USING_MASK** — Keep active until mask off
SET_CNT	Assert signal output when counter match this value 0x0 **START_ACTIVE** — Start as active

13.2.4.41 Panel Interface Signal Generator Register (LCDIF_PIGEON*n*)

This register contains parameters for timing signal generation

Address: 32E0_0000h base + 820h offset + (64d × i), where i=0d to 11d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																							SIG_ANOTHER					SIG_LOGIC			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF_PIGEON*n* field descriptions

Field	Description
31–9 RSVD	This field is reserved. Reserved
8–4 SIG_ANOTHER	Select another signal for logic operation or as mask or counter tick event 0 **CLEAR_USING_MASK** — Keep active until mask off
SIG_LOGIC	Logic operation with another signal: DIS/AND/OR/COND 0 **DIS** — No logic operation 1 **AND** — sigout = sig_another AND this_sig 2 **OR** — sigout = sig_another OR this_sig 3 **MASK** — mask = sig_another AND other_masks

13.3 CSI Bridge (CSI)

13.3.1 Overview

This chapter presents the CMOS Sensor Interface (CSI) architecture, operation principles, and programming model.

The CSI enables the chip to connect directly to external CMOS image sensors. CMOS image sensors are separated into two classes, dumb and smart. Dumb sensors are those that support only traditional sensor timing (Vertical SYNC and Horizontal SYNC) and output only Bayer and statistics data, while smart sensors support CCIR656 video decoder formats and perform additional processing of the image (for example, image compression, image pre-filtering, and various data output formats).

The capabilities of the CSI include:

- Configurable interface logic to support most commonly available CMOS sensors.
- Support for CCIR656 video interface as well as traditional sensor interface.
- 8-bit / 16-bit / 24-bit data port for YCbCr, YUV, or RGB data input.
- 8-bit / 10-bit / 16-bit data port for Bayer data input.
- Full control of 8-bit/pixel, 10-bit/pixel or 16-bit / pixel data format to 64-bit receive FIFO packing.
- 256 x 64 FIFO to store received image pixel data.
- Reveive FIFO overrun protection mechanism.
- Embedded DMA controllers to transfer data from receive FIFO or statistic FIFO through AHB bus.
- Support 2D DMA transfer from the receive FIFO to the frame buffers in the external memory.

- Support double bufferring two frames in the external memory.
- Single interrupt source to interrupt controller from maskable interrupt sources: Start of Frame, End of Frame, Change of Field, FIFO full, FIFO overrun, DMA transfer done, CCIR error and AHB bus response error.
- Configurable master clock frequency output to sensor.
- Statistic data generation for Auto Exposure (AE) and Auto White Balance (AWB) control of the camera (only for Bayer data and 8-bit/pixel format).
- Supports simple deinterlacing of interlaced input.

13.3.2 Principles of Operation

The information found here describes the modes of operation of the sensor interface.

The CSI is designed to support generic sensor interface timing as well as CCIR656 video interface timing. Traditional CMOS sensors typically use VSYNC (SOF), HSYNC (BLANK), and PIXCLK signals to output Bayer or YUV data. Smart CMOS sensors, that come with on-chip imaging processing, usually support video mode transfer. They use an embedded timing codec to replace the VSYNC and HSYNC signal. The timing codec is defined by the CCIR656 standard.

The CSI can support connection with the sensor as follows.

- To connect with one 8-bit sensor, the sensor data interface should connect to CSI_DATA[9:2].
- To connect with one 10-bit sensor, the sensor data interface should connect to CSI_DATA[9:0].
- To connect with one 16-bit sensor, the sensor data interface should connect to CSI_DATA[15:0].

Table 13-5. CSI input data format

Signal Name	TVdecoder YCbCr 1 Cycle	RGB888 1 Cycle	RGB888/ YUV4444 3 Cycle	RGB666 1 Cycle	RGB565 1 Cycle	YCbCr422 1 Cycle	YCbCr422 2 Cycle	Generic 10 bit	CCIR656
ipp_csi_d[23]	Y[7]	R[7]		R[5]					
ipp_csi_d[22]	Y[6]	R[6]		R[4]					
ipp_csi_d[21]	Y[5]	R[5]		R[3]					
ipp_csi_d[20]	Y[4]	R[4]		R[2]					
ipp_csi_d[19]	Y[3]	R[3]		R[1]					

Table continues on the next page...

Table 13-5. CSI input data format (continued)

Signal Name	TVdecoder YCbCr 1 Cycle	RGB888 1 Cycle	RGB888/ YUV4444 3 Cycle	RGB666 1 Cycle	RGB565 1 Cycle	YCbCr422 1 Cycle	YCbCr422 2 Cycle	Generic 10 bit	CCIR656
ipp_csi_d[18]	Y[2]	R[2]		R[0]					
ipp_csi_d[17]	Y[1]	R[1]		Y[5]					
ipp_csi_d[16]	Y[0]	R[0]		R[4]					
ipp_csi_d[15]	Cb[7]	G[7]		G[5]	R[4]	Y[7]			
ipp_csi_d[14]	Cb[6]	G[6]		G[4]	R[3]	Y[6]			
ipp_csi_d[13]	Cb[5]	G[5]		G[3]	R[2]	Y[5]			
ipp_csi_d[12]	Cb[4]	G[4]		G[2]	R[1]	Y[4]			
ipp_csi_d[11]	Cb[3]	G[3]		G[1]	R[0]	Y[3]			
ipp_csi_d[10]	Cb[2]	G[2]		G[0]	G[5]	Y[2]			
ipp_csi_d[9]	Cb[1]	G[1]	R/G/B[7]	G[5]	G[4]	Y[1]	Y/C[7]	Ge[9]	C/Y[7]
ipp_csi_d[8]	Cb[0]	G[0]	R/G/B[6]	G[4]	G[3]	Y[0]	Y/C[6]	Ge[8]	C/Y[6]
ipp_csi_d[7]	Cr[7]	B[7]	R/G/B[5]	B[5]	G[2]	C[7]	Y/C[5]	Ge[7]	C/Y[5]
ipp_csi_d[6]	Cr[6]	B[6]	R/G/B[4]	B[4]	G[1]	C[6]	Y/C[4]	Ge[6]	C/Y[4]
ipp_csi_d[5]	Cr[5]	B[5]	R/G/B[3]	B[3]	G[0]	C[5]	Y/C[3]	Ge[5]	C/Y[3]
ipp_csi_d[4]	Cr[4]	B[4]	R/G/B[2]	B[2]	B[4]	C[4]	Y/C[2]	Ge[4]	C/Y[2]
ipp_csi_d[3]	Cr[3]	B[3]	R/G/B[1]	B[1]	B[3]	C[3]	Y/C[1]	Ge[3]	C/Y[1]
ipp_csi_d[2]	Cr[2]	B[2]	R/G/B[0]	B[0]	B[2]	C[2]	Y/C[0]	Ge[2]	C/Y[0]
ipp_csi_d[1]	Cr[1]	B[1]		B[5]	B[1]	C[1]		Ge[1]	
ipp_csi_d[0]	Cr[0]	B[0]		B[4]	B[0]	C[0]		Ge[0]	

13.3.2.1 Data Transfer with the Embedded DMA Controllers

The CSI has two embedded DMA controllers, one for the receive FIFO and the other for the statistic FIFO. It supports 2D DMA transfer from the receive FIFO to the frame buffers in the external memory and linear DMA transfer from the statistic FIFO.

To transfer data from the RxFIFO to the external memory, the user should set the start address in the frame buffer where the transferred data is stored, the parameters of the frame buffers, and the parameters of the image coming from the sensor. The user can have two frame buffers in the external memory. Each one will store a frame of image coming from the sensor. The embedded DMA controller will first write the frame buffer1 and then frame buffer2. These two frame buffers will be written by turns. The start address should be aligned in double words and set in the CSIDMASA-FB1 and CSIDMASA-FB2 registers. In the CSIFBUF_PARA register, the user should set the stride of the frame buffer to show how many double words to skip before starting to write the next row of the image. In the CSIIMAG_PARA register, the user should set the width and height of the image coming from the sensor. The RxFF_LEVEL and DMA_REQ_EN_RFF bits in CSICR3 registers also need to be set before the data transfer starts. When the number of the data in the RxFIFO reaches the trigger level, a DMA request will be sent to the embedded DMA controller and the data will be read out from the RxFIFO and written through AHB bus into the external frame buffers. The burst type of transfer can be INCR4, INCR8 and INCR16 by setting DMA_BURST_TYPE_RFF bits in CSICR2 register. After all data in an image frame are transferred, the DMA_TSF_DONE_FB1 or DMA_TSF_DONE_FB2 bit will be set in CSISR register and the interrupt can be triggered if the corresponding enable bit is set in CSICR1 register. The DMA_REFLASH_RFF bit in CSICR3 can be used to activate or restart the embedded DMA controller.

The RxFIFO has the overrun protection mechanism in case the RxFIFO is overrun during data transfer. If the RxFIFO is full and more data needs to be received during the data transfer, the RxFIFO will be overwritten continuously and all 128 words of data in the RxFIFO before overrun occurred will be discarded; the corresponding 128 words memory space in the frame buffer will keep the previous values.

To transfer data from the statistic FIFO to the external memory, the user should set the start address of the externl memory where the transferred data is stored and the total transfer sizes. The start address and the transfer sizes are all aligned in double words and should be set in the CSIDMASA-STATFIFO and CSIDMATS-STATFIFO registers. The STATFF_LEVEL and DMA_REQ_EN_SFF bits in CSICR3 registers should also be set before the data transfer starts. When the number of the data in the STATFIFO reaches the trigger level, a dma request will be sent to the embedded DMA controller and the data will be read out from the STATFIFO and written through AHB bus into the external memory. The burst type of transfer can be INCR4, INCR8 and INCR16 by setting

DMA_BURST_TYPE_SFF bits in CSICR2 register. After all expected data (defined by the total transfer sizes) are transferred, the DMA_TSF_DONE_SFF bit will be set in CSISR register and an interrupt can be triggered if the SFF_DMA_DONE_INTEN is enabled in CSICR1 register. The DMA_REFLASH_SFF bit in CSICR3 can be used to activate or re-start the embedded DMA controller.

13.3.2.2 Gated Clock Mode

VSYNC, HSYNC, and PIXCLK signals are used in gated clock mode.

A frame starts with an active edge on VSYNC, then HSYNC asserts and holds for the entire line. The Pixel clock is valid as long as HSYNC is asserted. Data is latched at the active edge of the valid pixel clocks. HSYNC deasserts at the end of line. Pixel clocks then become invalid and CSI stops receiving data from the stream. For the next line the HSYNC timing repeats. For the next frame the VSYNC timing repeats.

13.3.2.3 Non-Gated Clock Mode

In non-gated clock mode, only the VSYNC and PIXCLK signals are used; the HSYNC signal is ignored.

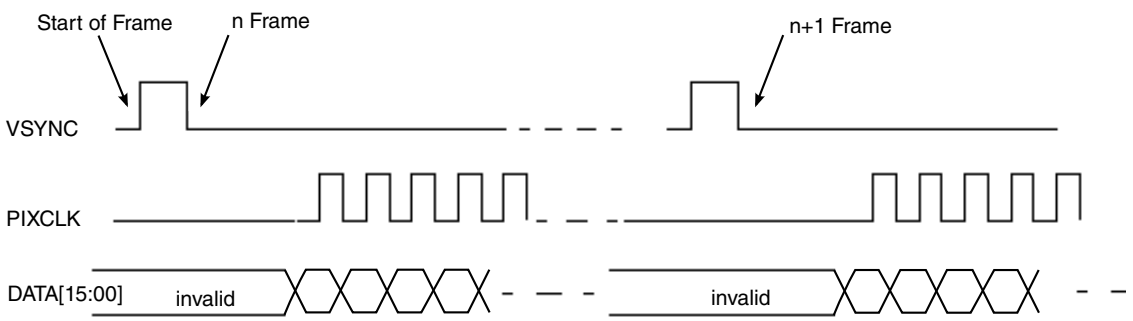


Figure 13-18. Non-Gated Clock Mode Timing Diagram

The overall timing of non-gated mode is the same as the gated-clock mode, except for the HSYNC signal. HSYNC signal is ignored by the CSI. All incoming pixel clocks are valid and cause data to be latched into RxFIFO. The PIXCLK signal is inactive (states low) until valid data is ready to be transmitted over the bus.

Figure 13-18 shows the timing of a typical sensor. Other sensors may have the slightly different timing from that shown. The CSI can be programed to support rising/falling-edge triggered VSYNC, active-high/low HSYNC, and rising/falling-edge triggered PIXCLK.

13.3.2.4 CCIR656 Interlace Mode

In CCIR656 interlace mode, only the PIXCLK and CSI_DATA[9:2] signals are used. The start of frame and blank signals are replaced by a timing codec which is embedded in the data stream. Each active line starts with an Start of Active Video (SAV) code and ends with an End of Active Video (EAV) code. In some cases, digital blanking is inserted in between EAV and SAV code. The CSI decodes and filters out the timing-coding from the data stream, recovering VSYNC and HSYNC signals for internal use, such as statistical block control. Data is forwarded to the data receive and packing block in a sequential manner without reordering-that is, field 1 followed by field 2. The fields must be reordered in software to get back the original image.

Change of Field (COF) interrupt is triggered upon every field change. The interrupt service routine reads the status register to check for the current field.

According to the CCIR656 specification, the image must be in 625/50 PAL or 525/60 NTSC format. In addition, the image is interlaced into odd and even fields with vertical and horizontal blank data being filled into certain lines. Data must be in YCbCr422 format, each pixel contains 2 bytes, either Y + Cr or Y + Cb. These requirements are set for TV systems. The CSI module supports PAL and NTSC format only.

The following figure describes the frame structure in PAL system, showing vertical and horizontal blanking.

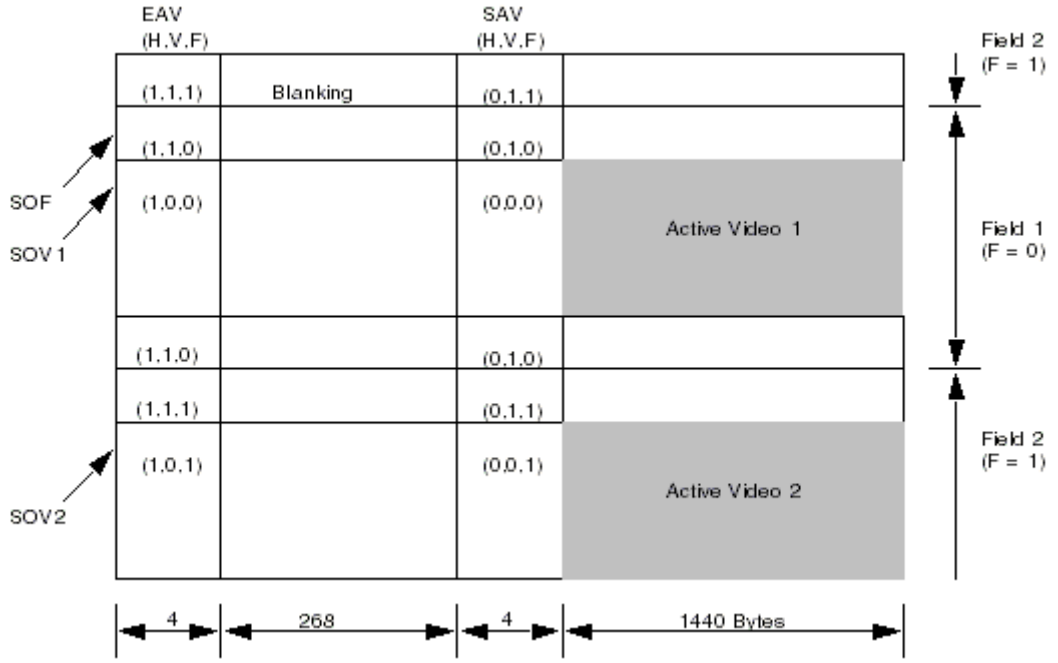


Figure 13-19. CCIR656 Interlace Mode (PAL)

The following figure describes the general timing for a single line, showing SAV and EAV.

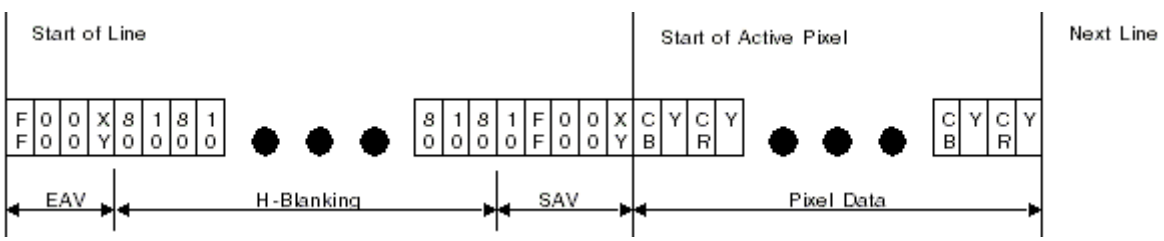


Figure 13-20. CCIR656 General Line Timing

The coding tables recommended by the CCIR656 specification are shown below. It is used in the CCIR656 mode to decode the video stream. An interrupt is generated for SOF, which is decoded from the embedded timing codec.

Table 13-6. Coding for SAV and EAV

Data Bit Number	1st Byte 0xFF	2nd Byte 0x00	3rd Byte 0x00	4th Byte 0xXY
7 (MSB)	1	0	0	1
6	1	0	0	F
5	1	0	0	V
4	1	0	0	H
3	1	0	0	P3
2	1	0	0	P2

Table continues on the next page...

Table 13-6. Coding for SAV and EAV (continued)

Data Bit Number	1st Byte 0xFF	2nd Byte 0x00	3rd Byte 0x00	4th Byte 0xXY
1	1	0	0	P1
0	1	0	0	P0

Table 13-7. Codes with Protection bits for Error Detection/Correction

F	V	H	P3	P2	P1	P0
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	1	1
1	0	1	1	0	1	0
1	1	0	1	1	0	0
1	1	1	0	0	0	1

Table 13-8. Representations by F-Bit

F-Bit	Representations
0	ODD FIELD (FIELD 1)
1	EVEN FIELD (FIELD 2)

13.3.2.5 CCIR656 Progressive Mode

For a CMOS camera system of VGA or CIF resolution, strict adherence to the interlace requirements stated in the CIR standard is not required.

The image is considered to have only 1 active field which is scanned in a progressive manner. This active field is regarded as field 1 and the F-bit in the timing codec is ignored by the decoder. Most sensors support CCIR timing in this mode (progressive) by default.

The following figure shows the typical flow of progressive mode.

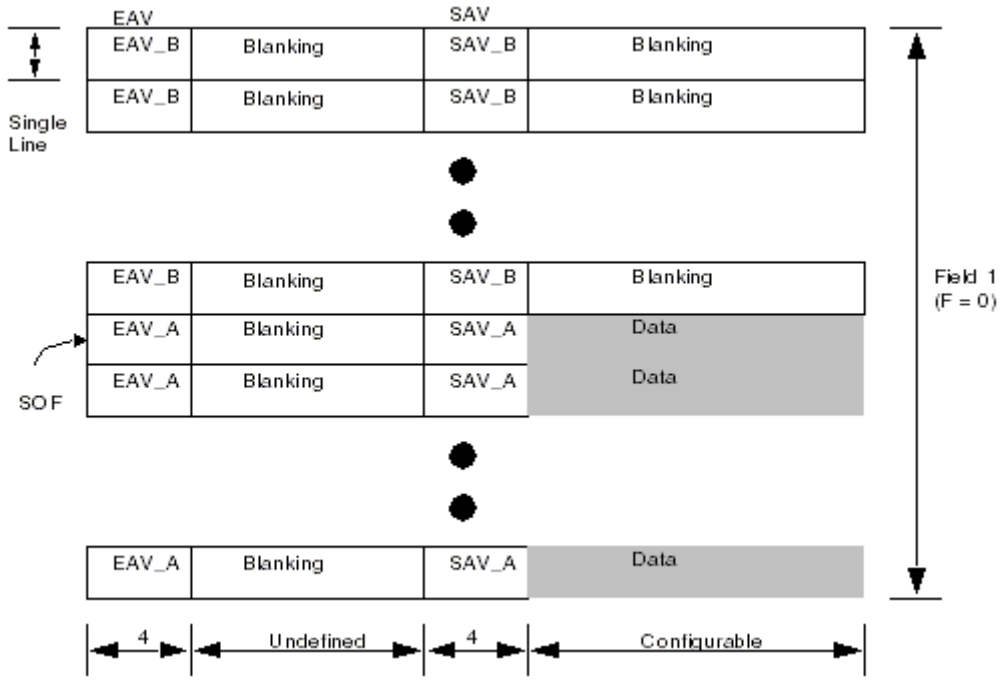


Figure 13-21. CCIR656 Progressive Mode (General Case)

An interrupt is generated for SOF but not for COF. In the general case, when SOF information is retrieved from the embedded coding, it is known as internal VSYNC mode. In other cases, when the VSYNC signal is provided by the sensor, it is known as external VSYNC mode. The CSI can be operated in internal or external VSYNC mode.

13.3.2.6 Error Correction for CCIR656 Coding

According to the algorithm for CCIR coding, protection bits in the SAV and EAV are encoded in the way that allows a 1-bit error to be corrected, or a 2-bit error to be detected by the decoder. This feature is supported by the interlace mode CCIR decoder in CSI.

For the 1-bit error case, users can select the error to be corrected automatically, or simply shown as a status flag instead. For the 2-bit error case, because the decoder is unable to make a correction, the error would be shown as a status flag only.

An interrupt can be generated upon the detection of an error. This signal can be enabled or disabled without affecting the operation of the status bit.

13.3.2.7 Deinterlacer

Deinterlacing is the process of converting interlaced video, such as CCIR656 input, into a non-interlaced form.

The CSI uses the weaving method to do deinterlacing. Weaving is done by adding consecutive fields together. CSI uses top-field detection function. No matter input is NTSC or PAL mode, the combined frame will always put the top-field first and then bottom-field.

13.3.3 Interrupt Generation

The information found here describes CSI events that generate interrupts.

13.3.3.1 Start Of Frame Interrupt (SOF_INT)

The source of an SOF interrupt is dependent on the mode of operation.

In traditional mode, VSYNC signal is taken from sensor and SOF_INT is generated at the rising or falling edge (programmable) of VSYNC.

In CCIR interlace mode, the SOF interrupt information is retrieved from the embedded coding and SOF_INT is generated.

In CCIR progressive mode, there are two sources of an SOF interrupt:

- In *internal* VSYNC mode, SOF is retrieved from the embedded coding.
- In *external* VSYNC mode, VSYNC is taken from the sensor and SOF is generated at the rising edge of VSYNC.

13.3.3.2 End Of Frame Interrupt (EOF_INT)

An EOF interrupt is generated when the frame ends and the complete frame data in RXFIFO is read.

The EOF event triggering works with the RX count register (CSIRXCNT). Software sets the RX count register to the frame size (in words). The CSI RX logic then counts the number of pixel data being received and compares it with the RX count. If the preset value is reached, an EOF interrupt is generated and the data in the RXFIFO are read. If a SOF event is detected before this happens, the EOF interrupt is not generated.

13.3.3.3 Change Of Field Interrupt (COF_INT)

The Change of Field interrupt is only valid in CCIR Interlace mode. The COF interrupt is generated when the field toggles, either from field 1 to field 2, or field 2 to field 1.

Software should first check COF_INT bit in the CSI Status Register (CSISTAT) before checking that F1_INT or F2_INT is turned on.

In PAL systems, the field changes at the beginning of the frame and coincides with SOF. For the first field, a COF interrupt is not generated, only an SOF. The COF interrupt is generated for the second field.

13.3.3.4 CCIR Error Interrupt (ECC_INT)

The CCIR Error Interrupt is only valid for CCIR Interlace mode. An ECC interrupt is generated when an error is found on the SAV or EAV codes in the incoming stream. When this happens, the ECC_INT status bit is set.

13.3.3.5 RxFIFO Full Interrupt (RxFF_INT)

A RxFIFO full interrupt is generated when the number of data in RXFIFO reaches the water mark defined by RxFF_LEVEL in CSICR3.

13.3.3.6 Statistic FIFO Full Interrupt (STATFF_INT)

A StatFIFO full interrupt is generated when the number of data in STATFIFO reaches the water mark defined by STATFF_LEVEL in CSICR3.

13.3.3.7 RxFIFO Overrun Interrupt (RFF_OR_INT)

A RxFIFO Overrun interrupt is generated when the RxFIFO has 128 words data and more data is being written in.

13.3.3.8 Statistic FIFO Overrun Interrupt (SFF_OR_INT)

A StatFIFO Overrun interrupt is generated when the STATFIFO has 64 words data and more data is being written in.

13.3.3.9 Frame Buffer1 DMA Transfer Done Interrupt (DMA_TSF_DONE_FB1)

A DMA transfer done interrupt of frame buffer1 is generated when one frame of data are transferred from RxFIFO to the frame buffer1 in the external memory.

13.3.3.10 Frame Buffer2 DMA Transfer Done Interrupt (DMA_TSF_DONE_FB2)

A DMA transfer done interrupt of frame buffer2 is generated when one frame of data are transferred from RxFIFO to the frame buffer2 in the external memory.

13.3.3.11 Statistic FIFO DMA Transfer Done Interrupt (DMA_TSF_DONE_SFF)

A StatFIFO DMA transfer done interrupt is generated when all the data are transferred from StatFIFO to the external memory. The transfer size is defined in the STATFIFO DMA Transfer Size Register.

13.3.3.12 AHB Bus Response Error Interrupt (HRESP_ERR_INT)

An AHB Bus response error interrupt is generated when a bus error is detected.

13.3.4 Data Packing Style

Careful attention to endianess is needed given the different port sizes at different stages of the image capture path.

To enable flexible packing of image data before storage in the FIFOs, the CSI module can swap data fields by use of the PACK_DIR and the SWAP16_EN bit in CSI Control Register 1 (CSICR1).

The CSI module accepts 8-bit, 10-bit or 16-bit data from the sensor by configuring PIXEL_BIT bit in CSI Control Register 1 (CSICR1) and TWO_8BIT_SENSOR bit in CSI Control Register3 (CSICR3). The input data is packed according to the setting of PACK_DIR bit. The packed data is stored in the RX FIFO according to the setting of the SWAP16_EN bit.

For 10-bit per pixel data format, each pixel is expanded to 16 bits by appending 6 zeros bits to the most significant bit.

13.3.4.1 RX FIFO Path

13.3.4.1.1 Bayer Data

Bayer data is a type of raw data from the image sensor. This byte-wide data must be converted to the RGB space or YUV space by software. The data path for Bayer data is from the CSI to memory. If the system is in little endian, then the PACK_DIR bit should be set to 0. 8-bit data format from a sensor is packed to 64 bits as P7.P6.P5.P4.P3.P2.P1.P0, where P0 is the pixel coming in time slot 0 (first data) and P3 is the pixel coming in time slot 3 (the last data in the 64-bit word). When the data is addressed as bytes by software, P0 is transferred first, P1 is transferred next, and so on. 10-bit data format is packed to 64 bits as 000000.P3.000000.P2.000000.P1.000000.P0, where P0 is the 10-bit data coming in time slot 0 (first pixel) and P3 is the 10-bit data coming in time slot 3 (fourth pixel). 16-bit data is packed to 64 bits as P7.P6.P5.P4.P3.P2.P1.P0.

13.3.4.1.2 RGB565 Data

RGB565 data is processed data from the image sensor, which can be put directly into the display buffer. The data is 16 bits wide. The data path is from CSI to memory to the display controller. On the sensor side, data must be transmitted as P0 first, followed by P1, and so on. For each pixel, whether the MSB or LSB is sent first depends on the endianness of the sensor. Data is 16 bits wide with the MSB labeled RG, and the LSB labeled GB. P0 is represented as RG0 and GB0.

CSI receives data in one of the following sequence:

- RG0, GB0, RG1, GB1, while RG0 comes out at time slot 0 (first data), and GB1 comes out at time slot 3 (last data)
- GB0, RG0, GB1, RG1

Using the first sequence as an example, and assuming the system is running in little endian, the data is presented as:

- 8-bit data from sensor: RG0, GB0, RG1, GB1, …
- 64-bit data before storage in the CSI RX FIFO (PACK_DIR bit = 1): RG0GB0RG1GB1RG2GB2RG3GB3
- 64-bitdata in CSI RX FIFO (SWAP16_EN bit enabled): RG3GB3RG2GB2RG1GB1RG0GB0
- 64-bit transfer to system memory: RG3GB3RG2GB2RG1GB1RG0GB0
- 16-bit read by display controller: RG0GB0, RG1GB1

13.3.4.1.3 RGB888 Data

This is another kind of processed data from image sensor, which can be used for further image processing directly. Each of the data consist of 8-bit Red, 8-bit Green, and 8-bit Blue data. An example of timing scheme is shown in the following figure.

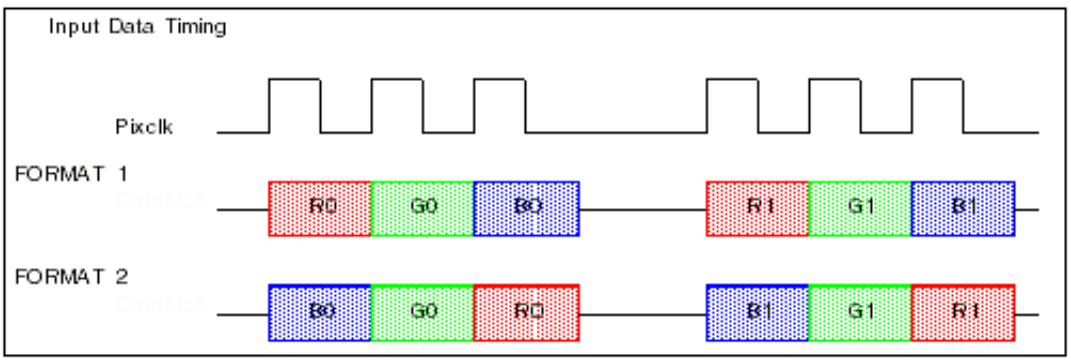
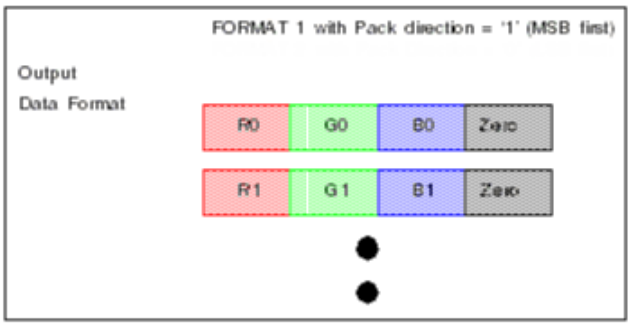


Figure 13-22. Sample Timing Diagram for RGB888 8 bits/cycle Data

An optional scheme to pack a dummy byte is provided. For every group of 3 bytes data, a dummy zero is packed to form a 32-bit word as shown in the following figure. The dummy zero can be packed at the LSB position or MSB position. Using RGB888A_FORMAT_SEL in CSI_CSICR18[18] to determine to put the dummy bytes packed at LSB or MSB position.


FORMAT 1 with Pack direction = '1' (MSB first)
Output
Data Format
R0
G0
B0
Zero
R1
G1
B1
Zero

13.3.4.2 STAT FIFO Path

Statistics only works for Bayer data in 8-bit per pixel format. It generates 16-bit statistical output from the 8-bit Bayer input (CSI_DATA[13:6]). The outputs are Sum of Green (G), Sum of Red (R), Sum of Blue (B), and Auto Focus (F). Each output is 16-bits wide.

The settings of PACK_DIR and SWAP16_EN bits in the CSICR1 register have no effect on the input path. The PACK_DIR only controls how the 16-bit stat output is packed into the 32-bit STAT FIFO.

When the PACK_DIR bit = 1, the stat data is packed as:

First 32-bit: RG

Second 32-bit: BF

…

When the PACK_DIR bit = 0, the stat data is packed as:

First 32-bit GR

Second 32-bit: FB

...

13.3.5 CSI Memory Map/Register Definition

All the 32-bit registers of the CSI module are summarized in the Memory Map below:

CSI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E2_0000	CSI Control Register 1 (CSI_CSICR1)	32	R/W	4000_0800h	13.3.5.1/ 3976
32E2_0004	CSI Control Register 2 (CSI_CSICR2)	32	R/W	0000_0000h	13.3.5.2/ 3980
32E2_0008	CSI Control Register 3 (CSI_CSICR3)	32	R/W	0000_0000h	13.3.5.3/ 3982
32E2_000C	CSI Statistic FIFO Register (CSI_CSISTATFIFO)	32	R	0000_0000h	13.3.5.4/ 3984

Table continues on the next page...

CSI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E2_0010	CSI RX FIFO Register (CSI_CSIRFIFO)	32	R	0000_0000h	13.3.5.5/ 3984
32E2_0014	CSI RX Count Register (CSI_CSIRXCNT)	32	R/W	0000_9600h	13.3.5.6/ 3985
32E2_0018	CSI Status Register (CSI_CSISR)	32	R/W	0000_4000h	13.3.5.7/ 3986
32E2_0020	CSI DMA Start Address Register - for STATFIFO (CSI_CSIDMASA_STATFIFO)	32	R/W	0000_0000h	13.3.5.8/ 3989
32E2_0024	CSI DMA Transfer Size Register - for STATFIFO (CSI_CSIDMATS_STATFIFO)	32	R/W	0000_0000h	13.3.5.9/ 3989
32E2_0028	CSI DMA Start Address Register - for Frame Buffer1 (CSI_CSIDMASA_FB1)	32	R/W	0000_0000h	13.3.5.10/ 3990
32E2_002C	CSI DMA Transfer Size Register - for Frame Buffer2 (CSI_CSIDMASA_FB2)	32	R/W	0000_0000h	13.3.5.11/ 3991
32E2_0030	CSI Frame Buffer Parameter Register (CSI_CSIFBUF_PARA)	32	R/W	0000_0000h	13.3.5.12/ 3991
32E2_0034	CSI Image Parameter Register (CSI_CSIIMAG_PARA)	32	R/W	0000_0000h	13.3.5.13/ 3992
32E2_0048	CSI Control Register 18 (CSI_CSICR18)	32	R/W	0002_D000h	13.3.5.14/ 3993

13.3.5.1 CSI Control Register 1 (CSI_CSICR1)

This register controls the sensor interface timing and interrupt generation. The interrupt enable bits in this register control the interrupt signals and the status bits. That means status bits will only function when the corresponding interrupt bits are enabled.

Address: 32E2_0000h base + 0h offset = 32E2_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SWAP16_EN	EXT_VSYNC	EOF_INT_EN	PrP_IF_EN	VIDEO_MODE	COF_INT_EN	SF_OR_INTEN	RF_OR_INTEN	Reserved	SFF_DMA_DONE_INTEN	STATFF_INTEN	FB2_DMA_DONE_INTEN	FB1_DMA_DONE_INTEN	RXFF_INTEN	SOF_POL	SOF_INTEN
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				HSYNC_POL	CCIR_EN	Reserved	FCC	PACK_DIR	CLR_STATFIFO	CLR_RXFIFO	GCLK_MODE	INV_DATA	INV_PCLK	REDGE	PIXEL_BIT
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

CSI_CSICR1 field descriptions

Field	Description
31 SWAP16_EN	SWAP 16-Bit Enable. This bit enables the swapping of 16-bit data. Data is packed from 8-bit or 10-bit to 32-bit first (according to the setting of PACK_DIR) and then swapped as 16-bit words before being put into the RX FIFO. The action of the bit only affects the RX FIFO and has no affect on the STAT FIFO. **NOTE:** Example of swapping enabled: Data input to FIFO = 0x11223344 Data in RX FIFO = 0x 33441122 **NOTE:** Example of swapping disabled: Data input to FIFO = 0x11223344 Data in RX FIFO = 0x11223344

Table continues on the next page...

CSI_CSICR1 field descriptions (continued)

Field	Description
	0 Disable swapping 1 Enable swapping
30 EXT_VSYNC	External VSYNC Enable. This bit controls the operational VSYNC mode. **NOTE:** This only works when the CSI is in CCIR progressive mode. 0 Internal VSYNC mode 1 External VSYNC mode
29 EOF_INT_EN	End-of-Frame Interrupt Enable. This bit enables and disables the EOF interrupt. 0 EOF interrupt is disabled. 1 EOF interrupt is generated when RX count value is reached.
28 PrP_IF_EN	CSI-PrP Interface Enable. This bit controls the CSI to PrP bus. When enabled the RxFIFO is detached from the AHB bus and connected to PrP. All CPU reads or DMA accesses to the RxFIFO register are ignored. All CSI interrupts are also masked. 0 CSI to PrP bus is disabled 1 CSI to PrP bus is enabled
27 VIDEO_MODE	Video mode select. This bit controls the video mode in CCIR mode. 0 Progressive mode is selected 1 Interlace mode is selected
26 COF_INT_EN	Change Of Image Field (COF) Interrupt Enable. This bit enables the COF interrupt. This bit works only in CCIR interlace mode which is when CCIR_EN = 1 and CCIR_MODE = 1. 0 COF interrupt is disabled 1 COF interrupt is enabled
25 SF_OR_INTEN	STAT FIFO Overrun Interrupt Enable. This bit enables the STATFIFO overrun interrupt. 0 STATFIFO overrun interrupt is disabled 1 STATFIFO overrun interrupt is enabled
24 RF_OR_INTEN	RxFIFO Overrun Interrupt Enable. This bit enables the RX FIFO overrun interrupt. 0 RxFIFO overrun interrupt is disabled 1 RxFIFO overrun interrupt is enabled
23 -	This field is reserved. Reserved. This bit is reserved and should read 0.
22 SFF_DMA_ DONE_INTEN	STATFIFO DMA Transfer Done Interrupt Enable. This bit enables the interrupt of STATFIFO DMA transfer done. 0 STATFIFO DMA Transfer Done interrupt disable 1 STATFIFO DMA Transfer Done interrupt enable
21 STATFF_INTEN	STATFIFO Full Interrupt Enable. This bit enables the STAT FIFO interrupt. 0 STATFIFO full interrupt disable 1 STATFIFO full interrupt enable
20 FB2_DMA_ DONE_INTEN	Frame Buffer2 DMA Transfer Done Interrupt Enable. This bit enables the interrupt of Frame Buffer2 DMA transfer done.

Table continues on the next page...

CSI_CSICR1 field descriptions (continued)

Field	Description
	0 Frame Buffer2 DMA Transfer Done interrupt disable 1 Frame Buffer2 DMA Transfer Done interrupt enable
19 FB1_DMA_DONE_INTEN	Frame Buffer1 DMA Transfer Done Interrupt Enable. This bit enables the interrupt of Frame Buffer1 DMA transfer done. 0 Frame Buffer1 DMA Transfer Done interrupt disable 1 Frame Buffer1 DMA Transfer Done interrupt enable
18 RXFF_INTEN	RxFIFO Full Interrupt Enable. This bit enables the RxFIFO full interrupt. 0 RxFIFO full interrupt disable 1 RxFIFO full interrupt enable
17 SOF_POL	SOF Interrupt Polarity. This bit controls the condition that generates an SOF interrupt. 0 SOF interrupt is generated on SOF falling edge 1 SOF interrupt is generated on SOF rising edge
16 SOF_INTEN	Start Of Frame (SOF) Interrupt Enable. This bit enables the SOF interrupt. 0 SOF interrupt disable 1 SOF interrupt enable
15–12 Reserved	This field is reserved. Reserved. This field is reserved.
11 HSYNC_POL	HSYNC Polarity Select. This bit controls the polarity of HSYNC. This bit only works in gated-clock-that is, GCLK_MODE = 1 and CCIR_EN = 0. 0 HSYNC is active low 1 HSYNC is active high
10 CCIR_EN	CCIR656 Interface Enable. This bit selects the type of interface used. When the CCIR656 timing decoder is enabled, it replaces the function of timing interface logic. 0 Traditional interface is selected. Timing interface logic is used to latch data. 1 CCIR656 interface is selected.
9 Reserved	This field is reserved. This field is reserved.
8 FCC	FIFO Clear Control. This bit determines how the RXFIFO and STATFIFO are cleared. When Synchronous FIFO clear is selected the RXFIFO and STATFIFO are cleared, and STAT block is reset, on every SOF. FIFOs and STAT block restarts immediately after reset. For information on the operation when Asynchronous FIFO clear is selected, refer to the descriptions for the CLR_RXFIFO and CLR_STATFIFO bits. 0 Asynchronous FIFO clear is selected. 1 Synchronous FIFO clear is selected.
7 PACK_DIR	Data Packing Direction. This bit Controls how 8-bit/10-bit image data is packed into 32-bit RX FIFO, and how 16-bit statistical data is packed into 32-bit STAT FIFO. 0 Pack from LSB first. For image data, 0x11, 0x22, 0x33, 0x44, it will appear as 0x44332211 in RX FIFO. For stat data, 0xAAAA, 0xBBBB, it will appear as 0xBBBBAAAA in STAT FIFO. 1 Pack from MSB first. For image data, 0x11, 0x22, 0x33, 0x44, it will appear as 0x11223344 in RX FIFO. For stat data, 0xAAAA, 0xBBBB, it will appear as 0xAAAABBBB in STAT FIFO.

Table continues on the next page...

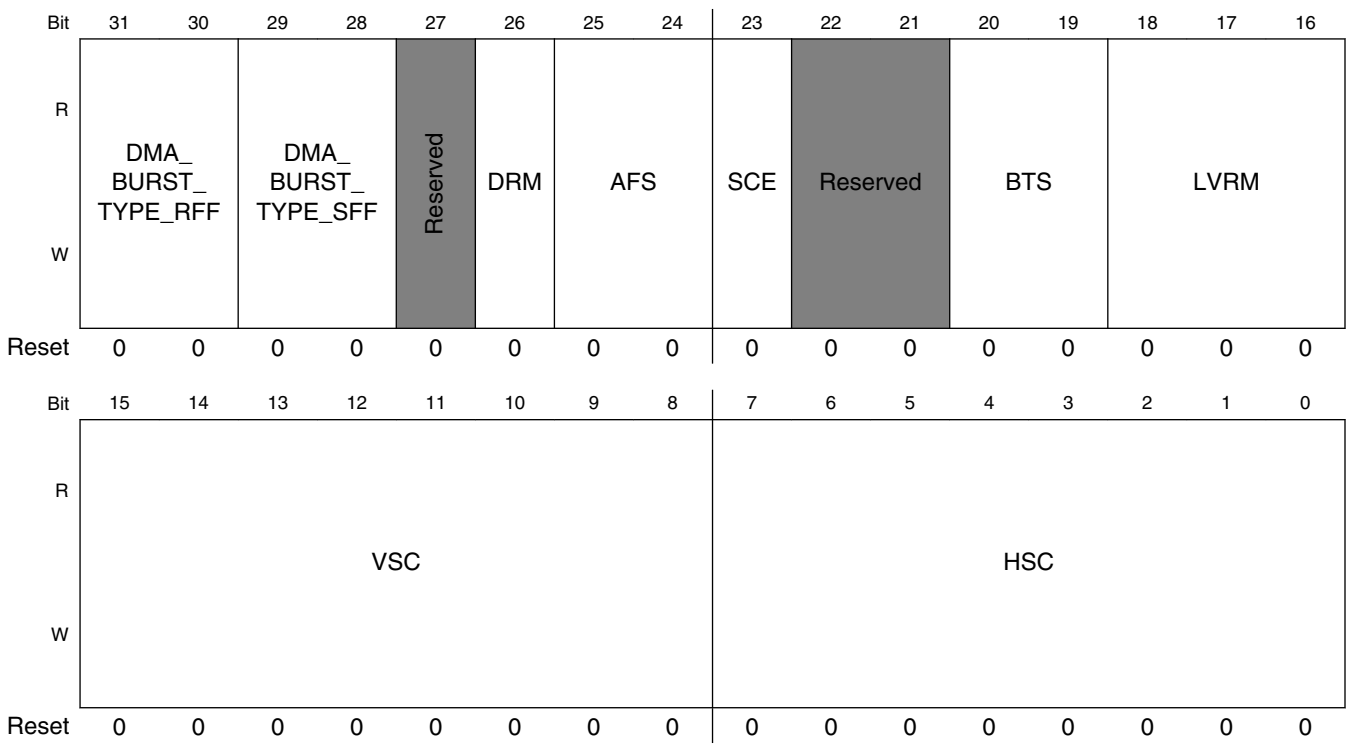
CSI_CSICR1 field descriptions (continued)

Field	Description
6 CLR_STATFIFO	Asynchronous STATFIFO Clear. This bit clears the STATFIFO and Reset STAT block. This bit works only in async FIFO clear mode-that is, FCC = 0. Otherwise this bit is ignored. Writing 1 will clear STATFIFO and reset STAT block immediately, STATFIFO and STAT block then wait and restart after the arrival of next SOF. The bit is restored to 0 automatically after finish. Normally reads 0.
5 CLR_RXFIFO	Asynchronous RXFIFO Clear. This bit clears the RXFIFO. This bit works only in async FIFO clear mode-that is, FCC = 0. Otherwise this bit is ignored. Writing 1 clears the RXFIFO immediately, RXFIFO restarts immediately after that. The bit is restored to 0 automatically after finish. Normally reads 0.
4 GCLK_MODE	Gated Clock Mode Enable. Controls if CSI is working in gated or non-gated mode. This bit works only in traditional mode-that is, CCIR_EN = 0. Otherwise this bit is ignored. 0 Non-gated clock mode. All incoming pixel clocks are valid. HSYNC is ignored. 1 Gated clock mode. Pixel clock signal is valid only when HSYNC is active.
3 INV_DATA	Invert Data Input. This bit enables or disables internal inverters on the data lines. 0 CSI_D[7:0] data lines are directly applied to internal circuitry 1 CSI_D[7:0] data lines are inverted before applied to internal circuitry
2 INV_PCLK	Invert Pixel Clock Input. This bit determines if the Pixel Clock (CSI_PIXCLK) is inverted before it is applied to the CSI module. 0 CSI_PIXCLK is directly applied to internal circuitry 1 CSI_PIXCLK is inverted before applied to internal circuitry
1 REDGE	Valid Pixel Clock Edge Select. Selects which edge of the CSI_PIXCLK is used to latch the pixel data. 0 Pixel data is latched at the falling edge of CSI_PIXCLK 1 Pixel data is latched at the rising edge of CSI_PIXCLK
0 PIXEL_BIT	Pixel Bit. This bit indicates the bayer data width for each pixel. This bit should be configured before activating or re-starting the embedded DMA controller. 0 8-bit data for each pixel 1 10-bit data for each pixel

13.3.5.2 CSI Control Register 2 (CSI_CSICR2)

This register provides the statistic block with data about which live view resolution is being used, and the starting sensor pixel of the Bayer pattern. It also contains the horizontal and vertical count used to determine the number of pixels to skip between the 64 x 64 blocks of statistics when generating statistics on live view image that are greater than 512 x 384.

Address: 32E2_0000h base + 4h offset = 32E2_0004h



CSI_CSICR2 field descriptions

Field	Description
31–30 DMA_BURST_ TYPE_RFF	Burst Type of DMA Transfer from RxFIFO. Selects the burst type of DMA transfer from RxFIFO. X0 INCR8 01 INCR4 11 INCR16
29–28 DMA_BURST_ TYPE_SFF	Burst Type of DMA Transfer from STATFIFO. Selects the burst type of DMA transfer from STATFIFO. X0 INCR8 01 INCR4 11 INCR16

Table continues on the next page...

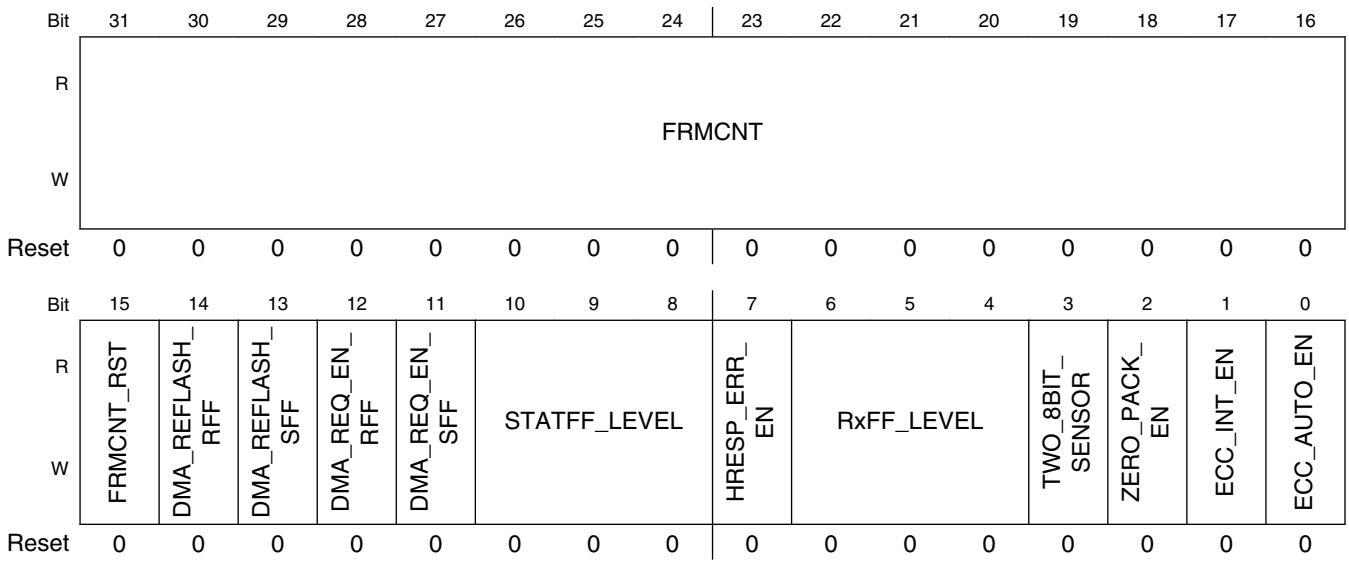
CSI_CSICR2 field descriptions (continued)

Field	Description
27 -	This field is reserved. Reserved. These bit is reserved and should read 0.
26 DRM	Double Resolution Mode. Controls size of statistics grid. 0 Stats grid of 8 x 6 1 Stats grid of 8 x 12
25–24 AFS	Auto Focus Spread. Selects which green pixels are used for auto-focus. 00 Abs Diff on consecutive green pixels 01 Abs Diff on every third green pixels 1x Abs Diff on every four green pixels
23 SCE	Skip Count Enable. Enables or disables the skip count feature. 0 Skip count disable 1 Skip count enable
22–21 -	This field is reserved. Reserved. These bits are reserved and should read 0.
20–19 BTS	Bayer Tile Start. Controls the Bayer pattern starting point. 00 GR 01 RG 10 BG 11 GB
18–16 LVRM	Live View Resolution Mode. Selects the grid size used for live view resolution. 0 512 x 384 1 448 x 336 2 384 x 288 3 384 x 256 4 320 x 240 5 288 x 216 6 400 x 300
15–8 VSC	Vertical Skip Count. Contains the number of rows to skip. SCE must be 1, otherwise VSC is ignored. 0-255 Number of rows to skip minus 1
HSC	Horizontal Skip Count. Contains the number of pixels to skip. SCE must be 1, otherwise HSC is ignored. 0-255 Number of pixels to skip minus 1

13.3.5.3 CSI Control Register 3 (CSI_CSICR3)

This read/write register acts as an extension of the functionality of the CSI Control register 1, adding additional control and features.

Address: 32E2_0000h base + 8h offset = 32E2_0008h



CSI_CSICR3 field descriptions

Field	Description
31–16 FRMCNT	Frame Counter. This is a 16-bit Frame Counter (Wraps around automatically after reaching the maximum)
15 FRMCNT_RST	Frame Count Reset. Resets the Frame Counter. (Cleared automatically after reset is done) 0 Do not reset 1 Reset frame counter immediately
14 DMA_REFLASH_RFF	Reflash DMA Controller for RxFIFO. This bit reflash the embedded DMA controller for RxFIFO. It should be reflashed before the embedded DMA controller starts to work. (Cleared automatically after reflashing is done) 0 No reflashing 1 Reflash the embedded DMA controller
13 DMA_REFLASH_SFF	Reflash DMA Controller for STATFIFO. This bit reflash the embedded DMA controller for STATFIFO. It should be reflashed before the embedded DMA controller starts to work. (Cleared automatically after reflashing is done) 0 No reflashing 1 Reflash the embedded DMA controller

Table continues on the next page...

CSI_CSICR3 field descriptions (continued)

Field	Description
12 DMA_REQ_EN_RFF	DMA Request Enable for RxFIFO. This bit enables the dma request from RxFIFO to the embedded DMA controller. 0 Disable the dma request 1 Enable the dma request
11 DMA_REQ_EN_SFF	DMA Request Enable for STATFIFO. This bit enables the dma request from STATFIFO to the embedded DMA controller. 0 Disable the dma request 1 Enable the dma request
10–8 STATFF_LEVEL	STATFIFO Full Level. When the number of data in STATFIFO reach this level, STATFIFO full interrupt is generated, or STATFIFO DMA request is sent. 000 4 Double words 001 8 Double words 010 12 Double words 011 16 Double words 100 24 Double words 101 32 Double words 110 48 Double words 111 64 Double words
7 HRESP_ERR_EN	Hresponse Error Enable. This bit enables the hresponse error interrupt. 0 Disable hresponse error interrupt 1 Enable hresponse error interrupt
6–4 RxFF_LEVEL	**RxFIFO Full Level**. When the number of data in RxFIFO reaches this level, a RxFIFO full interrupt is generated, or an RXFIFO DMA request is sent. 000 4 Double words 001 8 Double words 010 16 Double words 011 24 Double words 100 32 Double words 101 48 Double words 110 64 Double words 111 96 Double words
3 TWO_8BIT_SENSOR	16-bit Sensor Mode. This bit indicates one 16-bit sensor connected to the 16-bit data ports. This bit should be set if there is one 16-bit sensor connected. This bit should be configured before activating or restarting the embedded DMA controller. 0 Only one 8-bit sensor is connected. 1 One 16-bit sensor is connected.
2 ZERO_PACK_EN	Dummy Zero Packing Enable. This bit causes a dummy zero to be packed with every 3 incoming bytes, forming a 32-bit word. The dummy zero is always packed to the LSB position. This packing function is only available in 8-bit/pixel mode. 0 Zero packing disabled 1 Zero packing enabled

Table continues on the next page...

CSI_CSICR3 field descriptions (continued)

Field	Description
1 ECC_INT_EN	Error Detection Interrupt Enable. This bit enables and disables the error detection interrupt. This feature only works in CCIR interlace mode. 0 No interrupt is generated when error is detected. Only the status bit ECC_INT is set. 1 Interrupt is generated when error is detected.
0 ECC_AUTO_EN	Automatic Error Correction Enable. This bit enables and disables the automatic error correction. If an error occurs and error correction is disabled only the ECC_INT status bit is set. This feature only works in CCIR interlace mode. 0 Auto Error correction is disabled. 1 Auto Error correction is enabled.

13.3.5.4 CSI Statistic FIFO Register (CSI_CSISTATFIFO)

The StatFIFO is a read-only register containing statistic data from the sensor. Writing to this register has no effect.

Address: 32E2_0000h base + Ch offset = 32E2_000Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STAT																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CSI_CSISTATFIFO field descriptions

Field	Description
STAT	Static data from sensor

13.3.5.5 CSI RX FIFO Register (CSI_CSIRFIFO)

This read-only register contains received image data. Writing to this register has no effect.

Address: 32E2_0000h base + 10h offset = 32E2_0010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IMAGE																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CSI_CSIRFIFO field descriptions

Field	Description
IMAGE	Received image data

13.3.5.6 CSI RX Count Register (CSI_CSIRXCNT)

This register works for EOF interrupt generation. It should be set to the number of words to receive that would generate an EOF interrupt.

There is an internal counter that counts the number of words read from the RX FIFO. Whenever the RX FIFO is being read, by either the CPU or the embedded DMA controller, the counter value is updated and compared with this register. If the values match, then an EOF interrupt is triggered.

Address: 32E2_0000h base + 14h offset = 32E2_0014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										RXCNT																					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0

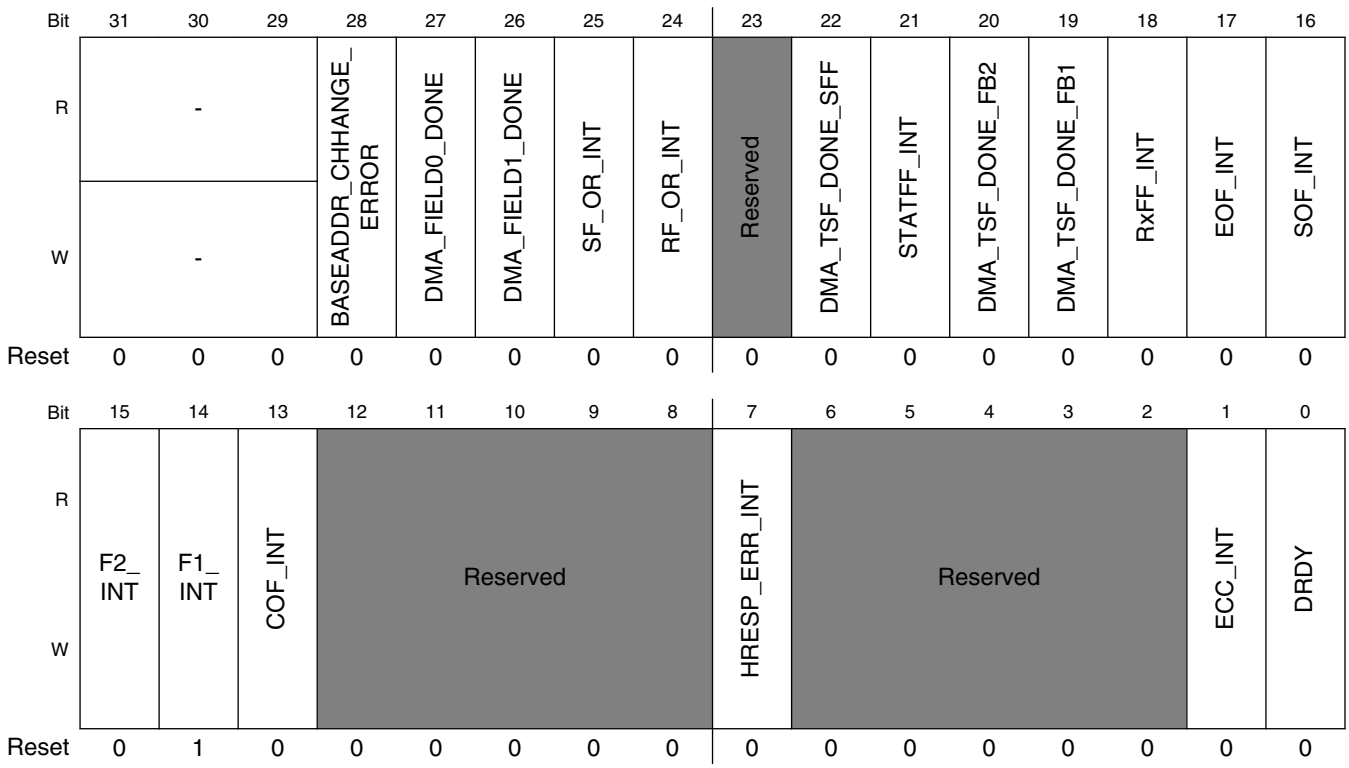
CSI_CSIRXCNT field descriptions

Field	Description
31–22 -	This field is reserved. Reserved. These bits are reserved and should read 0.
RXCNT	RxFIFO Count. This 22-bit counter for RXFIFO is updated each time the RXFIFO is read by CPU or DMA.This counter should be set to the expected number of words to receive that would generate an EOF interrupt.

13.3.5.7 CSI Status Register (CSI_CSISR)

This read/write register shows sensor interface status, and which kind of interrupt is being generated. The corresponding interrupt bits must be set for the status bit to function. Status bits should function normally even if the corresponding interrupt enable bits are not enabled.

Address: 32E2_0000h base + 18h offset = 32E2_0018h



CSI_CSISR field descriptions

Field	Description
31–29 -	Reserved. These bits are reserved and should read 0.
28 BASEADDR_CHHANGE_ERROR	When using base address switching enable, this bit will be 1 when switching occur before DMA complete. This bit will be clear by writing 1. When this interrupt happens, follow the steps listed below. 1. Unassert the CSI enable, CSIx_CSICR18 bit31, 2. Reflash the DMA, assert the CSIX_CSICR3 bit 14, 3. Assert the CSI enable, CSIx_CSICR18 bit31.
27 DMA_FIELD0_DONE	When DMA field 0 is complete, this bit will be set to 1(clear by writing 1).

Table continues on the next page...

CSI_CSISR field descriptions (continued)

Field	Description
26 DMA_FIELD1_ DONE	When DMA field 0 is complete, this bit will be set to 1(clear by writing 1).
25 SF_OR_INT	STATFIFO Overrun Interrupt Status. Indicates the overflow status of the STATFIFO register. (Cleared by writing 1) 0 STATFIFO has not overflowed. 1 STATFIFO has overflowed.
24 RF_OR_INT	RxFIFO Overrun Interrupt Status. Indicates the overflow status of the RxFIFO register. (Cleared by writing 1) 0 RXFIFO has not overflowed. 1 RXFIFO has overflowed.
23 -	This field is reserved. Reserved. This bit is reserved and should read 0.
22 DMA_TSF_ DONE_SFF	DMA Transfer Done from StatFIFO. Indicates that the dma transfer from StatFIFO is completed. It can trigger an interrupt if the corresponding enable bit is set in CSICR1. This bit can be cleared by writting 1 or reflashing the StatFIFO dma controller in CSICR3.(Cleared by writing 1) 0 DMA transfer is not completed. 1 DMA transfer is completed.
21 STATFF_INT	STATFIFO Full Interrupt Status. Indicates the number of data in the STATFIFO reaches the trigger level. (this bit is cleared automatically by reading the STATFIFO) 0 STATFIFO is not full. 1 STATFIFO is full.
20 DMA_TSF_ DONE_FB2	DMA Transfer Done in Frame Buffer2. Indicates that the DMA transfer from RxFIFO to Frame Buffer2 is completed. It can trigger an interrupt if the corresponding enable bit is set in CSICR1. This bit can be cleared by by writting 1 or reflashing the RxFIFO dma controller in CSICR3. (Cleared by writing 1) 0 DMA transfer is not completed. 1 DMA transfer is completed.
19 DMA_TSF_ DONE_FB1	DMA Transfer Done in Frame Buffer1. Indicates that the DMA transfer from RxFIFO to Frame Buffer1 is completed. It can trigger an interrupt if the corresponding enable bit is set in CSICR1. This bit can be cleared by by writting 1 or reflashing the RxFIFO dma controller in CSICR3. (Cleared by writing 1) 0 DMA transfer is not completed. 1 DMA transfer is completed.
18 RxFF_INT	RXFIFO Full Interrupt Status. Indicates the number of data in the RxFIFO reaches the trigger level. (this bit is cleared automatically by reading the RxFIFO) 0 RxFIFO is not full. 1 RxFIFO is full.
17 EOF_INT	End of Frame (EOF) Interrupt Status. Indicates when EOF is detected. (Cleared by writing 1) 0 EOF is not detected. 1 EOF is detected.
16 SOF_INT	Start of Frame Interrupt Status. Indicates when SOF is detected. (Cleared by writing 1)

Table continues on the next page...

CSI_CSISR field descriptions (continued)

Field	Description
	0 SOF is not detected. 1 SOF is detected.
15 F2_INT	CCIR Field 2 Interrupt Status. Indicates the presence of field 2 of video in CCIR mode. (Cleared automatically when current field does not match) **NOTE:** Only works in CCIR Interlace mode. 0 Field 2 of video is not detected 1 Field 2 of video is about to start
14 F1_INT	CCIR Field 1 Interrupt Status. Indicates the presence of field 1 of video in CCIR mode. (Cleared automatically when current field does not match) **NOTE:** Only works in CCIR Interlace mode. 0 Field 1 of video is not detected. 1 Field 1 of video is about to start.
13 COF_INT	Change Of Field Interrupt Status. Indicates that a change of the video field has been detected. Only works in CCIR Interlace mode. Software should read this bit first and then dispatch the new field from F1_INT and F2_INT. (Cleared by writing 1) 0 Video field has no change. 1 Change of video field is detected.
12–8 -	This field is reserved. Reserved. These bits are reserved and should read 0.
7 HRESP_ERR_ INT	Hresponse Error Interrupt Status. Indicates that a hresponse error has been detected. (Cleared by writing 1) 0 No hresponse error. 1 Hresponse error is detected.
6–2 -	This field is reserved. Reserved. These bits are reserved and should read 0.
1 ECC_INT	CCIR Error Interrupt. This bit indicates an error has occurred. This only works in CCIR Interlace mode. (Cleared by writing 1) 0 No error detected 1 Error is detected in CCIR coding
0 DRDY	RXFIFO Data Ready. Indicates the presence of data that is ready for transfer in the RxFIFO. (Cleared automatically by reading FIFO) 0 No data (word) is ready 1 At least 1 datum (word) is ready in RXFIFO.

13.3.5.8 CSI DMA Start Address Register - for STATFIFO (CSI_CSIDMASA_STATFIFO)

This register provides the start address for the embedded DMA controller of STATFIFO. The embedded DMA controller will read data from STATFIFO and write it to the external memory from the start address. This register should be configured before activating or restarting the embedded DMA controller.

Address: 32E2_0000h base + 20h offset = 32E2_0020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_START_ADDR_SFF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_START_ADDR_SFF														Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CSI_CSIDMASA_STATFIFO field descriptions

Field	Description
31–2 DMA_START_ ADDR_SFF	DMA Start Address for STATFIFO. Indicates the start address to write data. The embedded DMA controller will read data from STATFIFO and write it from this address through AHB bus. The address should be double words aligned.
-	This field is reserved. Reserved. These bits are reserved and should read 0.

13.3.5.9 CSI DMA Transfer Size Register - for STATFIFO (CSI_CSIDMATS_STATFIFO)

This register provides the total transfer size for the embedded DMA controller of STATFIFO. This register should be configured before activating or restarting the embedded DMA controller.

Address: 32E2_0000h base + 24h offset = 32E2_0024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_TSF_SIZE_SFF																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CSI_CSIDMATS_STATFIFO field descriptions

Field	Description
DMA_TSF_SIZE_SFF	DMA Transfer Size for STATFIFO. Indicates how many words to be transfered by the embedded DMA controller. The size should be double words aligned.

13.3.5.10 CSI DMA Start Address Register - for Frame Buffer1 (CSI_CSIDMASA_FB1)

This register provides the start address in the frame buffer1 for the embedded DMA controller of RxFIFO. The embedded DMA controller will read data from RxFIFO and write it to the frame buffer1 from the start address. This register should be configured before activating or restarting the embedded DMA controller.

Address: 32E2_0000h base + 28h offset = 32E2_0028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_START_ADDR_FB1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_START_ADDR_FB1														Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CSI_CSIDMASA_FB1 field descriptions

Field	Description
31–2 DMA_START_ADDR_FB1	DMA Start Address in Frame Buffer1. Indicates the start address to write data. The embedded DMA controller will read data from RxFIFO and write it from this address through AHB bus. The address should be double words aligned.
-	This field is reserved. Reserved. These bits are reserved and should read 0.

13.3.5.11 CSI DMA Transfer Size Register - for Frame Buffer2 (CSI_CSIDMASA_FB2)

This register provides the start address in the frame buffer2 for the embedded DMA controller of RxFIFO. The embedded DMA controller will read data from RxFIFO and write it to the frame buffer2 from the start address. This register should be configured before activating or restarting the embedded DMA controller.

Address: 32E2_0000h base + 2Ch offset = 32E2_002Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DMA_START_ADDR_FB2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DMA_START_ADDR_FB2														Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CSI_CSIDMASA_FB2 field descriptions

Field	Description
31–2 DMA_START_ ADDR_FB2	DMA Start Address in Frame Buffer2. Indicates the start address to write data. The embedded DMA controller will read data from RxFIFO and write it from this address through AHB bus. The address should be double words aligned.
-	This field is reserved. Reserved. These bits are reserved and should read 0.

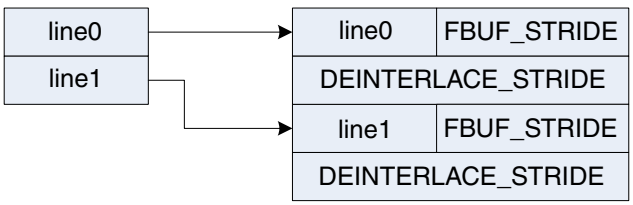
13.3.5.12 CSI Frame Buffer Parameter Register (CSI_CSIFBUF_PARA)

This register provides the stride of the frame buffer to show how many words to skip before starting to write the next row of the image. The width of the frame buffer minus the width of the image is the stride. This register should be configured before activating or restarting the embedded DMA controller.

Address: 32E2_0000h base + 30h offset = 32E2_0030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DEINTERLACE_STRIDE																FBUF_STRIDE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CSI_CSIFBUF_PARA field descriptions

Field	Description
31–16 DEINTERLACE_ STRIDE	DEINTERLACE_STRIDE is only used in the deinterlace mode. If line stride feature is supported in deinterlace mode, FBUF_STRIDE and DEINTERLACE_STRIDE need to be configured at the same time. DEINTERLACE_STRIDE is configured the same as line width. In normal line stride feature, only FBUF_STRIDE needs to be configured. 
FBUF_STRIDE	Frame Buffer Parameter. Indicates the stride of the frame buffer. The width of the frame buffer(in double words) minus the width of the image(in double words) is the stride. The stride should be double words aligned. The embedded DMA controller will skip the stride before starting to write the next row of the image.

13.3.5.13 CSI Image Parameter Register (CSI_CSIIMAG_PARA)

This register provides the width and the height of the image from the sensor. The width and height should be aligned in pixel. The width of the image multiplied by the height is the total pixel size that will be transfered in a frame by the embedded DMA controller. This register should be configured before activating or restarting the embedded DMA controller.

Address: 32E2_0000h base + 34h offset = 32E2_0034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMAGE_WIDTH																IMAGE_HEIGHT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CSI_CSIIMAG_PARA field descriptions

Field	Description
31–16 IMAGE_WIDTH	Image Width. Indicates how many pixels in a line of the image from the sensor. If the input data from the sensor is 8-bit/pixel format, the IMAGE_WIDTH should be a multiple of 8 pixels. If the input data from the sensor is 10-bit/pixel or 16-bit/pixel format, the IMAGE_WIDTH should be a multiple of 4 pixels.
IMAGE_HEIGHT	Image Height. Indicates how many pixels in a column of the image from the sensor.

13.3.5.14 CSI Control Register 18 (CSI_CSICR18)

This read/write register acts as an extension of the functionality of the CSI Control register 1

Address: 32E2_0000h base + 48h offset = 32E2_0048h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CSI_ENABLE	MIPI_DATA_FORMAT						LINE_STRIDE_EN	-	DATA_FROM_MIPI	MIPI_YU_SWAP	MIPI_DOUBLE_CMPNT	MASK_ OPTION		CSI_LCDIF_ BUFFER_ LINES	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AHB_HPROT				Reserved	RGB888A_FORMAT_SEL	BASEADDR_CHANGE_ERROR_IE	LAST_DMA_REQ_SEL	DMA_FIELD1_DONE_IE	FIELD0_DONE_IE	BASEADDR_SWITCH_SEL	BASEADDR_SWITCH_EN	PARALLEL24_EN	DEINTERLACE_EN	Reserved	
W																
Reset	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0

CSI_CSICR18 field descriptions

Field	Description
31 CSI_ENABLE	CSI global enable signal. Only when this bit is 1, CSI can start to receive the data and store to memory.
30–25 MIPI_DATA_FORMAT	Image Data Format Generic Short Packet: 0x08 ~ 0x0F Embedded 8-bit non Image: 0x12 YUV420 (8-bit) 0x18 YUV420 (10-bit) 0x19 YUV420 (8-bit legacy) 0x1A YUV420 (8-bit CSPS) 0x1C YUV420 (10-bit CSPS) 0x1D YUV422 (8-bit) 0x1E YUV422 (10-bit) 0x1F RGB444 0x20 (Not support) RGB555 0x21 (Not support) RGB565 0x22 RGB666 0x23 RGB888 0x24 RAW6 0x28 RAW7 0x29 RAW8 0x2A RAW10 0x2B RAW12 0x2C RAW14 0x2D User defined 1 0x30 User defined 2 0x31 User defined 3 0x32 User defined 4 0x33 User defined 5 0x34 User defined 6 0x35 User defined 7 0x36 User defined 8 0x37
24 LINE_STRIDE_EN	When the line width are not the multiple of the burst length, assert this bit.
23 -	Reserved
22 DATA_FROM_MIPI	0 Data from parallel sensor 1 Data from MIPI
21 MIPI_YU_SWAP	It only works in MIPI CSI YUV422 double component mode.
20 MIPI_DOUBLE_CMPNT	Double component per clock cycle in YUV422 formats. 0 Single component per clock cycle

Table continues on the next page...

CSI_CSICR18 field descriptions (continued)

Field	Description
	(half pixel per clock cycle) 1 Double component per clock cycle (a pixel per clock cycle)
19–18 MASK_OPTION	These bits used to choose the method to mask the CSI input. 00 Writing to memory from first completely frame, when using this option, the CSI_ENABLE should be 1. 01 Writing to memory when CSI_ENABLE is 1. 02 Writing to memory from second completely frame, when using this option, the CSI_ENABLE should be 1. 03 Writing to memory when data comes in, not matter the CSI_ENABLE is 1 or 0.
17–16 CSI_LCDIF_ BUFFER_LINES	The number of lines are used in handshake mode with LCDIF. 00 4 lines 01 8 lines 02 16 lines 03 16 lines
15–12 AHB_HPROT	Hprot value in AHB bus protocol.
11 -	This field is reserved.
10 RGB888A_ FORMAT_SEL	Output is 32-bit format. 0 {8'h0, data[23:0]} 1 {data[23:0], 8'h0}
9 BASEADDR_ CHANGE_ ERROR_IE	Base address change error interrupt enable signal.
8 LAST_DMA_ REQ_SEL	Choosing the last DMA request condition. 0 fifo_full_level 1 hburst_length
7 DMA_FIELD1_ DONE_IE	When in interlace mode, field 1 done interrupt enable. 0 Interrupt disabled 1 Interrupt enabled
6 FIELD0_DONE_ IE	In interlace mode, fileld 0 means interrupt enabled. 0 Interrupt disabled 1 Interrupt enabled
5 BASEADDR_ SWITCH_SEL	CSI 2 base addresses switching method. When using this bit, BASEADDR_SWITCH_EN is 1. 0 Switching base address at the edge of the vsync 1 Switching base address at the edge of the first data of each frame
4 BASEADDR_ SWITCH_EN	When this bit is enabled, CSI DMA will switch the base address according to BASEADDR_SWITCH_SEL rather than atomically by DMA completed.

Table continues on the next page...

CSI_CSICR18 field descriptions (continued)

Field	Description
3 PARALLEL24_ EN	When input is parallel rgb888/yuv444 24bit, this bit can be enabled.
2 DEINTERLACE_ EN	This bit is used to select the output method When input is standard CCIR656 video. 0 Deinterlace disabled 1 Deinterlace enabled
-	This field is reserved. Reserved.

13.4 MIPI CSI Host Controller (MIPI_CSI)

13.4.1 Overview

This chapter describes the camera interface for this chip. It works with the MIPI DPHY module to connect to the host processor.

13.4.1.1 Block Diagram

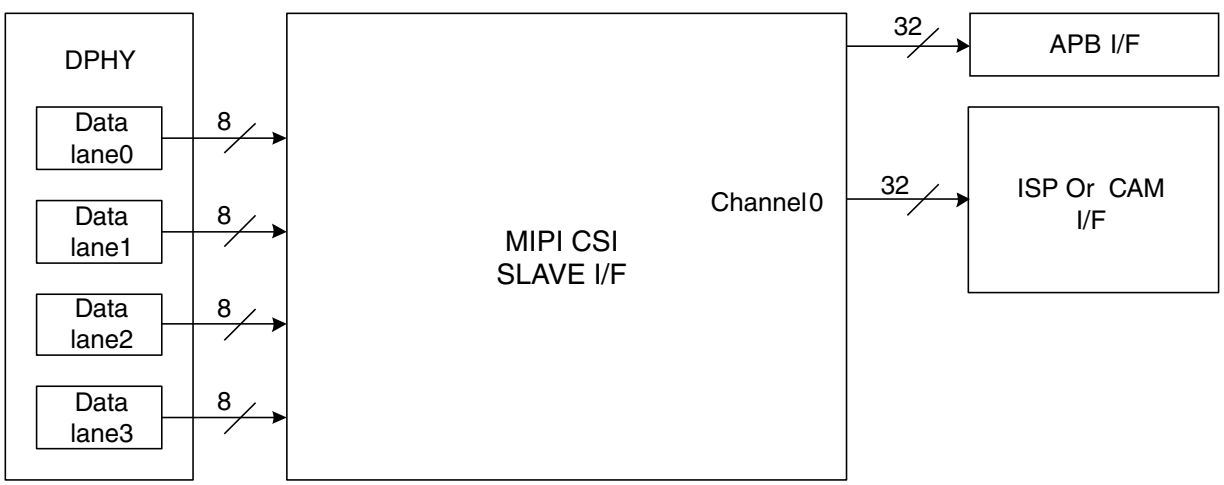


Figure 13-24. Block diagram of camera system with MIPI CSI Slave Interface

13.4.1.2 Features

- MIPI D-PHY specification V1.2 (Board Approved)

- Compliant to MIPI CSI2 Specification V1.3 except for C-PHY feature (Board Approved)
- Support primary and secondary Image format
  - YUV420, YUV420 (Legacy), YUV420 (CSPS), YUV422 of 8-bits and 10-bits
  - RGB565, RGB666, RGB888
  - RAW6, RAW7, RAW8, RAW10, RAW12, RAW14
  - All of User defined Byte-based Data packet
- Support up to 4 lanes of D-PHY
- Interfaces
  - Compatible to PPI(Protocol-to-PHY Interface) in MIPI D-PHY Specification
  - AMBA3.0 APB Slave for Register configuration.
  - Image output data buswidth : 32 bits
- Image memory
  - Size of SRAM is 4KB
- Pixel clock can be gated when no ppi data is coming.

13.4.1.3 MIPI CSI Slave and D-PHY I/F block diagram

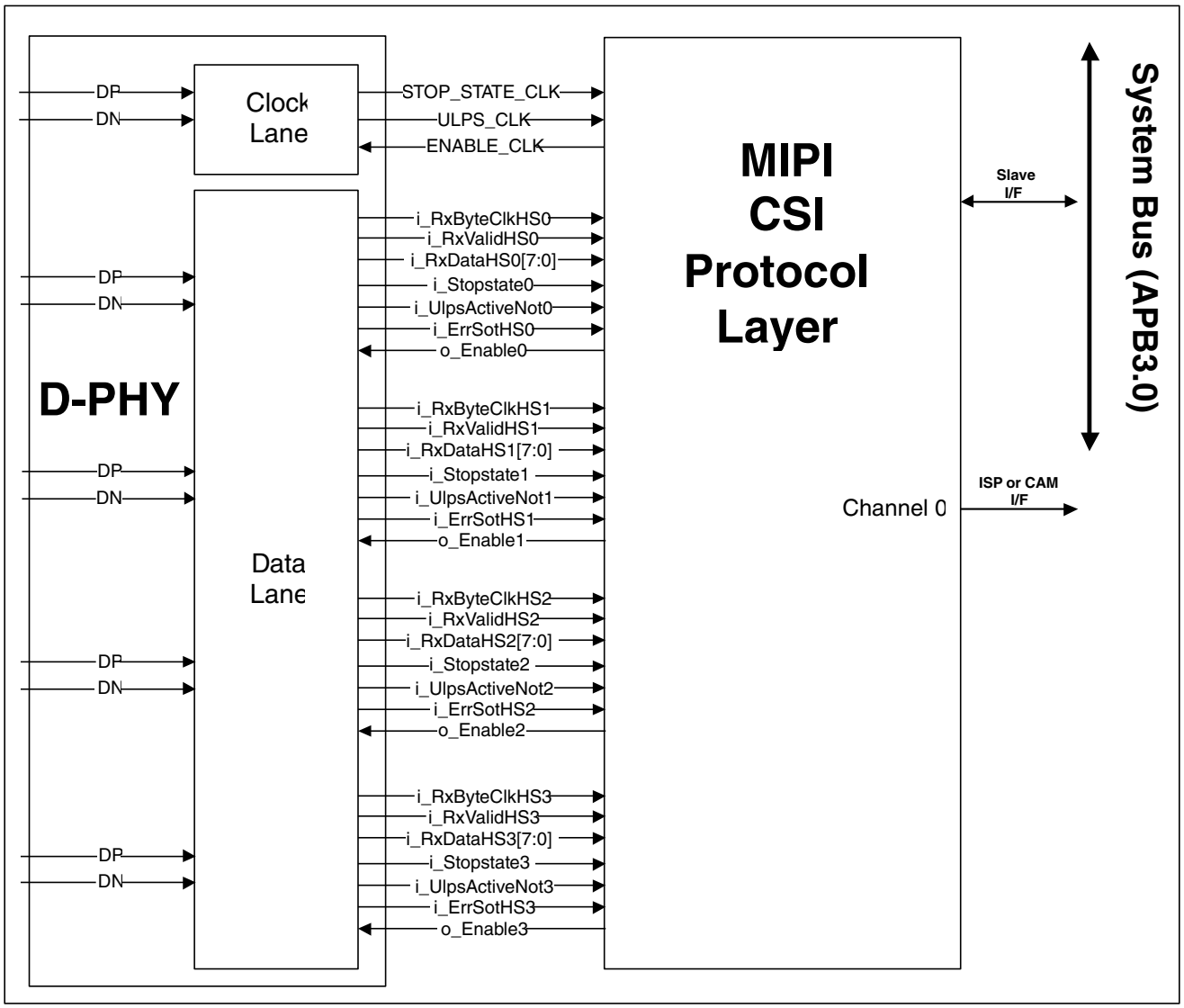


Figure 13-25. PPI I/F block diagram

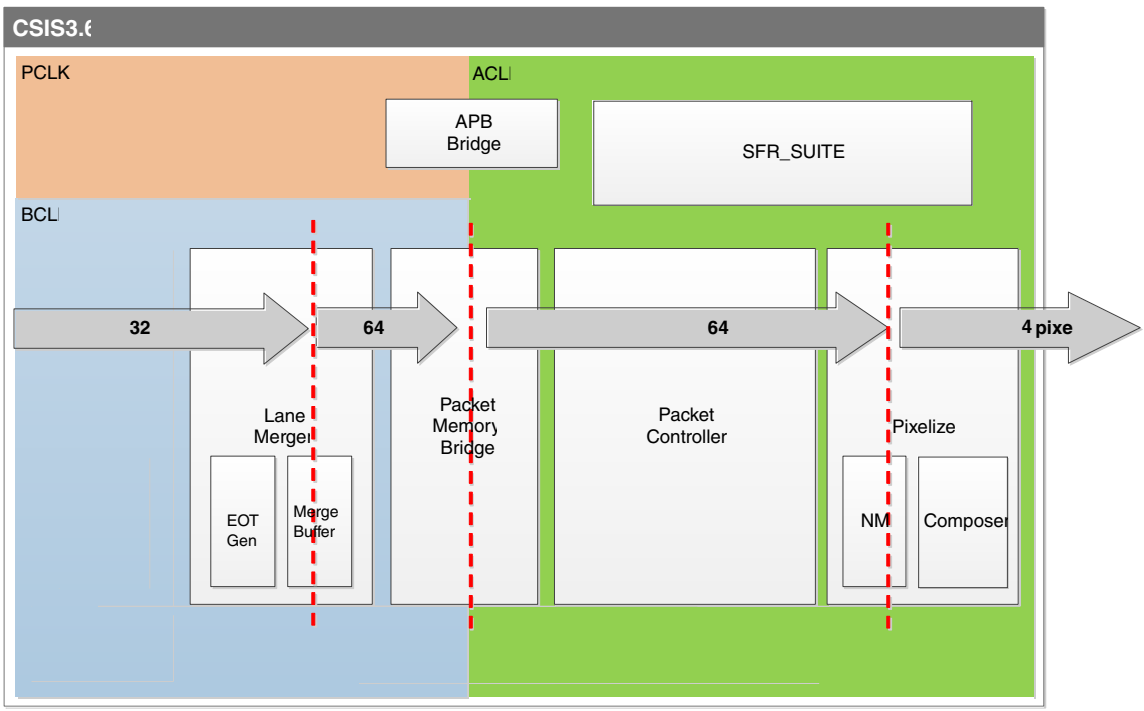


Figure 13-26. CSI Slave Module

13.4.1.4 Memory Bridge

The following table describes the configurable-sized memories.

Table 13-9. Memory bridge list

Usage	Memory Type	Size	Description
Image Data	2 port SRAM	4KB (1024 x 32bits)	Store the image data of virtual channel

There is the timing diagram for read memory in the figure below. If the mux for the split memories is needed, register for storing the address is necessary. Because the RDATA of memory is prepared at 1 clock later when the REN and RADDR are accepted.

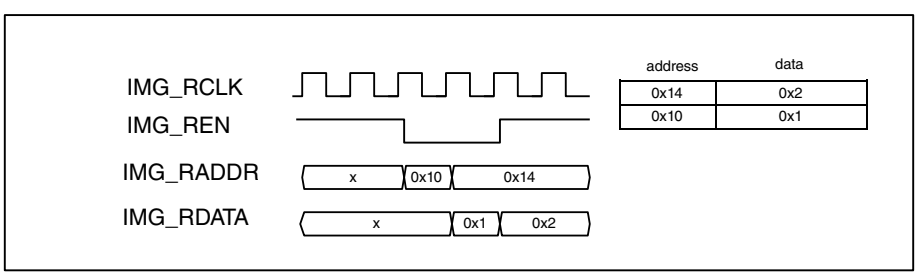


Figure 13-27. Timing diagram of Read memory

13.4.2 Initial Sequence

CSI Rx should be reset with D-PHY. Below Signal diagram is referred from Samsung D-PHY datasheet. This document describes the SFR setting and timing information.

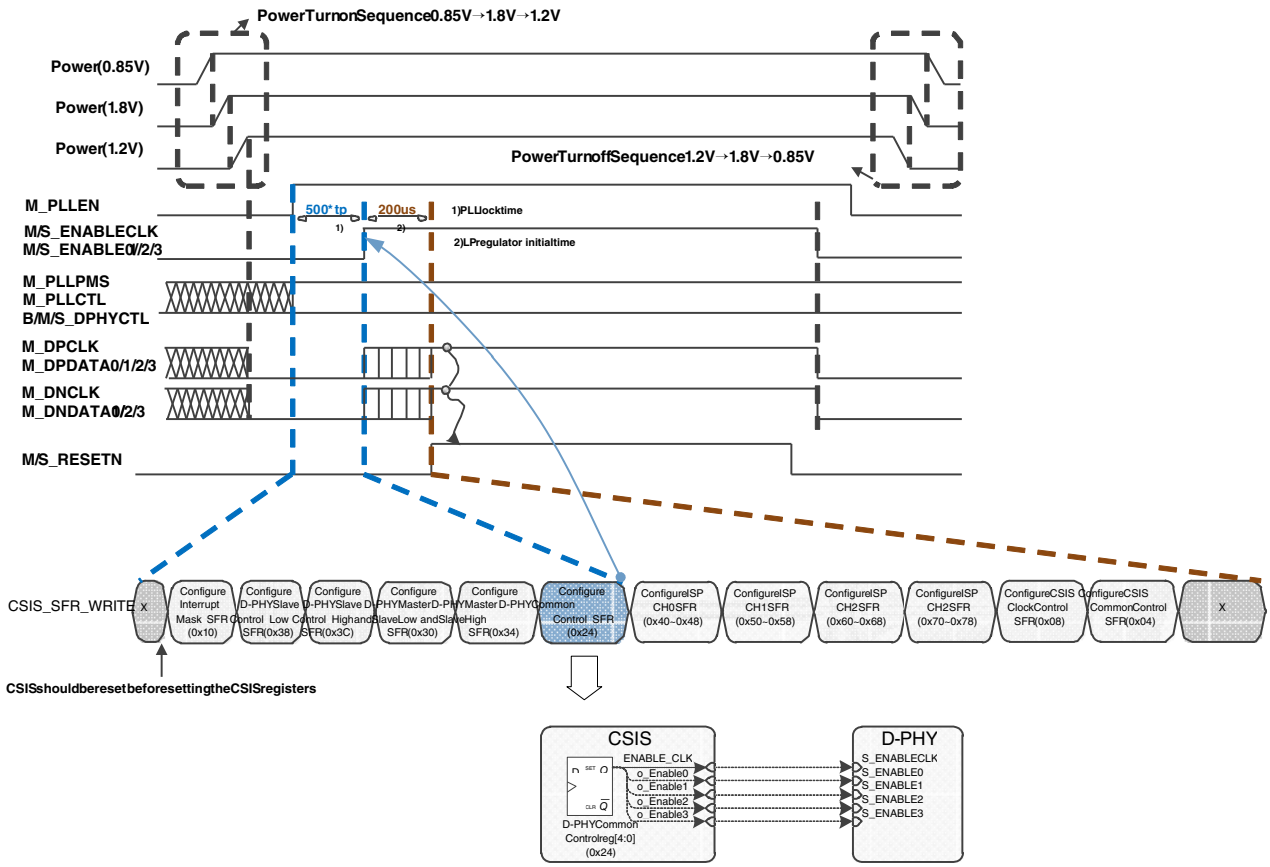


Figure 13-28. Initialize sequence

13.4.3 Protocol

13.4.3.1 D-PHY layer FSM

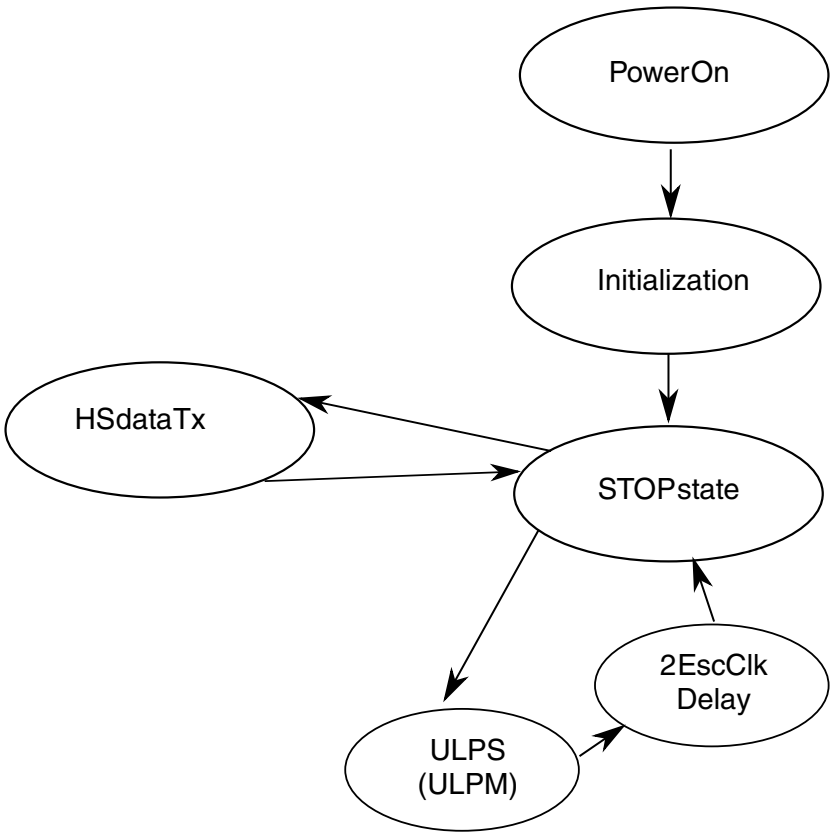


Figure 13-29. D-PHY Finite State Machine

CSIS V3.6.3 system supports HSDT(High Speed Data transfer) and ULPS(ULPM – Utra-Low Power State or Mode) only. There is no trigger function, LPDT and BTA.

13.4.3.2 High Speed Data Transfer

In the following figure, the upper 5 signals are related with clock lane and the lower 6 signals are related with data lane. Dp and Dn of upper signals are D-PHY channel of clock lane. BYTE_CLK is generated clock in the D-PHY. Dp and Dn of lower signals are Data lane. Another signals of lower are Phy Status signals. STOPstate (and STOPstateClk) indicates that differential channel state of D-PHY is LP11 (the voltage level of Dp and Dn is 1.2V)

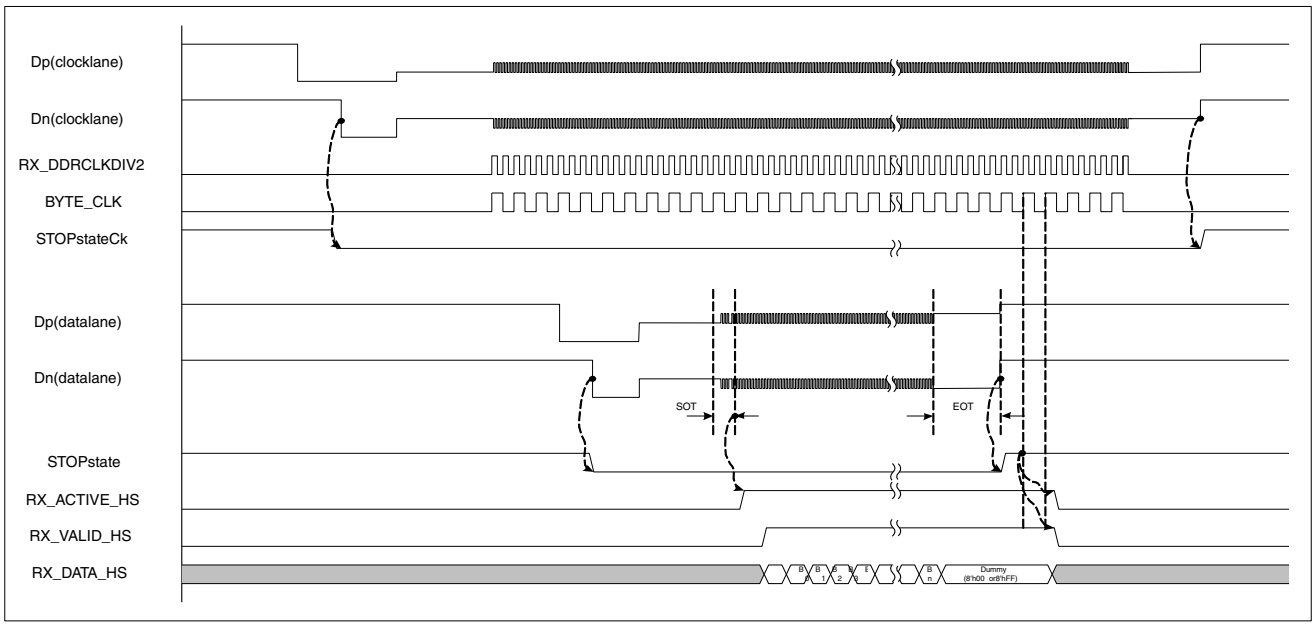


Figure 13-30. Timing Diagram of High Speed Data Transfer

13.4.3.3 Ultra-Low Power Mode

In the following figure, UlpsActiveNot* and STOPstate* is in PPI. ULPS command generated by D-PHY Tx is only for D-PHY Rx and it is decoded by D-PHY Rx and transferred to Link with Ulps data.

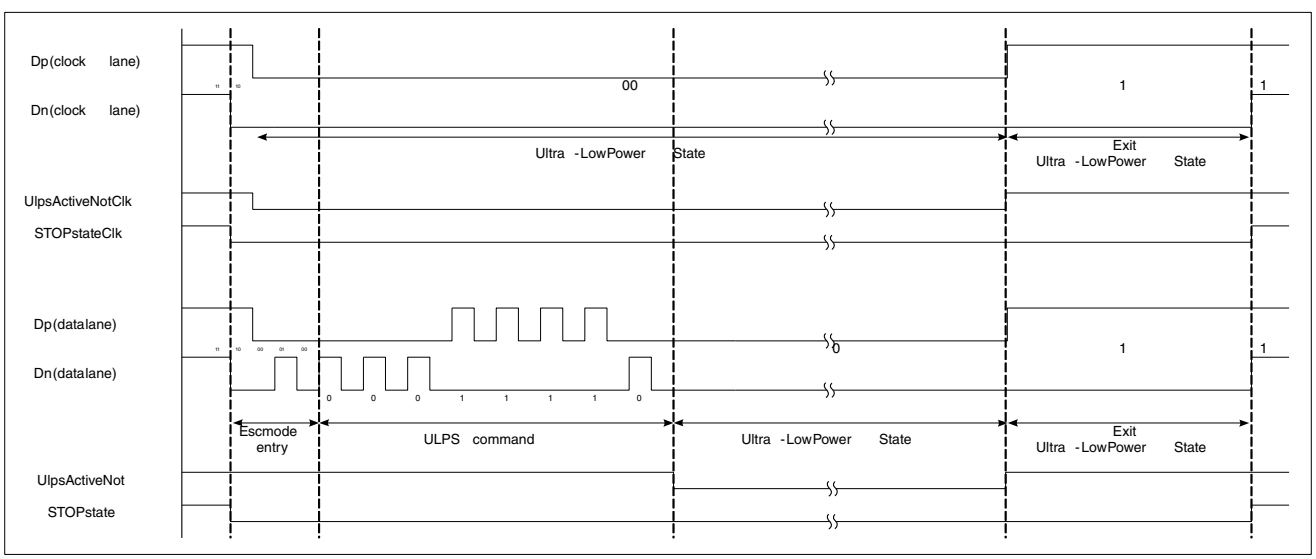


Figure 13-31. Timing Diagram of Ultra Low Power Mode

13.4.4 Interfaces

13.4.4.1 PPI Interface

Table 13-10. PPI I/F pin description

Port Name	I/O	Sync	Description
ENABLE_CLK	O	Async level	Enable clock lane module
STOP_STATE_CLK	I	Async level	Lanes in stop_state
ULPS_CLK	I	Async level	Clock lane is in Ultra Low-Power State (ULPS)
i_RxByteClkHS%	I	-	Rx byte clock in HS mode
i_RxValidHS%	I	Sync RX_BYTE_CLK_HS%	High-Speed receive data valid
i_RxDataHS%	I	Sync RX_BYTE_CLK_HS%	High-Speed receive data
i_ErrSotHS%	I	Sync RX_BYTE_CLK_HS%	Start of Transmission error (Pulse high)
o_Enable%	O	Async level	Enable Data lane module
i_Stopstate%	I	Async level	Lanes in stop_state
i_UlpsActiveNot%	I	Async level	Data Lane is in Ultra Low-Power State (ULPS)

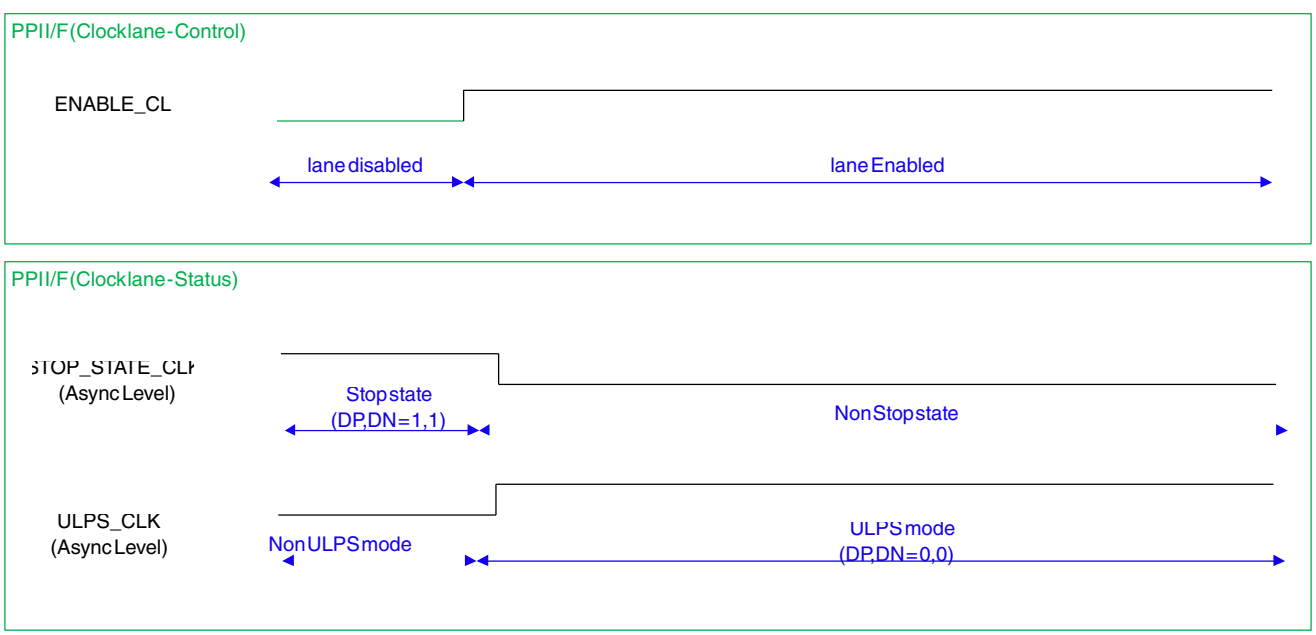


Figure 13-32. PPI I/F (Clock lane) timing diagram

NOTE

DPHY should supply i_RxByteClkHS% at least 15 clock period after falling i_RxValidHS%

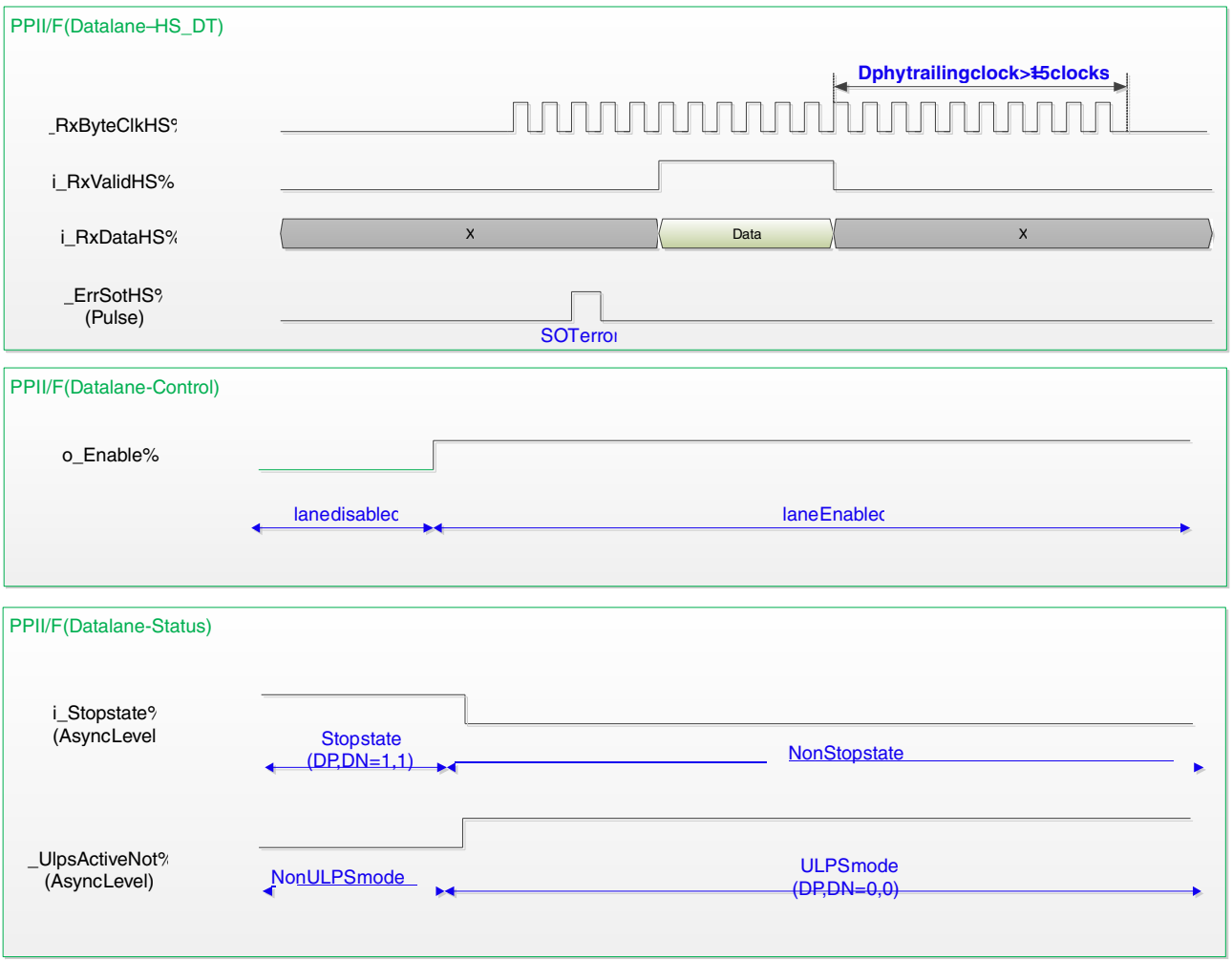


Figure 13-33. PPI I/F (Data lane) timing diagram

13.4.4.2 ISP Interface

CSIS V3.6.3 has output signals which are PIX_CLK, VVALID, HVALID, BVALID, and DATA.

Table 13-11. ISP I/F pin description

Port Name	I/O	Sync	Description
O_PIXCLK	O	-	Pixel clock. This signal is generated from I_ACLK.

Table continues on the next page...

Table 13-11. ISP I/F pin description (continued)

Port Name	I/O	Sync	Description
O_VVALID	O	O_PIX_CLK	Vertical sync signal. This signal is set during whole frame.
O_HVALID	O	O_PIX_CLK	Horizontal sync signal. This signal is set during each line.
O_BVALID	O	O_PIX_CLK	Byte valid signal. This signal indicates the valid number of byte and only valid in the User defined type and Parallel mode.
O_DATA	O	O_PIX_CLK	Pixel data bus. The pixel is aligned based on its data type.

The following figure describes the timing diagram of ISP I/F. All signals are synchronized with the rising edge of PIXCLK.

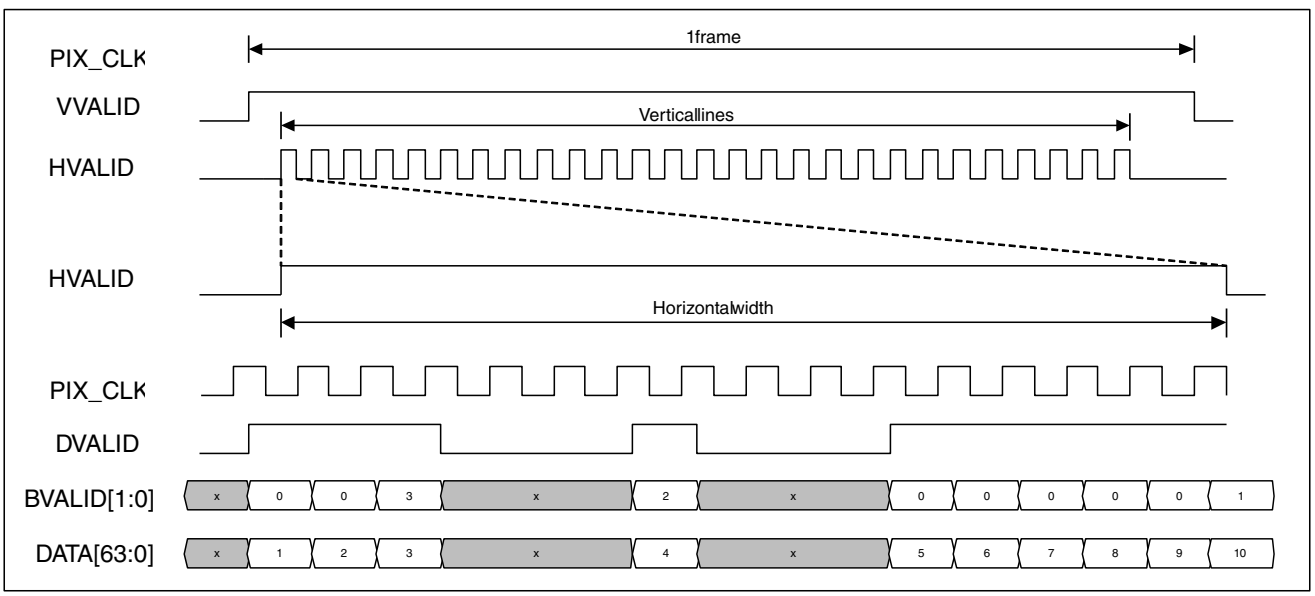


Figure 13-34. Output protocol of ISP Wrapper of CSIS C3.6.3

The following figure describes the sync timing of frame.

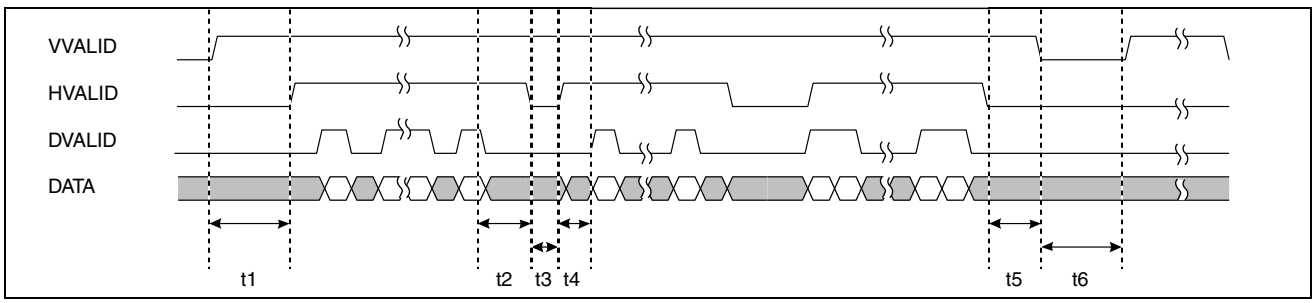


Figure 13-35. Frame sync timing diagram

The elements in the following table control the interval in Figure 13-35. Those elements are configurable by register in ISP SYNC register of CH0.

Table 13-12. Frame sync timing description

	Description	Minimum cycle (# PIX_CLK)	Maximum cycle (# PIX_CLK)
t1	Interval between rising of VVALID and first rising HVALID	1	-
t2	Interval between last falling of DVALID and falling HVALID	Hsync_Lintv + 1	-
t3	Interval between falling of HVALID and rising of next HVALID	1	-
t4	Interval between rising of HVALID and first rising of DVALID	0	-
t5	Interval between last falling of HVALID and falling of VVALID	1	-
t6	Interval between falling of VVALID and rising of next VVALID	1	-

13.4.4.3 SRAM Interface

CSIS V3.6.3 has output signals which are PIX_CLK, VVALID, HVALID, BVALID, and DATA.

Table 13-13. SRAM interface description

Port Name	I/O	Sync	Description
O_IMG_MEM_WCLK	O	-	Write clock for the SRAM of IMG_MEM
O_IMG_MEM_WEN	O	O_IMG_MEM_WCLK	Write enable for the SRAM of IMG_MEM (active low)

Table continues on the next page...

Table 13-13. SRAM interface description (continued)

Port Name	I/O	Sync	Description
O_IMG_MEM_WADDR	O	O_IMG_MEM_WCLK	Write address for the SRAM of IMG_MEM
O_IMG_MEM_WDATA	O	O_IMG_MEM_WCLK	Write data for the SRAM of IMG_MEM
O_IMG_MEM_RCLK	O	O_IMG_MEM_WCLK	Read clock for the SRAM of IMG_MEM
O_IMG_MEM_REN	O	O_IMG_MEM_RCLK	Read enable for the SRAM of IMG_MEM (active low)
O_IMG_MEM_RADDR	O	O_IMG_MEM_RCLK	Read address for the SRAM of IMG_MEM
O_IMG_MEM_RDATA	I	O_IMG_MEM_RCLK	Read data from the SRAM of IMG_MEM

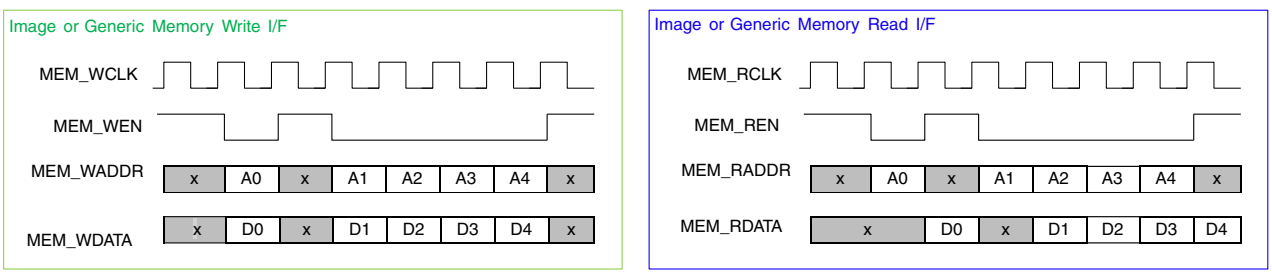


Figure 13-36. SRAM interface timing diagram

13.4.5 Image Data alignment

13.4.5.1 Normal Data alignment

The following figure describes the present of each bit in the data bus of ISP I/F.

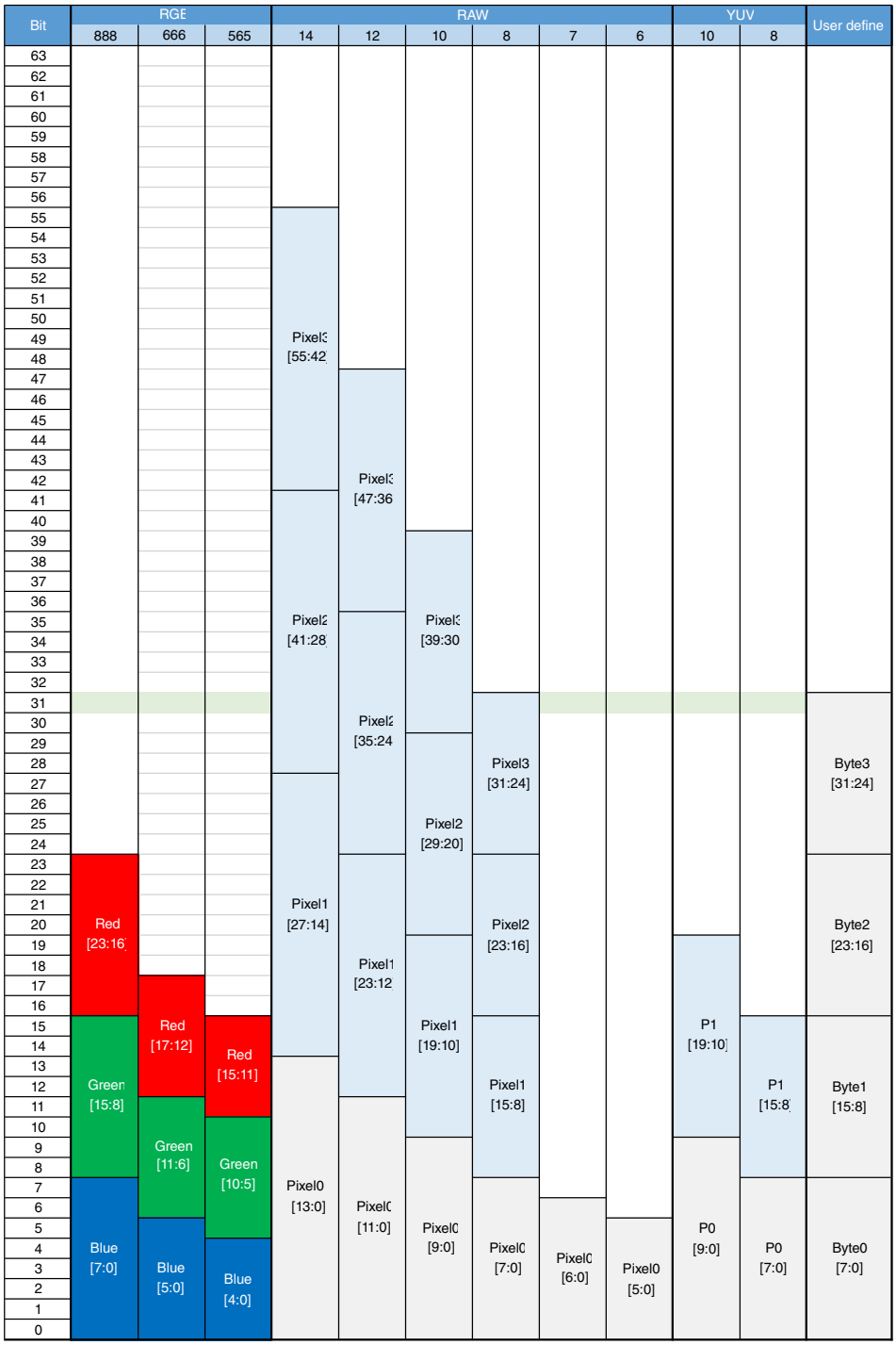


Figure 13-37. Pixel alignment

Pixel IF is adopted the little endian. If the current pixel mode is single, Pixel0 is a valid pixel. Higher pixels are the invalid signals. If the current pixel mode is dual, pixel0 and pixel1 is the valid pixels. In the case of Quad pixel mode, pixel0, pixel1, pixel2 and pixel3 is the valid pixels.

13.4.5.2 YUV data stream

Table 13-14. Output Image Stream of YUV formats

YUV format	Line	Image stream of content
YUV420 legacy	Odd	U1 Y1 Y2 U3 Y3 Y4 U5 Y5 ...
	Even	V1 Y1 Y2 V3 Y3 Y4 V5 Y5 ...
YUV420	Odd	Y1 Y2 Y3 Y4 Y5 Y6 Y7 Y8 ...
	Even	U1 Y1 V1 Y2 U3 Y3 V3 Y4 ...
YUV422	Odd/Even	U1 Y1 V1 Y2 U3 Y3 V3 Y4 ...

13.4.5.3 RGB swapped data alignment

When the “RGB_SWAP” bit of ISP_CONFIG_CH% register is set, the RGB data alignment of data bus is swapped. This is shown in the following figure.

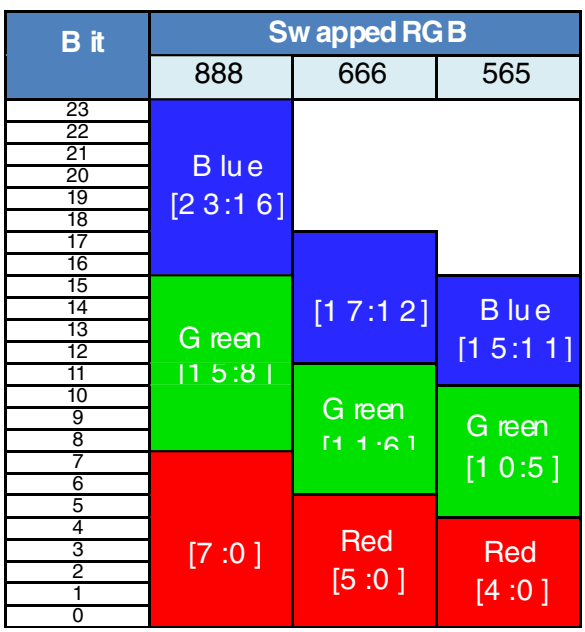


Figure 13-38. RGB swapped data alignment

13.4.5.4 Parallel-memory-storing data alignment

When the “Parallel” bit of ISP_CONFIG_CH% register is set, the outer bus width is 32 bits. User should refer to figures in chapter12 of MIPI CSI-2 standard spec for the alignment of output data of CSIS V3.6.3. The signal diagram of Parallel mode is identical to the User defined packet format (2.7.1). If the byte count of the payload is not a multiple of 4, by performing the 0 padding makes the last data to 32bit.

13.4.6 Configuration

13.4.6.1 Image resolution

MIPI CSI Slave block needs the configuration of Image resolution to measure one line length exactly and detect frame end. Vertical resolution register has 16bits (16’h0001 ~ 16’hFFFF) Horizontal resolution register has 16bits (16’h0004 ~ 16’hFFFC).

13.4.6.2 Image data format

In CSIS V3.6.3, Image data format should be configured for transferring image data to ISP or CAM I/F.

13.4.6.3 YUV format

MIPI CSI2 supports YUV formats: YUV420 8bit, YUV420 10bit, YUV420 Legacy, YUV420 8bit Chroma Shifted Pixel Sampling, YUV420 10bit Chroma Shifted Pixel Sampling, YUV422 8bit, and YUV422 10bit The following table describes Data type of YUV packets.

Table 13-15. Data type of YUV packets

Data type	Description
0x18	YUV420 8-bit
0x19	YUV420 10-bit
0x1A	Legacy YUV420 8-bit
0x1B	Reserved
0x1C	YUV420 8-bit CSPS (Chroma Shifted Pixel Sampling)
0x1D	YUV420 10-bit CSPS (Chroma Shifted Pixel Sampling)
0x1E	YUV422 8-bit
0x1F	YUV422 10-bit

13.4.6.4 RGB format

MIPI CSI2 supports RGB formats: RGB444, RGB555, RGB565, RGB666, and RGB888. The following table describes Data type of RGB packets.

Table 13-16. Data type of RGB packets

Data type	Description
0x20	RGB444 (not supported)
0x21	RGB555 (not supported)
0x22	RGB565
0x23	RGB666
0x24	RGB888
0x25	Reserved
0x26	
0x27	

13.4.6.5 RAW format (Bayer RGB)

MIPI CSI2 supports RAW formats: RAW6, RAW7, RAW8, RAW10, RAW12, and RAW14. The following table describes Data type of RAW packets.

Table 13-17. Data type of RAW packets

Data type	Description
0x28	RAW6
0x29	RAW7
0x2A	RAW8
0x2B	RAW10
0x2C	RAW12
0x2D	RAW14
0x2E	Reserved
0x2F	

13.4.6.6 User defined Byte-based format (like JPEG)

CSIS V3.6.3 supports 8 formats of User defined Byte-based Data packet. The following table describes Data type of User defined packets.

Table 13-18. Data type of User defined packets

Data type	Description
0x30	User defined Byte-based packet 1
0x31	User defined Byte-based packet 2
0x32	User defined Byte-based packet 3
0x33	User defined Byte-based packet 4
0x34	User defined Byte-based packet 5
0x35	User defined Byte-based packet 6
0x36	User defined Byte-based packet 7
0x37	User defined Byte-based packet 8

13.4.6.7 Null and Blanking Data

For both the null and blanking data types CSIS V3.6.3 ignore the content of the packet payload data.

Table 13-19. Null and Blanking Data type

Data type	Description
0x10	Null
0x11	Blanking Data

13.4.7 User defined packet format

The User Defined Data Type values shall be used to transmit arbitrary byte-based data, such as JPEG and 1226 MPEG4 data, over the CSI-2 bus. The packet data size in bits shall be divisible by 8, i.e. whole number of bytes shall be transmitted.

For User Defined data:

1. The frame is transmitted as a sequence of arbitrary sized packets.
2. The packet size may vary from packet to packet.
3. The packet spacing may vary between packets.

13.4.7.1 ISP/CAM I/F for User defined packet

For supporting User defined packet, MIPI CSIS puts out more signals : 2 bits of byte validation. Byte validation signal indicates what is valid byte in data of word size. In the following figure, n is 0, 1, 2, or 3. See below table for more information.

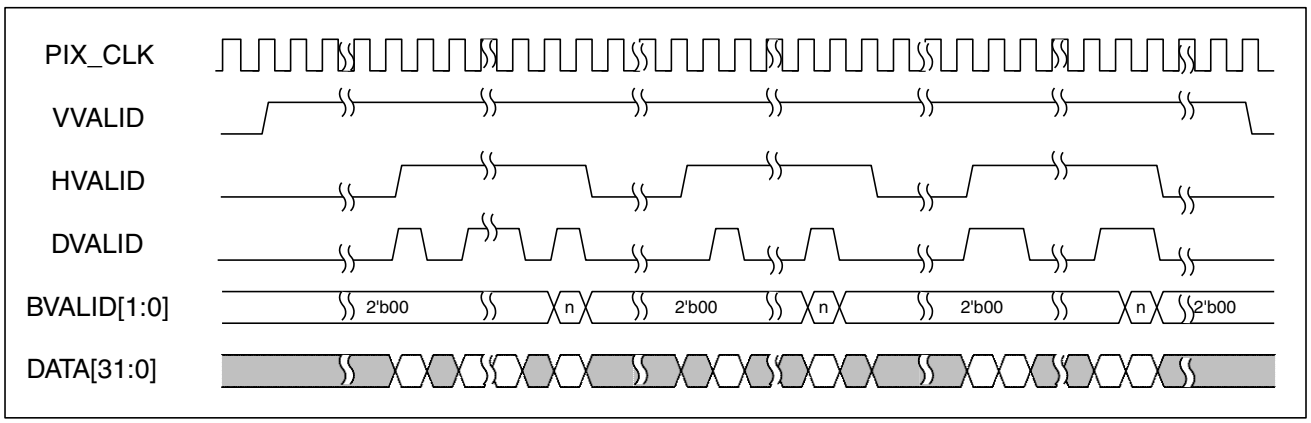


Figure 13-39. Output protocol of CSIS V3.6.3 in User defined data type and Parallel mode

Table 13-20. Memory bridge list

BVALID[1:0]	Description
2'b00	All bytes in data bus are valid
2'b01	Only byte 0 is valid
2'b10	2 bytes from LSB are valid
2'b11	3 bytes from LSB are valid

13.4.8 Synchronization Short Packet Data type

Image Frame begins with a Frame Start and finishes with a Frame End packet. With these Frame synch packet, CSIS V3.6.3 generate Vertical Sync signal (VVALID). The Line synchronization packet (Line Start and Line End) generate Horizontal Sync signal (HVALID). But it can be generated by image data packet so that CSIS V3.6.3 doesn't support the Line Synchronization packets (0x02~0x03) and ignore it if it is received.

Table 13-21. Synchronization short packet data type

Data Type	Description
0x00	Frame Start Code
0x01	Frame End Code
0x02	Line Start Code (Not supported)

Table continues on the next page...

Table 13-21. Synchronization short packet data type (continued)

Data Type	Description
0x03	Line End Code (Not supported)
0x04 to 0x07	Reserved

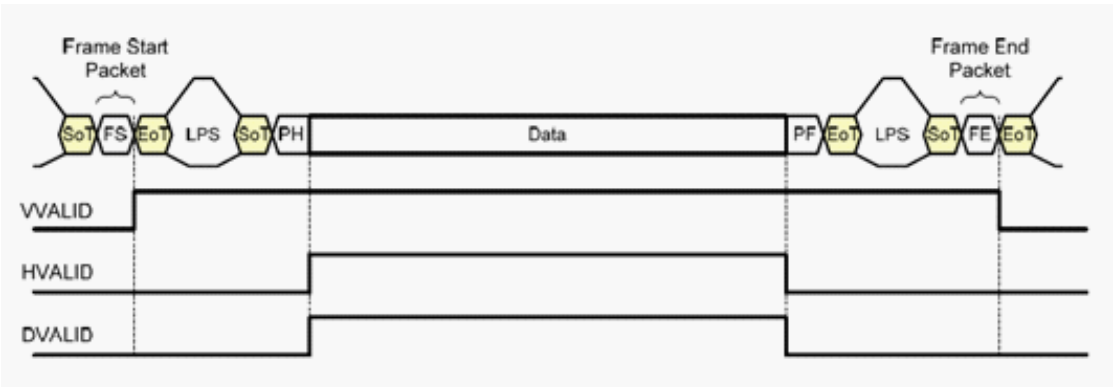


Figure 13-40. Synchronization example

13.4.9 Synchronization Short Packet Data type

Image Frame begins with a Frame Start and finishes with a Frame End packet. With these Frame synch packet, CSIS V3.6.3 generate Vertical Sync signal (VVALID). The Line synchronization packet (Line Start and Line End) generate Horizontal Sync signal (HVALID). But it can be generated by image data packet so that CSIS V3.6.3 doesn't support the Line Synchronization packets (0x02~0x03) and ignore it if it is received.

Table 13-22. Synchronization short packet data type

Data Type	Description
0x00	Frame Start Code
0x01	Frame End Code
0x02	Line Start Code (Not supported)
0x03	Line End Code (Not supported)
0x04 to 0x07	Reserved

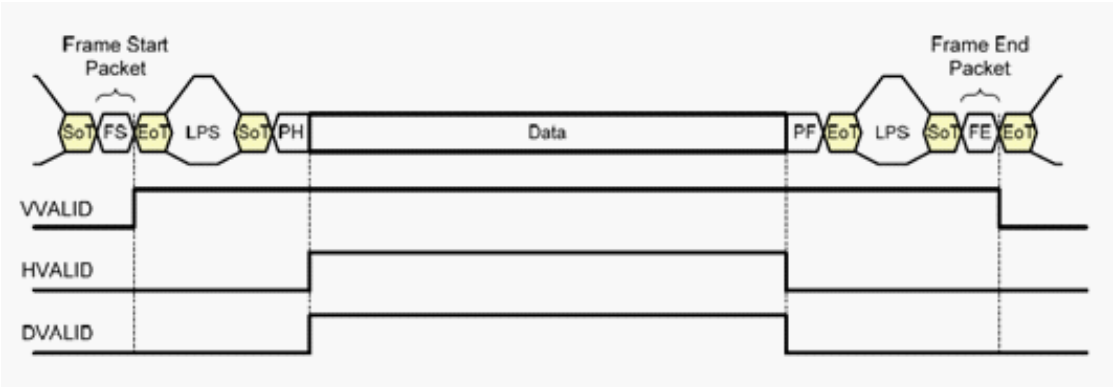


Figure 13-41. Synchronization example

13.4.10 Shadow Update

The configurations related to ISP registers (which are dataformat, vci, resolution and so on.) should not be varied during the run-time. So, they are updated by programming 'update_shadow' (in CSIS_CMN_CTRL register) in idle state. The scheme is described in the following figure.

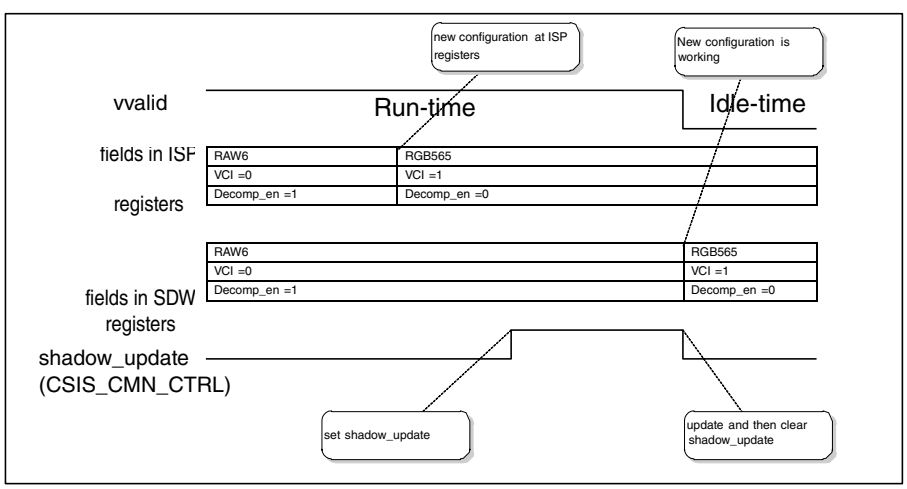


Figure 13-42. Operation of shadow_update

If the frame interleaving scheme (interleave_mode in CSIS_CMN_CTRL = 2 or 3) is used then the instance of entering idle_state at each ISP channels may be different. In this case, updating shadow registers can be delayed until that all isp channels are in idle state by setting update_shadow_ctrl.

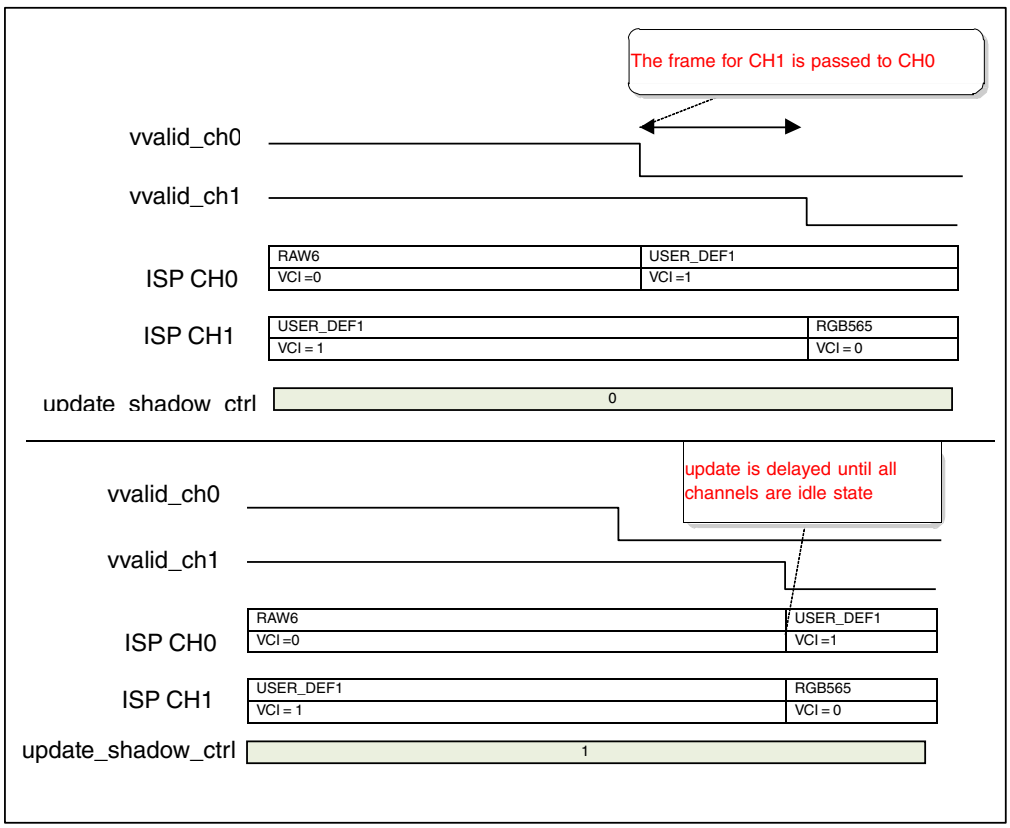


Figure 13-43. Operation of shadow_update_ctrl

13.4.11 Clock Specification

MIPI CSI have 3 clock sources: RX_BYTE_CLK_HS%, I_PCLK, and I_WRAP_CLK.

Table 13-23. Clock sources

Clock source	Description
I_PCLK	PCLK is system clock generated by general processor PLL.
i_RxByteClkHS%	This signal comes from data lane 0,1,2,3 of D-PHY.
I_ACLK	I_ACLK is external clock for pixel clock what is for using ISP or CAM I/F

13.4.12 Clock Domain

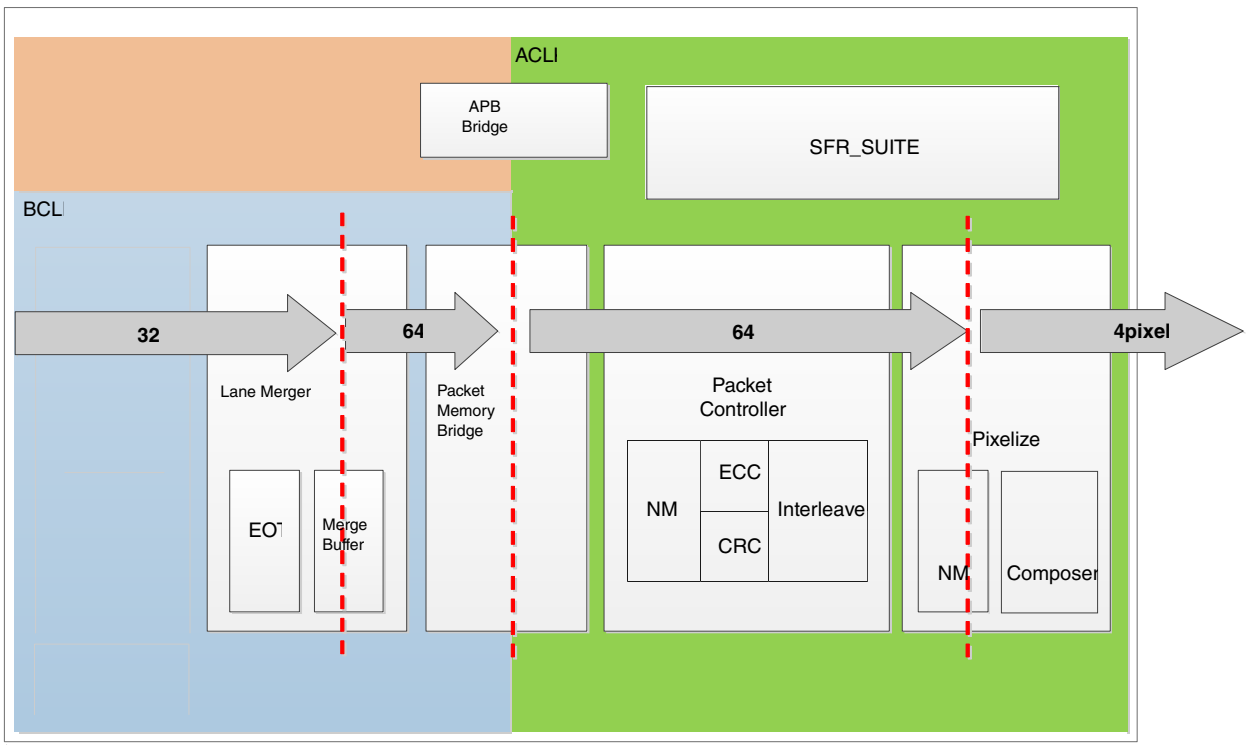


Figure 13-44. Block diagram of clock domain

The number of clock domain of CSIS V3.6.3 is 3:

- BYTE clock (RxByteCLKSH0)
- Pixel clock (I_ACLK)
- APB clock (I_PCLK)

The relationship between input and output bandwidth is that the output bandwidth should be faster than input bandwidth. There is an equation of previous relationship: **(FREQ. OF RX_BYTE_CLK_HS) X (NUMBER OF DATA LANE) X 8BITS ≤ (FREQ. OF PIXEL CLOCK) X (BITWIDTH OF IMAGE FORMAT) X (NUMBER OF PIXEL PER CLOCK)**

When parallel mod is on, **(BITWIDTH OF IMAGE FORMAT) X (NUMBER OF PIXEL PER CLOCK) = 32**

13.4.13 Interface Signals

13.4.13.1 Scalable

Our IP can change by setting parameters. Interface signals are generated by the parameter values. Parameter CH_NUM is the number of PHYsical channels of cam interface and affects cam if signals.

13.4.13.2 Reset and Clock signal

Table 13-24. Variation of clock and reset

Clock and reset	Frequency	Clock generator	Usage
I_PRESETn	-	-	Global Asynchronous reset
I_PCLK	-	System controller	Main control clock (system dependent)
i_RxByteClkHs0	Up to 250 Mhz (2G), 187.5MHz (1.5G), and 125MHz (1G)	Data lane 0 of D-PHY	MIPI CSI processes data with ByteCLK
I_ACLK	Up to 550Mhz	External clock source	Output pixel clock

13.4.13.3 PPI (PHY-Protocol Interface) Signals

BandCtrl and HSzeroCtl signals of following table are additional signals for D-PHY timing configuration. The other PPI signals in the table are compatible with MIPI D-PHY spec. v0.84r00 (See Annex A PPI Description informative).

Table 13-25. Protocol-to-PHY interface signal list

Signal	I/O	Description
Clock lane		
STOP_STATE_CLK	I	Stop state indicator (active high)
ULPS_CLK	I	ULPS state indicator (active low)
ENABLE_CLK	O	Clock lane enabler
Data lane (LANE -1 = %, 0<%<3, if LANE-1 is under 3, Lane3 is unused and should tie 0 to lane3's input ports)		
i_RxByteCLKHS%	I	Byte Clock
i_RxValidHS%	I	Received Data is valid.
i_RxDataHS%	I	Data bus for HS transfer
i_Stopstate%	I	Stop state indicator (active high)
i_UlpsActiveNot%	I	ULPS state indicator (active low)
i_ErrSotHS%	I	Error flag of Start of transmission in HS mode.

Table continues on the next page...

Table 13-25. Protocol-to-PHY interface signal list (continued)

Signal	I/O	Description
o_Enable%	O	Data lane enabler
D-PHY test		
I_DPHY_TESTMODE	I	D-PHY test mode
I_DTLPCLK	I	D-PHY Test clock
TXESCCLK%	O	Escape mode clock for D-PHY test
D-PHY config		
O_HS_SETTLE[7:0]	O	Configuration for tuning HS settle time
O_S_CLKSETTLECTRL[1:0]	O	Configurationfor D-PHY control
O_B_DPHYCTRL[63:0]	O	Configuration for D-PHY ac characteristics
O_S_DPHYCTRL[63:0]	O	Configuration for D-PHY ac characteristics
O_S_DPDN_SWAP_CLK	O	Swapping Dp and Dn channel of clock lane in D-PHY.
O_S_DPDN_SWAP_DAT	O	Swapping Dp and Dn channel of data lanes in D-PHY.

13.4.13.4 AMBA Signal

Table 13-26. APB slave Interface signal list

Signal	I/O	Description
APB Slave		
I_PCLK	I	Main system clock
I_PRESETn	I	Main system reset
I_PADDR[31:0]	I	System address
I_PSEL	I	Module select strobe
I_PWRITE	I	Write strobe
I_PENABLE	I	enable strobe
I_PWDATA[31:0]	I	Write data to slave module
O_PREADY	O	Slave module is ready. (but tied high : always ready)
O_PSLVERR	O	Slave module with error
O_PRDATA[31:0]	O	Read data from slave module

13.4.13.5 Image Signal

Image output signals are duplicated for output channel 1 to 3.

Table 13-27. Image Output Interface signal list

Signal	I/O	Description
Pixel I/F (CH_NUM -1 = %, 0<%<3, if CH_NUM-1 is under 3, Channel3 is unused and that is not generated)		
O_PIXCLK[%]	O	Pixel clock for ISP or CAM I/F
		This is pixel clock for transferring image date to ISP or CAM I/F.
		The relationship of bandwidth between input and output is that output bandwidth should be faster than input bandwidth.
		The frequency of this clock is determined as following equation.
		(# of data lane) X (frequency of i_RxByteCLKHS0) ≤ (bitwidths of image format) X (frequency of PIXCLK).
		O_PIX_CLK[0]~[3] are worked when the corresponding channels are used
O_VVALID[%]	O	Vertical sync (level)
		O_VVALID[0]~[3] are worked when the corresponding channels are used
O_HVALID[%]	O	Horizontal sync (level)
		O_HVALID[0]~[3] are worked when the corresponding channels are used
O_DVALID[%]	O	Data valid
		O_DVALID[0]~[3] are worked when the corresponding channels are used
O_BVALID[CH_NUM*2-	O	Byte valid
1:0]		O_BVALID[1:0]~[7:6] are worked when the corresponding channels are used
O_DATA[CH_NUM*64- 1:0]	O	Pixel data
		O_DATA[0+:64]~[3*64+:64] are worked when the corresponding channels are used

13.4.13.6 Sideband Signal

Table 13-28. Sideband signal list

Signal	I/O	Description
Interrupt		
O_INT_CSIS	O	Interrupt request signal
I_INTGEN_SEL	I	This port shall be integrated to 1.

Table 13-28. Sideband signal list

Signal	I/O	Description
		1 : level interrupt generation

13.4.13.7 Memory I/F

Table 13-29. Memory I/F

Signal	I/O	Description
Memory for Image data		
O_IMG_MEM_WCLK	O	Write clock (ByteClk)
O_IMG_MEM_WEN	O	Image data Write enable strobe
O_IMG_MEM_WADDR[8:0]	O	Image data Write Address
O_IMG_MEM_WDATA[63:0]	O	Image data Write Data bus
O_IMG_MEM_RCLK	O	Read clock (PIXCLK)
O_IMG_MEM_RADDR[8:0]	O	Image data Read Address
O_IMG_MEM_REN	O	Image data Read enable strobe
I_IMG_MEM_RDATA[63:0]	I	Image data Read Data bus

13.4.14 Application Scenario

13.4.14.1 Programming Model

Table 13-30. Behavior programming each case

No.	Pseudo Code	Description
1	APB_WRITE(CSIS_INTR_MSK, -)	set Interrupt Mask
2	APB_WRITE(DPHY_BCTRL_L, -)	set DPHYBCTRL LSB
3	APB_WRITE(DPHY_BCTRL_H, -)	set DPHYBCTRL MSB
4	APB_WRITE(DPHY_CMN_CTRL, 0x11)	set DPHY HSSETTLE
5	APB_WRITE(ISP_CONFIG_CH0, -)	set ISP_CH0 configuration
6	APB_WRITE(ISP_RESOL_CH0, -)	set ISP_CH0 resolution
7	APB_WRITE(ISP_SYNC_CH0, -)	set ISP_CH0 sync
8	APB_WRITE(ISP_CONFIG_CH1, -)	set ISP_CH1 configuration
9	APB_WRITE(ISP_RESOL_CH1, -)	set ISP_CH1 resolution
10	APB_WRITE(ISP_SYNC_CH1, -)	set ISP_CH1 sync
11	APB_WRITE(CSIS_CLK_CTRL, -)	set CSIS clock control
12	APB_WRITE(CSIS_CMN_CTRL, -)	set CSIS common control(enable CSI)

13.4.15 MIPI CSI Memory Map/Register Definition

MIPI_CSI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E3_0004	CSIS Common Control Register (MIPI_CSI_CSIS_COMMON_CTRL)	32	R/W	0000_4000h	13.4.15.1/ 4024
32E3_0008	CSIS Clock Control Register (MIPI_CSI_CSIS_CLOCK_CTRL)	32	R/W	0000_00F0h	13.4.15.2/ 4025
32E3_0010	Interrupt mask register 0 (MIPI_CSI_INTERRUPT_MASK_0)	32	R/W	0000_0000h	13.4.15.3/ 4026
32E3_0014	Interrupt source register 0 (MIPI_CSI_INTERRUPT_SOURCE_0)	32	R/W	0000_0000h	13.4.15.4/ 4028
32E3_0018	Interrupt mask register 1 (MIPI_CSI_INTERRUPT_MASK_1)	32	R/W	0000_0000h	13.4.15.5/ 4030
32E3_001C	Interrupt source register 1 (MIPI_CSI_INTERRUPT_SOURCE_1)	32	R/W	0000_0000h	13.4.15.6/ 4030
32E3_0020	D-PHY status register (MIPI_CSI_DPHY_STATUS)	32	R/W	0000_00F1h	13.4.15.7/ 4031
32E3_0024	D-PHY common control register (MIPI_CSI_DPHY_COMMON_CTRL)	32	R/W	0000_0000h	13.4.15.8/ 4033
32E3_0030	D-PHY Master and Slave Control register Low (MIPI_CSI_DPHY_MASTER_SLAVE_CTRL_LOW)	32	R/W	0000_0000h	13.4.15.9/ 4034
32E3_0034	D-PHY Master and Slave Control register HIGH (MIPI_CSI_DPHY_MASTER_SLAVE_CTRL_HIGH)	32	R/W	0000_0000h	13.4.15.10/ 4034
32E3_0038	D-PHY Slave Control register Low (MIPI_CSI_DPHY_SLAVE_CTRL_LOW)	32	R/W	0000_0000h	13.4.15.11/ 4035
32E3_003C	D-PHY Slave Control register HIGH (MIPI_CSI_DPHY_SLAVE_CTRL_HIGH)	32	R/W	0000_0000h	13.4.15.12/ 4035
32E3_0040	ISP Configuration Register (MIPI_CSI_ISP_CONFIG_CH0)	32	R/W	0000_08FCh	13.4.15.13/ 4036
32E3_0044	ISP Resolution Register (MIPI_CSI_ISP_RESOLUTION_CH0)	32	R/W	8000_8000h	13.4.15.14/ 4037
32E3_0048	ISP SYNC Register (MIPI_CSI_ISP_SYNC_CH0)	32	R/W	0000_0000h	13.4.15.15/ 4038
32E3_0050	ISP Configuration Register (MIPI_CSI_ISP_CONFIG_CH1)	32	R/W	0000_08FCh	13.4.15.13/ 4036
32E3_0054	ISP Resolution Register (MIPI_CSI_ISP_RESOLUTION_CH1)	32	R/W	8000_8000h	13.4.15.14/ 4037
32E3_0058	ISP SYNC Register (MIPI_CSI_ISP_SYNC_CH1)	32	R/W	0000_0000h	13.4.15.15/ 4038
32E3_0060	ISP Configuration Register (MIPI_CSI_ISP_CONFIG_CH2)	32	R/W	0000_08FCh	13.4.15.13/ 4036

Table continues on the next page...

MIPI_CSI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E3_0064	ISP Resolution Register (MIPI_CSI_ISP_RESOLUTION_CH2)	32	R/W	8000_8000h	13.4.15.14/ 4037
32E3_0068	ISP SYNC Register (MIPI_CSI_ISP_SYNC_CH2)	32	R/W	0000_0000h	13.4.15.15/ 4038
32E3_0070	ISP Configuration Register (MIPI_CSI_ISP_CONFIG_CH3)	32	R/W	0000_08FCh	13.4.15.13/ 4036
32E3_0074	ISP Resolution Register (MIPI_CSI_ISP_RESOLUTION_CH3)	32	R/W	8000_8000h	13.4.15.14/ 4037
32E3_0078	ISP SYNC Register (MIPI_CSI_ISP_SYNC_CH3)	32	R/W	0000_0000h	13.4.15.15/ 4038
32E3_0080	Shadow Configuration Register (MIPI_CSI_SHADOW_CONFIG_CH0)	32	R/W	0000_08FCh	13.4.15.16/ 4039
32E3_0084	Shadow Resolution Register (MIPI_CSI_SHADOW_RESOLUTION_CH0)	32	R/W	8000_8000h	13.4.15.17/ 4040
32E3_0088	Shadow SYNC Register (MIPI_CSI_SHADOW_SYNC_CH0)	32	R/W	0000_0000h	13.4.15.18/ 4040
32E3_0090	Shadow Configuration Register (MIPI_CSI_SHADOW_CONFIG_CH1)	32	R/W	0000_08FCh	13.4.15.16/ 4039
32E3_0094	Shadow Resolution Register (MIPI_CSI_SHADOW_RESOLUTION_CH1)	32	R/W	8000_8000h	13.4.15.17/ 4040
32E3_0098	Shadow SYNC Register (MIPI_CSI_SHADOW_SYNC_CH1)	32	R/W	0000_0000h	13.4.15.18/ 4040
32E3_00A0	Shadow Configuration Register (MIPI_CSI_SHADOW_CONFIG_CH2)	32	R/W	0000_08FCh	13.4.15.16/ 4039
32E3_00A4	Shadow Resolution Register (MIPI_CSI_SHADOW_RESOLUTION_CH2)	32	R/W	8000_8000h	13.4.15.17/ 4040
32E3_00A8	Shadow SYNC Register (MIPI_CSI_SHADOW_SYNC_CH2)	32	R/W	0000_0000h	13.4.15.18/ 4040
32E3_00B0	Shadow Configuration Register (MIPI_CSI_SHADOW_CONFIG_CH3)	32	R/W	0000_08FCh	13.4.15.16/ 4039
32E3_00B4	Shadow Resolution Register (MIPI_CSI_SHADOW_RESOLUTION_CH3)	32	R/W	8000_8000h	13.4.15.17/ 4040
32E3_00B8	Shadow SYNC Register (MIPI_CSI_SHADOW_SYNC_CH3)	32	R/W	0000_0000h	13.4.15.18/ 4040
32E3_0100	Frame Counter (MIPI_CSI_FRAME_COUNTER_CH0)	32	R/W	0000_0000h	13.4.15.19/ 4041
32E3_0104	Frame Counter (MIPI_CSI_FRAME_COUNTER_CH1)	32	R/W	0000_0000h	13.4.15.19/ 4041
32E3_0108	Frame Counter (MIPI_CSI_FRAME_COUNTER_CH2)	32	R/W	0000_0000h	13.4.15.19/ 4041
32E3_010C	Frame Counter (MIPI_CSI_FRAME_COUNTER_CH3)	32	R/W	0000_0000h	13.4.15.19/ 4041
32E3_0110	Line Interrupt Ratio (MIPI_CSI_LINE_INTERRUPT_RATIO_CH0)	32	R/W	0000_0000h	13.4.15.20/ 4041

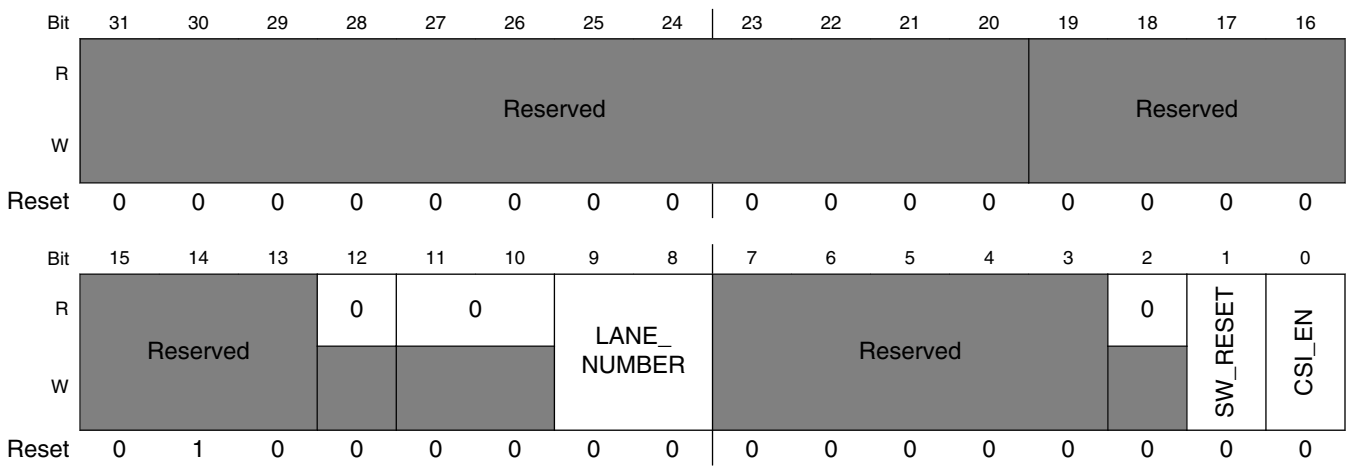
Table continues on the next page...

MIPI_CSI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
32E3_0114	Line Interrupt Ratio (MIPI_CSI_LINE_INTERRUPT_RATIO_CH1)	32	R/W	0000_0000h	13.4.15.20/ 4041
32E3_0118	Line Interrupt Ratio (MIPI_CSI_LINE_INTERRUPT_RATIO_CH2)	32	R/W	0000_0000h	13.4.15.20/ 4041
32E3_011C	Line Interrupt Ratio (MIPI_CSI_LINE_INTERRUPT_RATIO_CH3)	32	R/W	0000_0000h	13.4.15.20/ 4041

13.4.15.1 CSIS Common Control Register (MIPI_CSI_CSIS_COMMON_CTRL)

Address: 32E3_0000h base + 4h offset = 32E3_0004h



MIPI_CSI_CSIS_COMMON_CTRL field descriptions

Field	Description
31–20 -	This field is reserved.
19–16 UPDATE_ SHADOW	Strobe of updating shadow registers This field is reserved. After configuration, User has to set this bit for updating shadow registers. This bit is cleared automatically after updating shadow registers. [19] CH3 [18] CH2 [17] CH1 [16] CH0
15–13 -	This field is reserved.
12 Reserved	This read-only field is reserved and always has the value 0.
11–10 Reserved	This read-only field is reserved and always has the value 0.
9–8 LANE_NUMBER	Number of data lane

Table continues on the next page...

MIPI_CSI_CSIS_COMMON_CTRL field descriptions (continued)

Field	Description
	00 1 data lane 01 2 data lane 10 3 data lane 11 4 data lane
7–3 -	This field is reserved. -
2 Reserved	This read-only field is reserved and always has the value 0.
1 SW_RESET	Software reset All writable registers in CSI2 go back to initial state. This bit is de-asserted automatically after the software reset is done. **NOTE:** Notice : Almost MIPI CSI2 block uses “ByteClk” from D- PHY. This “ByteClk” is not continuous clock. User has to assert software reset when Camera module is turned off. 0 Ready 1 Reset
0 CSI_EN	MIPI CSI2 system enable 0 Disable 1 Enable

13.4.15.2 CSIS Clock Control Register (MIPI_CSI_CSIS_CLOCK_CTRL)

Address: 32E3_0000h base + 8h offset = 32E3_0008h

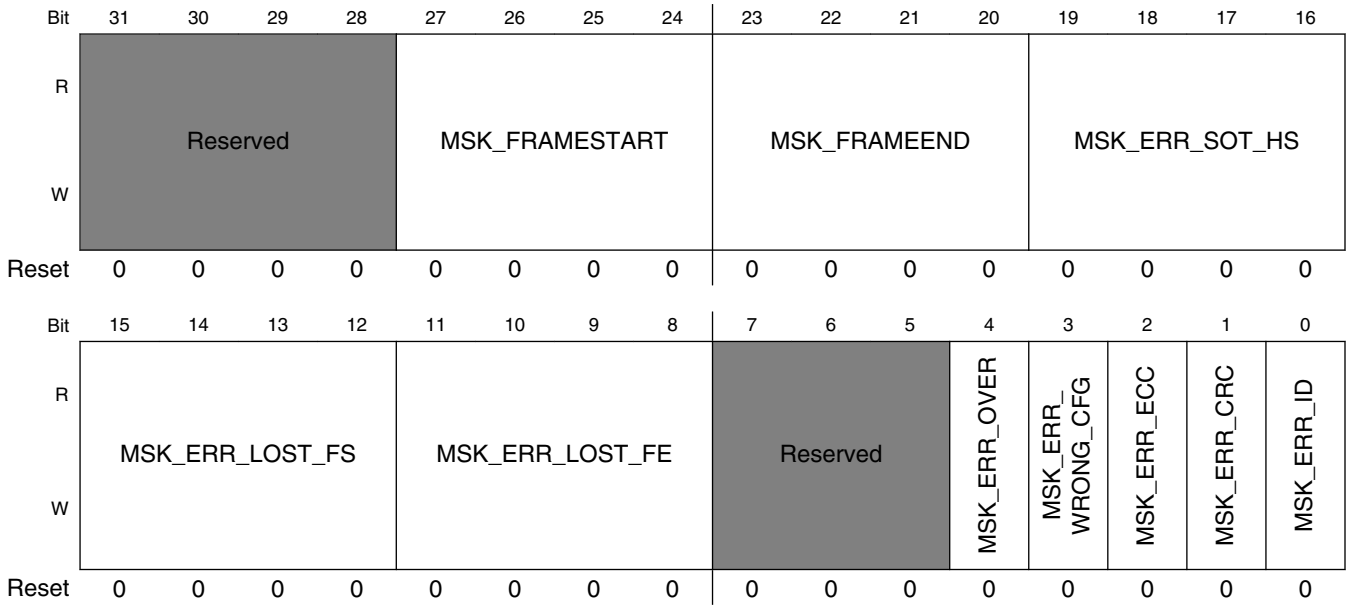
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																Reserved								CLKGATE_EN				Reserved			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

MIPI_CSI_CSIS_CLOCK_CTRL field descriptions

Field	Description
31–16 CLKGATE_TRAIL	0 ~ 15 (1~16 Trailing clocks) This field is reserved. [31:28] CH3 [27:24] CH2 [23:20] CH1 [19:16] CH0 **NOTE:** Trailing clocks used for poping the F/F of ISP or CAM
15–8 -	This field is reserved.
7–4 CLKGATE_EN	0 Pixel clock is always alive 1 Pixel clock is alive during the interval of frame [7] CH3 [6] CH2 [5] CH1 [4] CH0 (Refer 2.9)
-	This field is reserved.

13.4.15.3 Interrupt mask register 0 (MIPI_CSI_INTERRUPT_MASK_0)

Address: 32E3_0000h base + 10h offset = 32E3_0010h



MIPI_CSI_INTERRUPT_MASK_0 field descriptions

Field	Description
31–28 -	This field is reserved.
27–24 MSK_ FRAMESTART	FS packet is received, [CH3,CH2,CH1,CH0] 0 Disable (masked) 1 Enable (unmasked)
23–20 MSK_ FRAMEEND	FE packet is received, [CH3,CH2,CH1,CH0] 0 Disable (masked) 1 Enable (unmasked)
19–16 MSK_ERR_ SOT_HS	Start of transmission error [Lane3, Lane2, Lane1, Lane0] 0 Disable (masked) 1 Enable (unmasked)
15–12 MSK_ERR_ LOST_FS	Lost of Frame Start packet, [CH3,CH2,CH1,CH0] 0 Disable (masked) 1 Enable (unmasked)
11–8 MSK_ERR_ LOST_FE	Lost of Frame End packet, [CH3,CH2,CH1,CH0] 0 Disable (masked) 1 Enable (unmasked)
7–5 -	This field is reserved.

Table continues on the next page...

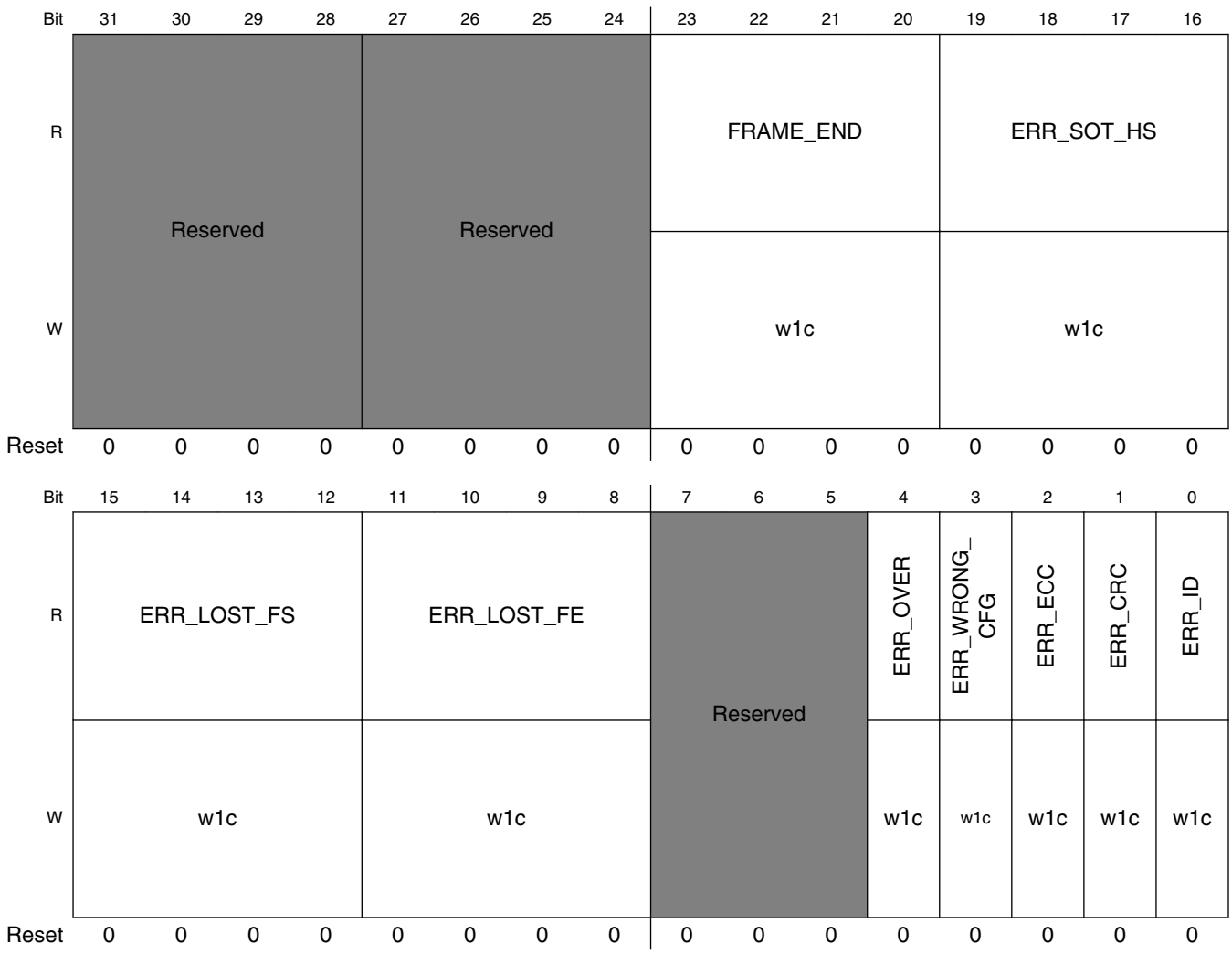
MIPI_CSI_INTERRUPT_MASK_0 field descriptions (continued)

Field	Description
4 MSK_ERR_ OVER	Image FIFO overflow interrupt 0 Disable (masked) 1 Enable (unmasked)
3 MSK_ERR_ WRONG_CFG	Wrong configuration 0 Disable (masked) 1 Enable (unmasked)
2 MSK_ERR_ECC	ECC error 0 Disable (masked) 1 Enable (unmasked)
1 MSK_ERR_CRC	CRC error 0 Disable (masked) 1 Enable (unmasked)
0 MSK_ERR_ID	Unknown ID error 0 Disable (masked) 1 Enable (unmasked)

13.4.15.4 Interrupt source register 0 (MIPI_CSI_INTERRUPT_SOURCE_0)

Write ‘1’ clears status bit and write ‘0’ has no effect. This register is valid only during csis is enabled. All of register fields should be cleared after csis is enabled.

Address: 32E3_0000h base + 14h offset = 32E3_0014h



MIPI_CSI_INTERRUPT_SOURCE_0 field descriptions

Field	Description
31–28 -	This field is reserved.
27–24 FRAME_START	This field is reserved. FS packet is received, [CH3,CH2,CH1,CH0]
23–20 FRAME_END	FE packet is received, [CH3,CH2,CH1,CH0]

Table continues on the next page...

MIPI_CSI_INTERRUPT_SOURCE_0 field descriptions (continued)

Field	Description
19–16 ERR_SOT_HS	Start of transmission error, [Lane3,Lane2,Lane1,Lane0]
15–12 ERR_LOST_FS	Indication of lost of Frame Start packet, [CH3,CH2,CH1,CH0] This field is set to '1' when the image data is received without frame start packet.
11–8 ERR_LOST_FE	Indication of lost of Frame End packet, [CH3,CH2,CH1,CH0]
7–5 -	This field is reserved.
4 ERR_OVER	Overflow is caused in image FIFO. Outer bandwidth has to be faster than imputer bandwidth. But image FIFO can be overflow because of user's fault. There are 2 ways for preventing overflow. • Tune output pixel clock faster than current. • Tune input byte clock slowr than current. First case : WCLK_Src in CSIS_CTRL register shoud be set 1, and then assign faster clock Second case : user can set register in camera module through I2C channel. When this interrupt is generated: • Turn camera off • Assert software reset, if you didn't assert software reset, MIPI CSIS could not receive any more data. • Tune the clock frequency and re-configure all related registers. MIPI CSIS module is ready for operating.
3 ERR_WRONG_CFG	Wrong configuration **NOTE:** The received packet is not allocated to the PHYsical channel at any PHYsical channels (due to wrong data type or virtual channel id).
2 ERR_ECC	ECC error
1 ERR_CRC	CRC error
0 ERR_ID	Unknown ID error

13.4.15.5 Interrupt mask register 1 (MIPI_CSI_INTERRUPT_MASK_1)

Address: 32E3_0000h base + 18h offset = 32E3_0018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												MSK_LINE_END			
W													w1c			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_CSI_INTERRUPT_MASK_1 field descriptions

Field	Description
31–4 -	This field is reserved.
MSK_LINE_END	End of specific line [CH3, CH2, CH1, CH0]

13.4.15.6 Interrupt source register 1 (MIPI_CSI_INTERRUPT_SOURCE_1)

Write ‘1’ clears status bit and write ‘0’ has no effect. This register is valid only during csis is enabled. All of register fields should be cleared after csis is enabled.

Address: 32E3_0000h base + 1Ch offset = 32E3_001Ch

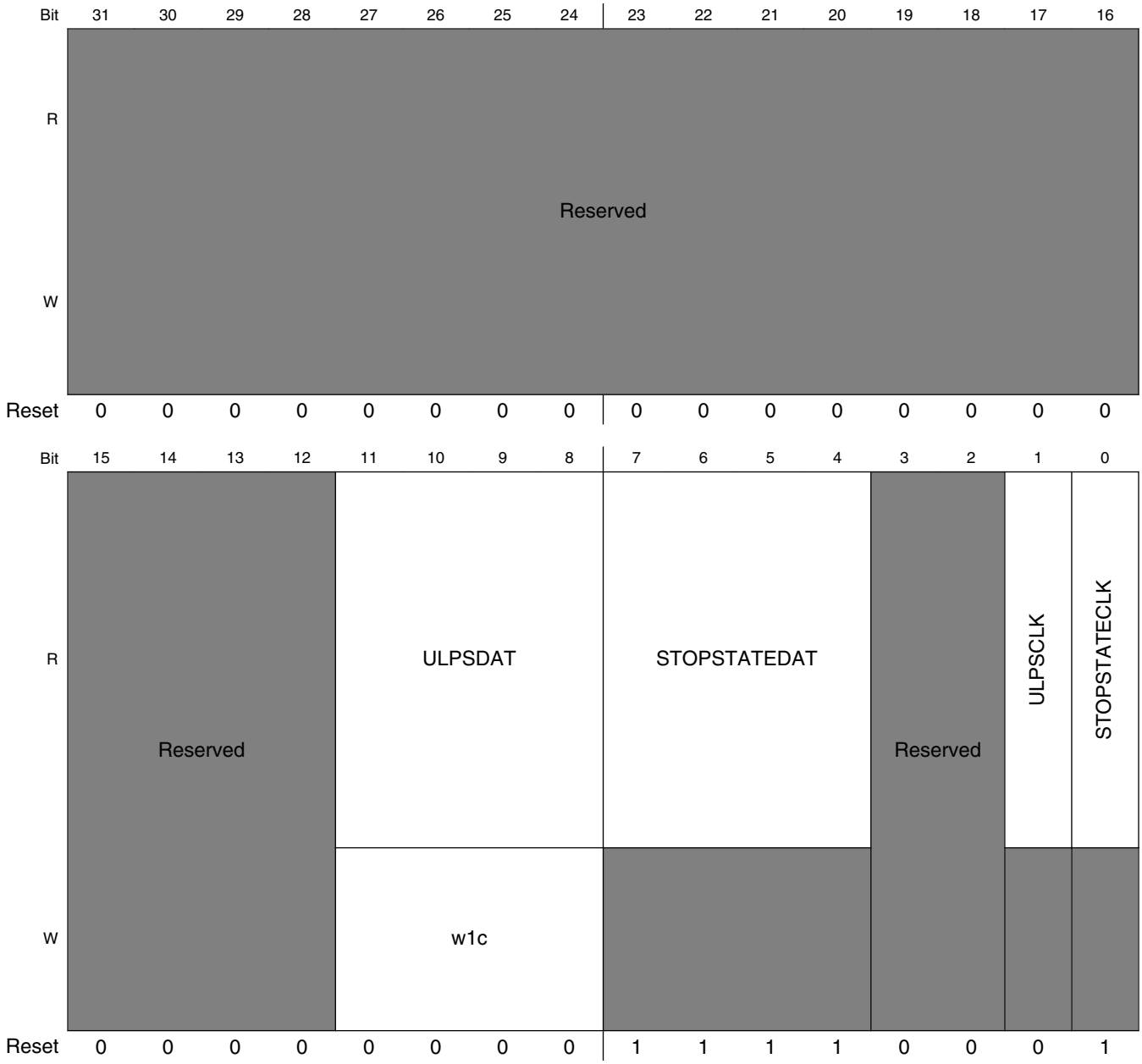
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																												LINE_END			
W																													w1c			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_CSI_INTERRUPT_SOURCE_1 field descriptions

Field	Description
31–4 -	This field is reserved.
LINE_END	End of specific line [CH3, CH2, CH1, CH0]

13.4.15.7 D-PHY status register (MIPI_CSI_DPHY_STATUS)

Address: 32E3_0000h base + 20h offset = 32E3_0020h



MIPI_CSI_DPHY_STATUS field descriptions

Field	Description
31–12 -	This field is reserved.
11–8 ULPSDAT	Data lane [3:0] is in ULPS [7] : data lane 3

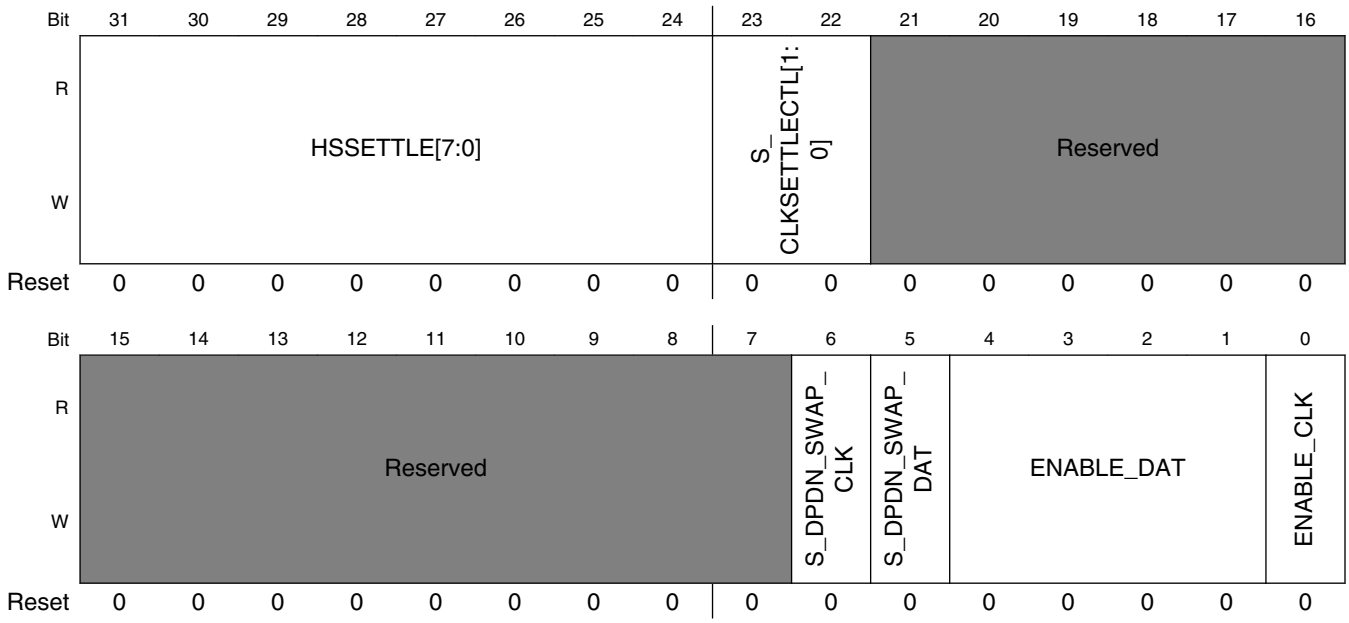
Table continues on the next page...

MIPI_CSI_DPHY_STATUS field descriptions (continued)

Field	Description
	[6] : data lane 2 [5] : data lane 1 [4] : data lane 0 0 Not ULPS 1 ULPS
7–4 STOPSTATEDAT	Data lane [3:0] is in Stop State [7] : data lane 3 [6] : data lane 2 [5] : data lane 1 [4] : data lane 0 0 Not Stop state 1 Stop state
3–2 -	This field is reserved.
1 ULPSCLK	0 Not ULPS 1 ULPS
0 STOPSTATECLK	Clock lane is in Stop State 0 Not Stop state 1 Stop state

13.4.15.8 D-PHY common control register (MIPI_CSI_DPHY_COMMON_CTRL)

Address: 32E3_0000h base + 24h offset = 32E3_0024h



MIPI_CSI_DPHY_COMMON_CTRL field descriptions

Field	Description
31–24 HSSETTLE[7:0]	HS-RX settle time control register. Refer to user guide of D-PHY
23–22 S_ CLKSETTLECTL[1:0]	D-PHY control register for standard spec v0.9 of MIPI CSI2 Refer to user guide of D-PHY
21–7 -	This field is reserved.
6 S_DPDN_SWAP_CLK	Swapping Dp and Dn channel of clock lane. 0 Default 1 Swapped
5 S_DPDN_SWAP_DAT	Swapping Dp and Dn channel of data lanes. 0 Default 1 Swapped
4–1 ENABLE_DAT	D-PHY enable [3] : data lane 3 [2] : data lane 2 [1] : data lane 1 [0] : data lane 0

Table continues on the next page...

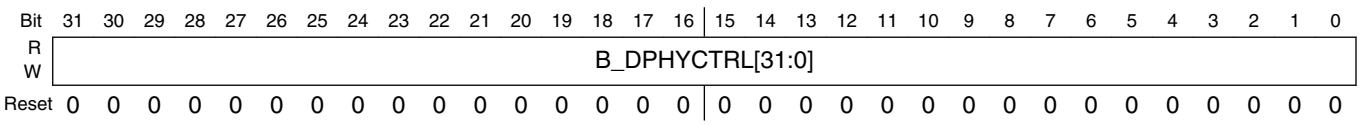
MIPI_CSI_DPHY_COMMON_CTRL field descriptions (continued)

Field	Description
	0 Disable 1 Enable
0 ENABLE_CLK	D-PHY clock lane enable 0 Disable 1 Enable

13.4.15.9 D-PHY Master and Slave Control register Low (MIPI_CSI_DPHY_MASTER_SLAVE_CTRL_LOW)

Depending on D-PHY, the bit-width of D-PHY control register is different. Since DPHY_1.5G_LN20LPE, it requires more than 32bits and DPHY_BCTRL_H (0x34) is usable.

Address: 32E3_0000h base + 30h offset = 32E3_0030h

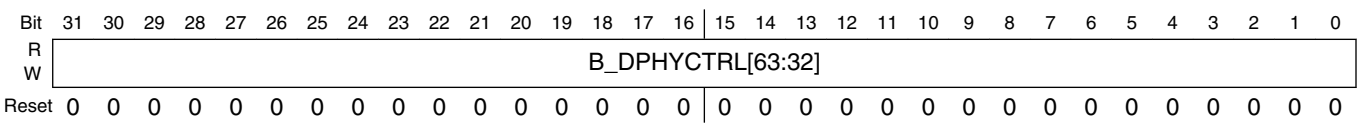


MIPI_CSI_DPHY_MASTER_SLAVE_CTRL_LOW field descriptions

Field	Description
B_ DPHYCTRL[31:0]	D-PHY Master and Slave control register Low part Refer to 'User Guide of D-PHY' for more information.

13.4.15.10 D-PHY Master and Slave Control register HIGH (MIPI_CSI_DPHY_MASTER_SLAVE_CTRL_HIGH)

Address: 32E3_0000h base + 34h offset = 32E3_0034h



MIPI_CSI_DPHY_MASTER_SLAVE_CTRL_HIGH field descriptions

Field	Description
B_DPHYCTRL[63:32]	D-PHY Master and Slave control register High part Refer to 'User Guide of D-PHY' for more information.

13.4.15.11 D-PHY Slave Control register Low (MIPI_CSI_DPHY_SLAVE_CTRL_LOW)

Depending on D-PHY, the bit-width of D-PHY control register is different. Since DPHY_1.5G_LN20LPE, it requires more than 32bits and DPHY_SCTRL0_H(0x3C) is usable

Address: 32E3_0000h base + 38h offset = 32E3_0038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	S_DPHYCTRL[31:0]																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_CSI_DPHY_SLAVE_CTRL_LOW field descriptions

Field	Description
S_ DPHYCTRL[31:0]	D-PHY Slave control register Low part Refer to ‘User Guide of D-PHY’ for more information.

13.4.15.12 D-PHY Slave Control register HIGH (MIPI_CSI_DPHY_SLAVE_CTRL_HIGH)

Address: 32E3_0000h base + 3Ch offset = 32E3_003Ch

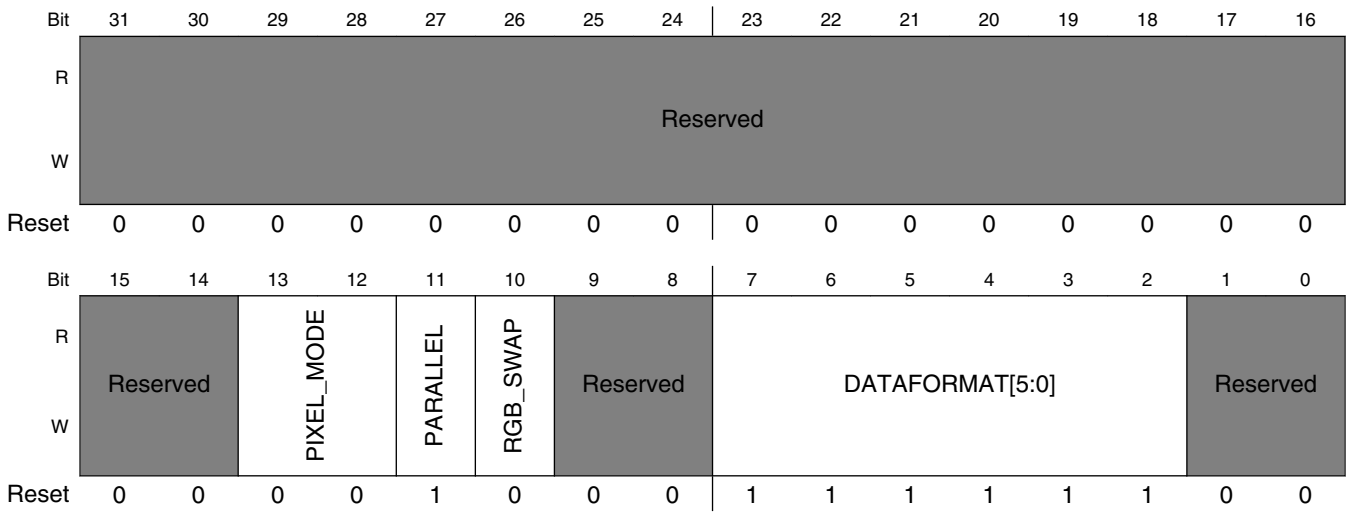
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	S_DPHYCTRL[63:32]																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_CSI_DPHY_SLAVE_CTRL_HIGH field descriptions

Field	Description
S_DPHYCTRL[63:32]	D-PHY Slave control register High part Refer to ‘User Guide of D-PHY’ for more information.

13.4.15.13 ISP Configuration Register (MIPI_CSI_ISP_CONFIG*n*)

Address: 32E3_0000h base + 40h offset + (16d × i), where i=0d to 3d



MIPI_CSI_ISP_CONFIG*n* field descriptions

Field	Description
31–14 -	This field is reserved.
13–12 PIXEL_MODE	Pixel mode selection, 'b11 : Invalid 'b10 : Quad pixel mode (RAW8/10/12) 'b01 : Dual pixel mode (RAW8/10/12, YUV422) 'b00 : Single pixel mode
11 PARALLEL	Output bus width of CH0 is 32 bits. When this bit is set, the outer bus width of CSIS V3.3 is 32. Refer to figures in chapter 12 of MIPI CSI-2 standard spec. 0 Normal output 1 32bit data alignment
10 RGB_SWAP	Swapping RGB sequence 0 MSB is R and LSB is B 1 MSB is B and LSB is R (swapped)
9–8 -	This field is reserved.
7–2 DATAFORMAT[5:0]	Image Data Format 0x18 - YUV420 (8bit) 0x19 - YUV420 (10bit) 0x1A - YUV420 (8bit legacy)

Table continues on the next page...

MIPI_CSI_ISP_CONFIG*n* field descriptions (continued)

Field	Description
	0x1C - YUV420 (8bit CSPS) 0x1D - YUV420 (10bit CSPS) 0x1E - YUV422 (8bit) 0x1F - YUV422 (10bit) 0x20 - (NOT SUPPORTED) RGB444 0x21 - (NOT SUPPORTED) RGB555 0x22 - RGB565 0x23 - RGB666 0x24 - RGB888 0x28 - RAW6 0x29 - RAW7 0x2A - RAW8 0x2B - RAW10 0x2C - RAW12 0x2D - RAW14 0x30 - User defined 1 0x31 - User defined 2 0x32 - User defined 3 0x33 - User defined 4 0x34 - User defined 5 0x35 - User defined 6 0x36 - User defined 7 0x37 - User defined 8 **NOTE:** Not described types are ignored.
-	This field is reserved.

13.4.15.14 ISP Resolution Register (MIPI_CSI_ISP_RESOLUTION*n*)

Address: 32E3_0000h base + 44h offset + (16d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VRESOL																HRESOL															
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_CSI_ISP_RESOLUTION*n* field descriptions

Field	Description
31–16 VRESOL	Vertical Image resolution

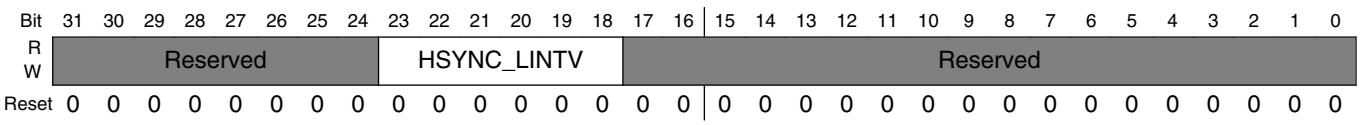
Table continues on the next page...

MIPI_CSI_ISP_RESOLUTION*n* field descriptions (continued)

Field	Description
	Input boundary : 0x0001 ~ 0xFFFF
HRESOL	Horizontal Image resolution Input boundary of each image format. Hresol value must comply with the terms of the standard spec and it must be a multiple of four. Input boundary : 0x0004 ~ 0xFFFC

13.4.15.15 ISP SYNC Register (MIPI_CSI_ISP_SYNC*n*)

Address: 32E3_0000h base + 48h offset + (16d × i), where i=0d to 3d

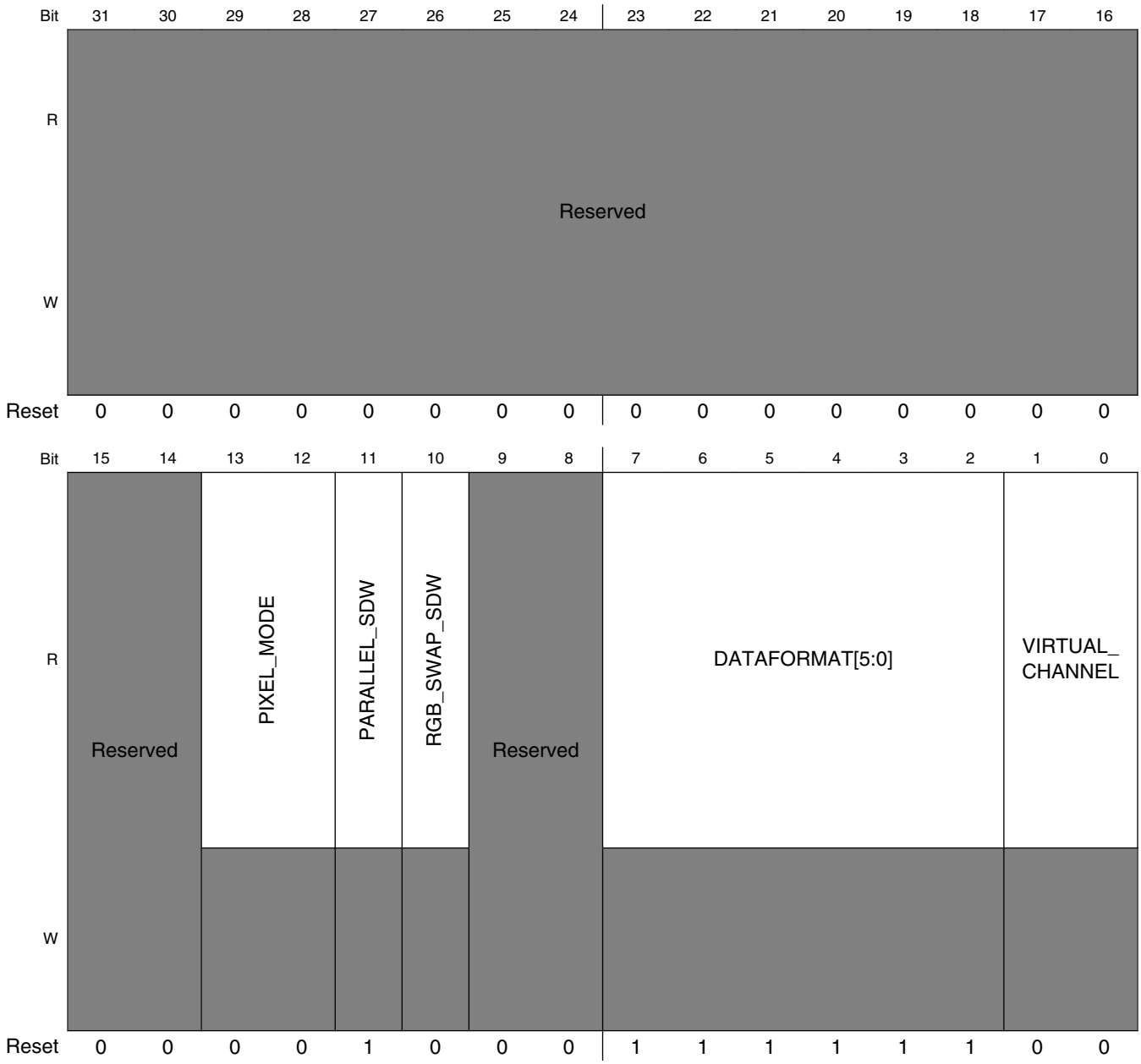


MIPI_CSI_ISP_SYNC*n* field descriptions

Field	Description
31–24 -	This field is reserved.
23–18 HSYNC_LINTV	Interval between last falling of DVALID and falling of HSync. In Figure 2-12, t2 is this. 6'h00 ~ 6'h3F cycle of Pixel clock
-	This field is reserved.

13.4.15.16 Shadow Configuration Register (MIPI_CSI_SHADOW_CONFIG*n*)

Address: 32E3_0000h base + 80h offset + (16d × i), where i=0d to 3d



MIPI_CSI_SHADOW_CONFIG*n* field descriptions

Field	Description
31–14 -	This field is reserved.

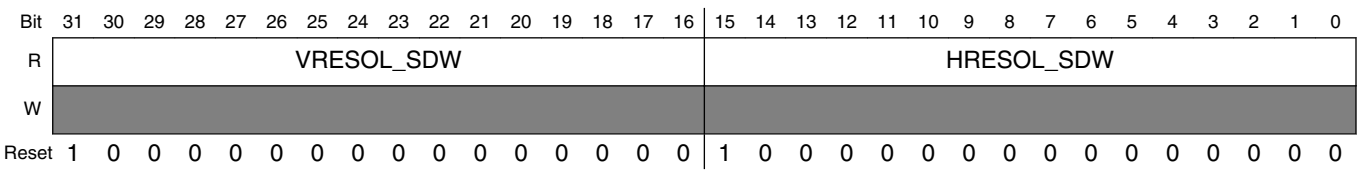
Table continues on the next page...

MIPI_CSI_SHADOW_CONFIG*n* field descriptions (continued)

Field	Description
13–12 PIXEL_MODE	Current pixel_mode
11 PARALLEL_SDW	Current Parallel
10 RGB_SWAP_SDW	Current RGB_SWAP
9–8 -	This field is reserved.
7–2 DATAFORMAT[5:0]	Current Image Data Format
VIRTUAL_CHANNEL	Current Virtual channel

13.4.15.17 Shadow Resolution Register (MIPI_CSI_SHADOW_RESOLUTION*n*)

Address: 32E3_0000h base + 84h offset + (16d × i), where i=0d to 3d

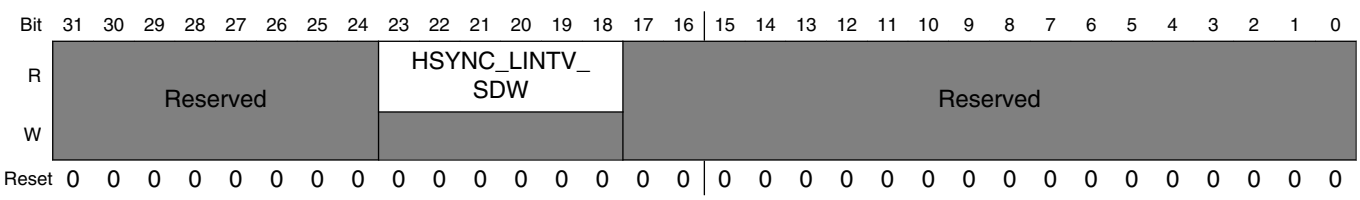


MIPI_CSI_SHADOW_RESOLUTION*n* field descriptions

Field	Description
31–16 VRESOL_SDW	Current Vertical Image resolution
HRESOL_SDW	Current Horizontal Image resolution

13.4.15.18 Shadow SYNC Register (MIPI_CSI_SHADOW_SYNC*n*)

Address: 32E3_0000h base + 88h offset + (16d × i), where i=0d to 3d



MIPI_CSI_SHADOW_SYNC*n* field descriptions

Field	Description
31–24 -	This field is reserved.
23–18 HSYNC_LINTV_SDW	Current interval between Hsync falling and Hsync rising (Line interval)
-	This field is reserved.

13.4.15.19 Frame Counter (MIPI_CSI_FRAME_COUNTER*n*)

Address: 32E3_0000h base + 100h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FRM_CNT																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_CSI_FRAME_COUNTER*n* field descriptions

Field	Description
FRM_CNT	Number of input frames

13.4.15.20 Line Interrupt Ratio (MIPI_CSI_LINE_INTERRUPT_RATIO*n*)

Address: 32E3_0000h base + 110h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LINE_INTR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_CSI_LINE_INTERRUPT_RATIO*n* field descriptions

Field	Description
LINE_INTR	The line number that user wants to set the line_end interrupt.

13.5 MIPI DSI Host Controller (MIPI_DSI)

13.5.1 Overview

This chapter describes in detail about the MIPI Display Serial Interface (DSI) controller. This is a flexible, high-performance core that allows communication with MIPI DSI compliant peripherals.

13.5.1.1 Block Diagram

13.5.1.1.1 Total system block diagram

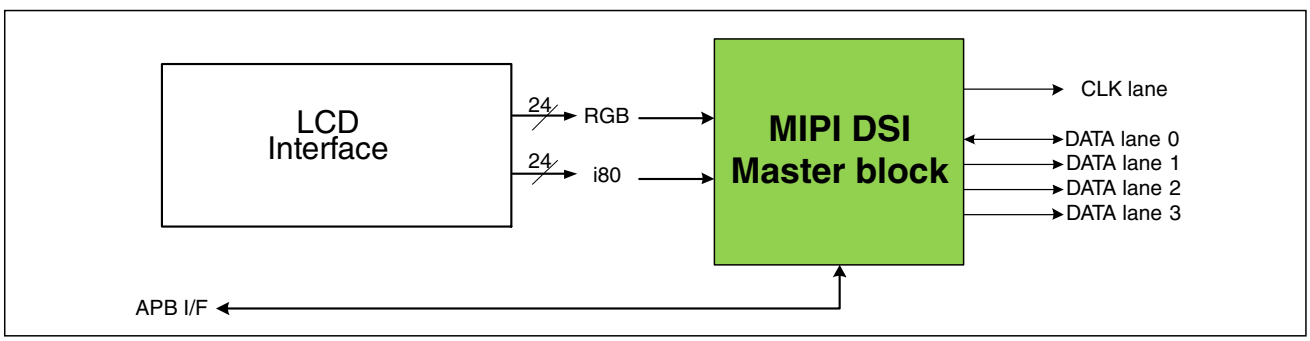


Figure 13-45. MIPI DSI master system block diagram

13.5.1.1.2 MIPI DSI master and D-PHY I/F block diagram

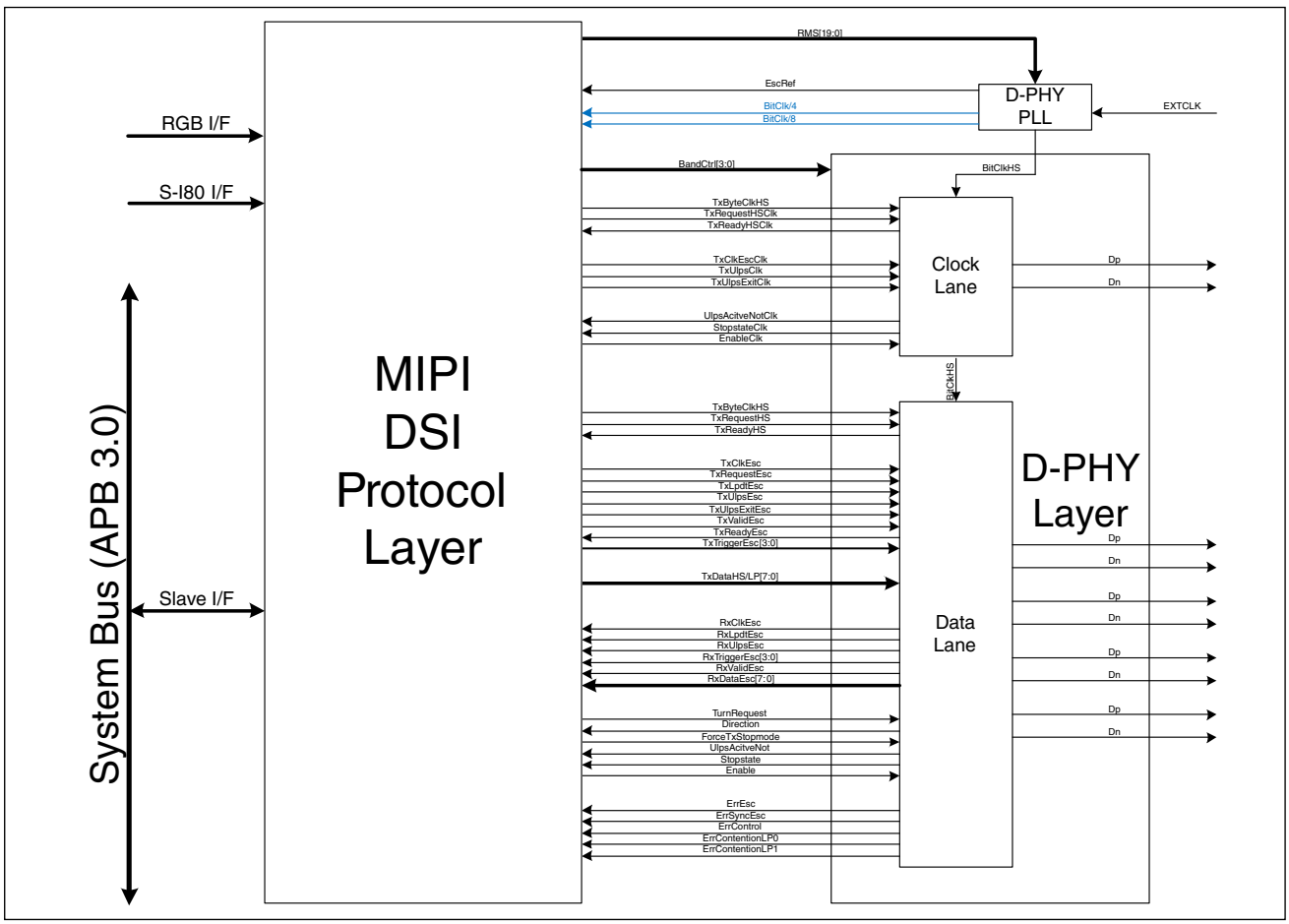


Figure 13-46. PPI I/F block diagram

13.5.1.1.3 MIPI DSI master block diagram

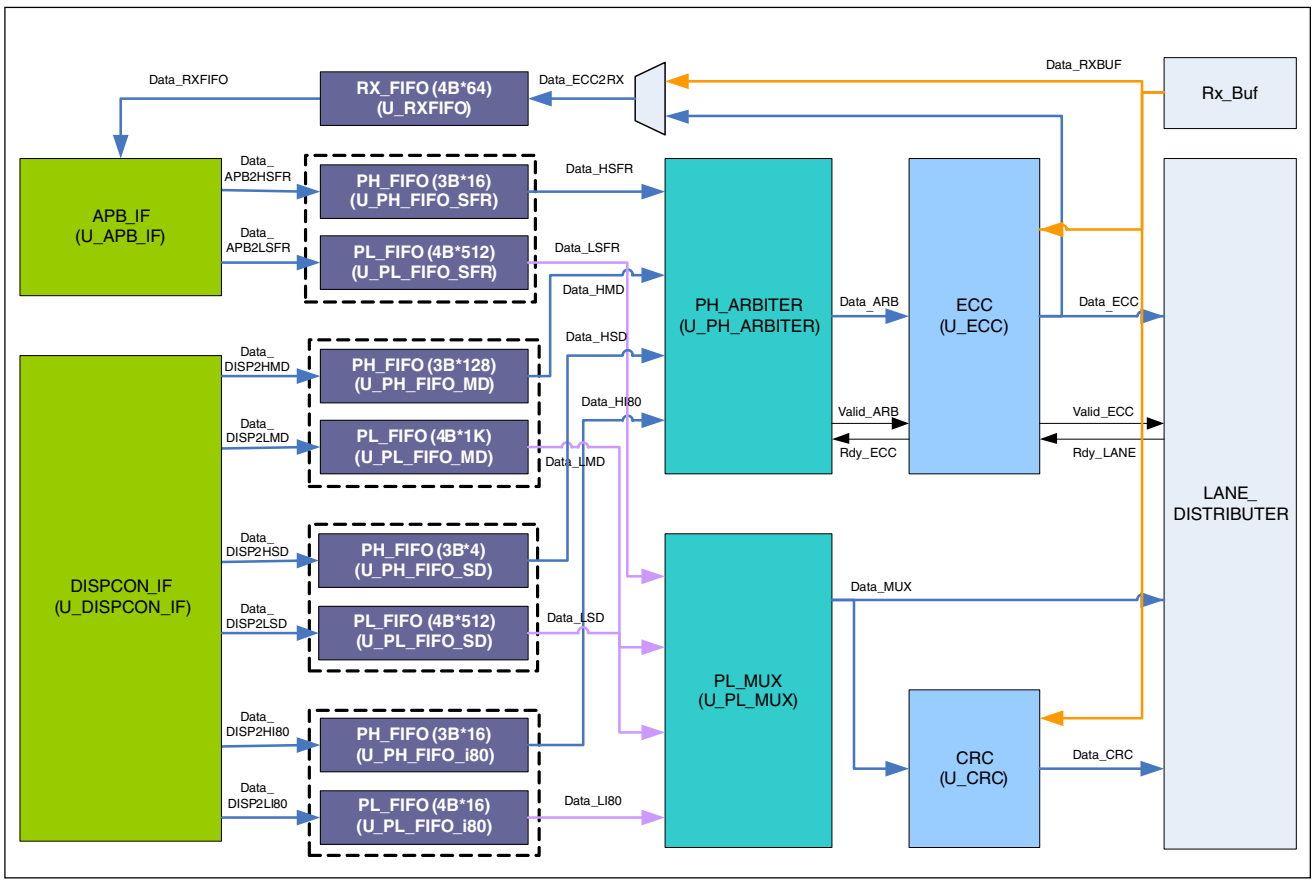


Figure 13-47. MIPI DSI master module block diagram

13.5.2 Features

The key features of MIPI DSI include:

- Complies to MIPI DSI Standard Specification V1.01r11
  - Maximum resolution ranges up to WQHD (1920x1080p60, 24bpp)
    - It should be decided on bandwidth between input clock (video clock) and output clock (D-PHY HS clock).
  - Supports 1, 2, 3, or 4 data lanes
  - Supports pixel format: 16bpp, 18bpp packed, 18bpp loosely packed (3 byte format), and 24bpp
- Interfaces
  - Complies with Protocol-to-PHY Interface (PPI) in 1.0Gbps / 1.5Gbps MIPI D-PHY
  - Supports RGB Interface for Video Image Input from general display controller

- Supports S-i80(Synchronous i80) Interface for Command Mode Image input from display controller
- Supports PMS control interface for PLL to configure byte clock frequency
- Supports Prescaler to generate escape clock from byte clock

13.5.3 Functional description

The following sections describe functional details of the MIPI DSI module.

13.5.3.1 Internal primary FIFOs

Table 13-31 describes configurable-sized primary FIFOs.

Table 13-31. Internal primary FIFO list

Port	FIFO type	Size	Description
Main display for RGB I/F and S-i80 I/F	Packet Header FIFO	3Byte X 128 depth	Packet header FIFO for main display
	Payload FIFO	4Byte X 1024 depth	Payload FIFO for main display image (SRAM used)
Sub display for S-i80 I/F image data	Packet Header FIFO	3Byte X 4 depth	Packet header FIFO for S-i80 I/F sub display
	Payload FIFO	4Byte X 512 depth	Payload FIFO for S-i80 I/F sub display image (SRAM used)
Command for S-i80 I/F command	Packet Header FIFO	3Byte X 16 depth	Packet header FIFO for S-i80 I/F command packet
	Payload FIFO	4Byte X 16 depth	Payload FIFO for S-i80 I/F command long packet payload
SFR for general packets	Packet Header FIFO	3Byte X 16 depth	Packet header FIFO for general packet
	Payload FIFO	4Byte X 512 depth	Payload FIFO for general long packet (SRAM used)
RX FIFO	Packet header and Payload FIFO	4Byte X 64 depth	Rx FIFO for LPDR. This FIFO is common for packet header and payload

13.5.3.2 Packet header arbitration

There are four-packet headers FIFOs for Tx, namely, main display, sub display, S-i80 INTERFACE command, and SFR FIFO. The main and sub display FIFO packet headers contain the image data, while the S-i80 INTERFACE command FIFO packet header contains the command packets. On the other hand, the SFR FIFO packet header contains command packets, sub display image data (in Video mode), and so on.

The packet header arbiter has a “Fixed priority” algorithm. Priority order is fixed as main display, sub display, S-i80 INTERFACE command, and SFR FIFO packet header.

In the Video mode, sub display and S-i80 INTERFACE command FIFO are not used. The SFR FIFO packet header checks if the main display FIFO is empty (no request) in not-active image region and then sends its request.

13.5.3.3 RxFIFO Structure

To read the packets received via low power data receiving mode, RxFIFO acts like an SFR. RxFIFO is an asynchronous FIFO with ByteClk and PCLK domains as input clock and output clock domains respectively. The Rx data is synchronized to RxClk. RXBUF has four Rx Byte buffers for aligning byte to word.

The packet headers of all the packets stored in RXFIFO are word-aligned, that is, the first byte of a packet is always stored in LSB. For example, if a long packet has 7-byte payload, the last byte is filled with dummy byte and the next packet is stored in the next word, as shown in Figure 13-48.

NOTE

CRC data is not stored in RxFIFO.

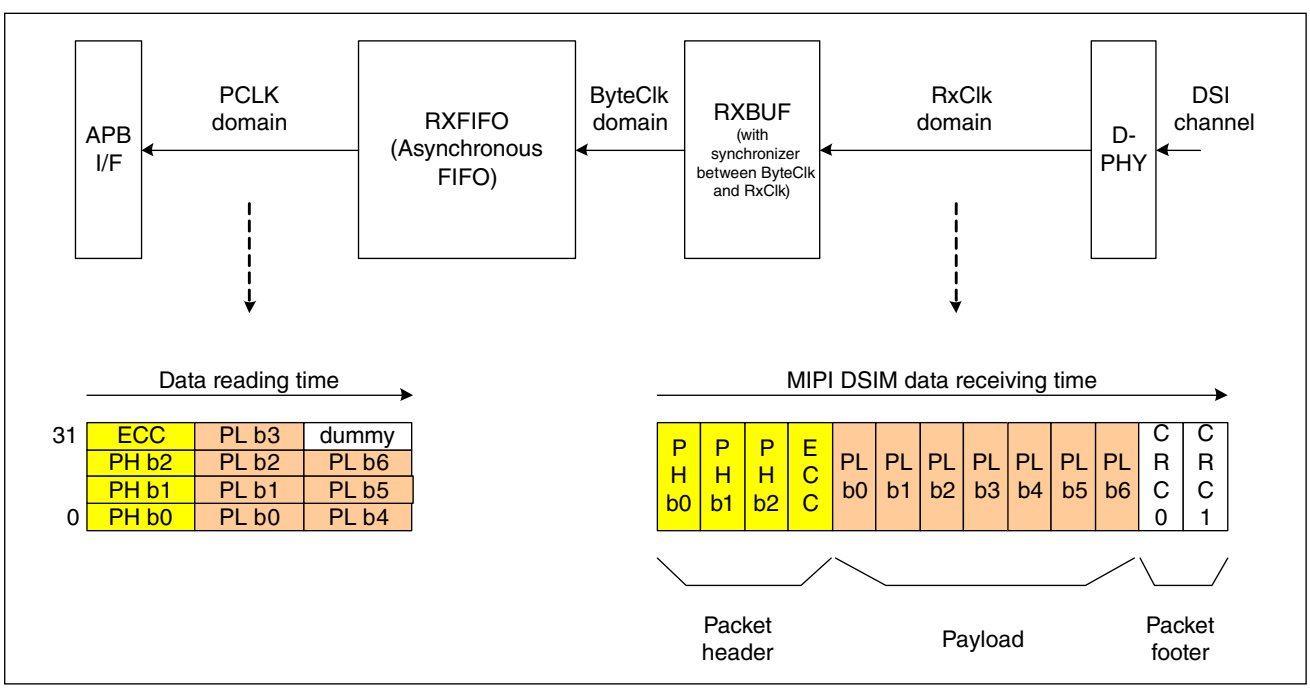


Figure 13-48. Rx data word alignment

13.5.3.4 Interfaces and protocol

Display Controller-to-DSI Conceptual Signal Converting Diagram in Video Mode.

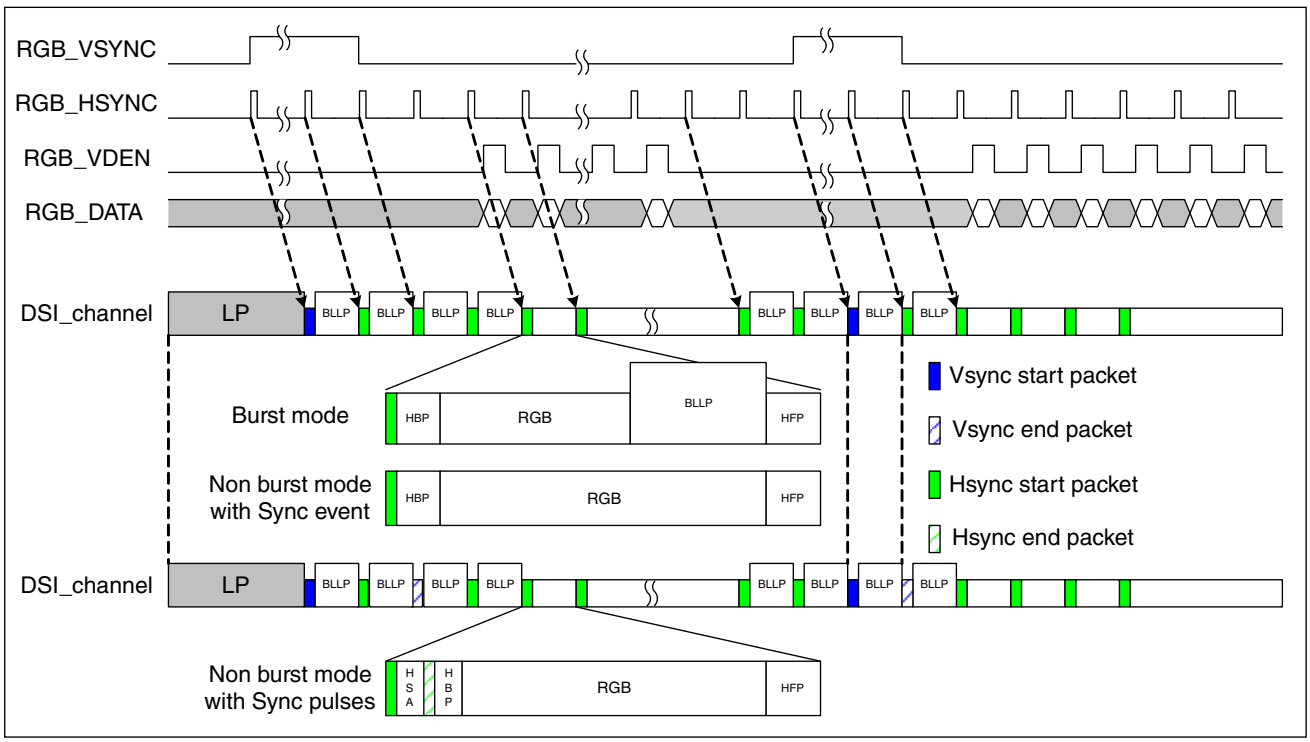


Figure 13-49. Signal converting diagram in video mode

13.5.3.5 Interface timing and protocol

13.5.3.5.1 Display controller interface

MIPI DSI Master has two-display controller interfaces, namely, RGB INTERFACE for main display and CPU INTERFACE (S-i80 INTERFACE) for main/ sub display. The Video mode uses RGB INTERFACE while the Command mode uses CPU INTERFACE.

The RGB image data is loaded on the data bus of RGB INTERFACE and S-i80 INTERFACE with the same order: RGB_VD[23:0] or SYS_VDOUT[23:0] is {R[7:0],G[7:0],B[7:0]}. Each byte aligns to the most significant bit. For instance, in the 12-bit mode, only three 4-bit values are valid as R, G, and B each, that is, data[23:20], data[15:12], and data[7:4]. The DSIM ignores rest of the bits.

13.5.3.5.2 RGB interface

Vsync, Hsync, and VDEN are active high signals. Among the three signals, Vsync and Hsync are pulse types that spend several video clocks. RGB_VD[23:0] is {R[7:0],G[7:0],B[7:0]}. All sync signals are synchronized to the rising edge of RGB_VCLK. The display controller sends minimum one horizontal line length of Vsync pulse, V back porch, and V front porch. Hsync pulse width should be longer than 1-byte clock cycle.

13.5.3.5.2.1 HSA disable mode

HSA mode specifies the Horizontal Sync Pulse area disable mode.

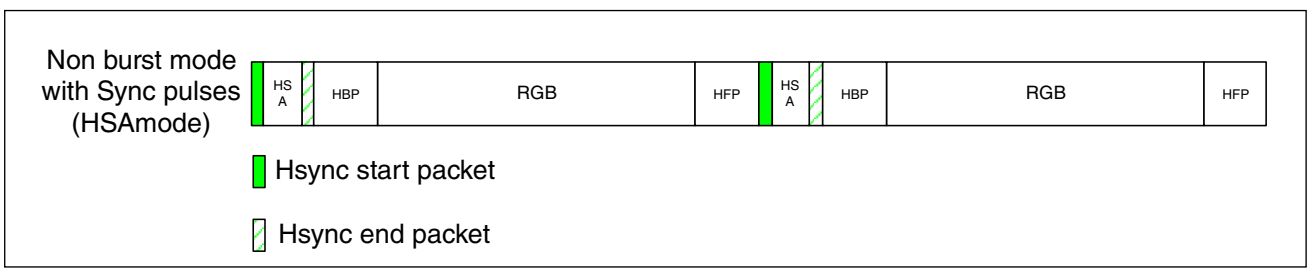


Figure 13-50. Block timing diagram of HSA disable mode (HSA mode reset : DSIM_CONFIG[20] = 0)

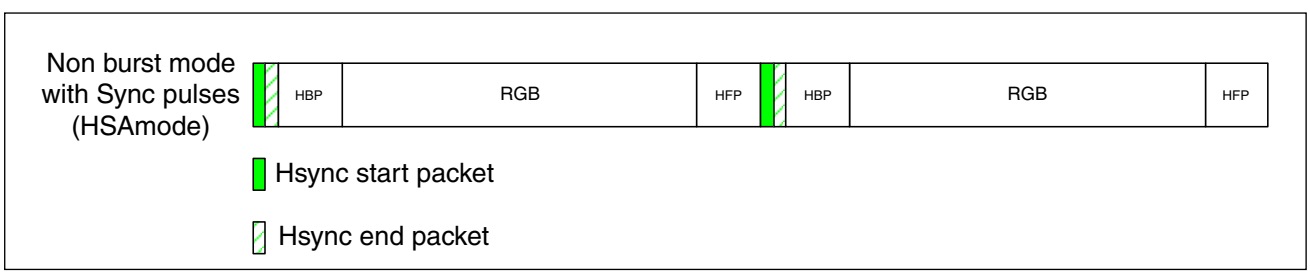


Figure 13-51. Block timing diagram of HSA disable mode (HSA mode set : DSIM_CONFIG[20] = 1)

13.5.3.5.2.2 HBP disable mode

This is optional spec what is used only burst mode. HBP mode is Horizontal Back Porch disable mode.

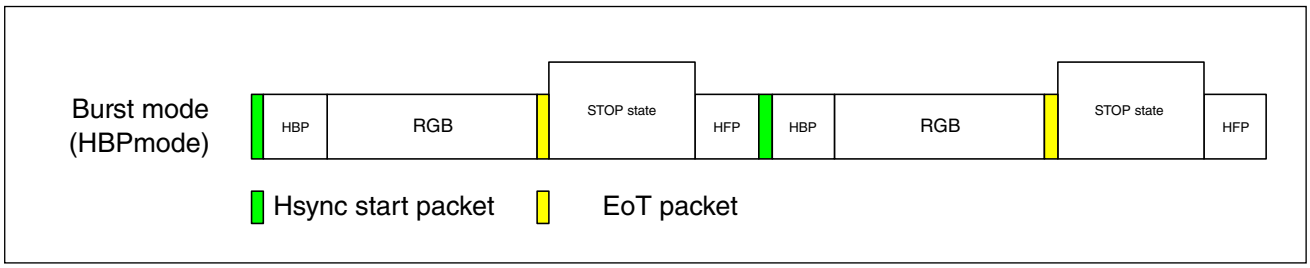


Figure 13-52. Block timing diagram of HBP disable mode (HBP mode reset : DSIM_CONFIG[21] = 0)

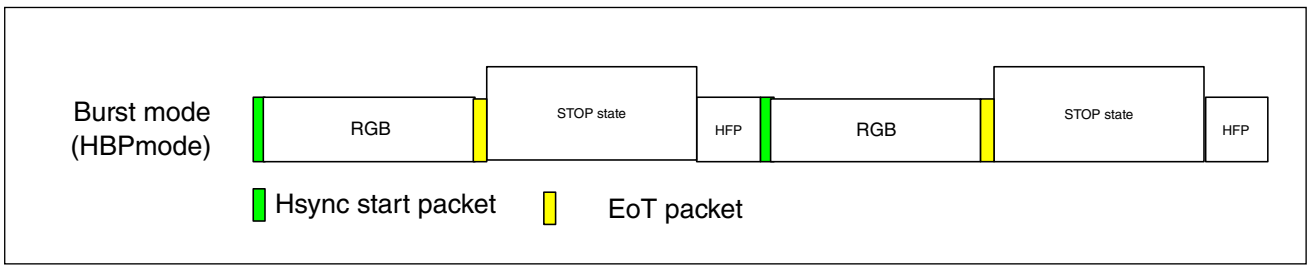


Figure 13-53. Block timing diagram of HBP disable mode (HBP mode set : DSIM_CONFIG[21] = 1)

13.5.3.5.2.3 HFP disable mode

This is optional spec what is used only burst mode. HFP mode is Horizontal Front Porch disable mode.

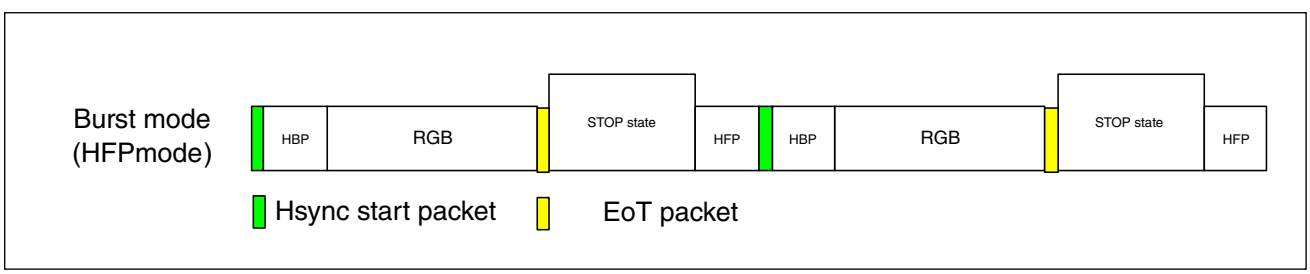


Figure 13-54. Block timing diagram of HFP disable mode (HFP mode set : DSIM_CONFIG[22] = 1)

13.5.3.5.2.4 HSE disable mode

HSE mode is Horizontal Sync End packet enable mode in Vsync pulse or Vporch area.

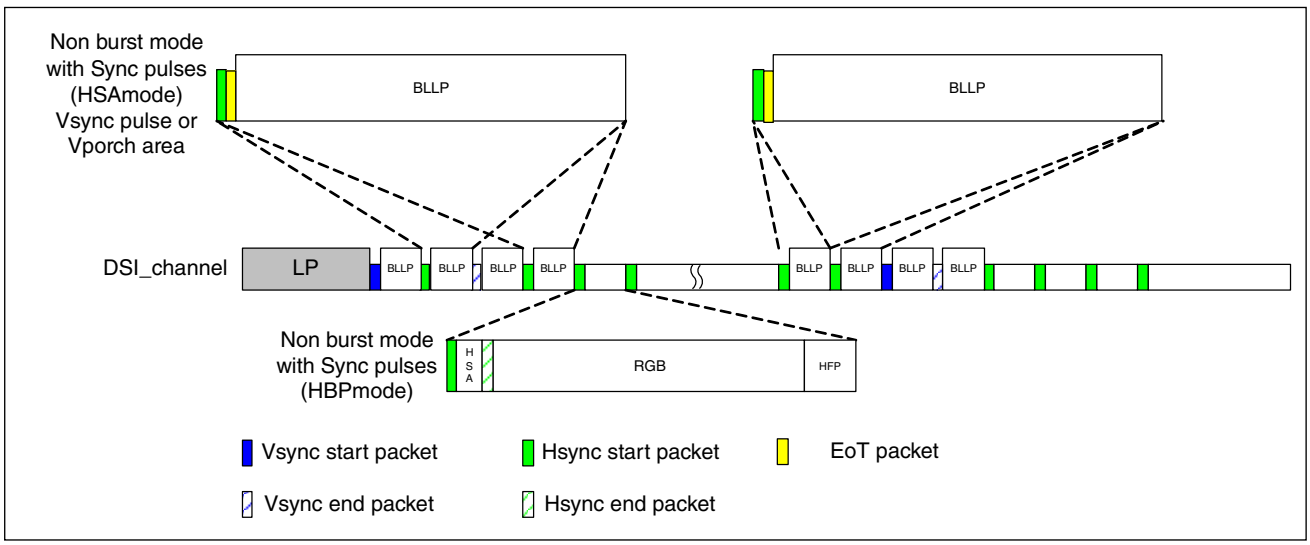


Figure 13-55. Block timing diagram of HSE disable mode (HSE mode reset : DSIM_CONFIG[23] = 0)

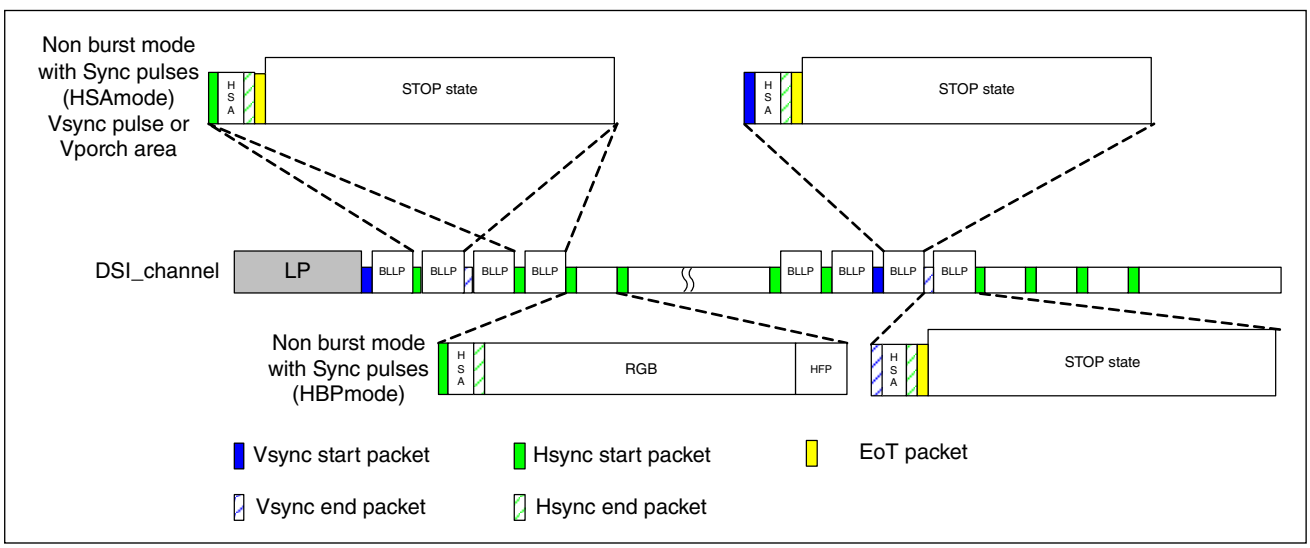


Figure 13-56. Block timing diagram of HSE disable mode (HSE mode set : DSIM_CONFIG[23] = 1)

13.5.3.5.2.5 Transfer general data in video mode

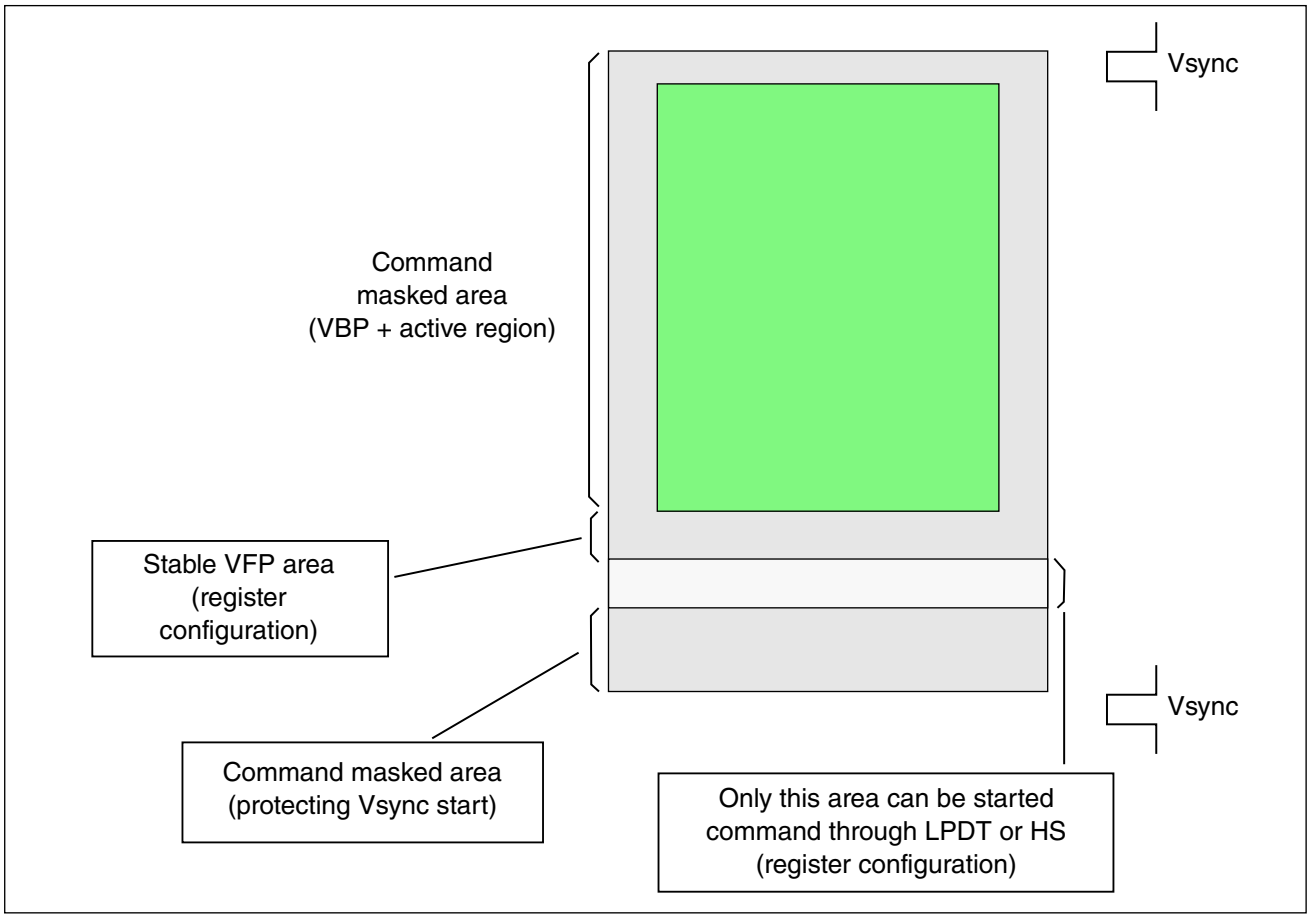


Figure 13-57. Stable VFP Area Before Command Transfer Allowing Area

MIPI DSIM Converts RGB Interface to Video Mode.

Vsync and Hsync packets are extremely important to protect image in Video mode. MIPI DSIM allows several lines in VFP area to transfer general data transfer. As shown in Figure 13-57, the vertical front porch is divided into three areas, namely, Stable VFP area, Command allowed area, and Command masked area.

The register configures stable VFP area. Configuration boundary is 11'h000 ~ 11'h7FFF in DSIM_MVPORCH.

The register also configures the command allowed area. Configuration boundary is 4'h0 ~ 4'hF in DSIM_MVPORCH. Only this area is allowed to start "command transfer" through HS mode or LPDT. In LPDT, data transferring takes a long time to complete (approximately hundreds of microseconds or more). In this time, Hsync packet does not arrive due to LPDT long packet. MIPI DSIM comprises of big size FIFO for lost Hsync packet. After LPDT, MIPI DSIM transfers these Hsync packets immediately through HS mode.

To protect Vsync, command masked area is masked area. This area is calculated using LPDT bandwidth. For example, if EscClk is 10MHz, the maximum long packet payload size is 1KB and LPDT, LPDT transferring time is 824us (packet size: 1030byte, LPDT maximum bandwidth: 10Mbps). If one line time is 20us, the line timing violation occurs in 42 lines. Therefore, command masked area is larger than 42 + "block". This "block" is transferring time of the violated Hsync packets.

Display controller should be configured in such a way that VFP lines are sum of stable VFP, command allowed area, and command masked area.

13.5.3.5.3 S-i80 interface

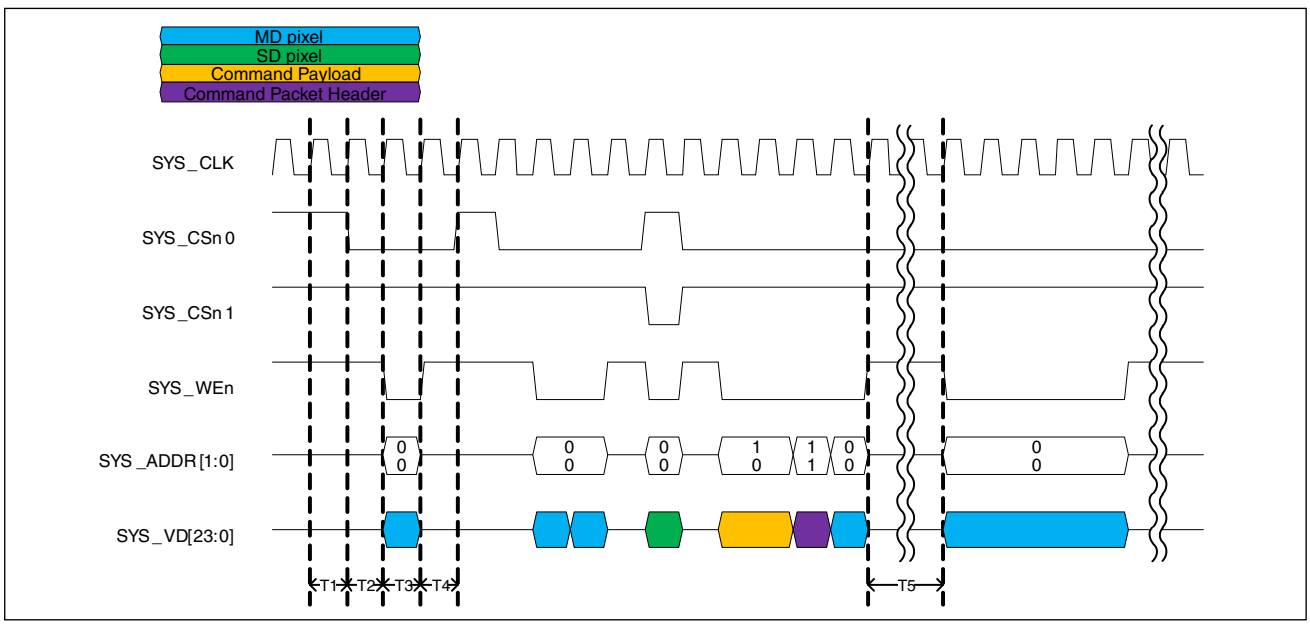


Figure 13-58. S-i80 I/F timing diagram

- T1 ≥ 1 clock cycle.
- T2 ≥ 0 clock cycle.
- T3 ≥ 1 clock cycle.
- T4 ≥ 0 clock cycle.
- T2+T3+T4 > 1 cycle of Byte clock.
- T5 ≥ 10 clock cycle (every horizontal blank period).

A display controller generates these signals with its internal clock: SYS_CS0/CS1, SYS_WE, and SYS_VD. MIPI DSI master decodes the SYS_ADDR. Table 13-32 describes the Command mode Interface address map.

Table 13-32. Command mode interface address map

SYS_ADDR[1:0]	Description
2'b00	Image data
2'b01	Reserved
2'b10	Payload data
2'b11	Packet Header

Figure 13-59 shows how MIPI DSI Master makes packet from the image data stream via S-i80 INTERFACE in Command mode. MIPI DSI master makes packet from the first line with DCS command "write_memory_start" and the other lines with DCS command "write_memory_continue".

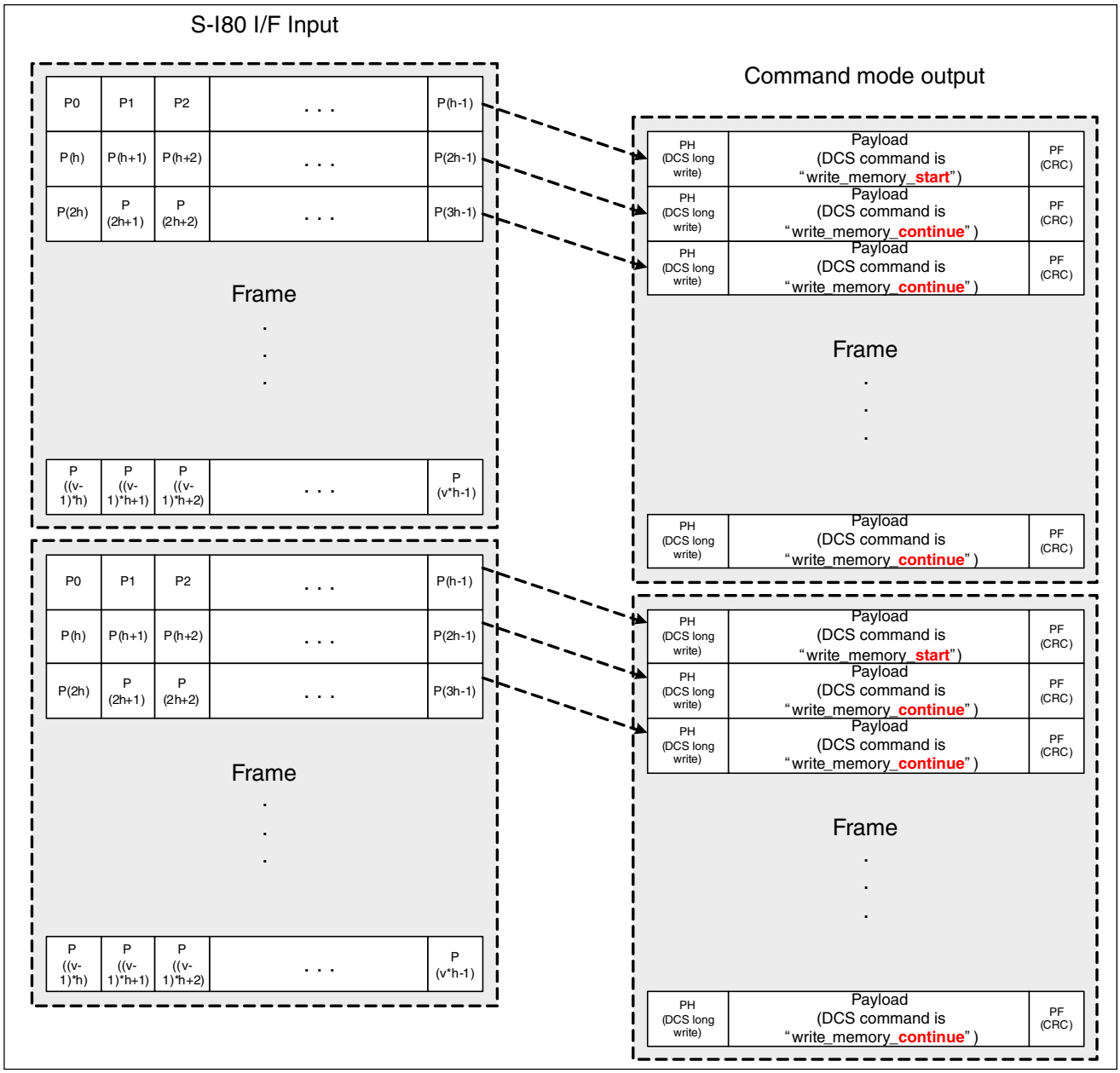


Figure 13-59. Packetizing for MIPI DSI command mode from S-i80 I/F

13.5.3.5.4 Relation between input transactions and DSI transactions

Table 13-33. Relation between input transactions and DSI transactions

Input Interface	Input transaction	DSI Transaction
RGB	RGB transaction	Specifies the RGB Packet. 888, 666, 666 (loosely packed), and 565 should be specified via register configuration.
RGB / S-i80	RGB / S-i80 Image Transaction	Specifies the Data type, that is, "DCS Long Write packet".

Table continues on the next page...

Table 13-33. Relation between input transactions and DSI transactions (continued)

Input Interface	Input transaction	DSI Transaction
		(DCS command is "memory write start/continue".)
S-i80	S-i80 Command Transaction	Specifies any DSI packet. Bytes in S-i80 transaction should be the same bytes in DSI packets.
SFR	Header and Payload FIFO access	Specifies any DSI Packets. Bytes in APB transaction should be the same bytes in DSI packets.

13.5.3.5.5 PPI interface timing and protocol

13.5.3.5.5.1 Initialization after power-on / reset

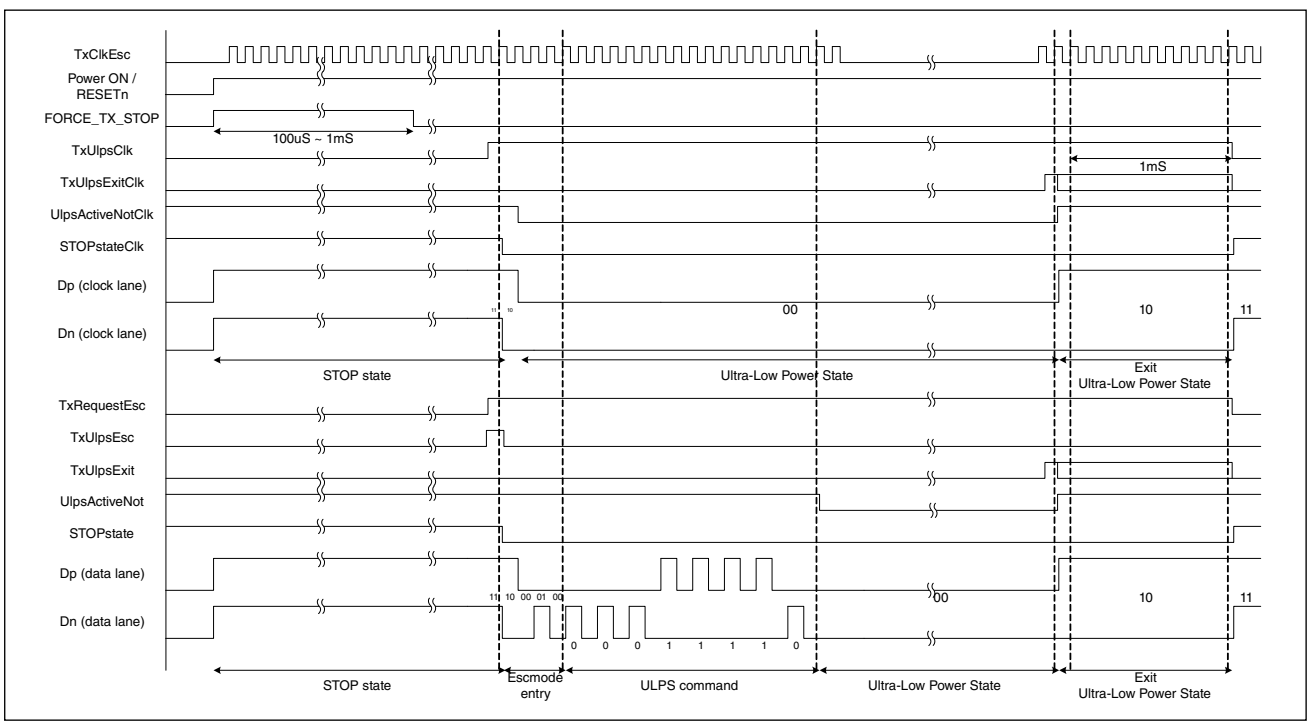


Figure 13-60. Timing diagram of initialization

13.5.3.5.5.2 High speed data transfer

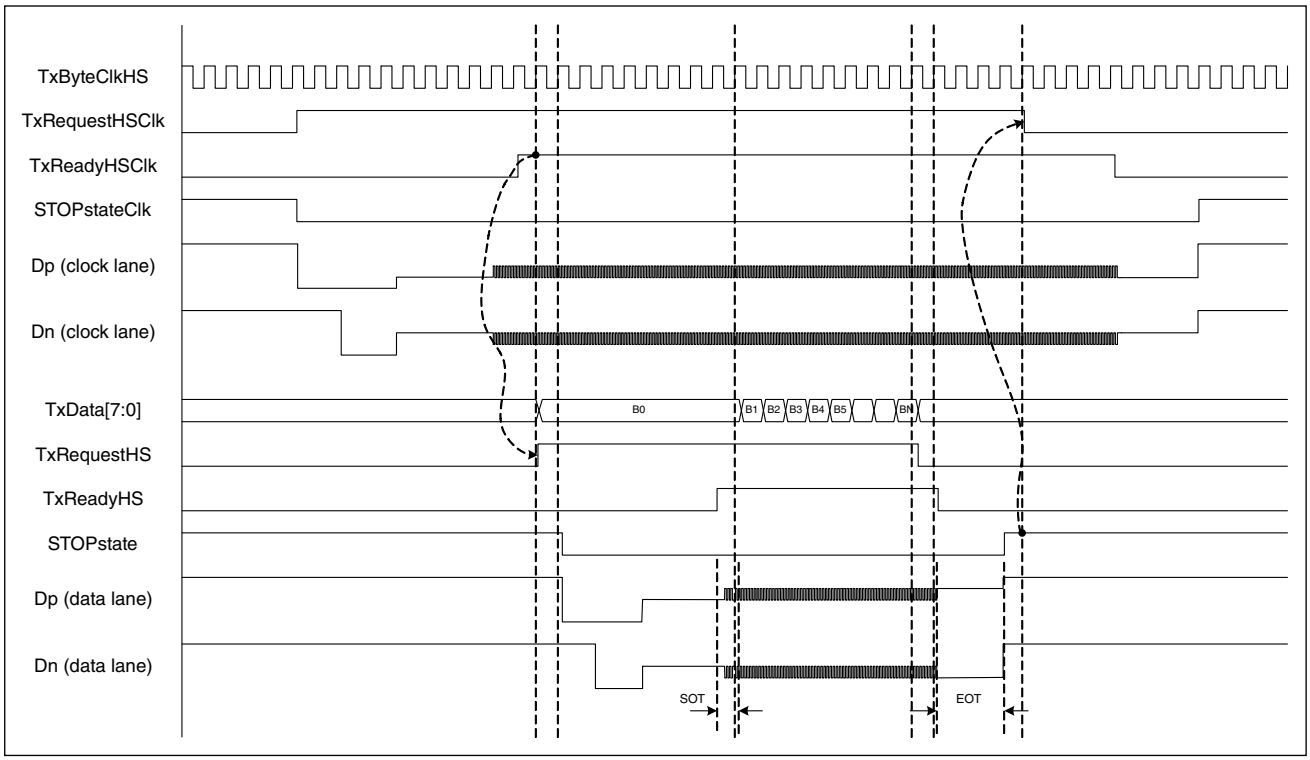


Figure 13-61. Timing diagram of high speed data transfer

13.5.3.5.5.3 Low power data transfer

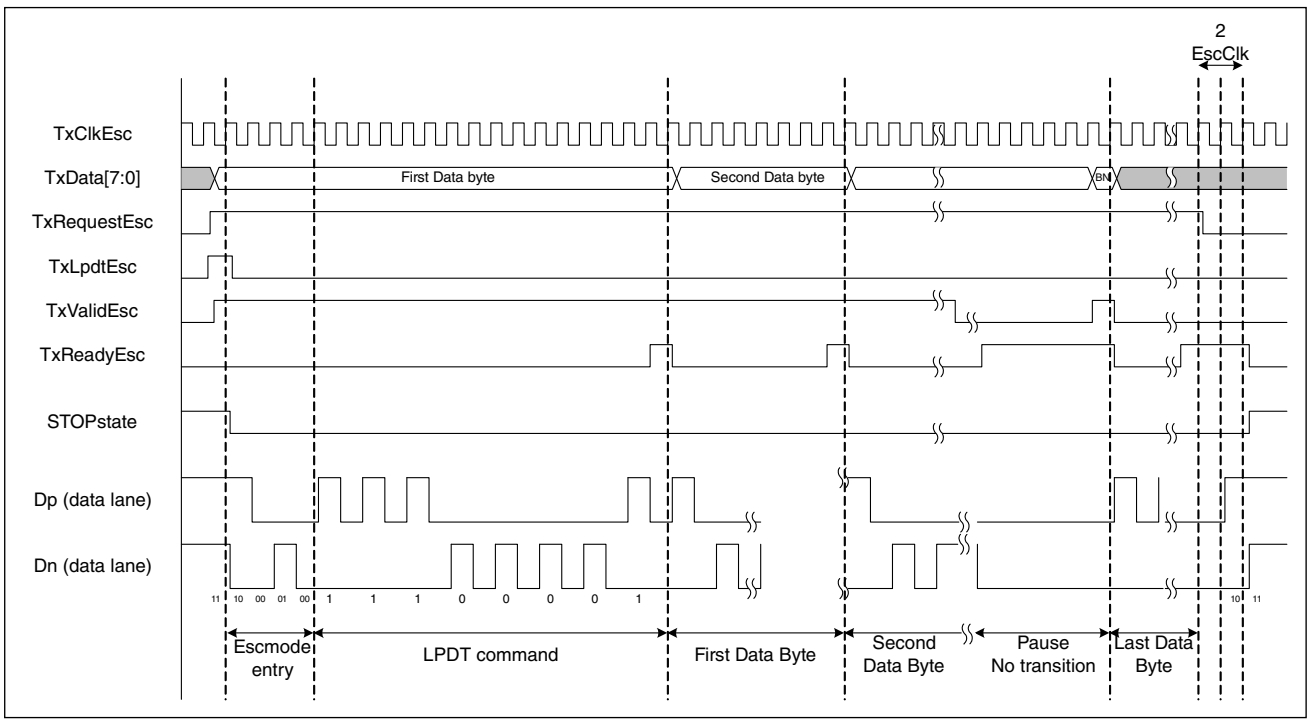


Figure 13-62. Timing diagram of low power data transfer

13.5.3.5.5.4 Ultra-Low power state

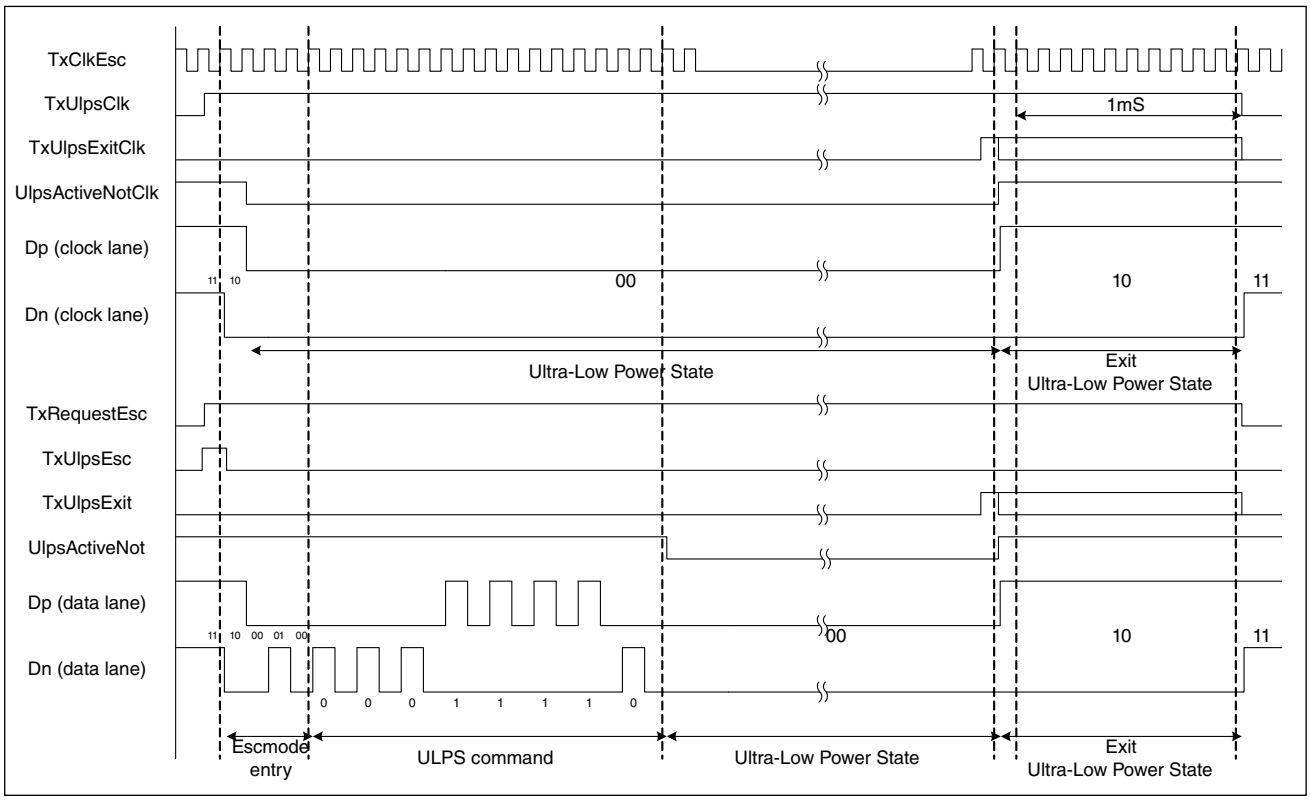


Figure 13-63. Timing diagram of ultra low power state

13.5.3.5.5.5 Trigger function

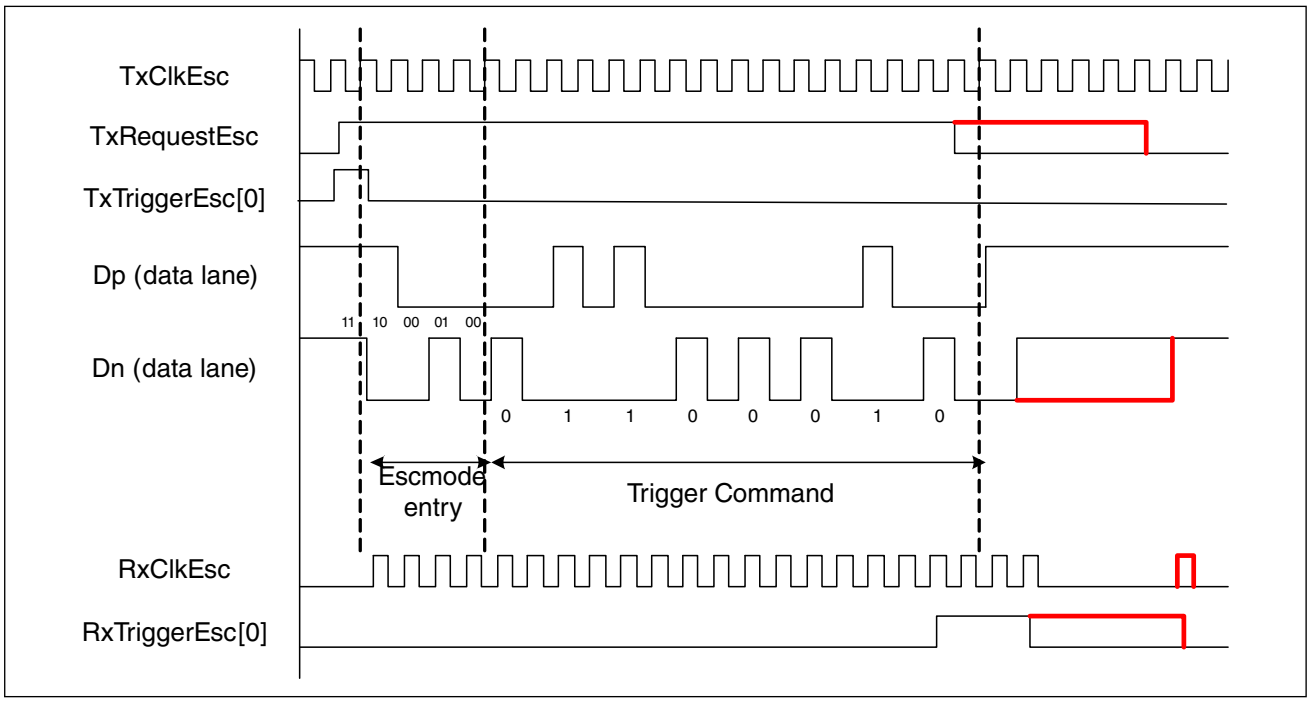


Figure 13-64. Diagram of trigger function "Remote Reset"

There are four types of trigger commands:

- 01100010 : remote reset – forward direction
- 01011101 : Tearing Effect – reverse direction
- 00100001 : Acknowledge – reverse direction
- 10100000 : Unknown

If DSI D-PHY received these signals, D-PHY indicates trigger function to protocol layer.

13.5.3.5.5.6 Turn-around and low power data receiving

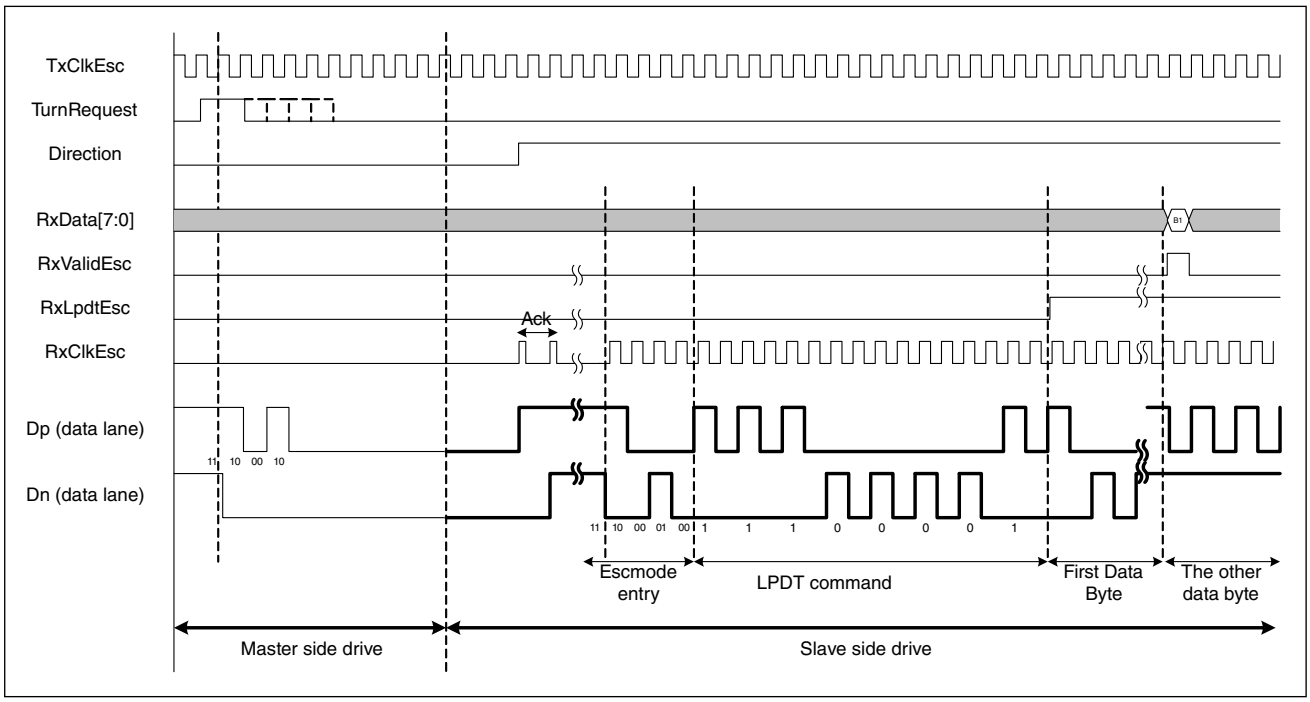


Figure 13-65. Timing diagram of Bus-Turn-Around

13.5.3.5.5.7 TE signaling

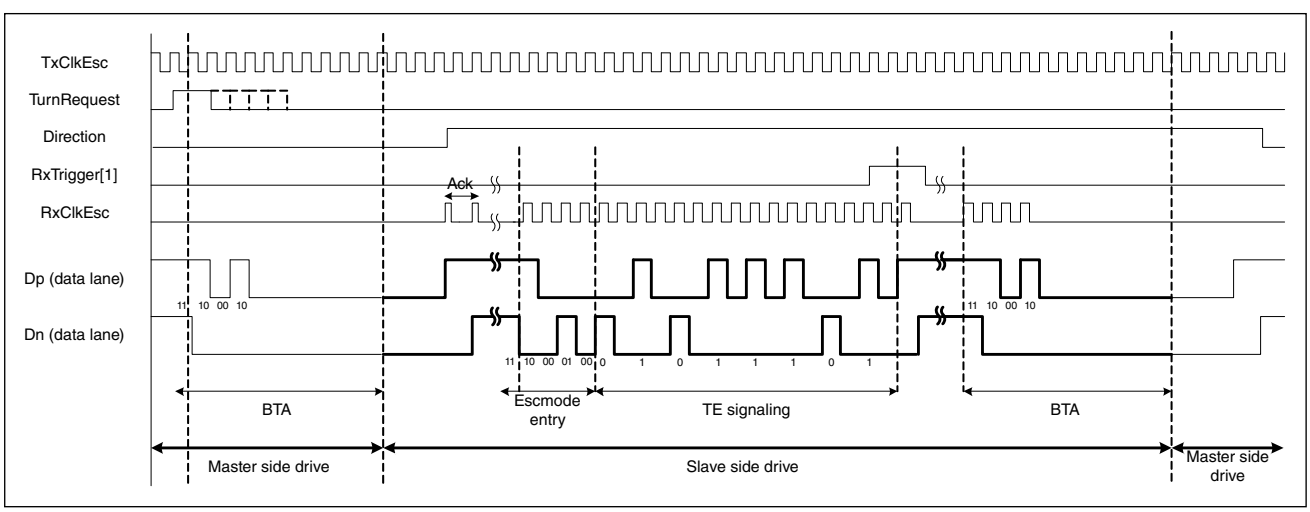


Figure 13-66. TE signaling

13.5.3.6 Configuration

13.5.3.6.1 Video mode VS Command mode

MIPI DSI Master Block supports two modes, namely, Video mode and Command mode.

Command mode uses RGB interface and S-i80 interface. Mode can change via SFRs.

13.5.3.6.2 Dual display VS Single display

13.5.3.6.2.1 Dual display

MIPI DSI Master supports dual display configuration in Command mode only, that is, both main and sub display image should be transmitted via S-i80 INTERFACE.

13.5.3.6.2.2 Single display

For single display configuration, use video mode or command mode.

13.5.3.6.2.3 PLL

To transmit Image data, MIPI DSI Master Block needs high frequency clock (80MHz ~ 1GHz) generated by PLL.

To configure PLL, MIPI DSI Master comprises of SFRs and corresponding interface signals. PLL is embedded in PHY module.

13.5.3.6.2.4 Buffer

In MIPI DSI standard specification, DSI Master sends image stream in burst mode. The image stream transmits in high-speed and bit-clock frequency. This mode allows the device to stay in stop state longer to reduce power consumption. For this mode, MIPI DSI Master has a line buffer to store one complete line and send it faster at the next line time.

13.5.4 Application Scenario

13.5.4.1 Display mode

13.5.4.1.1 Video mode

Video mode starting sequence:

1. Set RGB I/F in Display Controller(RGB888), also set Video mode at MIPI DSI Master.
2. Configure all parameters for displaying image (define resolution, back porch, front porch, data lane number, HS clock frequency(byte clock), etc).
3. Start image transmission.

13.5.4.1.2 Command mode

Command mode starting sequence:

1. Set S-i80 I/F in Display Controller (RGB888), also set Command mode at MIPI DSI Master.
2. Configure all parameters for displaying image.
3. Set the command at MIPI DSI registers using MIPI DCS.
4. S-i80 I/F is just used to transmit image data. The users have to read display peripheral status and write command through MIPI DSI registers.

13.5.4.2 Programming Model

Table 13-34. Behavior programming each case

No.	Mode	Case	Pseudo Code

1	Video	Initialize	1. reset or power on
			2. SFR_Write(Display Controller_registers, RGB I/F_888);
			3. SFR_Write(DSIM_CLKCNT,{16'h1000,Esc_prescaler}); //EscClk = ByteClk/Esc_prescaler;
			4. SFR_Write(DSIM_MDRESOL,{M_vert,M_horizon});
			5. SFR_Write(DSIM_MVPORCH,M_Vback);
			6. SFR_Write(DSIM_MHPORCH,{M_Hfront,M_Hback});
			7. SFR_Write(DSIM_MBLANK,{M_SBLLP,M_LBLLP});
			8. SFR_Write(DSIM_MSYNC,{M_VSA,M_HSA}); // In SyncEvent mode, Ignore it
			9. SFR_Write(DSIM_CONFIG,{HFPdismode, MainVC,MPix, SubVC,SPix,Sync_inform,BLLP,Burst, Videomode(1),NumofDatlane,Lane_EN); // set the value of each bit
			10. SFR_Write(DSIM_PLLCTRL, {4'h0, Freq_band,4'h8,PMS}); // PLL enable
			11. Wait for 100uS ~ 1mS using system timer after reset or power on.
			In this time, ForceSTOPstate at DSIM_CONFIG is assigned to D-PHY by default.
			100uS ~ 1mS is not mandatory. Please see the DSI slave spec.
			12. SFR_Write(DSIM_CONFIG, ForceSTOPstate(0)); // disable
			13. while(~SFR_Read(DSIM_STATUS, StopstateClk, StopstateDat));
			D-PHY & protocol layer(MIPI DSI Master block) in STOP state

Table continues on the next page...

Table 13-34. Behavior programming each case (continued)

			14. SFR_Write(DSIM_ESCMODE, TxUlps(1),TxUlpsClk(1)); D-PHY & protocol layer(MIPI DSI Master block) in ULPS In ULPS, User can turn off Byte/Esc clock for Low power estimation SFR_Write(DSIM_CLKCNT, ESCClk_EN(0), BytClk_En(0)); Or SFR_Write(DSIM_CLKCNT,ByteClk_EN(0),Lane_EscClkEn[4 :0](0)); If user wants to turn on display, Turn on EscClk and 15. SFR_Write(DSIM_ESCMODE, TxUlpsExit(1),TxUlpsClkExit(1)); 16. while(~SFR_Read(DSIM_STATUS, UlpsClk, UlpsDat)); 17. SFR_Write(DSIM_ESCMODE, TxUlpsExit(0), TxUlpsClkExit(0)); 18. Wait for about 1mS using system timer. 1mS is not mandatory, just an example. Please see the DSI slave spec. 19. SFR_Write(DSIM_ESCMODE, TxUlps(0),TxUlpsClk(0)); 20. while(~SFR_Read(DSIM_STATUS, StopstateClk, StopstateDat)); D-PHY & protocol layer(MIPI DSI Master block) in STOP state 21. MIPI DSI Master is ready for HS/LP operation.
2	Video	HS Data Transfer	In Video mode, there is no pseudo code. After initializing, SFR_Write(Display Controller_register, START); If line image is full in line buffer, MIPI DSI master transmit data through long data packet. If user wants to send general command to display or transmit image to sub display from memory, user will have to set following sequence. 1. SFR_Write(DSIM_PAYLOAD,PAYLOAD_DATA); 2. SFR_Write(DSIM_PKTHDR,PKTHDR_DATA); If user wants to send command to Sub display, User shall set “Sub display Virtual Channel field” in Packet Header(PKTHDR[7:6]).
3	Video	LP Data Transfer	If user wants to send general command to display or transmit image to sub display from memory via LPDT, 1. SFR_Write(DSIM_ESCMODE, TxLpdt); 2. If DSIM transfer data in HS mode, DSIM will wait STOPstate and transfer mode change HS to LP. 3. same as the previous test scenario cases. If user wants to send command to Sub display, User shall set “Sub display Virtual Channel field” in Packet Header(PKTHDR[7:6]).
4	Video	ULPS	ULPS sequence is same the initialization process (test scenario #1), 13~20
5	Video	Trigger function	The process is as follows: 1. SFR_Write(DSIM_ESCMODE, TxTriggerRst); 2. If DSI master state is not STOP state, DSIM will wait for LP stop mode and transmit trigger automatically.
6	Video	BTA	There are two cases of BTA. One case is after transferring general read data packet or DCS command write(set_tear_on) and the other one is BTA_req bit setting by software.

Table continues on the next page...

Table 13-34. Behavior programming each case (continued)

			Previous case is automatically setting BTA. Next case is as follows: If user reads the state or data of display panel, user will use BTA. 1. SFR_Write(DSIM_ESCMODE, BTA); 2. If DSIM transfer data, DSIM will wait STOPstate. 3. After BTA assert, DSIM release DSI channel. In reverse direction (DSI slave has DSI channel grant), all DSIM FIFO is flushed and masked Video/SFR inputs. Only LPRX data input is unmasked. 4. If DSIS releases DSI channel to DSIM, Int_BusTurnOver is generated. After ISR, DSIM can transfer data.

7	Command	Initialize	1. reset or power on
			2. SFR_Write(Display Controller_registers, S-i80 I/F_888);
			3. SFR_Write(DSIM_CLKCNT,{16'h1000,Esc_prescaler});
			4. SFR_Write(DSIM_MDRESOL,{M_vert,M_horizon});
			5. SFR_Write(DSIM_SDRESOL,{S_vert,S_horizon});
			6. SFR_Write(DSIM_CONFIG,{ MainVC,MPix,SubVC,SPix,
			Sync_inform,NumofDatlane,Videomode(0), Lane_EN); // set the value of each bit
			7. SFR_Write(DSIM_PLLCTRL, {4'h0, Freq_band,4'h8,PMS}); // PLL enable
			8. Wait for 100uS ~ 1mS using system timer after reset or power on. In this time, ForceSTOPstate at DSIM_CONFIG is assigned to D-PHY by default. 100uS ~ 1mS is not mandatory. Please see the DSI slave spec.
			9. SFR_Write(DSIM_CONFIG, ForceSTOPstate(0)); // disable
			10. while(~SFR_Read(DSIM_STATUS, StopstateClk, StopstateDat));
			D-PHY & protocol layer(MIPI DSI Master block) in STOP state
			11. SFR_Write(DSIM_ESCMODE, TxUlps(1),TxUlpsClk(1));
			D-PHY & protocol layer(MIPI DSI Master block) in ULPS
			In ULPS, User can turn off Byte/Esc clock for Low power estimation
			SFR_Write(DSIM_CLKCNT, ESCClk_EN(0), BytClk_En(0));
			Or
			SFR_Write(DSIM_CLKCNT,ByteClk_EN(0),Lane_EscClkEn[4 :0](0));
			If user wants to turn on display, Turn on EscClk and
			12. SFR_Write(DSIM_ESCMODE, TxUlpsExit(1),TxUlpsClkExit(1));
			13. while(~SFR_Read(DSIM_STATUS, UlpsClk, UlpsDat));
			14. SFR_Write(DSIM_ESCMODE, TxUlpsExit(0), TxUlpsClkExit(0));
			15. Wait for about 1mS using system timer.
			1mS is not mandatory, just an example. Please see the DSI slave spec.
			16. SFR_Write(DSIM_ESCMODE, TxUlps(0),TxUlpsClk(0));
			17. while(~SFR_Read(DSIM_STATUS, StopstateClk, StopstateDat));
			D-PHY & protocol layer(MIPI DSI Master block) in STOP state

Table continues on the next page...

Table 13-34. Behavior programming each case (continued)

<table>
<tr><td></td><td></td><td></td><td>18. MIPI DSI Master is ready for HS/LP operation.</td></tr>
<tr><td>8</td><td>Command</td><td>HS Data Transfer</td><td>In Command mode, there is no pseudo code.
After initializing,
SFR_Write(Display Controller_register, START);
If line image is full in line buffer, MIPI DSI master transmit data through long data packet.
<table>
<tr><th>SYS_ ADD[1]</th><th>SYS_ ADD[0]</th><th>SYS_ CS0</th><th>SYS_ CS1</th><th>Contents</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>Main display Image</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>Sub display Image</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>Payload data</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>Packet Header data
User can control main display and sub display via virtual channel field in Packet Header.
(Virtual channel field : PKTHDR[7:6])</td></tr>
<tr><td colspan="4">The other cases</td><td>Reserved</td></tr>
</table>
If user wants to send general command to display or transmit image to sub display from memory, user will have to set following sequence.
SFR_Write(DSIM_PAYLOAD,PAYLOAD_DATA);
SFR_Write(DSIM_PKTHDR,PKTHDR_DATA);
If user wants to update partial image,
wait after Display Controller transfer last line.
wait(DSIM_INTSRC, Frame_done); or check STOPstate.
SFR_Write(DSIM_MDRESOL,{M_vert,M_horizon}); or
SFR_Write(DSIM_SDRESOL,{S_vert,S_horizon});
Turn on Display Controller for Image transfer
If user wants to send command to Sub display, User shall set “Sub display Virtual Channel field” in Packet Header(PKTHDR[7:6]).</td></tr>
<tr><td>9</td><td>Command</td><td>LP Data Transfer</td><td>If user wants to send general command to display or transmit image to sub display from memory via LPDT.
SFR_Write(DSIM_ESCMODE, TxLpdt);
If DSIM transfer data in HS mode, DSIM will wait STOPstate and transfer mode change HS to LP.
same as the previous test scenario cases.
If user wants to send command to Sub display, User shall set “Sub display Virtual Channel field” in Packet Header(PKTHDR[7:6]).</td></tr>
<tr><td>10</td><td>Command</td><td>ULPS</td><td>ULPS sequence is same the initialization process (test scenario #1), 13~20</td></tr>
<tr><td>11</td><td>Command</td><td>Trigger function</td><td>The process is as follows:</td></tr>
</table>

Table continues on the next page...

Table 13-34. Behavior programming each case (continued)

			SFR_Write(DSIM_ESCMODE, TxTriggerRst); If DSI master state is not STOP state, DSIM will wait for LP stop mode and transmit trigger automatically.
12	Command	BTA	There are two cases of BTA. One case is after transferring general read data packet or DCS command write(set_tear_on) and the other one is BTA_req bit setting by software. Previous case is automatically setting BTA. Next case is as follows: If user read the state or data of display panel, user will use BTA. SFR_Write(DSIM_ESCMODE, BTA); If DSIM transfer data, DSIM will wait STOPstate. After BTA assert, DSIM release DSI channel. In reverse direction (DSI slave has DSI channel grant), all DSIM FIFO is flushed and masked Video/SFR inputs. Only LPRX data input is unmasked. If DSIS releases DSI channel to DSIM, “Int_BusTurnOver” is generated. After ISR, DSIM can transfer data.

13	ISR
	(Error)
			SFR_Write(Global_INTC_MASK, INTMASK(DSIM));
			SFR_Read(Global_INTC_SRC);
			SFR_Write(DSIM_INTMSK, all_int_mask);
			SFR_Read(DSIM_INTSRC);
			Error reporting to application layer
			SFR_Write(DSIM_INTSRC, intsrc); // clear interrupt source
			SFR_Write(DSIM_INTMSK, all_int_unmask);
			SFR_Write(Global_INT_SRC, intsrc); // clear interrupt source
			SFR_Write(Global_INT_MASK, INTUNMASK(DSIM));
			SFR_Write(DSIM_CONFIG, ForceSTOPstate(1));
			enough time later
			SFR_Write(DSIM_CONFIG, ForceSTOPstate(0));

14	ISR
	(Rx Data)
			After BTA, wait interrupt “Int_BusTurnOver”. The process is as follows:
			SFR_Write(Global_INTC_MASK, INTMASK(DSIM));
			SFR_Read(Global_INTC_SRC);
			SFR_Write(DSIM_INTMSK, all_int_mask);
			SFR_Read(DSIM_INTSRC);
			SFR_Write(DSIM_INTSRC, {BusTurnOver,RxDone}); // clear interrupt source
			SFR_Write(DSIM_INTMSK, all_int_unmask);
			SFR_Write(Global_INT_SRC, intsrc); // clear interrupt source
			SFR_Write(Global_INT_MASK, INTUNMASK(DSIM));
			SFR_Read(DSIM_RXFIFO);

15	ISR (trigger flag)		After BTA, wait interrupt “Int_BusTurnOver”. The process is as follows: SFR_Write(Global_INTC_MASK, INTMASK(DSIM)); SFR_Read(Global_INTC_SRC);

Table continues on the next page...

Table 13-34. Behavior programming each case (continued)

			SFR_Write(DSIM_INTMSK, all_int_mask);
			SFR_Read(DSIM_INTSRC);
			flag reporting to application layer
			SFR_Write(DSIM_INTSRC, {BusTurnOver, trigflags}); // clear interrupt source
			SFR_Write(DSIM_INTMSK, all_int_unmask);
			SFR_Write(Global_INT_SRC, intsrc); // clear interrupt source
			SFR_Write(Global_INT_MASK, INTUNMASK(DSIM));

16	Software reset	Function reset	If you want to assert Software reset, all clocks must be turned on because software reset is a synchronous reset. Turn on RGB_VCLK only from Display controller SFR_Write(DSIM_PLLTMR, PllTimer); SFR_Write(DSIM_CLKCTRL, {EscClkEn, EscPrescaler}); SFR_Write(DSIM_PLLCTRL, {PllEn,PMS}); wait Pll stable (polling status register or ISR) SFR_Write(DSIM_SWRST, FuncRst); wait for software reset release (polling status register or ISR) ready for operation
		Software reset	If you want to assert Software reset, all clocks must be turned on because software reset is synchronous reset. Turn on RGB_VCLK only from Display controller SFR_Write(DSIM_PLLTMR, PllTimer); SFR_Write(DSIM_CLKCTRL, {EscClkEn, EscPrescaler}); SFR_Write(DSIM_PLLCTRL, {PllEn,PMS}); wait Pll stable (polling status register or ISR) SFR_Write(DSIM_SWRST, SwRst); wait software reset release (polling status register or ISR) re-configure all registers ready for operation
17	Clock source changing from PLL to external clock source		Turn on RGB_VCLK(only clock) from Display controller and STOP data inputting to MIPI DSIM SFR_Write(DSIM_SWRST, FuncRst); or SFR_Write(DSIM_SWRST, SwRst); Turn off any inputs in MIPI DSIM. SFR_Write(DSIM_PLLCTRL, PLL disable); SRF_Write(DSIM_CLKCNTRL, ByteClkSel); SFR_Write(DSIM_SWRST, FuncRst); or SFR_Write(DSIM_SWRST, SwRst); reconfiguration Turn on Display controller or input data to MIPI DSIM ready for operation
18	Usage of EoT packet		Video mode after turning on display controller FSM of Display controller I/F of MIPI DSIM generates EoT packet automatically

Table 13-34. Behavior programming each case

	when user wants to transfer command packet through HS mode.		Just write command or general packet in SFR FIFO Video mode after turning off/before turning on display controller Write command or general packet in SFR FIFO If user does not want to transfer general or command packet, user should write EoT packet in SFR FIFO. If general or command packets are transferred through LP mode, previous programming step is "don't care".

13.5.5 Register Map

This section includes the MIPI DSI module memory map and detailed descriptions of the registers.

13.5.5.1 MIPI DSI register descriptions

13.5.5.1.1 MIPI_DSI Memory map

MIPI_DSI base address: 32E1_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Specifies the DSI version register. (DSI_VERSION)	32	RO	0106_0100h
4h	Specifies the status register. (DSI_STATUS)	32	RO	0010_010Fh
8h	Specifies the RGB FSM status register. (DSI_RGB_STATUS)	32	RO	0000_0001h
Ch	Specifies the software reset register. (DSI_SWRST)	32	RW	0000_0000h
10h	Specifies the clock control register. (DSI_CLKCTRL)	32	RW	0000_FFFFh
14h	Specifies the time out register. (DSI_TIMEOUT)	32	RW	00FF_FFFFh
18h	Specifies the configuration register. (DSI_CONFIG)	32	RW	0200_0000h
1Ch	Specifies the escape mode register. (DSI_ESCMODE)	32	RW	0000_0000h
20h	Specifies the main display image resolution register. (DSI_MDRESOL)	32	RW	0300_0400h
24h	Specifies the main display Vporch register. (DSI_MVPORCH)	32	RW	F000_0000h
28h	Specifies the main display Hporch register. (DSI_MHPORCH)	32	RW	0000_0000h
2Ch	Specifies the main display Sync Area register. (DSI_MSYNC)	32	RW	0000_0000h
30h	Specifies the sub display image resolution register. (DSI_SDRESOL)	32	RW	0300_0400h
34h	Specifies the interrupt source register. (DSI_INTSRC)	32	RW	0000_0000h
38h	Specifies the interrupt mask register. (DSI_INTMSK)	32	RW	BB37_FFFFh

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
3Ch	Specifies the packet header FIFO register. (DSI_PKTHDR)	32	WO	0000_0000h
40h	Specifies the payload FIFO register. (DSI_PAYLOAD)	32	WO	0000_0000h
44h	Specifies the read FIFO register. (DSI_RXFIFO)	32	RO	Table 13-208
48h	Specifies the FIFO threshold level register. (DSI_FIFOTHLD)	32	RW	0000_01FFh
4Ch	Specifies the FIFO status and control register. (DSI_FIFOCTRL)	32	RW	0155_551Fh
50h	Specifies the FIFO memory AC characteristic register. (DSI_MEMACCHR)	32	RW	0000_4040h
78h	Specifies the Multi Packet, Packet Go register. (DSI_MULTI_PKT)	32	RW	0000_0002h
90h	Specifies the 1Gbps D-PHY PLL control register. (DSI_PLLCTRL_1G)	32	RW	0000_0000h
94h	Specifies the PLL control register. (DSI_PLLCTRL)	32	RW	0000_0000h
98h	Specifies the PLL control register 1. (DSI_PLLCTRL1)	32	RW	0000_0000h
9Ch	Specifies the PLL control register 2. (DSI_PLLCTRL2)	32	RW	0000_0000h
A0h	Specifies the PLL timer register. (DSI_PLLTMR)	32	RW	FFFF_FFFFh
A4h	Specifies the D-PHY control register 1. (DSI_PHYCTRL_B1)	32	RW	0000_0000h
A8h	Specifies the D-PHY control register 2. (DSI_PHYCTRL_B2)	32	RW	0000_0000h
ACh	Specifies the D-PHY control register 1. (DSI_PHYCTRL_M1)	32	RW	0000_0000h
B0h	Specifies the D-PHY control register 2. (DSI_PHYCTRL_M2)	32	RW	0000_0000h
B4h	Specifies the D-PHY timing register. (DSI_PHYTIMING)	32	RW	0000_0000h
B8h	Specifies the D-PHY timing register 1. (DSI_PHYTIMING1)	32	RW	0000_0000h
BCh	Specifies the D-PHY timing register 2. (DSI_PHYTIMING2)	32	RW	0000_0000h

13.5.5.1.2 Specifies the DSI version register. (DSI_VERSION)

13.5.5.1.2.1 Offset

Register	Offset
DSI_VERSION	0h

13.5.5.1.2.2 Function

Specifies the DSI version register.

13.5.5.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Version															
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Version															
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

13.5.5.1.2.4 Fields

Field	Function
31-0 Version	Specifies the DSI version information

13.5.5.1.3 Specifies the status register. (DSI_STATUS)

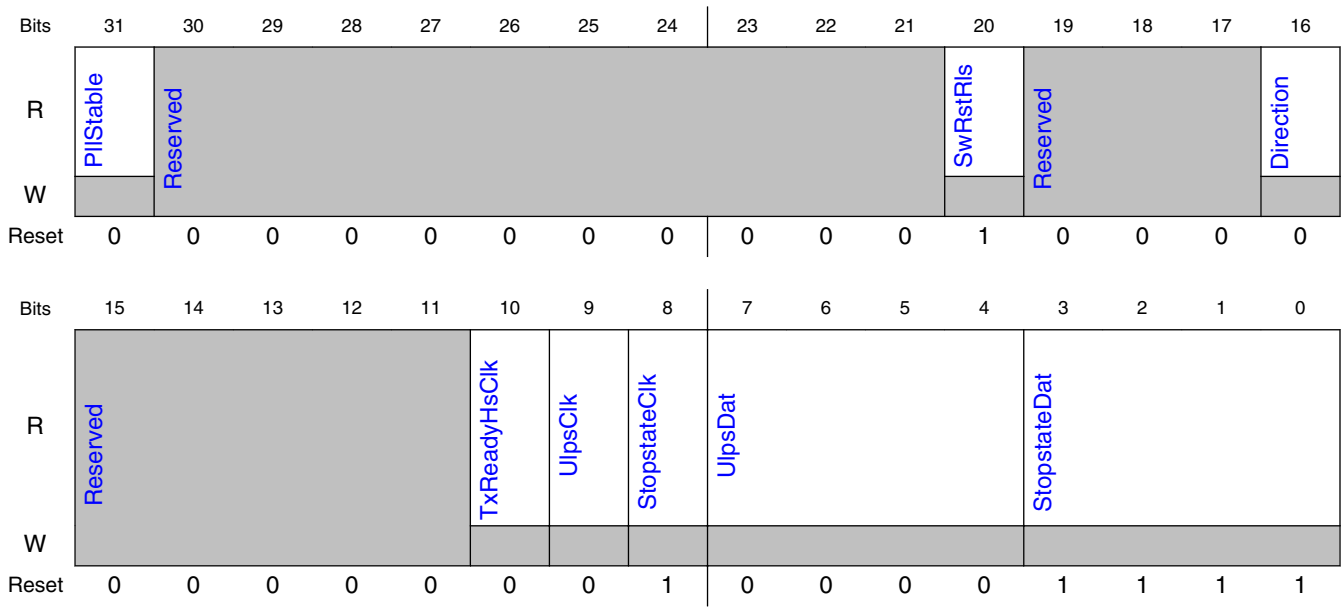
13.5.5.1.3.1 Offset

Register	Offset
DSI_STATUS	4h

13.5.5.1.3.2 Function

This register reads and checks internal and interface status. It also checks FSM status, Line buffer status, current image line number, and so on.

13.5.5.1.3.3 Diagram



13.5.5.1.3.4 Fields

Field	Function
31 PllStable	D-phy pll generates stable byteclk.
30-21 —	Reserved
20 SwRstRls	Specifies the software reset status. 0 = Reset state 1 = Release state
19-17 —	Reserved
16 Direction	Specifies the data direction indicator. 0 = Forward direction 1 = Backward direction
15-11 —	Reserved
10 TxReadyHsClk	Specifies the HS clock ready at clock lane. 0 = Not ready for transmitting HS data at clock lane 1 = Ready for transmitting HS data at clock lane
9 UlpsClk	Specifies the ULPS indicator at clock lane. 0 = No ULPS in clock lane 1 = ULSP in clock lane

Table continues on the next page...

Field	Function
8 StopstateClk	Specifies the stop state indicator at clock lane. 0 = No stop state in clock lane 1 = Stop state in clock lane
7-4 UlpsDat	Specifies the ULPS indicator at data lanes. UlpsDat[0]: Data lane 0 UlpsDat[1]: Data lane 1 UlpsDat[2]: Data lane 2 UlpsDat[3]: Data lane 3 0 = No ULPS in each data lane 1 = ULPS in each data lane
3-0 StopstateDat	Specifies the stop state indicator at data lane. StopstateDat[0]: Data lane 0 StopstateDat[1]: Data lane 1 StopstateDat[2]: Data lane 2 StopstateDat[3]: Data lane 3 0 = No stop state in each data lane 1 = Stop state in each data lane

13.5.5.1.4 Specifies the RGB FSM status register. (DSI_RGB_STATUS)

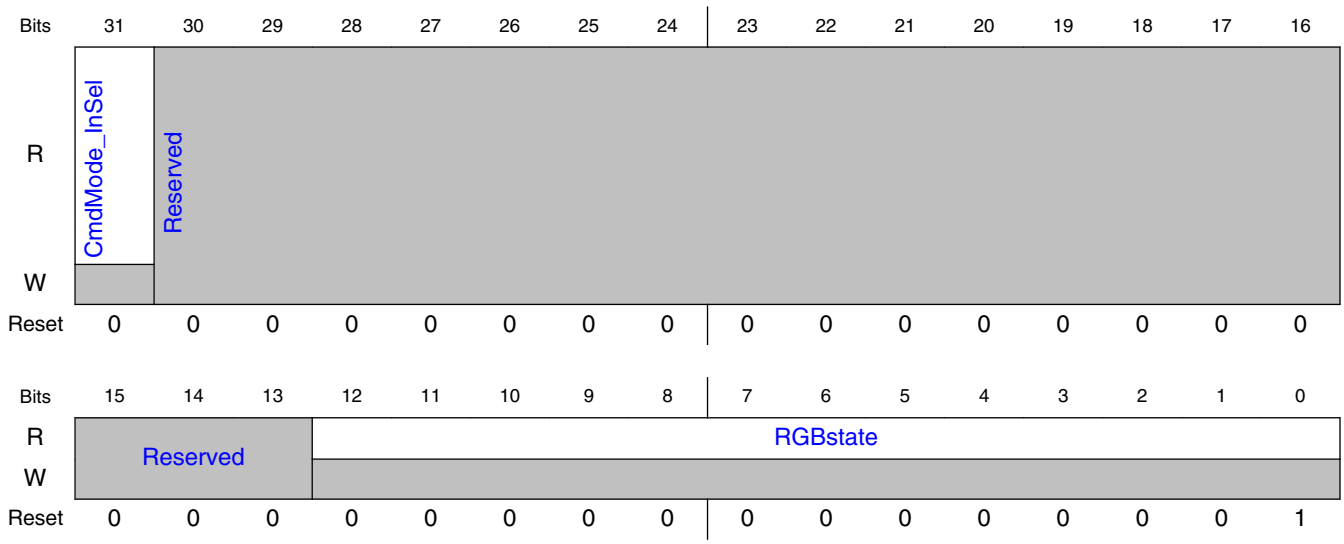
13.5.5.1.4.1 Offset

Register	Offset
DSI_RGB_STATUS	8h

13.5.5.1.4.2 Function

Specifies the RGB FSM status register.

13.5.5.1.4.3 Diagram



13.5.5.1.4.4 Fields

Field	Function
31 CmdMode_InSel	Specifies the command mode input selection 0 = using RGB video interface 1 = using S-i80 interface
30-13 —	Reserved
12-0 RGBstate	Specifies the RGB packetize FSM status. 0x0001 = IDLE 0x0002 = STOP 0x0004 = VSyS 0x0008 = VSE 0x0010 = HSS 0x0020 = HSA 0x0040 = HSE 0x0080 = HBP 0x0100 = RGB 0x0200 = NULL 0x0400 = HFP 0x0800 = EOT 0x1000 = NHOLD

13.5.5.1.5 Specifies the software reset register. (DSI_SWRST)

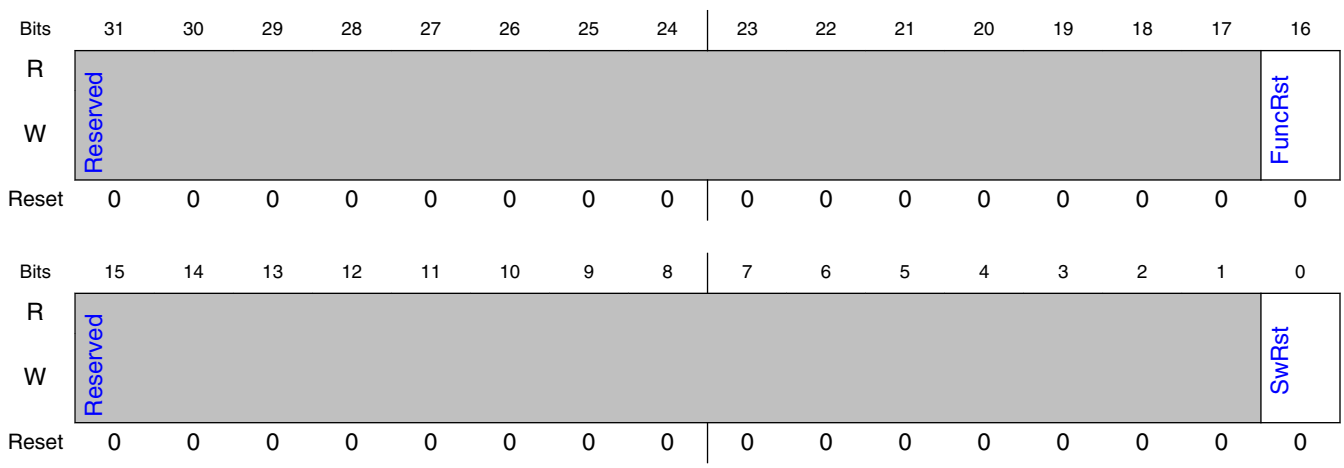
13.5.5.1.5.1 Offset

Register	Offset
DSI_SWRST	Ch

13.5.5.1.5.2 Function

Specifies the software reset register.

13.5.5.1.5.3 Diagram



13.5.5.1.5.4 Fields

Field	Function
31-17 —	Reserved
16 FuncRst	Specifies the software reset (High active). "Function reset" resets all FF in MIPI DSI (except SFRs: STATUS, RGB_STATUS, SWRST, CLKCTRL, TIMEOUT, CONFIG, ESCMODE[1], MDRESOL, MVPORCH, MHPORCH, MSYNC, INTMSK, SDRESOL, S3D_CTL, P3D_CTL, MIC_CTL, P3D_ON_MIC_OFF_HSA, P3D_OFF_MIC_ON_HSA, P3D_ON_MIC_ON_HSA, P3D_ON_MIC_OFF_HFP, P3D_OFF_MIC_ON_HFP, P3D_ON_MIC_ON_HFP, MULTI_PKT, FIFOTHLD, FIFOCTRL, [2], MEMACCHR, PLLCTRL, PLLTMR, PHYACCHR, and VERINFORM). 0 = Standby 1 = Reset
15-1	Reserved

Table continues on the next page...

Field	Function
—	
0 SwRst	Specifies the software reset (High active). “Software reset” resets all FF in MIPI DSI (except some SFRs: STATUS, SWRST, CLKCTRL, PLLCTRL, PLLTMR, PHYTUNE and VERINFORM). 0 = Standby 1 = Reset

1. ForceStopstate, CmdLpdt, TxLpdt
2. nInitRx, nInitSfr, nInitI80, nInitSub, nInitMD

13.5.5.1.6 Specifies the clock control register. (DSI_CLKCTRL)

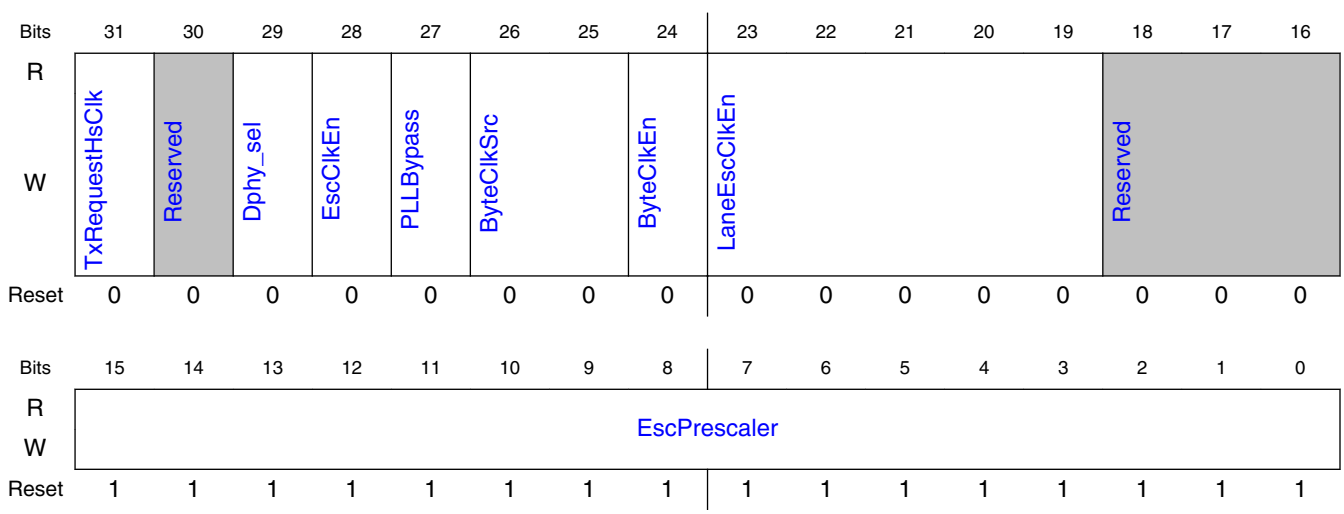
13.5.5.1.6.1 Offset

Register	Offset
DSI_CLKCTRL	10h

13.5.5.1.6.2 Function

Specifies the clock control register.

13.5.5.1.6.3 Diagram



13.5.5.1.6.4 Fields

Field	Function
31 TxRequestHsClk	Specifies the HS clock request for HS transfer at clock lane (Turn on HS clock)
30 —	Reserved
29 Dphy_sel	D-PHY select 0 = 1.5Gbps D-PHY (default) 1 = 1Gbps D-PHY
28 EscClkEn	Enables the escape clock generating prescaler. 0 = Disables 1 = Enables
27 PLLBypass	Sets the PLLBypass signal connected to D-PHY module input for selecting clock source bit. 0 = PLL output 1 = External Serial clock This bit must be set to 0.
26-25 ByteClkSrc	Selects byte clock source. (It must be 2'b00.) 00 = D-PHY PLL (default). PLL_out clock is used to generate ByteClk by dividing 8.
24 ByteClkEn	Enables byte clock. 0 = Disables 1 = Enables
23-19 LaneEscClkEn	Enables escape clock for D-phy lane. LaneEscClkEn[0] = Clock lane LaneEscClkEn[1] = Data lane 0 LaneEscClkEn[2] = Data lane 1 LaneEscClkEn[3] = Data lane 2 LaneEscClkEn[4] = Data lane 3 0 = Disables 1 = Enables
18-16 —	Reserved
15-0 EscPrescaler	Specifies the escape clock prescaler value. The escape clock frequency range varies up to 20MHz. **NOTE:** The requirement for BTA is that the Host Escclk frequency should range between 66.7 ~ 150% of the peripheral escape clock frequency. EscClk = ByteClk / (EscPrescaler)

13.5.5.1.7 Specifies the time out register. (DSI_TIMEOUT)

13.5.5.1.7.1 Offset

Register	Offset
DSI_TIMEOUT	14h

13.5.5.1.7.2 Function

Specifies the time out register.

13.5.5.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								BtaTout							
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LpdrTout															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

13.5.5.1.7.4 Fields

Field	Function
31-24 —	Reserved
23-16 BtaTout	Specifies the timer for BTA. This register specifies time out from BTA request to change the direction with respect to Tx escape clock.
15-0 LpdrTout	Specifies the timer for LP Rx mode timeout. This register specifies time out on how long RxValid deasserts, after RxLpdt asserts with respect to Tx escape clock. RxValid specifies Rx data valid indicator. RxLpdt specifies an indicator that D-phy is under RxLpdt mode. RxValid and RxLpdt specifies signal from D-phy.

13.5.5.1.8 Specifies the configuration register. (DSI_CONFIG)

13.5.5.1.8.1 Offset

Register	Offset
DSI_CONFIG	18h

13.5.5.1.8.2 Function

This register configures MIPI DSI master such as data lane number, input interface, porch area, frame rate, BTA, LPDT, ULPS, and so on.

The figure below shows the timing diagram of the clock lane:

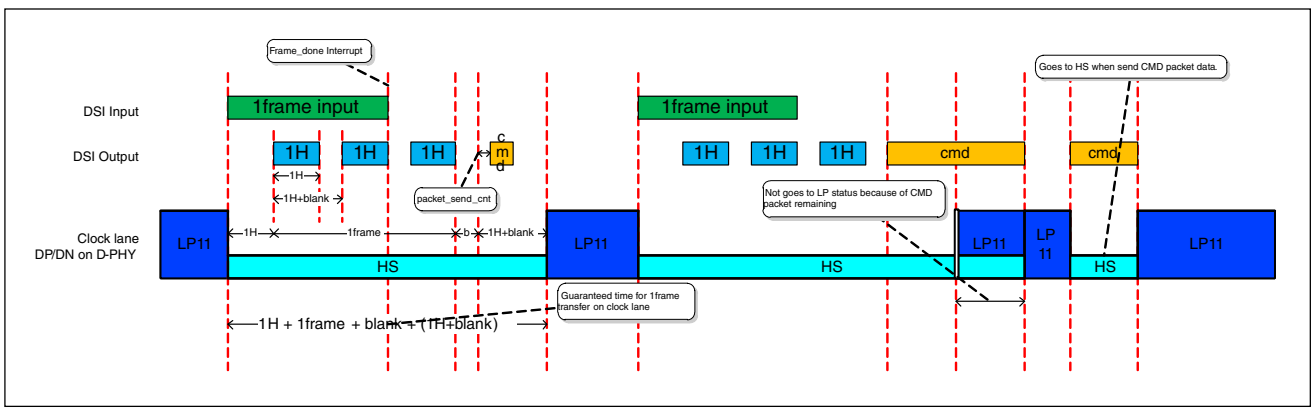
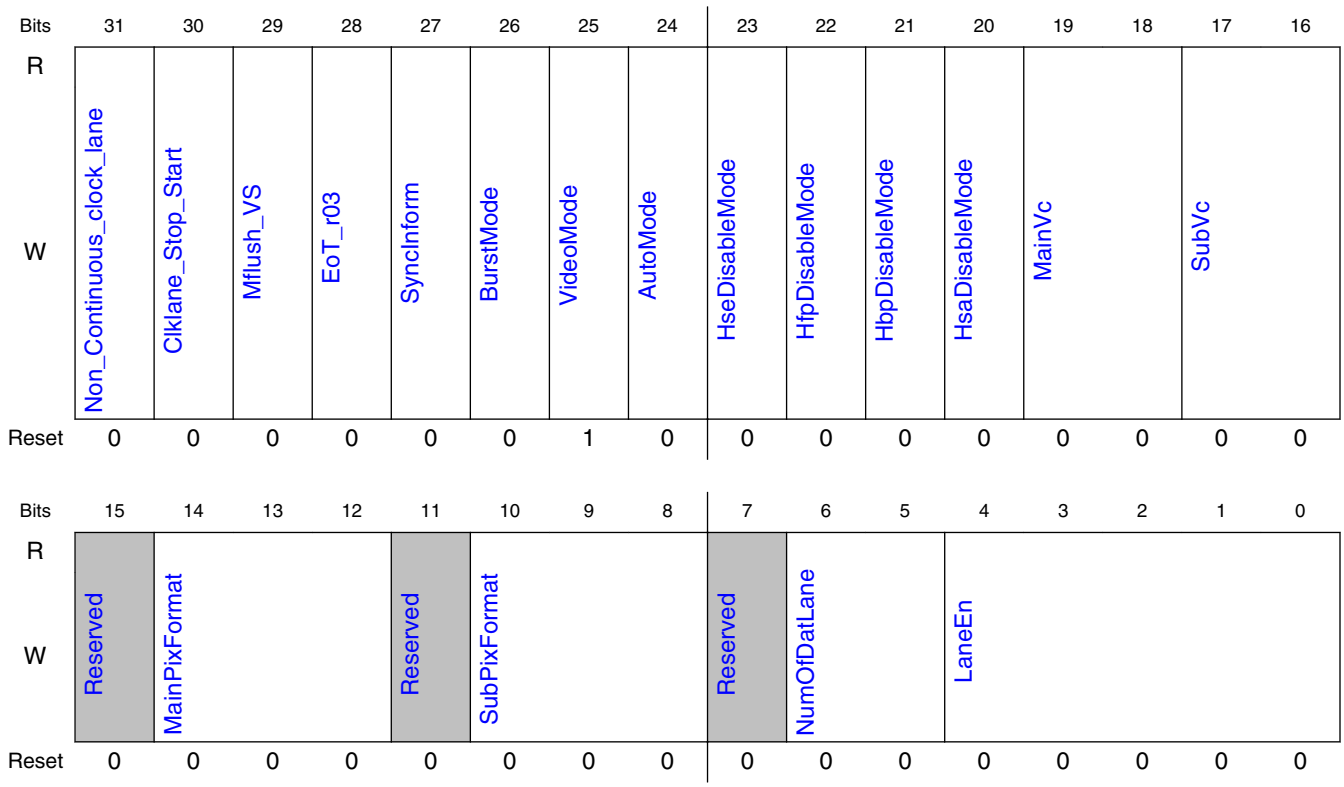


Figure 13-67. Non-continuous clock lane timing diagram

13.5.5.1.8.3 Diagram



13.5.5.1.8.4 Fields

Field	Function
31 Non_Continuous_clock_lane	Non-continuous clock mode. This feature is D-PHY clock lane control DP/DN Please refer to the diagram above to view the clock lane timing diagram. Please refer to the DSI_MULTI_PKT register, which shows the configuration of count values. 0 = Disable (default) 1 = Enable
30 Clklane_Stop_Start	PHY clock lane On/Off for ESD. It is operated when first HBP to second HFP in every VSYNC. This feature is not guaranteed on Command mode. 0 = Disable (default) 1 = Enable
29 Mflush_VS	Auto flush of MD FIFO using Vsync pulse. It needs that Main display FIFO should be flushed for deleting garbage data. 0 = Enable (default) 1 = Disable

Table continues on the next page...

Field	Function
28 EoT_r03	Disables EoT packet in HS mode. 0 = Enables EoT packet generation for V1.01r11 1 = Disables EoT packet generation for V1.01r03
27 SyncInform	Selects Sync Pulse or Event mode in Video mode. 0 = Event mode (non burst, burst) 1 = Pulse mode (non burst only) In command mode, this bit is ignored.
26 BurstMode	Selects Burst mode in Video mode In Non-burst mode, RGB data area is filled with RGB data and Null packets, according to input bandwidth of RGB interface. In Burst mode, RGB data area is filled with RGB data only. 0 = Non-burst mode 1 = Burst mode In command mode, this bit is ignored.
25 VideoMode	Specifies display configuration. 0 = Command mode 1 = Video mode
24 AutoMode	Specifies auto vertical count mode. In Video mode, the vertical line transition uses line counter configured by VSA, VBP, and Vertical resolution. If this bit is set to ‘1’, the line counter does not use VSA and VBP registers. 0 = Configuration mode 1 = Auto mode In command mode, this bit is ignored.
23 HseDisableMode	In Vsync pulse and Vporch area, MIPI DSI master transfers only Hsync start packet to MIPI DSI slave at MIPI DSI spec 1.1r02. This bit transfers Hsync end packet in Vsync pulse and Vporch area (optional). 0 = Disables transfer 1 = Enables transfer In command mode, this bit is ignored.
22 HfpDisableMode	Specifies HFP disable mode. If this bit set, DSI master ignores HFP area in Video mode. 0 = Enables 1 = Disables In command mode, this bit is ignored.
21 HbpDisableMode	Specifies HBP disable mode. If this bit set, DSI master ignores HBP area in Video mode. 0 = Enables 1 = Disables In command mode, this bit is ignored.
20 HsaDisableMode	Specifies HSA disable mode. If this bit set, DSI master ignores HSA area in Video mode. 0 = Enables 1 = Disables In command mode, this bit is ignored.

Table continues on the next page...

Field	Function
19-18 MainVc	Specifies virtual channel number for main display.
17-16 SubVc	Specifies virtual channel number for sub display.
15 —	Reserved
14-12 MainPixFormat	Specifies pixel stream format for main display. 000 = 3bpp (for Command mode only) 001 = 8bpp (for Command mode only) 010 = 12bpp (for Command mode only) 011 = 16bpp (for Command mode only) 100 = 16-bit RGB (565) (for Video mode only) 101 = 18-bit RGB (666: packed pixel stream) (for Video mode only) 110 = 18-bit RGB (666: loosely packed pixel stream) for Common 111 = 24-bit RGB (888) for common
11 —	Reserved
10-8 SubPixFormat	Specifies pixel stream format for sub display. 000 = 3bpp (for Command mode only) 001 = 8bpp (for Command mode only) 010 = 12bpp (for Command mode only) 011 = 16bpp (for Command mode only) 100 = 16-bit RGB (565) (for Video mode only) 101 = 18-bit RGB (666: packed pixel stream) for Video mode only) 110 = 18-bit RGB (666: loosely packed pixel stream) for common) 111 = 24-bit RGB (888) (for Common)
7 —	Reserved
6-5 NumOfDatLane	Sets the data lane number. 00 = Data lane 0 (1 data lane) 01 = Data lane 0 ~ 1 (2 data lanes) 10 = Data lane 0 ~ 2 (3 data lanes) 11 = Data lane 0 ~ 3 (4 data lanes)
4-0 LaneEn	Enables the lane. If Lane_EN is disabled, the lane ignores input and drives initial value through output port. 0 = Lane is off. 1 = Lane is on. LaneEn[0] = Clock lane enabler LaneEn[1] = Data lane 0 enabler

Field	Function
	LaneEn[2] = Data lane 1 enabler LaneEn[3] = Data lane 2 enabler LaneEn[4] = Data lane 3 enabler

13.5.5.1.9 Specifies the escape mode register. (DSI_ESCMODE)

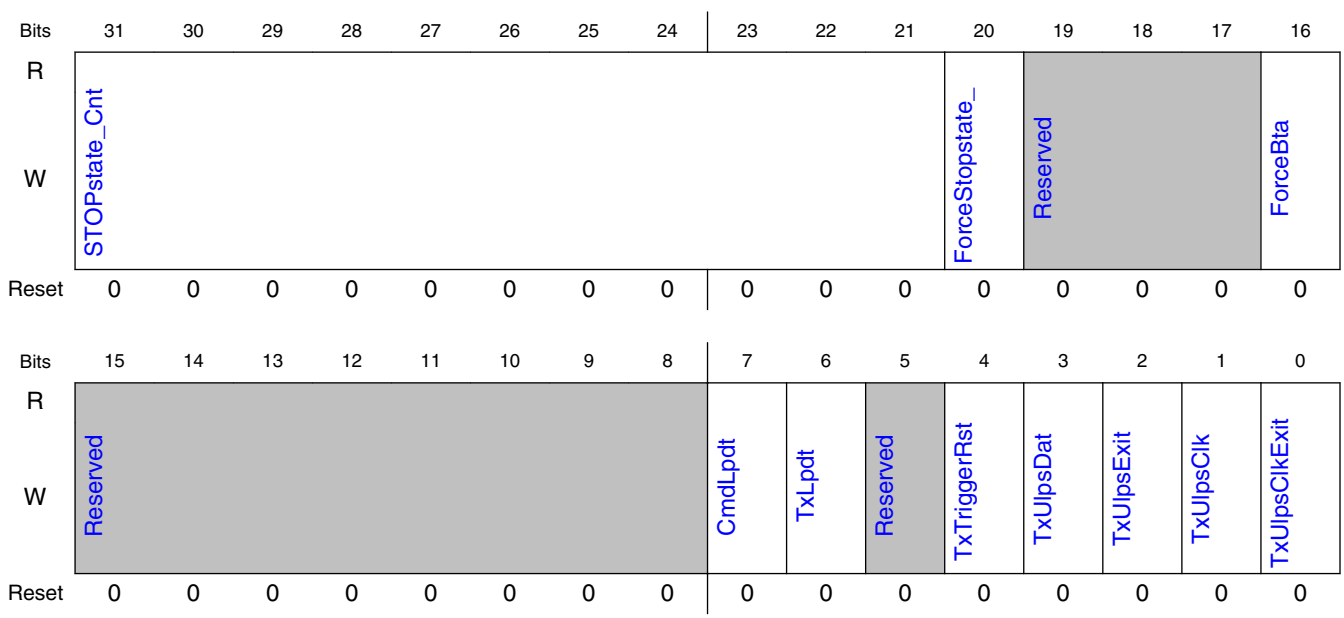
13.5.5.1.9.1 Offset

Register	Offset
DSI_ESCMODE	1Ch

13.5.5.1.9.2 Function

This register configures MIPI DSI master.

13.5.5.1.9.3 Diagram



13.5.5.1.9.4 Fields

Field	Function
31-21 STOPstate_Cnt	After transmitting read packet or write “set_tear_on” command, BTA requests to D-phy automatically. This counter value specifies the interval value between transmitting read packet (or write “set_tear_on” command) and BTA request. 11'h000 = 2 EscClk 11'h001 = 2 EscClk + 1 EscClk 11'h3FF = 2 EscClk + 1023 EscClk
20 ForceStopstate_	Forces Stopstate for D-PHY.
19-17 —	Reserved
16 ForceBta	Forces Bus Turn Around. 1 = Sends the protocol layer request to D-PHY. MIPI DSI peripheral becomes master after BTA sequence. This bit clears automatically after receiving BTA acknowledge from MIPI DSI peripheral.
15-8 —	Reserved
7 CmdLpdt	Specifies LPDT transfers command in SFR FIFO. 0 = HS Mode 1 = LP Mode
6 TxLpdt	Specifies data transmission in LP mode (all data transfer in LPDT). 0 = HS Mode 1 = LP Mode
5 —	Reserved
4 TxTriggerRst	Specifies remote reset trigger function. After trigger operation, these bits will be cleared automatically.
3 TxUlpsDat	Specifies ULPS request for data lane. Manually clears after ULPS exit.
2 TxUlpsExit	Specifies ULPS exit request for data lane. Manually clears after ULPS exit.
1 TxUlpsClk	Specifies ULPS request for clock lane. Manually clears after ULPS exit.
0 TxUlpsClkExit	Specifies ULPS exit request for clock lane. Manually clears after ULPS exit.

13.5.5.1.10 Specifies the main display image resolution register. (DSI_MDRESOL)

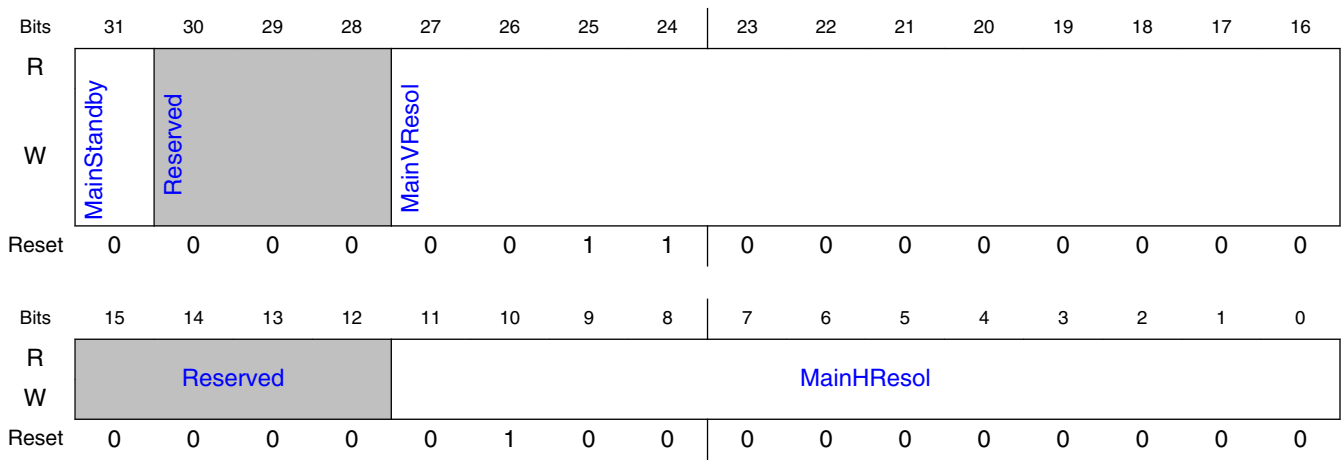
13.5.5.1.10.1 Offset

Register	Offset
DSI_MDRESOL	20h

13.5.5.1.10.2 Function

Specifies the main display image resolution register.

13.5.5.1.10.3 Diagram



13.5.5.1.10.4 Fields

Field	Function
31 MainStandby	Specifies standby for receiving DISPCON output in Command mode after setting all configuration. 0 = Not ready 1 = Stand by Standby should be set after configuration (resolution, reqtype, pixelform, and so on) is set for command mode. In Video mode, if this bit value is 0, data is not transferred.
30-28 —	Reserved
27-16 MainVResol	Specifies Vertical resolution (1 ~ 2047).
15-12 —	Reserved
11-0 MainHResol	Specifies Horizontal resolution (1 ~ 2047).

13.5.5.1.11 Specifies the main display Vporch register. (DSI_MVPORCH)

13.5.5.1.11.1 Offset

Register	Offset
DSI_MVPORCH	24h

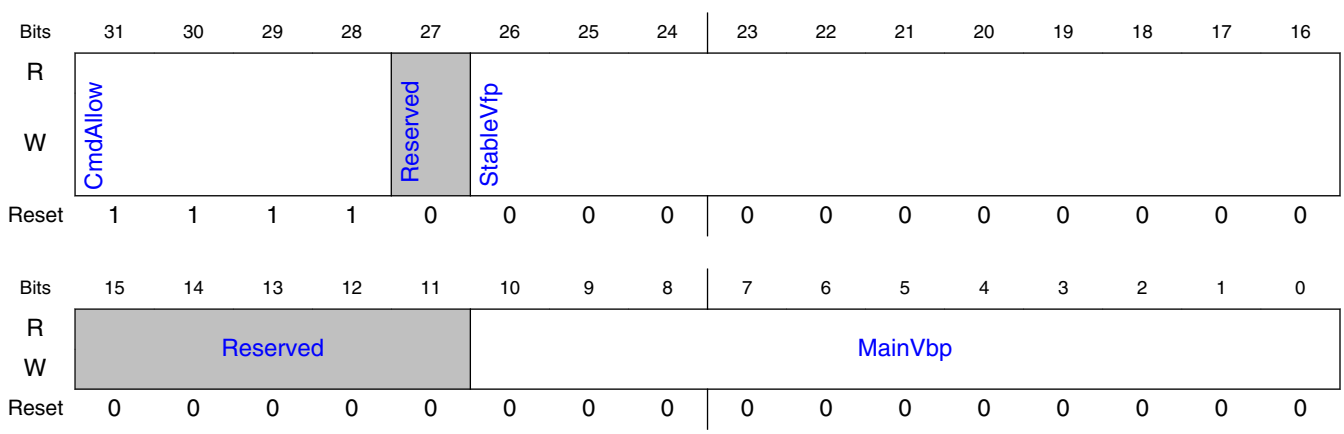
13.5.5.1.11.2 Function

Specifies the main display Vporch register.

NOTE

Transfers command packets after Stable VFP area. Display controller VFP lines should be set based on sum of these values: Stable VFP, command allowing area and command masked area. Please refer to Transfer general data in video mode for more information

13.5.5.1.11.3 Diagram



13.5.5.1.11.4 Fields

Field	Function
31-28 CmdAllow	Specifies the number of horizontal lines, where command packet transmission is allowed after Stable VFP period. For more information, please see Figure 13-57.
27 —	Reserved

Table continues on the next page...

Field	Function
26-16 StableVfp	Specifies the number of horizontal lines, where command packet transmission is not allowed after end of active region. For more information, please see Figure 13-57. **NOTE:** In Command mode, these bits are ignored.
15-11 —	Reserved
10-0 MainVbp	Specifies vertical back porch width for Video mode (line count). In Command mode, these bits are ignored.

13.5.5.1.12 Specifies the main display Hporch register. (DSI_MHPORCH)

13.5.5.1.12.1 Offset

Register	Offset
DSI_MHPORCH	28h

13.5.5.1.12.2 Function

Specifies the main display Hporch register.

13.5.5.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MainHfp															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MainHbp															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.12.4 Fields

Field	Function
31-16 MainHfp	Specifies the horizontal front porch width for Video mode. HFP is specified using blank packet. These bits specify the word counts for blank packet in HFP. In Command mode, these bits are ignored.
15-0 MainHbp	Specifies the horizontal back porch width for Video mode. HBP is specified using blank packet. These bits specify the word counts for blank packet in HBP. In Command mode, these bits are ignored.

13.5.5.1.13 Specifies the main display Sync Area register. (DSI_MSYNC)

13.5.5.1.13.1 Offset

Register	Offset
DSI_MSYNC	2Ch

13.5.5.1.13.2 Function

Specifies the main display Sync Area register.

13.5.5.1.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MainVsa										Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MainHsa															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.13.4 Fields

Field	Function
31-22 MainVsa	Specifies the vertical sync pulse width for Video mode (Line count). In command mode, these bits are ignored.
21-16 —	Reserved
15-0 MainHsa	Specifies the horizontal sync pulse width for Video mode. HSA is specified using blank packet. These bits specify word counts for blank packet in HSA. In command mode, these bits are ignored.

13.5.5.1.14 Specifies the sub display image resolution register. (DSI_SDRESOL)

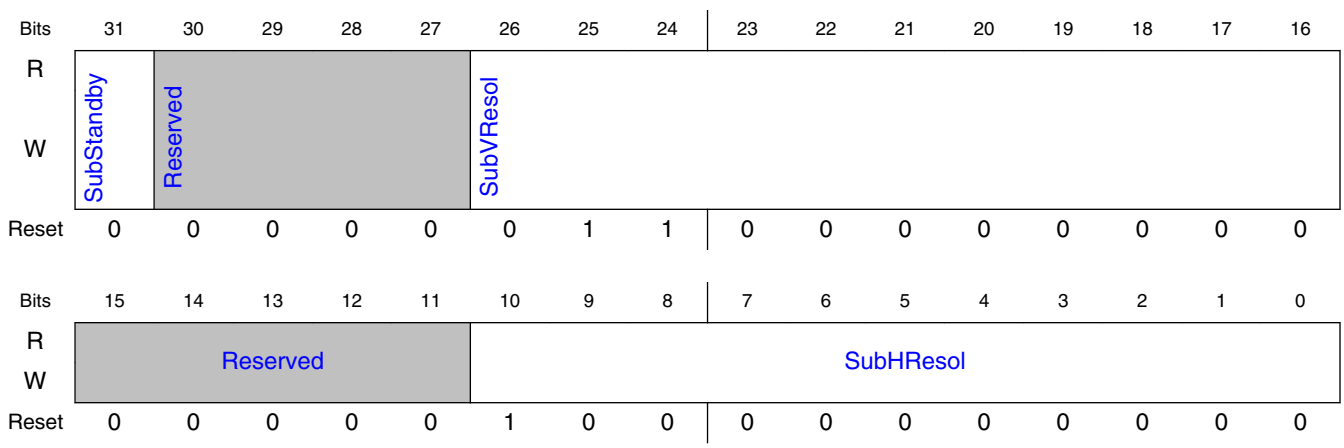
13.5.5.1.14.1 Offset

Register	Offset
DSI_SDRESOL	30h

13.5.5.1.14.2 Function

Specifies the sub display image resolution register.

13.5.5.1.14.3 Diagram



13.5.5.1.14.4 Fields

Field	Function
31 SubStandby	Specifies standby for receiving DISPCON output in Command mode after setting all configuration. 0 = Not ready 1 = Standby Standby should be set after configuration (resolution, reqtype, pixelform, and so on) is set for command mode. In Video mode, this bit is ignored.
30-27 —	Reserved
26-16 SubVResol	Specifies the Vertical resolution (1 ~ 1024).
15-11	Reserved

Table continues on the next page...

Field	Function
—	
10-0 SubHResol	Specifies the Horizontal resolution (1 ~ 1024).

13.5.5.1.15 Specifies the interrupt source register. (DSI_INTSRC)

13.5.5.1.15.1 Offset

Register	Offset
DSI_INTSRC	34h

13.5.5.1.15.2 Function

This register identifies interrupt sources. Internal block error, data transmit interrupt, inter-layer (D_PHY) error, etc.

The bits are set even if they are masked off by DSI_INTMSK_REG.

Write '1' to clear the Interrupt.

13.5.5.1.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PllStable	SwRstRelease	SFRPLFifoEmpty	SFRPHFifoEmpty	SyncOverride	Reserved	BusTurnOver	FrameDone	Reserved		LpdrTout	TaTout	Reserved	RxDatDone	RxTE	RxAck
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ErrRxECC	ErrRxCRC	ErrEsc3	ErrEsc2	ErrEsc1	ErrEsc0	ErrSync3	ErrSync2	ErrSync1	ErrSync0	ErrControl3	ErrControl2	ErrControl1	ErrControl0	ErrContentLP0	ErrContentLP1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.15.4 Fields

Field	Function
31 PllStable	Indicates that D-phy PLL is stable.
30 SwRstRelease	Releases the software reset.
29 SFRPLFifoEmpty	Specifies the SFR payload FIFO empty.
28 SFRPHFifoEmpty	Specifies the SFR Packet Header FIFO empty
27 SyncOverride	Indicates that other DSI command transfer have overridden sync timing.
26 —	Reserved
25 BusTurnOver	Indicates when bus grant turns over from DSI slave to DSI master.
24 FrameDone	Indicates when MIPI DSI transfers the whole image frame. **NOTE:** If Hsync is not received during two line times, internal timer is timed out and this bit is flagged.
23-22 —	Reserved
21 LpdrTout	Specifies the LP Rx timeout. Please refer to the DSI_TIMEOUT register.
20 TaTout	Turns around Acknowledge Timeout. Please refer to the DSI_TIMEOUT register.
19 —	Reserved
18 RxDatDone	Completes receiving data.
17 RxTE	Receives TE Rx trigger.
16 RxAck	Receives ACK Rx trigger.
15 ErrRxECC	Specifies the ECC multi bit error in LPDR.
14 ErrRxCRC	Specifies the CRC error in LPDR.
13 ErrEsc3	Specifies the escape mode entry error lane 3. For more information, please refer to the standard D-PHY specification.

Table continues on the next page...

Field	Function
12 ErrEsc2	Specifies the escape mode entry error lane 2. For more information, please refer to the standard D-PHY specification.
11 ErrEsc1	Specifies the escape mode entry error lane 1. For more information, please refer to the standard D-PHY specification.
10 ErrEsc0	Specifies the escape mode entry error lane 0. For more information, please refer to the standard D-PHY specification.
9 ErrSync3	Specifies the LPDT sync error lane 3. For more information, please refer to the standard D-PHY specification.
8 ErrSync2	Specifies the LPDT Sync Error lane2. For more information, refer to standard D-PHY specification.
7 ErrSync1	Specifies the LPDT Sync Error lane1. For more information, refer to standard D-PHY specification.
6 ErrSync0	Specifies the LPDT Sync Error lane0. For more information, refer to standard D-PHY specification.
5 ErrControl3	Controls Error lane3. For more information, please refer to the standard D-PHY specification.
4 ErrControl2	Controls Error lane2. For more information, please refer to the standard D-PHY specification.
3 ErrControl1	Controls Error lane1. For more information, please refer to the standard D-PHY specification.
2 ErrControl0	Controls Error lane0. For more information, please refer to the standard D-PHY specification.
1 ErrContentLP0	Specifies the LP0 Contention Error (only lane0, because BTA occurs at lane0 only). For more information, please refer to the standard D-PHY specification.
0 ErrContentLP1	Specifies the LP1 Contention Error (only lane0, because BTA occurs at lane0 only). For more information, please refer to the standard D-PHY specification.

13.5.5.1.16 Specifies the interrupt mask register. (DSI_INTMSK)

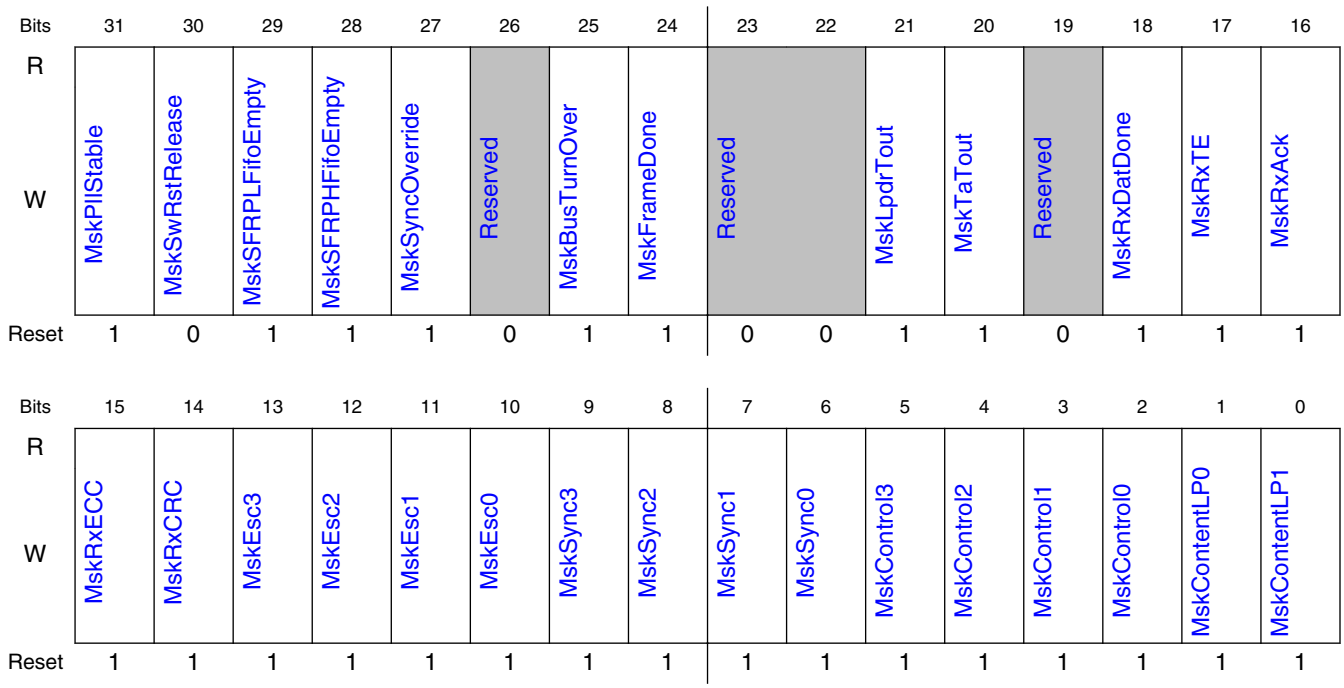
13.5.5.1.16.1 Offset

Register	Offset
DSI_INTMSK	38h

13.5.5.1.16.2 Function

This register masks interrupt sources.

13.5.5.1.16.3 Diagram



13.5.5.1.16.4 Fields

Field	Function
31 MskPllStable	Indicates that D-PHY PLL is stable.
30 MskSwRstRelease	Releases software reset.
29 MskSFRPLFifoEmpty	Empties SFR payload FIFO.
28 MskSFRPHFifoEmpty	Interrupt Mask for SFR packet header FIFO empty
27 MskSyncOverride	Indicates that other DSI command transfer have overridden sync timing.
26 —	Reserved
25 MskBusTurnOver	Indicates when bus grant turns over from DSI slave to DSI master.

Table continues on the next page...

Field	Function
24 MskFrameDone	Indicates when MIPI DSI transfers whole image frame.
23-22 —	Reserved
21 MskLpdrTout	Specifies LP Rx timeout. Please refer to the DSI_TIMEOUT register.
20 MskTaTout	Specifies turnaround acknowledge timeout. Please refer to the DSI_TIMEOUT register.
19 —	Reserved
18 MskRxDatDone	Specifies completion of data receiving.
17 MskRxTE	Specifies receipt of TE Rx trigger.
16 MskRxAck	Specifies receipt of ACK Rx trigger.
15 MskRxECC	Specifies ECC multibit error in LPDR.
14 MskRxCRC	Specifies CRC error in LPDR.
13 MskEsc3	Specifies escape mode entry error in lane3. For more information, please refer to the standard D-PHY specification.
12 MskEsc2	Specifies escape mode entry error in lane2. For more information, please refer to the standard D-PHY specification.
11 MskEsc1	Specifies escape mode entry error in lane1. For more information, please refer to the standard D-PHY specification.
10 MskEsc0	Specifies escape mode entry error in lane0. For more information, please refer to the standard D-PHY specification.
9 MskSync3	Specifies LPDT sync error in lane3. For more information, refer to standard D-PHY specification.
8 MskSync2	Specifies LPDT sync error in lane2. For more information, refer to standard D-PHY specification.
7 MskSync1	Specifies LPDT sync error in lane1. For more information, refer to standard D-PHY specification.
6 MskSync0	Specifies LPDT sync error in lane0. For more information, refer to standard D-PHY specification.
5 MskControl3	Controls error in lane3. For more information, please refer to the standard D-PHY specification.
4	Controls error in lane2. For more information, please refer to the standard D-PHY specification.

Table continues on the next page...

Field	Function
MskControl2	
3 MskControl1	Controls error in lane1. For more information, please refer to the standard D-PHY specification.
2 MskControl0	Controls error in lane0. For more information, please refer to the standard D-PHY specification.
1 MskContentLP0	Specifies LP0 contention error. For more information, refer to standard D-PHY specification.
0 MskContentLP1	Specifies LP1 contention error. For more information, refer to standard D-PHY specification.

13.5.5.1.17 Specifies the packet header FIFO register. (DSI_PKTHDR)

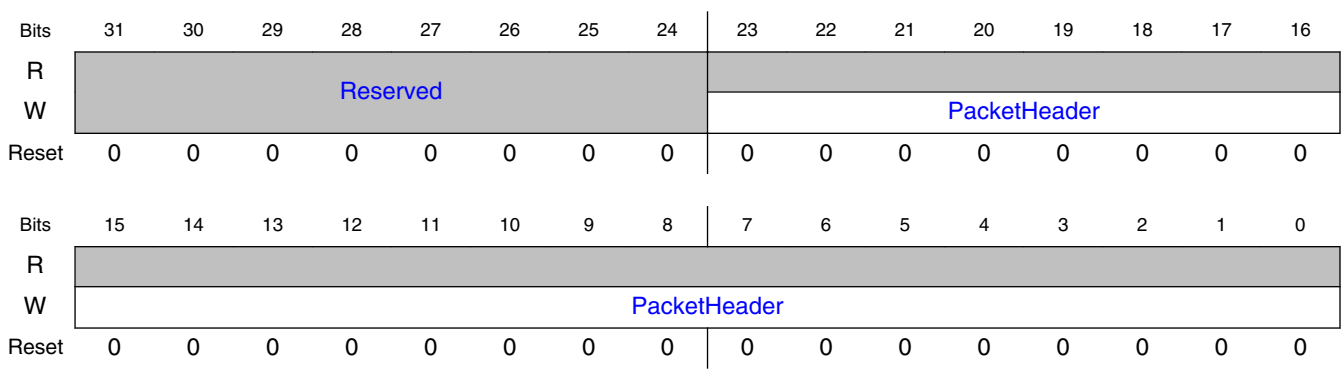
13.5.5.1.17.1 Offset

Register	Offset
DSI_PKTHDR	3Ch

13.5.5.1.17.2 Function

This register is the FIFO for packet header to send DSI packets.

13.5.5.1.17.3 Diagram



13.5.5.1.17.4 Fields

Field	Function
31-24	Reserved

Table continues on the next page...

Field	Function
—	
23-0 PacketHeader	Writes the packet header of Tx packet. [7:0] = DI [15:8] = Dat0 (Word Count lower byte for long packet) [23:16] = Dat1 (Word Count upper byte for long packet)

13.5.5.1.18 Specifies the payload FIFO register. (DSI_PAYLOAD)

13.5.5.1.18.1 Offset

Register	Offset
DSI_PAYLOAD	40h

13.5.5.1.18.2 Function

This register specifies the FIFO for payload to send DSI packets.

13.5.5.1.18.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	Payload															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	Payload															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.18.4 Fields

Field	Function
31-0 Payload	Writes the Payload of Tx packet.

13.5.5.1.19 Specifies the read FIFO register. (DSI_RXFIFO)

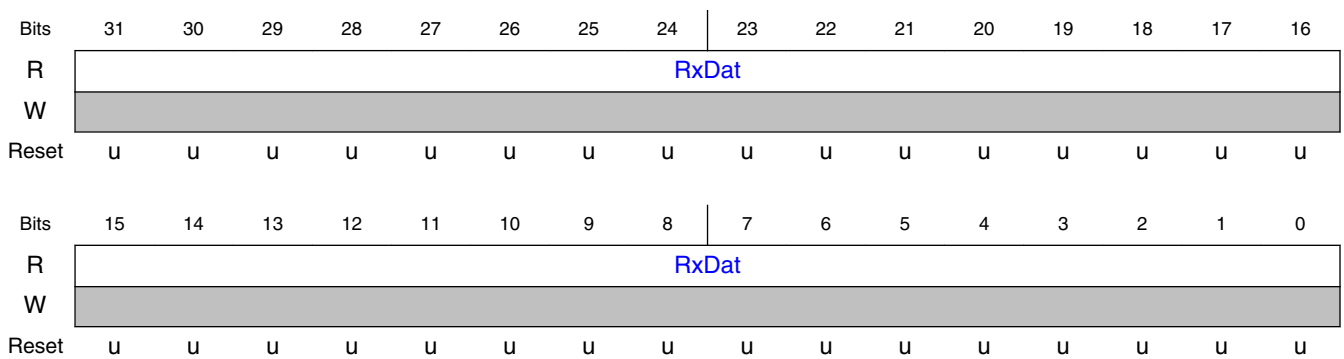
13.5.5.1.19.1 Offset

Register	Offset
DSI_RXFIFO	44h

13.5.5.1.19.2 Function

This register is the gate of FIFO read.

13.5.5.1.19.3 Diagram



13.5.5.1.19.4 Fields

Field	Function
31-0 RxDat	In the Rx mode, you can read Rx data through this register. **NOTE:** The CRC in packet is not stored in RxFIFO.

13.5.5.1.20 Specifies the FIFO threshold level register. (DSI_FIFOTHLD)

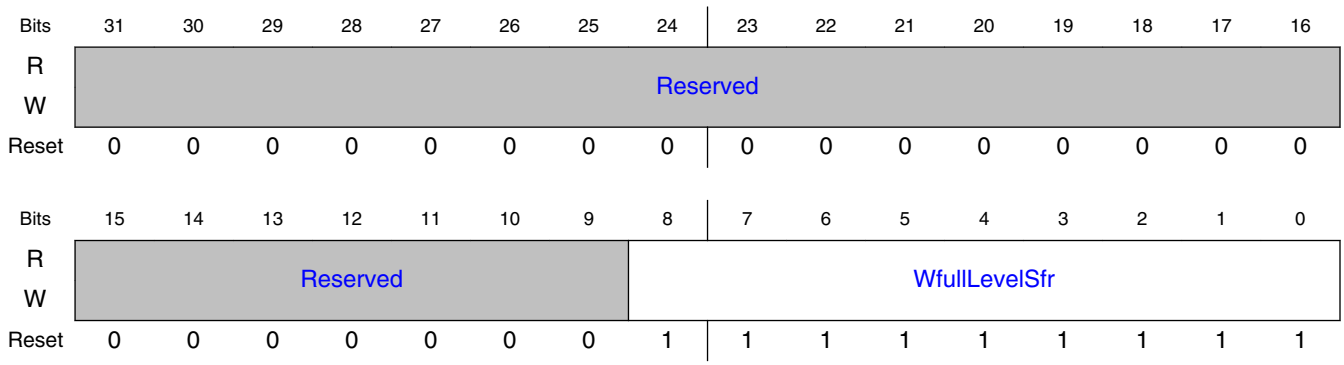
13.5.5.1.20.1 Offset

Register	Offset
DSI_FIFOTHLD	48h

13.5.5.1.20.2 Function

Specifies the FIFO threshold level register.

13.5.5.1.20.3 Diagram



13.5.5.1.20.4 Fields

Field	Function
31-9 —	Reserved
8-0 WfullLevelSfr	Almost full level of SFR payload FIFO

13.5.5.1.21 Specifies the FIFO status and control register. (DSI_FIFO CTRL)

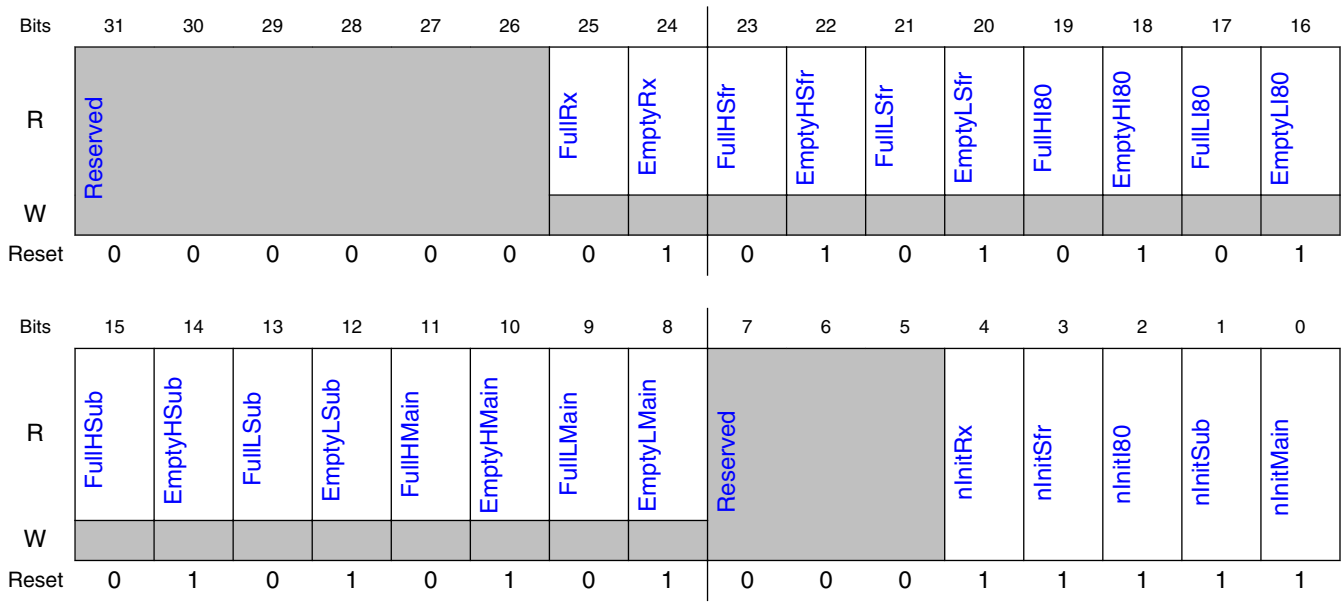
13.5.5.1.21.1 Offset

Register	Offset
DSI_FIFOCTRL	4Ch

13.5.5.1.21.2 Function

Specifies the FIFO status and control register.

13.5.5.1.21.3 Diagram



13.5.5.1.21.4 Fields

Field	Function
31-26 —	Reserved
25 FullRx	Rx FIFO full
24 EmptyRx	Rx FIFO empty
23 FullHSfr	SFR packet header FIFO full
22 EmptyHSfr	SFR packet header FIFO empty
21 FullLSfr	SFR payload FIFO full
20 EmptyLSfr	SFR payload FIFO empty
19 FullHI80	S-i80 packet header FIFO full
18 EmptyHI80	S-i80 packet header FIFO empty
17 FullLI80	S-i80 payload FIFO full

Table continues on the next page...

Field	Function
16 EmptyLI80	S-i80 payload FIFO empty
15 FullHSub	Sub display packet header FIFO full
14 EmptyHSub	Sub display packet header FIFO empty
13 FullLSub	Sub display payload FIFO full
12 EmptyLSub	Sub display payload FIFO empty
11 FullHMain	Main display packet header FIFO full
10 EmptyHMain	Main display packet header FIFO empty
9 FullLMain	Main display payload FIFO full
8 EmptyLMain	Main display payload FIFO empty
7-5 —	Reserved
4 nInitRx	MD FIFO read point initialize
3 nInitSfr	SFR FIFO write point initialize
2 nInitI80	S-i80 FIFO write point initialize
1 nInitSub	SD FIFO write point initialize
0 nInitMain	MD FIFO write point initialize

13.5.5.1.22 Specifies the FIFO memory AC characteristic register. (DSI_MEMACCHR)

13.5.5.1.22.1 Offset

Register	Offset
DSI_MEMACCHR	50h

13.5.5.1.22.2 Function

Specifies the FIFO memory AC characteristic register.

NOTE

In the current design, these memory port controls were disabled. User can ignore the function of this register.

13.5.5.1.22.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PGEN_SD	RETN_SD	EMAB_SD			EMAA_SD			PGEN_MD	RETN_MD	EMAB_MD			EMAA_MD		
Reset	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0

13.5.5.1.22.4 Fields

Field	Function
31-16 —	Reserved
15 PGEN_SD	Sub display FIFO memory power gating
14 RETN_SD	Sub display FIFO memory Retention
13-11 EMAB_SD	Sub display FIFO memory B port margin adjustment
10-8 EMAA_SD	Sub display FIFO memory A port margin adjustment
7 PGEN_MD	Main display FIFO memory power gating
6 RETN_MD	Main display FIFO memory Retention
5-3 EMAB_MD	Main display FIFO memory B port margin adjustment

Table continues on the next page...

Field	Function
2-0 EMAA_MD	Main display FIFO memory A port margin adjustment

13.5.5.1.23 Specifies the Multi Packet, Packet Go register. (DSI_MULTI_PKT)

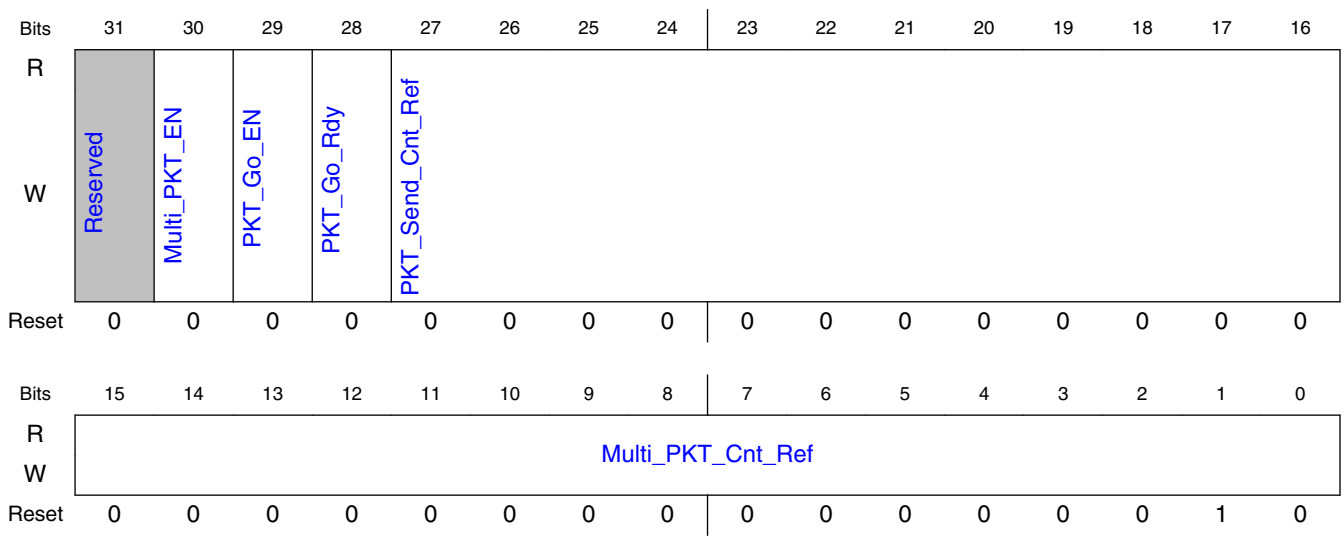
13.5.5.1.23.1 Offset

Register	Offset
DSI_MULTI_PKT	78h

13.5.5.1.23.2 Function

Specifies the Multi Packet, Packet Go register.

13.5.5.1.23.3 Diagram



13.5.5.1.23.4 Fields

Field	Function
31 —	Reserved
30	Specifies the send multi command packets on single transmission.

Table continues on the next page...

Field	Function
Multi_PKT_EN	Multi command packet can transfer by only HS mode. ‘0’ = Send single command packet. ‘1’ = Send multi command packets.
29 PKT_Go_EN	Specifies the send command packet(s) per frame enable Packet go can transfer by only HS mode. ‘0’ = Send command packet(s) during every VFP. ‘1’ = Send command packet(s) during 1frame VFP.
28 PKT_Go_Rdy	Specifies the send command packet(s) on this frame VFP. If packet transferred or PKT_Go_EN is disabled, this bit goes to 1'b0.
27-16 PKT_Send_Cnt_Ref	Specifies the command packet(s) send point indicator. That value is 1H time + D-PHY LP-HS-LP latency by ByteCLK. Do not use '0'.
15-0 Multi_PKT_Cnt_Ref	Specifies the number of packets on single transmission. Do not use '0' and '1'. 0 and 1 are same as '2'

13.5.5.1.24 Specifies the 1Gbps D-PHY PLL control register. (DSI_PLLCTRL_1G)

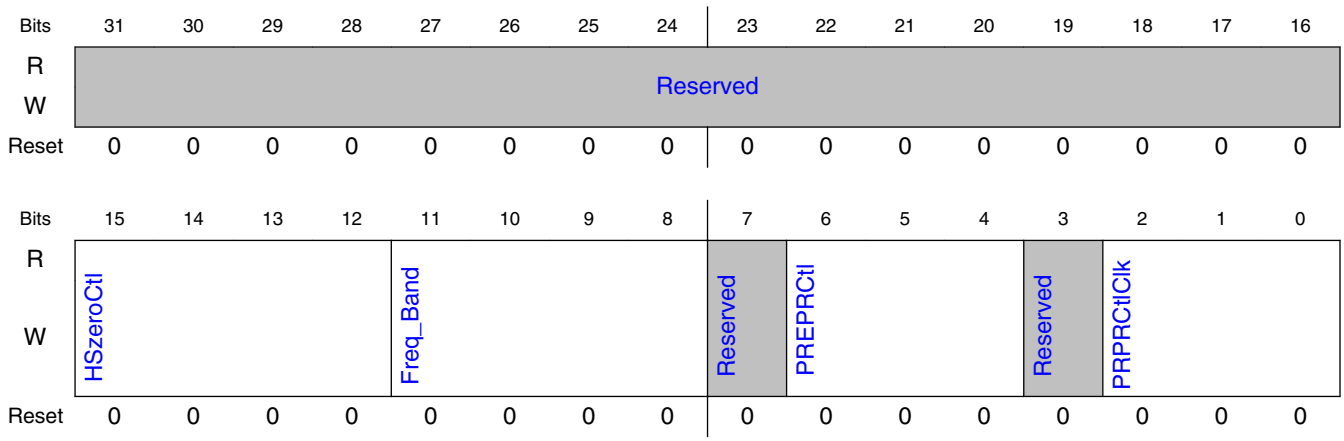
13.5.5.1.24.1 Offset

Register	Offset
DSI_PLLCTRL_1G	90h

13.5.5.1.24.2 Function

This register configures 1Gbps D-PHY PLL control, D-PHY, clock range indication, and so on.

13.5.5.1.24.3 Diagram



13.5.5.1.24.4 Fields

Field	Function
31-16 —	Reserved
15-12 HSzeroCtl	1Gbps D-PHY HS-Zero driving timing control.
11-8 Freq_Band	1Gbps D-PHY Timing control for D-PHY global operation timing.
7 —	Reserved
6-4 PREPRCtl	1Gbps D-PHY PLL Tclk-prepare and Ths-prepare driving control.
3 —	Reserved
2-0 PRPRCtlClk	1Gbps D-PHY PLL Ths-prepare driving time control.

13.5.5.1.25 Specifies the PLL control register. (DSI_PLLCTRL)

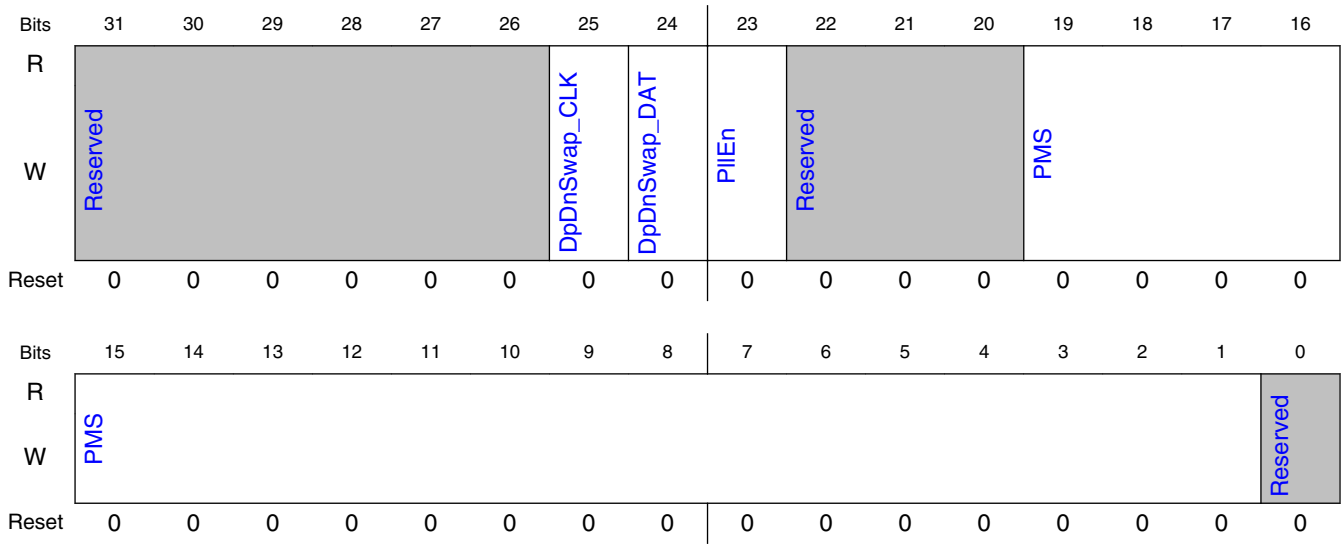
13.5.5.1.25.1 Offset

Register	Offset
DSI_PLLCTRL	94h

13.5.5.1.25.2 Function

This register configures PLL control, D-PHY, clock range indication, and so on.

13.5.5.1.25.3 Diagram



13.5.5.1.25.4 Fields

Field	Function
31-26 —	Reserved
25 DpDnSwap_CLK	Swaps Dp/Dn channel of clock lane. If this bit is set, Dp and Dn channel swap each other.
24 DpDnSwap_DAT	Swaps Dp/Dn channel of Data lanes. If this bit is set, Dp and Dn channel swap each other.
23 PllEn	Enables PLL.
22-20 —	Reserved
19-1 PMS	Specifies the PLL PMS value.
0 —	Reserved

13.5.5.1.26 Specifies the PLL control register 1. (DSI_PLLCTRL1)

13.5.5.1.26.1 Offset

Register	Offset
DSI_PLLCTRL1	98h

13.5.5.1.26.2 Function

This register configures D-PHY PLL control (M_PLLCTL[31:0]).

13.5.5.1.26.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M_PLLCTL0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M_PLLCTL0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.26.4 Fields

Field	Function
31-0 M_PLLCTL0	M_PLLCTL[31:0] to D-PHY

13.5.5.1.27 Specifies the PLL control register 2. (DSI_PLLCTRL2)

13.5.5.1.27.1 Offset

Register	Offset
DSI_PLLCTRL2	9Ch

13.5.5.1.27.2 Function

This register configures D-PHY PLL control (M_PLLCTL[39:32]).

13.5.5.1.27.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								M_PLLCTL1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.27.4 Fields

Field	Function
31-8 —	Reserved
7-0 M_PLLCTL1	M_PLLCTL[39:32] to D-PHY

13.5.5.1.28 Specifies the PLL timer register. (DSI_PLLTMR)

13.5.5.1.28.1 Offset

Register	Offset
DSI_PLLTMR	A0h

13.5.5.1.28.2 Function

Specifies the PLL timer register.

13.5.5.1.28.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PllTimer															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PllTimer															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

13.5.5.1.28.4 Fields

Field	Function
31-0 PllTimer	Specifies the PLL Timer for stability of the generated clock (System clock cycle base). If the timer value goes to 0x00000000, the clock stable bit of status and interrupt register is set.

13.5.5.1.29 Specifies the D-PHY control register 1. (DSI_PHYCTRL_B1)

13.5.5.1.29.1 Offset

Register	Offset
DSI_PHYCTRL_B1	A4h

13.5.5.1.29.2 Function

D-PHY Master and Slave Analog block characteristics control registers (B_DPHYCTL).

13.5.5.1.29.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B_DPHYCTL0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B_DPHYCTL0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.29.4 Fields

Field	Function
31-0 B_DPHYCTL0	B_DPHYCTL[31:0] to D-PHY

13.5.5.1.30 Specifies the D-PHY control register 2. (DSI_PHYCTRL_B2)

13.5.5.1.30.1 Offset

Register	Offset
DSI_PHYCTRL_B2	A8h

13.5.5.1.30.2 Function

D-PHY Master and Slave Analog block characteristics control registers (B_DPHYCTL).

13.5.5.1.30.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B_DPHYCTL1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B_DPHYCTL1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.30.4 Fields

Field	Function
31-0 B_DPHYCTL1	B_DPHYCTL[63:32] to D-PHY

13.5.5.1.31 Specifies the D-PHY control register 1. (DSI_PHYCTRL_M1)

13.5.5.1.31.1 Offset

Register	Offset
DSI_PHYCTRL_M1	ACh

13.5.5.1.31.2 Function

D-PHY Master Analog block characteristics control registers (M_DPHYCTL).

13.5.5.1.31.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M_DPHYCTL0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M_DPHYCTL0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.31.4 Fields

Field	Function
31-0 M_DPHYCTL0	M_DPHYCTL[31:0] to D-PHY

13.5.5.1.32 Specifies the D-PHY control register 2. (DSI_PHYCTRL_M2)

13.5.5.1.32.1 Offset

Register	Offset
DSI_PHYCTRL_M2	B0h

13.5.5.1.32.2 Function

D-PHY Master Analog block characteristics control registers (M_DPHYCTL).

13.5.5.1.32.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M_DPHYCTL1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M_DPHYCTL1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.32.4 Fields

Field	Function
31-0 M_DPHYCTL1	M_DPHYCTL[63:32] to D-PHY

13.5.5.1.33 Specifies the D-PHY timing register. (DSI_PHYTIMING)

13.5.5.1.33.1 Offset

Register	Offset
DSI_PHYTIMING	B4h

13.5.5.1.33.2 Function

D-PHY Master global operating timing registers.

13.5.5.1.33.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M_TLPXCTL								M_THSEXITCTL							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.33.4 Fields

Field	Function
31-16 —	Reserved
15-8 M_TLPXCTL	M_TLPXCTL[7:0] to D-PHY
7-0 M_THSEXITCTL	M_THSEXITCTL[7:0] to D-PHY

13.5.5.1.34 Specifies the D-PHY timing register 1. (DSI_PHYTIMING1)

13.5.5.1.34.1 Offset

Register	Offset
DSI_PHYTIMING1	B8h

13.5.5.1.34.2 Function

D-PHY Master global operating timing registers.

13.5.5.1.34.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M_TCLKPRPRCTL								M_TCLKZEROCTL							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M_TCLKPOSTCTL								M_TCLKTRAILCTL							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.34.4 Fields

Field	Function
31-24 M_TCLKPRPRCTL	M_TCLKPRPRCTL[7:0] to D-PHY
23-16 M_TCLKZEROCTL	M_TCLKZEROCTL[7:0] to D-PHY
15-8 M_TCLKPOSTCTL	M_TCLKPOSTCTL[7:0] to D-PHY
7-0 M_TCLKTRAILCTL	M_TCLKTRAILCTL[7:0] to D-PHY

13.5.5.1.35 Specifies the D-PHY timing register 2. (DSI_PHYTIMING2)

13.5.5.1.35.1 Offset

Register	Offset
DSI_PHYTIMING2	BCh

13.5.5.1.35.2 Function

D-PHY Master global operating timing registers.

13.5.5.1.35.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								M_THSPRPRCTL							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M_THSZEROCTL								M_THSTRAILCTL							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.5.5.1.35.4 Fields

Field	Function
31-24 —	Reserved
23-16 M_THSPRPRCTL	M_THSPRPRCTL[7:0] to D-PHY
15-8 M_THSZEROCTL	M_THSZEROCTL[7:0] to D-PHY
7-0 M_THSTRAILCTL	M_THSTRAILCTL[7:0] to D-PHY

13.6 MIPI D-PHY (MIPI_DPHY)

13.6.1 Overview

This chapter describes integrating the MIPI D-PHY core into the ASIC or SoC chips and testing the MIPI D-PHY core. The MIPI D-PHY core IP is a flexible, low-cost, and high-speed serial interface that connects a mobile display driver or a camera sensor to a host processor

The MIPI D-PHY has simplified functions for DSI2 Master and CSI Slave.

The functions of the Master Clock Lane of the core are:

- HS-TX

- LP-TX
- CIL-MCNN

The functions of Slave Clock Lane Module of core are:
- HS-RX
- LP-RX
- CIL-SCNN

The functions of Master Data Lane 0 Module of the core are:
- HS-TX
- LP-TX
- LP-RX
- LP-CD
- CIL-MFAA (Bi-directional Escape Mode, Bi-directional LPDT).

The functions of Master Data Lane 1/2/3 Module of core are:
- HS-TX
- LP-TX
- CIL-MFEN (forward ULPS only, no LPDT)

The functions of the Slave Data Lane 0/1/2/3 Module of the core are:
- HS-RX
- LP-RX
- CIL-SFEN (forward ULPS only)

13.6.2 Features

The maximum high-speed clock frequency of MIPI D-PHY core is 1.5 GHz.

The features of MIPI D-PHY are:
- D-PHY spec v1.2 compatible.
- Synchronous link between Master (data source) and Slave (data sink).
- All lanes support high-speed transmission in the forward direction.
- Bi-directional data transmission in Low-Power mode at the Master Data Lane 0 only.
- Uses token passing to control the communication direction of the link.
- High-Speed mode for fast data traffic and Low-Power mode for controls and low-speed data transmission.
- High-Speed mode: Differential and terminated, 200 mV swing: 80 – 1500 Mbps.
- Low-Power mode: Single-ended and non-terminated, 1.2 V swing: 10 Mbps maximum.

NOTE

Use this mode for low-speed asynchronous data communications or controls.

13.6.3 PLL and Clock Lane Connection

The following figure illustrates the PLL and Clock Lane connection.

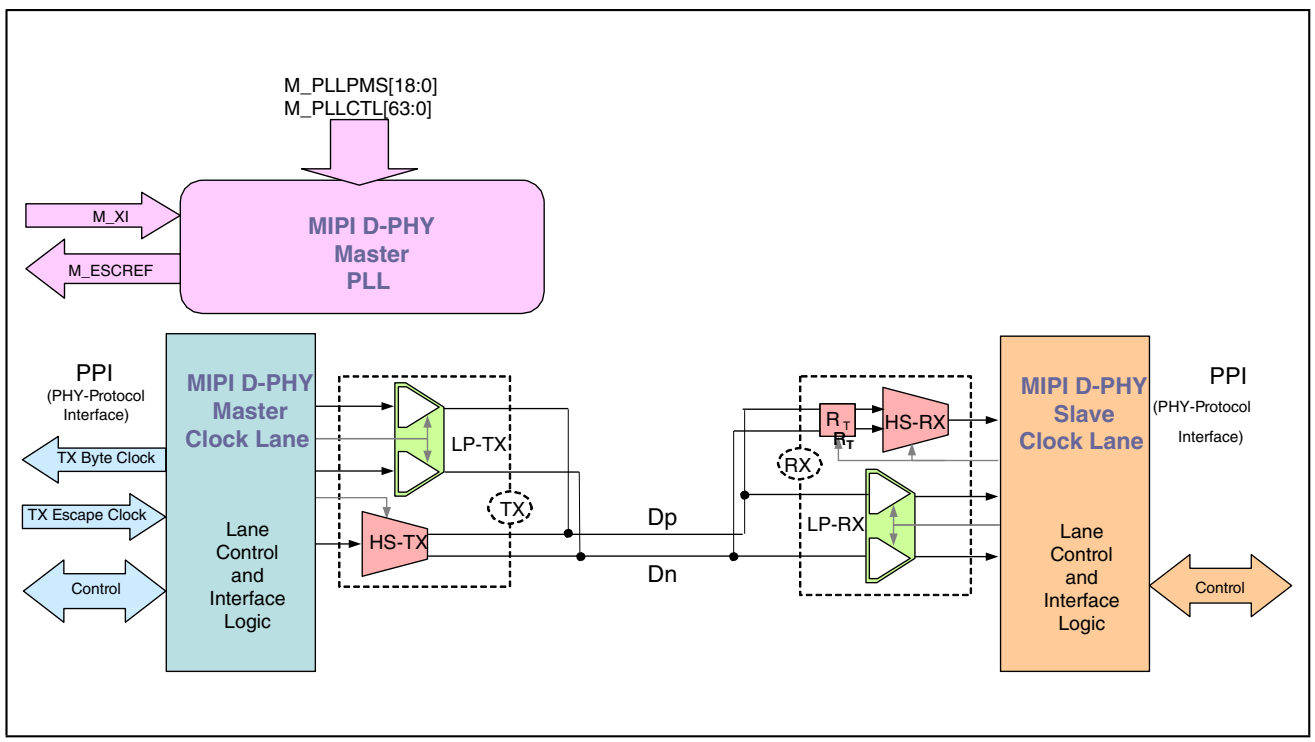


Figure 13-68. PLL and Clock Lane Connection

13.6.4 Data Lane Connection

The following figure illustrates the Data0 Lane connection between Master and the Slave.

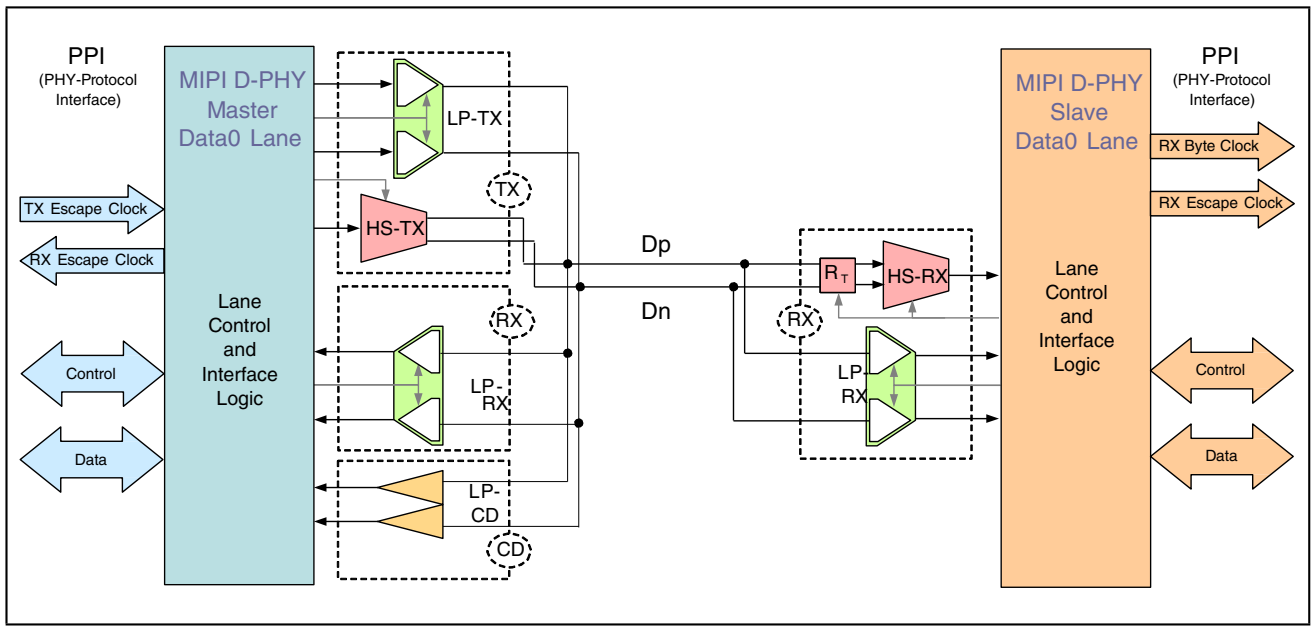


Figure 13-69. Data0 Lane Connection

The following figure illustrates the Data1 Lane connection between Master and the Slave.

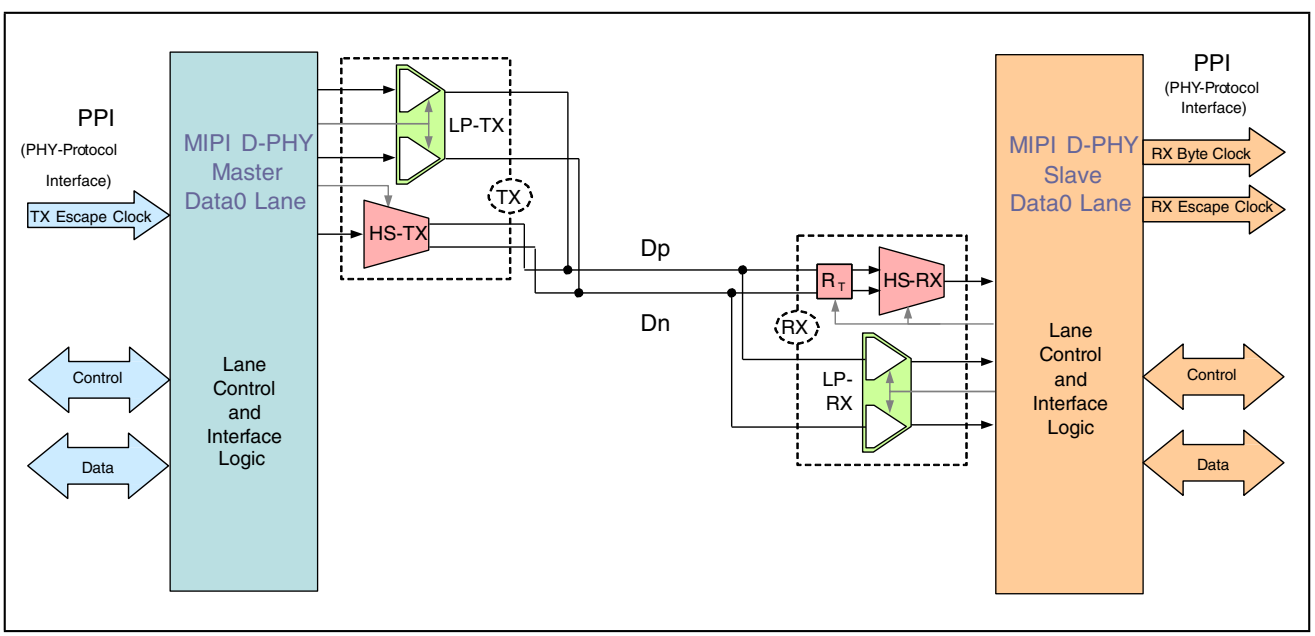


Figure 13-70. Data1 Lane Connection

The following figure illustrates the Data2 Lane connection between Master and the Slave.

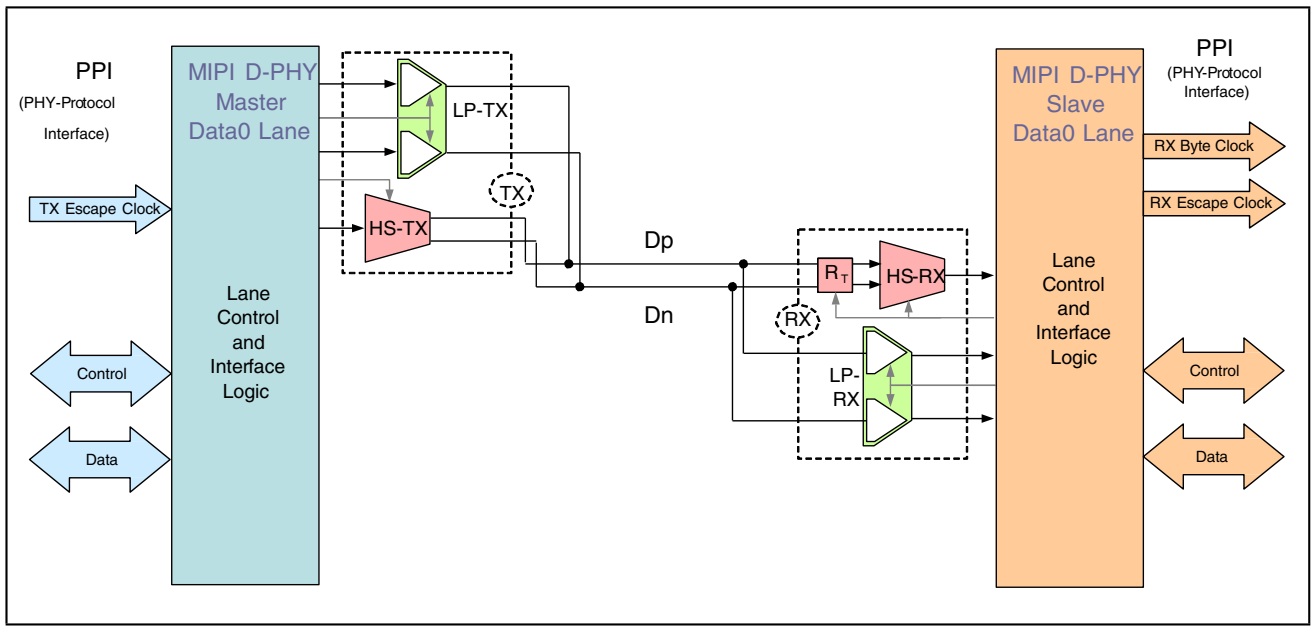


Figure 13-71. Data2 Lane Connection

The following figure illustrates the Data3 Lane connection between Master and the Slave.

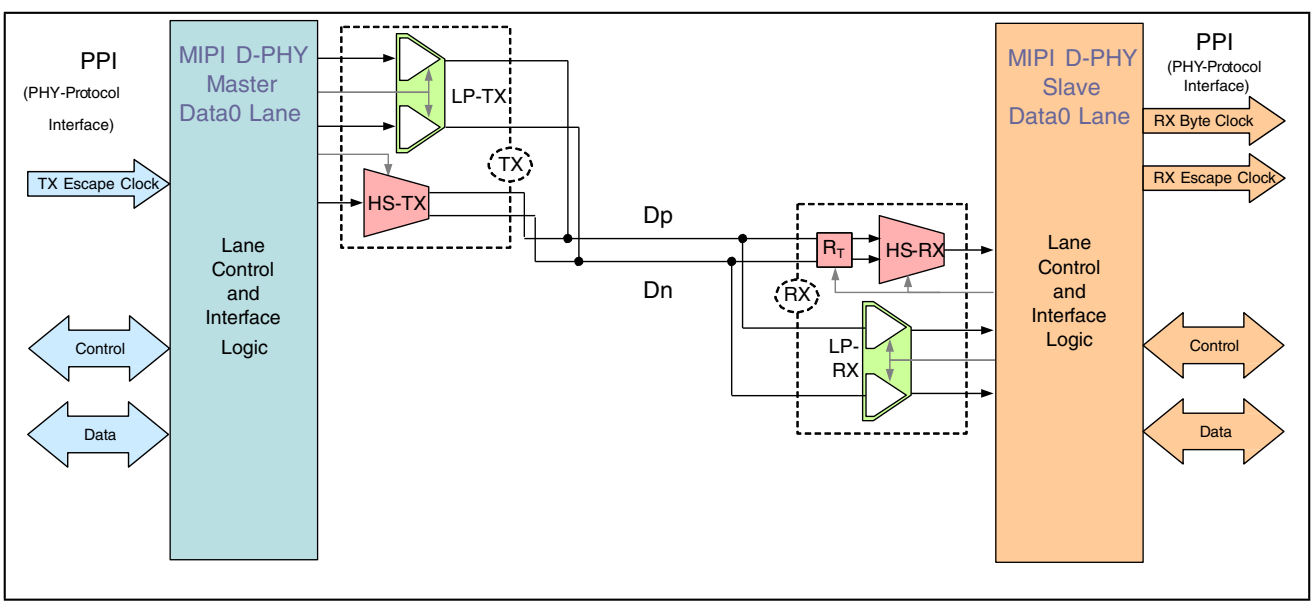


Figure 13-72. Data3 Lane Connection

13.6.5 IP Structure

The MIPI D-PHY core consists of five modules. The modules are:

- Master PLL

- Master Clock lane
- Master Data lane
- Slave Clock lane
- Slave Data lane

NOTE

It is strongly recommended to use the Master and Slave lane as a pair for loop-back test.

The following figure illustrates the IP structure.

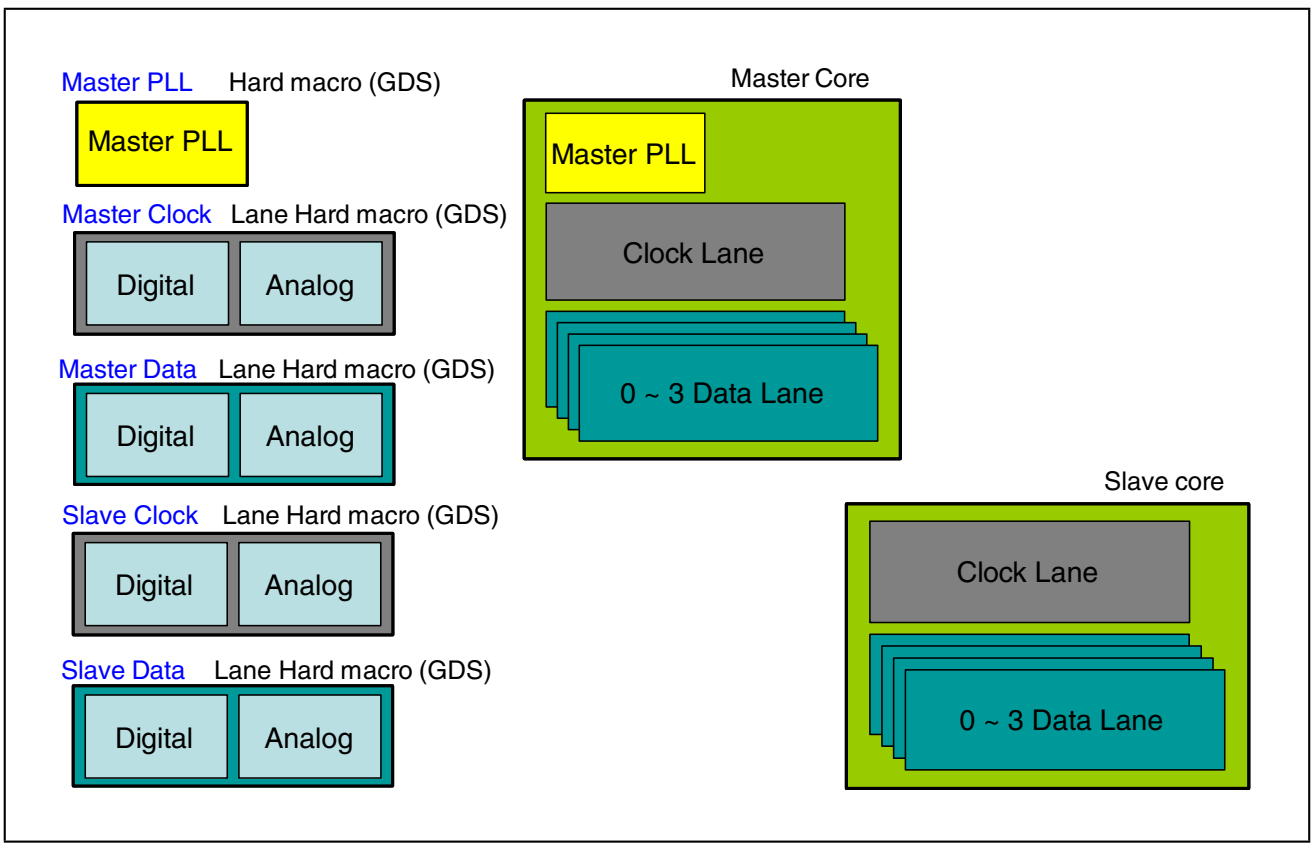


Figure 13-73. IP Structure

13.6.6 D-PHY Architecture

The 4-lane D-PHY architecture including lanes and common block is shown in the following figure.

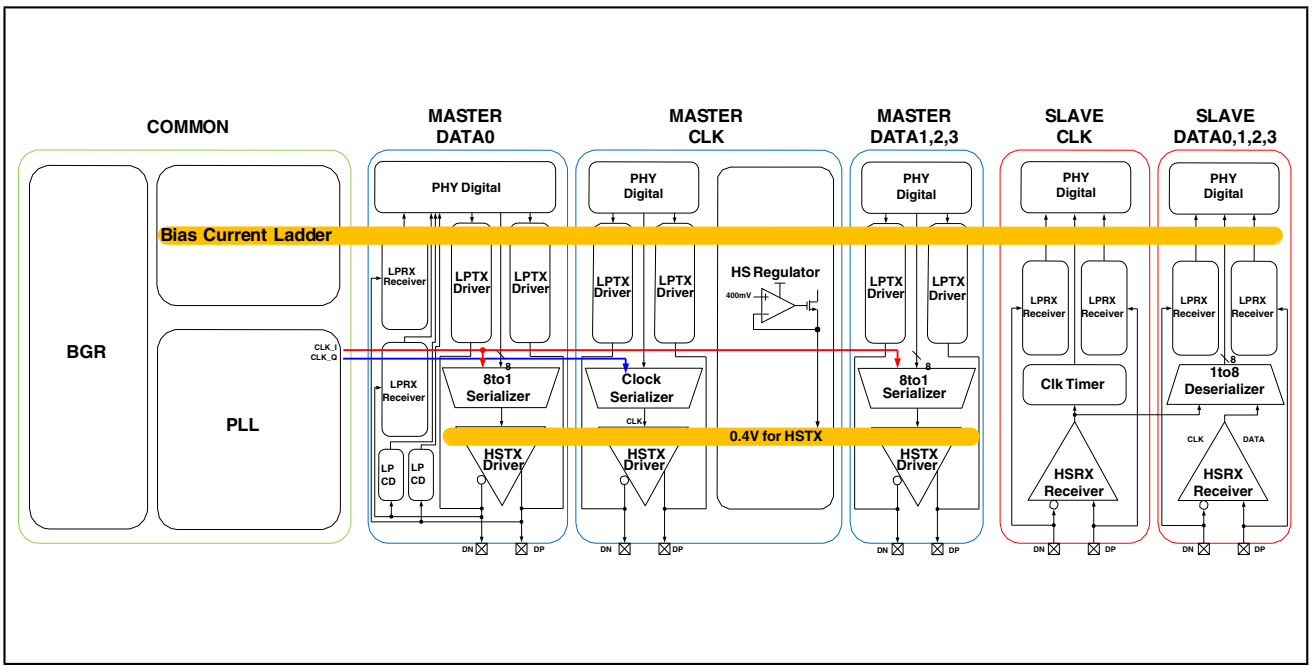


Figure 13-74. D-PHY Architecture

13.6.7 Protocol Support Summary

A summary of supported standards with corresponding data rates is listed in the below table.

Table 13-35. Supported Standards

Standard	Lane	Data Rate
MIPI D-PHY V1.2	4	80Mbps~1500Mbps

13.6.8 IP Interface

MIPI D-PHY configuration contains a Clock Lane Module and 1–4 Data Lane Modules. "0/1/2/3" in signals signify the data lane 0/1/2/3.

13.6.8.1 I/O Diagram

The following figure illustrates the 4-line D-PHY I/O diagram.

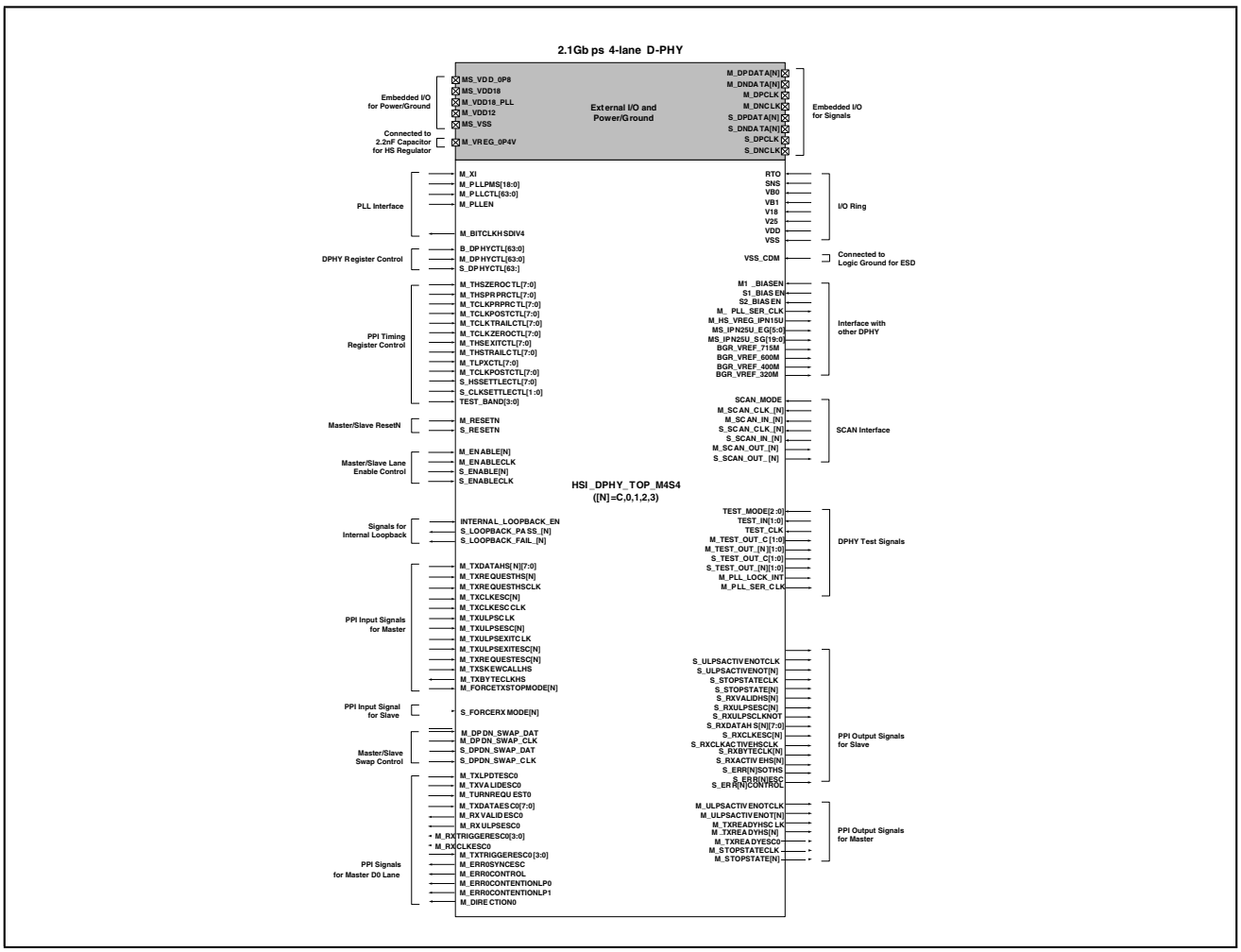


Figure 13-75. 4-Lane D-PHY Pin Description

Here, [N] means the number of data lanes. The power and the ground pins described with gray box in upper side and the other signal pins described in lower side.

13.6.8.2 Pad Signals

The MIPI D-PHY core includes the I/O pads. I/O pads are dedicated to D-PHY and you cannot share them with any other signals. You can determine the number of power pads and signal pads from the core module configurations.

The following table describes the pad signals.

Table 13-36. Pad Signals

Mode	I/O	Active Level	Description
M_VREG_0P4V	I/O	N/A	Regulator capacitor connection
M_DPCLK	O	N/A	Master CLK Lane DP
M_DNCLK	O	N/A	Master CLK Lane DN
M_DPDATA0	O	N/A	Master DATA0 Lane DP
M_DNDATA0	O	N/A	Master DATA0 Lane DN
M_DPDATA1	O	N/A	Master DATA1 Lane DP
M_DNDATA1	O	N/A	Master DATA1 Lane DN
M_DPDATA2	O	N/A	Master DATA2 Lane DP
M_DNDATA2	O	N/A	Master DATA2 Lane DN
M_DPDATA3	O	N/A	Master DATA3 Lane DP
M_DNDATA3	O	N/A	Master DATA3 Lane DN
S_DPCLK	I	N/A	Slave CLK Lane DP
S_DNCLK	I	N/A	Slave CLK Lane DN
S_DPDATA0	I	N/A	Slave DATA0 Lane DP
S_DNDATA0	I	N/A	Slave DATA0 Lane DN
S_DPDATA1	I	N/A	Slave DATA1 Lane DP
S_DNDATA1	I	N/A	Slave DATA1 Lane DN
S_DPDATA2	I	N/A	Slave DATA2 Lane DP
S_DNDATA2	I	N/A	Slave DATA2 Lane DN
S_DPDATA3	I	N/A	Slave DATA3 Lane DP
S_DNDATA3	I	N/A	Slave DATA3 Lane DN
VDD18_PLL	Supply	N/A	1.8V Power for PLL
DVDD08D_PLL	Supply	N/A	0.9V Power for PLL
MS_VDD_0P8	Supply	N/A	0.9 V Power for Internal Logic
MS_VDD18	Supply	N/A	1.8 V Power for Analog
M_VDD12	Supply	N/A	1.2 V Power for Analog
MS_VSS	GND	N/A	Ground
RTO/SNS/VB0/VB1/V18/V25	I/O	N/A	These signals are I/O pad's ring signals. If you want to get more detail information about these signals, check the datasheet of GPIO.

13.6.8.3 ESD Protection Port

The anti-parallel diode is located between the D-PHY ground and the SoC logic ground for ESD protection.

Table 13-37. ESD Protection Port

Mode	I/O	Active Level	Description
VSS_CDM	Supply	N/A	This port is connected to the SoC logic ground with wide metal. See Figure 21.

13.6.8.4 Master PLL: PHY-Protocol Interface (PPI) Signals

The following table describes the PPI signals of Master PLL.

Table 13-38. Master PLL: PPI Signals

Name	I/O	Active Level	Description
M_XI	I	N/A	PLL reference input clocks. You should shield and route the inputs shortly.
M_PLLPMS[18:0]	I	N/A	80 MHz to 2.1 GHz Frequency Control Signals from the register: PMS setting.
M_PLLCTL[63:0]	I	N/A	PLL control.
M_BITCLKHSDIV4	O	N/A	Used for Protocol Layer multiple Lanes FIFO controls (maximum 525 MHz).
M_PLLEN	I	High	Initializes PLL during LOW state and enable when it is HIGH state. NOTE: You must assert to Low state during power-up sequence for a stable initialization.

13.6.8.5 Master Clock Lane: PPI Signals

The following table describes the PPI signals of Master Clock Lane.

Table 13-39. Master Clock Lane: PPI Signals

Name	I/O	Active Level	Description
M_TXBYTECLKHS	O	N/A	To Protocol Layer, PPI clock 10 MHz to 262.5 MHz Periodic Jitter: tM_TXBYTECLKHS*(+/-3%)

Table continues on the next page...

Table 13-39. Master Clock Lane: PPI Signals (continued)

Name	I/O	Active Level	Description
M_TXREQUESTHSCLK	I	High	This signal causes the Clock Lane Module to transmit a high-speed clock (M_TXBYTECLKHS , Level High).
M_TXREADYHSCLK	O	High	High-Speed Transmit Ready on clock Lane (M_TXBYTECLKHS , Level High).
M_TXCLKESCCLK	I	N/A	Escape Mode Clock (£ 20 MHz). Used for LP mode operations.
M_TXULPSCLK	I	High	Transmits Ultra Low-Power mode on clock Lane (M_TXCLKESCCLK, Level High).
M_TXULPSEXITCLK	I	High	Transmits the ULP Exit Sequence (M_TXCLKESCCLK , Level High).
M_ULPSACTIVENOTCLK	O	Low	Ultra Low Power State (not) Active (M_TXCLKESCCLK , Level Low).
M_STOPSTATECLK	O	High	Lane is in the Stop state (asynchronous, Level High).
M_ENABLECLK	I	High	Enables the Clock Lane Module (asynchronous, Level High). NOTE: You must assert to Low state during power-up sequence for a stable initialization.
M_TXSKEWCALHS	I	High	This signal causes the Master Module to begin the Skew-Calibration Mode (M_TXBYTECLKHS , Level or Pulse High).

13.6.8.6 Master Data0 Lane: PPI Signals

The following table describes the PPI signals of Master Data0 Lane.

Table 13-40. Master Data0 Lane: PPI Signals

Name	I/O	Active Level	Description
M_TXDATAHS0[7:0]	I	N/A	High-Speed Transmit Data (M_TXBYTECLKHS , captured). Transmits M_TXDATAHS [0 position bit] first.
M_TXREQUESTHS0	I	High	High-Speed Transmit Request and Data Valid (M_TXBYTECLKHS , Level High).
M_TXREADYHS0	O	High	High-Speed Transmit Ready (M_TXBYTECLKHS , Level High).
M_TXCLKESC0	I	N/A	Escape mode Clock (£ 20 MHz).
M_TXREQUESTESC0	I	High	Escape mode Transmit Request (M_TXCLKESC0 , Level High).
M_TXULPSESC0	I	High	Escape mode Transmit Ultra Low Power (M_TXCLKESC0 , Pulse High).
M_TXULPSEXITESC	I	High	Transmits ULP Exit Sequence (M_TXCLKESC0 , Pulse High).
M_TXTRIGGERESC0[3:0]	I	High	Escape mode Transmit Trigger 0-3 (M_TXCLKESC0 , Pulse High only 1-bit).
M_TXLPDTESC0	I	High	Escape mode Transmit Low Power Data (M_TXCLKESC0 , Pulse High).
M_TXDATAESC0[7:0]	I	High	Escape mode Transmit Data (M_TXCLKESC0 , captured). Transmits M_TXDATAESC [0 position bit] first.
M_TXVALIDESC0	I	High	Escape mode Transmit Data Valid (M_TXCLKESC0 , Pulse High or Level High).
M_TXREADYESC0	O	High	Escape mode Transmit Ready (M_TXCLKESC0 , Pulse High or Level High).
M_RXCLKESC0	O	N/A	Escape mode Receive Clock (This is edge detection clock from both Dp, Dn lines).
M_RXLPDTESC0	O	High	Escape Low Power Data Receive mode (M_RXCLKESC0 , Level High).

Table continues on the next page...

Table 13-40. Master Data0 Lane: PPI Signals (continued)

Name	I/O	Active Level	Description
M_RXULPSESC0	O	High	Escape Ultra Low Power Receive mode (M_RXCLKESC0 , Level High).
M_RXTRIGGERESC0[3:0]	O	High	Escape Receive Trigger 0–3 (M_RXCLKESC0 , Level High only 1-bit).
M_RXDATAESC0[7:0]	O	High	Escape mode Receive Data (M_RXCLKESC0 , captured) Receives M_RXDATAESC [0] first.
M_RXVALIDESC0	O	High	Escape mode Receive Data Valid (M_RXCLKESC0 , Pulse High).
M_TURNREQUEST0	I	High	Turn Around Request (M_TXCLKESC0 , Level High).
M_DIRECTION0	O	High	Transmit/Receive Direction (M_TXCLKESC0 or M_RXCLKESC0 , Level-Low: Transmit, High: Receive).
M_FORCETXSTOPMODE0	I	High	Force Lane into Transmit mode Generate Stop state (asynchronous, Level High).
M_ULPSACTIVENOT0	O	Low	Ultra Low Power State (not) Active (M_TXCLKESC0 or M_RXCLKESC0 , Level Low).
M_STOPSTATE0	O	High	Data Lane is in Stop state (asynchronous, Level High).
M_ENABLE0	I	High	Enable Data Lane Module (Asynchronous, Level High). NOTE: You must assert to Low state during power-up sequence for a stable initialization.
M_ERR0ESC	O	High	Escape Entry Error (M_RXCLKESC0 , Level High).
M_ERR0SYNCESC	O	High	Low Power Data Transmission Synchronization Error (M_RXCLKESC0 , Pulse High).
M_ERR0CONTROL	O	High	Control Error (M_RXCLKESC0 , Pulse High).
M_ERR0CONTENTIONLP0	O	High	LP0 Contention Error (M_TXCLKESC0 , Pulse High or Level High).

Table continues on the next page...

Table 13-40. Master Data0 Lane: PPI Signals (continued)

Name	I/O	Active Level	Description
M_ERR0CONTENTIONLP1	O	High	LP1 Contention Error (M_TXCLKESC0 , Pulse High or Level High).

13.6.8.7 Master Data1 Lane: PPI Signals

The following table describes the PPI signals of Master Data1 Lane.

Table 13-41. Master Data1 Lane: PPI Signals

Name	I/O	Active Level	Description
M_TXDATAHS1[7:0]	I	N/A	High-Speed Transmit Data (M_TXBYTECLKHS , captured). Transmits M_TXDATAHS [0 position bit] first.
M_TXREQUESTHS1	I	High	High-Speed Transmit Request and Data Valid (M_TXBYTECLKHS , Level High).
M_TXREADYHS1	O	High	High-Speed Transmit Ready (M_TXBYTECLKHS , Level High).
M_TXCLKESC1	I	N/A	Escape mode Clock (£ 20 MHz).
M_TXREQUESTESC1	I	High	Escape mode Transmit Request (M_TXCLKESC1 , Level High).
M_TXULPSESC1	I	High	Escape mode Transmit Ultra Low Power (M_TXCLKESC1, Pulse High).
M_TXULPSEXITESC1	I	High	Transmit ULP Exit Sequence (M_TXCLKESC1 , Pulse High).
M_FORCETXSTOPMODE1	I	High	Force Lane into Transmit mode. Generates the Stop state (asynchronous, Level High).
M_ULPSACTIVENOT1	O	Low	Ultra Low Power State is (not) Active (M_TXCLKESC1 , Level Low).
M_STOPSTATE1	O	High	Data Lane is in the Stop state (asynchronous, Level High).
M_ENABLE1	I	High	Enables Data Lane Module (Asynchronous, Level High). NOTE: You should assert to

Table 13-41. Master Data1 Lane: PPI Signals

Name	I/O	Active Level	Description
			Low state during power-up sequence for stable initialization.

13.6.8.8 Master Data2 Lane: PPI Signals

The following table describes the PPI signals of Master Data2 Lane.

Table 13-42. Master Data2 Lane: PPI Signals

Name	I/O	Active Level	Description
M_TXDATAHS2[7:0]	I	N/A	High-Speed Transmit Data (M_TXBYTECLKHS , captured). Transmits M_TXDATAHS [0 position bit] first.
M_TXREQUESTHS2	I	High	High-Speed Transmit Request and Data Valid (M_TXBYTECLKHS , Level High).
M_TXREADYHS2	O	High	High-Speed Transmit Ready (M_TXBYTECLKHS , Level High).
M_TXCLKESC2	I	N/A	Escape mode Clock (£ 20 MHz).
M_TXREQUESTESC2	I	High	Escape mode Transmit Request (M_TXCLKESC2 , Level High).
M_TXULPSESC2	I	High	Escape mode Transmit Ultra Low Power (M_TXCLKESC2, Pulse High).
M_TXULPSEXITESC2	I	High	Transmits the ULP Exit Sequence (M_TXCLKESC2 , Pulse High).
M_FORCETXSTOPMODE2	I	High	Forces the Lane into the Transmit mode. Generates the Stop state (asynchronous, Level High).
M_ULPSACTIVENOT2	O	Low	Ultra Low Power State (not) Active (M_TXCLKESC2 , Level Low).
M_STOPSTATE2	O	High	Data Lane is in the Stop state (asynchronous, Level High).
M_ENABLE2	I	High	Enables the Data Lane Module (asynchronous, Level High). NOTE: You must

Table 13-42. Master Data2 Lane: PPI Signals

Name	I/O	Active Level	Description
			assert to Low state during power-up sequence for stable initialization.

13.6.8.9 Master Data3 Lane: PPI Signals

The following table describes the PPI signals of Master Data3 Lane.

Table 13-43. Master Data3 Lane: PPI Signals

Name	I/O	Active Level	Description
M_TXDATAHS3[7:0]	I	N/A	High-Speed Transmit Data (M_TXBYTECLKHS , captured). Transmits M_TXDATAHS [0 position bit] first.
M_TXREQUESTHS3	I	High	High-Speed Transmit Request and Data Valid (M_TXBYTECLKHS , Level High).
M_TXREADYHS3	O	High	High-Speed Transmit Ready (M_TXBYTECLKHS , Level High).
M_TXCLKESC3	I	N/A	Escape mode Clock (£ 20 MHz).
M_TXREQUESTESC3	I	High	Escape mode Transmit Request (M_TXCLKESC3 , Level High).
M_TXULPSESC3	I	High	Escape mode Transmit Ultra Low Power (M_TXCLKESC3 , Pulse High).
M_TXULPSEXITESC3	I	High	Transmits the ULP Exit Sequence (M_TXCLKESC3 , Pulse High).
M_FORCETXSTOPMODE3	I	High	Force Lane into Transmit mode Generate Stop state (asynchronous, Level High).
M_ULPSACTIVENOT3	O	Low	Ultra Low Power State (not) Active (M_TXCLKESC3 , Level Low).
M_STOPSTATE3	O	High	Data Lane is in the Stop state (asynchronous, Level High).
M_ENABLE3	I	High	Enable Data Lane Module (a synchronous, Level High) NOTE: You must assert to Low state during power-up sequence for a stable initialization.

13.6.8.10 Slave Clock Lane: PPI Signals

The following table describes the PPI signals of Slave Clock Lane.

Table 13-44. Slave Clock Lane: PPI Signals

Name	I/O	Active Level	Description
S_RXCLKACTIVEHSCLK	O	High	Receiver Clock Active (asynchronous, Level High).
S_RXULPSCLKNOT	O	Low	Receive Ultra Low-Power State on Clock Lane (asynchronous, Level Low).
S_ULPSACTIVENOTCLK	O	Low	Ultra Low Power State (not) Active (asynchronous, Level Low).
S_STOPSTATECLK	O	High	Lane is in Stop state (asynchronous, Level High).
S_ENABLECLK	I	High	Enables Clock Lane Module (asynchronous, Level High). NOTE: You must assert to Low state during power-up sequence for a stable initialization.

13.6.8.11 Slave Data0 Lane: PPI Signals

The following table describes the PPI signals of Slave Data0 Lane.

Table 13-45. Slave Data0 Lane: PPI Signals

Name	I/O	Active Level	Description
S_RXBYTECLKHS0	O	N/A	"RXDDR/4" Clock. For HS Digital operation and for PPI. 10 MHz to 262.5 MHz Periodic jitter(@D-PHY): tS_RXBYTECLKHS0 * +/-5.5%
S_RXDATAHS0[7:0]	O	N/A	High-Speed Receive Data (S_RXBYTECLKHS0 , captured). This matches accurately with M_TXDATAHS[7:0] bits order.
S_RXACTIVEHS0	O	High	High-Speed Receive Active (S_RXBYTECLKHS0 , Level High).

Table continues on the next page...

Table 13-45. Slave Data0 Lane: PPI Signals (continued)

Name	I/O	Active Level	Description
S_RXVALIDHS0	O	High	High-Speed Receive Data including garbage data during HS- RX trail period (S_RXBYTECLKHS0 , Level High).
S_RXCLKESC0	O	N/A	Escape mode Receive Clock (this is edge detection clock from both DP and DN lines).
S_RXULPSESC0	O	High	Escape Ultra Low Power Receive mode (S_RXCLKESC0 , Level High).
S_FORCERXMODE0	I	High	Force Lane into Receive mode/Wait for Stop state (asynchronous, Level High)
S_ULPSACTIVENOT0	O	Low	Ultra Low Power State (not) Active (S_TXCLKESC0 or S_RXCLKESC0 , Level Low).
S_STOPSTATE0	O	High	Data Lane is in Stop state (asynchronous, Level High).
S_ENABLE0	I	High	Enable Data Lane Module (asynchronous, Level High). NOTE: You must assert to Low state during power-up sequence for a stable initialization.
S_ERR0SOTHS	O	High	Start-of-Transmission Error (S_RXBYTECLKHS0 , Pulse High).
S_ERR0ESC	O	High	Escape Entry Error (S_RXCLKESC0 , Level High).
S_ERR0CONTROL	O	High	Control Error (S_RXCLKESC0 , Pulse High).

13.6.8.12 Slave Data1 Lane: PPI Signals

The following table describes the PPI signals of Slave Data1 Lane.

Table 13-46. Slave Data1 Lane: PPI Signals

Name	I/O	Active Level	Description
S_RXBYTECLKHS1	O	N/A	"RXDDR/4" Clock. For HS Digital operation and for PPI. 10 MHz to 262.5 MHz

Table continues on the next page...

Table 13-46. Slave Data1 Lane: PPI Signals (continued)

Name	I/O	Active Level	Description
			Periodic jitter(@D-PHY): tS_RXBYTECLKHS1 * +/-5.5%
S_RXDATAHS1[7:0]	O	N/A	High-Speed Receive Data (S_RXBYTECLKHS1 , captured). It matches accurately with M_TXDATAHS[7:0] bits order.
S_RXACTIVEHS1	O	High	High-Speed Receive Active (S_RXBYTECLKHS1 , Level High).
S_RXVALIDHS1	O	High	High-Speed Receive Data including garbage data during HS- RX trail period (S_RXBYTECLKHS1 , Level High).
S_RXCLKESC1	O	N/A	Escape mode Receive Clock (this is the edge detection clock from both DP and DN lines).
S_RXULPSESC1	O	High	Escape Ultra Low Power Receive mode (S_RXCLKESC1 , Level High).
S_FORCERXMODE1	I	High	Force Lane into Receive mode/Wait for Stop state (asynchronous, Level High).
S_ULPSACTIVENOT1	O	Low	Ultra Low Power State (not) Active (S_TXCLKESC1 or S_RXCLKESC1 , Level Low).
S_STOPSTATE1	O	High	Data Lane is in Stop state (asynchronous, Level High).
S_ENABLE1	I	High	Enable Data Lane Module (asynchronous, Level High) NOTE: You must assert to Low state during power-up sequence for a stable initialization.
S_ERR1SOTHS	O	High	Start-of-Transmission Error (S_RXBYTECLKHS1 , Pulse High).
S_ERR1ESC	O	High	Escape Entry Error (S_RXCLKESC1 , Level High).
S_ERR1CONTROL	O	High	Control Error (S_RXCLKESC1 , Pulse High).

13.6.8.13 Slave Data2 Lane: PPI Signals

The following table describes the PPI signals of Slave Data2 Lane.

Table 13-47. Slave Data2 Lane: PPI Signals

Name	I/O	Active Level	Description
S_RXBYTECLKHS2	O	N/A	"RXDDR/4" Clock. For HS Digital operation and for PPI. 10 MHz to 262.5 MHz Periodic jitter(@D-PHY): tS_RXBYTECLKHS2 * +/-5.5%
S_RXDATAHS2[7:0]	O	N/A	High-Speed Receive Data (S_RXBYTECLKHS2 , captured). It matches accurately with M_TXDATAHS[7:0] bits order.
S_RXACTIVEHS2	O	High	High-Speed Receive Active (S_RXBYTECLKHS2 , Level High).
S_RXVALIDHS2	O	High	High-Speed Receive Data including garbage data during HS- RX trail period (S_RXBYTECLKHS2 , Level High).
S_RXCLKESC2	O	N/A	Escape mode Receive Clock (This is edge detection clock from both DP and DN lines).
S_RXULPSESC2	O	High	Escape Ultra Low Power Receive mode (S_RXCLKESC2 , Level High).
S_FORCERXMODE2	I	High	Force Lane into Receive mode/Wait for Stop state (asynchronous, Level High).
S_ULPSACTIVENOT2	O	Low	Ultra Low Power State (not) Active (S_TXCLKESC2 or S_RXCLKESC2 , Level Low).
S_STOPSTATE2	O	High	Data Lane is in Stop state (asynchronous, Level High).
S_ENABLE2	I	High	Enable Data Lane Module (asynchronous, Level High). NOTE: You should assert to Low state during power-up sequence for stable initialization.
S_ERR2SOTHS	O	High	Start-of-Transmission Error (S_RXBYTECLKHS2 , Pulse High).
S_ERR2ESC	O	High	Escape Entry Error (S_RXCLKESC2 , Level High).

Table continues on the next page...

Table 13-47. Slave Data2 Lane: PPI Signals (continued)

Name	I/O	Active Level	Description
S_ERR2CONTROL	O	High	Control Error (S_RXCLKESC2 , Pulse High).

13.6.8.14 Slave Data3 Lane: PPI Signals

The following table describes the PPI signals of Slave Data3 Lane.

Table 13-48. Slave Data3 Lane: PPI Signals

Name	I/O	Active Level	Description
S_RXBYTECLKHS3	O	N/A	"RXDDR/4" Clock. For HS Digital operation and for PPI. 10 MHz to 262.5 MHz Periodic jitter(@D-PHY): tS_RXBYTECLKHS3 * +/-5.5%
S_RXDATAHS3[7:0]	O	N/A	High-Speed Receive Data (S_RXBYTECLKHS3 , captured). It matches accurately with M_TXDATAHS[7:0] bits order.
S_RXACTIVEHS3	O	High	High-Speed Receive Active (S_RXBYTECLKHS3 , Level High).
S_RXVALIDHS3	O	High	High-Speed Receive Data including garbage data during HS-RX trail period (S_RXBYTECLKHS3 , Level High).
S_RXCLKESC3	O	N/A	Escape mode Receive Clock (This is edge detection clock from both DP and DN lines).
S_RXULPSESC3	O	High	Escape Ultra Low Power Receive mode (S_RXCLKESC3 , Level High).
S_FORCERXMODE3	I	High	Force Lane into Receive mode/Wait for Stop state (asynchronous, Level High).
S_ULPSACTIVENOT3	O	Low	Ultra Low Power State (not) Active (S_TXCLKESC3 or S_RXCLKESC3 , Level Low).
S_STOPSTATE3	O	High	Data Lane is in Stop state (asynchronous, Level High).

Table continues on the next page...

Table 13-48. Slave Data3 Lane: PPI Signals (continued)

Name	I/O	Active Level	Description
S_ENABLE3	I	High	Enable Data Lane Module (asynchronous, Level High). NOTE: You must assert to Low state during power-up sequence for a stable initialization.
S_ERR3SOTHS	O	High	Start-of-Transmission Error (S_RXBYTECLKHS3 , Pulse High).

13.6.8.15 Control Register: PPI Signals

13.6.8.15.1 Timing Control Register of High-Speed Data Transmission

Before High-Speed Data Transmission, you should set these registers. For more information,see register descriptions.

The following table describes the Timing Control Registers of High-Speed Data Transmission.

Table 13-49. Timing Control Registers of High-Speed Data Transmission

Name	I/O	Active Level	Description
M_TCLKPRPRCTL[7:0]	I	N/A	Master Clock Lane Control Register for TCLK-PREPARE.
M_TCLKZEROCTL[7:0]	I	N/A	Master Clock Lane Control Register for TCLK-ZERO.
M_TCLKPOSTCTL[7:0]	I	N/A	Master Clock Lane Control Register for TCLK-POST.
M_TCLKTRAILCTL[7:0]	I	N/A	Master Clock Lane Control Register for TCLK-TRAIL.
M_THSPRPRCTL[7:0]	I	N/A	Master Data Lanes Control Register for THS-PREPARE.
M_THSZEROCTL[7:0]	I	N/A	Master Data Lanes Control Register for THS-ZERO.
M_THSTRAILCTL[7:0]	I	N/A	Master Data Lanes Control Register for THS-TRAIL.
M_TLPXCTL[7:0]	I	N/A	Master Clock and Data Lanes Control Register for TLPX.
M_THSEXITCTL[7:0]	I	N/A	Master Clock and Data Lanes Control Register for THS-EXIT.

Table continues on the next page...

Table 13-49. Timing Control Registers of High-Speed Data Transmission (continued)

Name	I/O	Active Level	Description
S_CLKSETTLECTL[1:0]	I	N/A	Slave Clock Lane Control Register for TCLK-SETTLE. 2'b0x = 110 ns to 280ns (v0.87 to v1.00) 2'b10 = 150 ns to 430ns (v0.83 to v0.86) 2'b11 = 60 ns to 140ns (v0.82)
S_HSSETTLECTL[7:0]	I	N/A	Slave Clock Lane Control Register for THS-SETTLE.

13.6.8.15.2 Reset and Test Mode Signals

The following table describes the reset and test mode signals.

Table 13-50. Reset and Test Mode: PPI Signals

Name	I/O	Active Level	Description
M_RESETN	I	Low	Master Clock and Data Lane Reset (Active Low)
S_RESETN	I	Low	Slave Clock and Data Lane Reset (Active Low)
INTERNAL_LOOPBACK_EN	I	High	Internal or External Loopback selection signal • 1'b0 = External Loopback Mode (default) • 1'b1 = Internal Loopback Mode
SCAN_MODE	I	High	Scan Test mode selection
M_SCAN_CLK_0	I	N/A	Master Data0 Lane Scan Test Mode Clock
M_SCAN_CLK_1	I	N/A	Master Data1 Lane Scan Test Mode Clock
M_SCAN_CLK_2	I	N/A	Master Data2 Lane Scan Test Mode Clock
M_SCAN_CLK_3	I	N/A	Master Data3 Lane Scan Test Mode Clock
M_SCAN_IN_0	I	N/A	Master Data0 Lane Scan Test Mode Input
M_SCAN_IN_1	I	N/A	Master Data1 Lane Scan Test Mode Input
M_SCAN_IN_2	I	N/A	Master Data2 Lane Scan Test Mode Input
M_SCAN_IN_3	I	N/A	Master Data3 Lane Scan Test Mode Input
M_SCAN_OUT_0	O	N/A	Master Data0 Lane Scan Test Mode Output

Table continues on the next page...

Table 13-50. Reset and Test Mode: PPI Signals (continued)

Name	I/O	Active Level	Description
M_SCAN_OUT_1	O	N/A	Master Data1 Lane Scan Test Mode Output
M_SCAN_OUT_2	O	N/A	Master Data2 Lane Scan Test Mode Output
M_SCAN_OUT_3	O	N/A	Master Data3 Lane Scan Test Mode Output
S_SCAN_CLK_0	I	N/A	Slave Data0 Lane Scan Test Mode Clock
S_SCAN_CLK_1	I	N/A	Slave Data1 Lane Scan Test Mode Clock
S_SCAN_CLK_2	I	N/A	Slave Data2 Lane Scan Test Mode Clock
S_SCAN_CLK_3	I	N/A	Slave Data3 Lane Scan Test Mode Clock
S_SCAN_IN_0	I	N/A	Slave Data0 Lane Scan Test Mode Input
S_SCAN_IN_1	I	N/A	Slave Data1 Lane Scan Test Mode Input
S_SCAN_IN_2	I	N/A	Slave Data2 Lane Scan Test Mode Input
S_SCAN_IN_3	I	N/A	Slave Data3 Lane Scan Test Mode Input
S_SCAN_OUT_0	O	N/A	Slave Data0 Lane Scan Test Mode Output
S_SCAN_OUT_1	O	N/A	Slave Data1 Lane Scan Test Mode Output
S_SCAN_OUT_2	O	N/A	Slave Data2 Lane Scan Test Mode Output
S_SCAN_OUT_3	O	N/A	Slave Data3 Lane Scan Test Mode Output
TEST_MODE[2:0]	I	N/A	Test Mode selection
TEST_BAND[3:0]	I	N/A	Select the band by High Speed Clock Frequency for Loopback Test • Data Rate 80 Mbps to 159 Mbps = 4'd0 • Data Rate 160 Mbps to 249 Mbps = 4'd1 • Data Rate 250 Mbps to 299 Mbps = 4'd2 • Data Rate 300 Mbps to 399 Mbps = 4'd3 • Data Rate 400 Mbps to 499 Mbps = 4'd4 • ... • Data Rate 1200 Mbps to 1299 Mbps = 4'd12 • Data Rate 1300 Mbps to 1399 Mbps = 4'd13

Table continues on the next page...

Table 13-50. Reset and Test Mode: PPI Signals (continued)

Name	I/O	Active Level	Description
			• Data Rate 1400 Mbps to 1499 Mbps = 4'd14 • Data Rate 1500 Mbps to 2100 Mbps = 4'd15 • For more information, refer to the "TEST_BAND" sheet in the LN14LPP_MIPI D-PHY_Supplement file. Contact the Samsung Engineers to get the document.
TEST_CLK	I	N/A	Test Mode Clock. All Lanes.
TEST_IN[1:0]	I	High	Test Mode Input. All Lanes.
M_TEST_OUT_C[1:0]	O	N/A	Master Clock Lane Test Mode Output
M_TEST_OUT_0[1:0]	O	N/A	Master Data0 Lane Test Mode Output
M_TEST_OUT_1[1:0]	O	N/A	Master Data1 Lane Test Mode Output
M_TEST_OUT_2[1:0]	O	N/A	Master Data2 Lane Test Mode Output
M_TEST_OUT_3[1:0]	O	N/A	Master Data3 Lane Test Mode Output
M_PLL_LOCK_INT	O	N/A	Dummy (NC)
M_PLL_SER_CLK	O	N/A	Dummy (NC)
S_TEST_OUT_C[1:0]	O	N/A	Slave Clock Lane Test Mode Output
S_TEST_OUT_0[1:0]	O	N/A	Slave Data0 Lane Test Mode Output
S_TEST_OUT_1[1:0]	O	N/A	Slave Data1 Lane Test Mode Output
S_TEST_OUT_2[1:0]	O	N/A	Slave Data2 Lane Test Mode Output
S_TEST_OUT_3[1:0]	O	N/A	Slave Data3 Lane Test Mode Output
S_LOOPBACK_PASS_0	O	High	Slave Data0 Lane Loopback Test Mode Output
S_LOOPBACK_PASS_1	O	High	Slave Data1 Lane Loopback Test Mode Output
S_LOOPBACK_PASS_2	O	High	Slave Data2 Lane Loopback Test Mode Output
S_LOOPBACK_PASS_3	O	High	Slave Data3 Lane Loopback Test Mode Output
S_LOOPBACK_FAIL_0	O	High	Slave Data0 Lane Loopback Test Mode Output

Table continues on the next page...

Table 13-50. Reset and Test Mode: PPI Signals (continued)

Name	I/O	Active Level	Description
S_LOOPBACK_FAIL_1	O	High	Slave Data1 Lane Loopback Test Mode Output
S_LOOPBACK_FAIL_2	O	High	Slave Data2 Lane Loopback Test Mode Output
S_LOOPBACK_FAIL_3	O	High	Slave Data3 Lane Loopback Test Mode Output
S_ATB_OUT_0/1/2/3[31:30]	O	N/A	Dummy
S_ATB_OUT_0/1/2/3[29:24]	O	N/A	RX Skew Calibration results value (Data Lane)
S_ATB_OUT_0/1/2/3[23:22]	O	N/A	Dummy
S_ATB_OUT_0/1/2/3[21:16]	O	N/A	RX Skew Calibration results value (Clock Lane)
S_ATB_OUT_0/1/2/3[15:2]	O	N/A	Dummy
S_ATB_OUT_0/1/2/3[1]	O	N/A	RX Skew Calibration done signal
S_ATB_OUT_0/1/2/3[0]	O	N/A	RX Skew Calibration compare pass signal

13.6.8.16 PPI Requests and Timing Conditions

13.6.8.16.1 PPI Request Conditions

Following are the request conditions invoked by the D-PHY:

- The Correct request timing window is in the Stop state.
- Accepts only one request from protocol.
- D-PHY selects only one request based on the service priority when you assert two or more requests.

13.6.8.16.2 PPI Requests and D-PHY Timing

HS-TX:

- **Latency** - The minimum required FIFO depth is 128, because the HS-TX requests the HS-TX ready latency.
- **Data Lane** - D-PHY ensures $T_{HS-EXIT}$ (minimum 100 ns) timing.
- **Clock Lane** - D-PHY must guarantee the $T_{CLK-PRE}$ and $T_{CLK-POST}$ timings.

NOTE

The protocol layer need not control these timings.

- **HS-2-HS Request Interval** - Not implied.

LP:

- **LP–2–LP Request** - The LP-2_LP request wait for the 2-ESC CLK.

LP and HS:

- **LP–2–HS Request Interval** - LP-2_HS request interval wait for the 2-ESC CLK in the Stop state.
- **HS-2-LP Request Interval** - Not implied.

13.6.8.16.3 Protocol Watchdog Timers

Following are the features of the Protocol Watchdog Timers:

- **HS RX Timeout** - Protocol specific.
- The Protocol times-out when there is no EoT received within a certain period in the HS RX mode.
- **HS TX Timeout** - Protocol specific.
- The maximum transmission length in HS TX is bounded.
- **Escape Mode Timeout** - Protocol specific.
- The Escape mode timeout should be greater than the Escape mode Silence Limit of the other device.
- **Escape Mode Silence Timeout** - Protocol specific.
- A device has a bound length for LP TX-00 during Escape mode, after which the other device may timeout. For example, a display module must have an Escape Mode Silence Limit, after which the host processor can timeout.
- **Turnaround Errors**
- The Protocol times out initiates appropriate action when there is no acknowledgement observed within a certain period of time.

13.6.8.16.4 Abnormal Conditions

The abnormal conditions are:

- M_FORCETXSTOPMODE
- M_FORCERXMODE
- M (S)_ERRCONTENTIONLP0 and M (S)_ERRCONTENTIONLP1

13.6.8.17 Core Interface Timing Diagrams

13.6.8.17.1 Clock Lane: HS-TX and HS-RX Function

The following figure illustrates the HS-TX and HS-RX function of the Clock Lane.

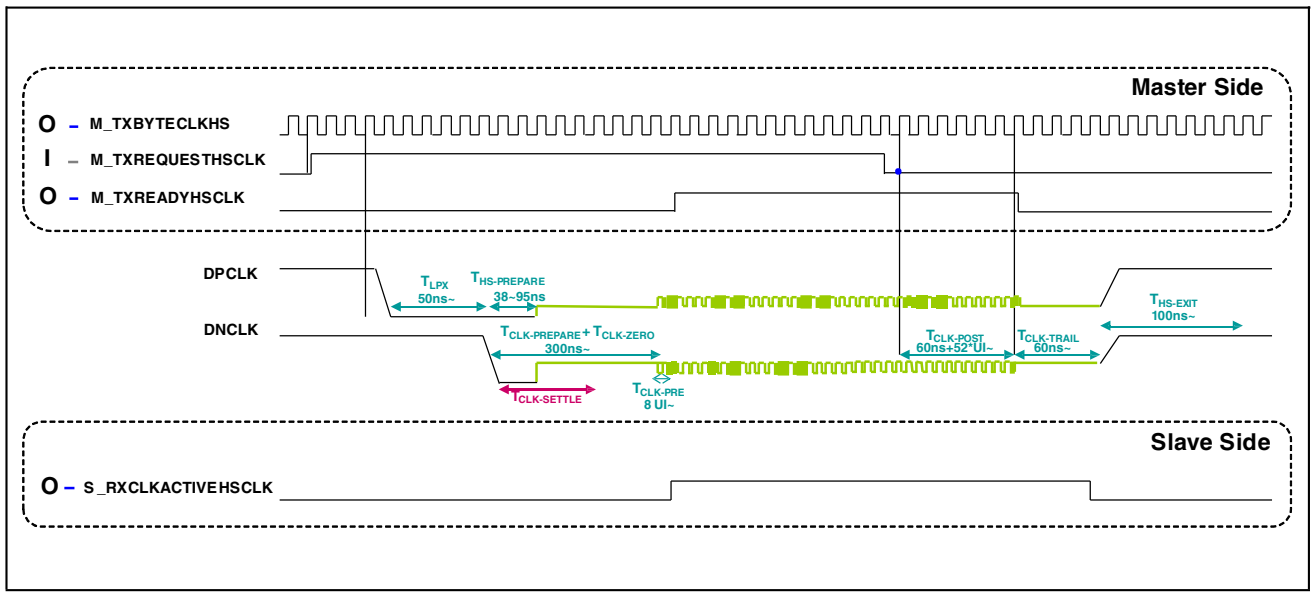


Figure 13-76. Clock Lane: HS-TX and HS-RX Function

13.6.8.17.2 Data Lane: HS-TX and HS-RX Function

The following figure illustrates the HS-TX and HS-RX function of the Data Lane.

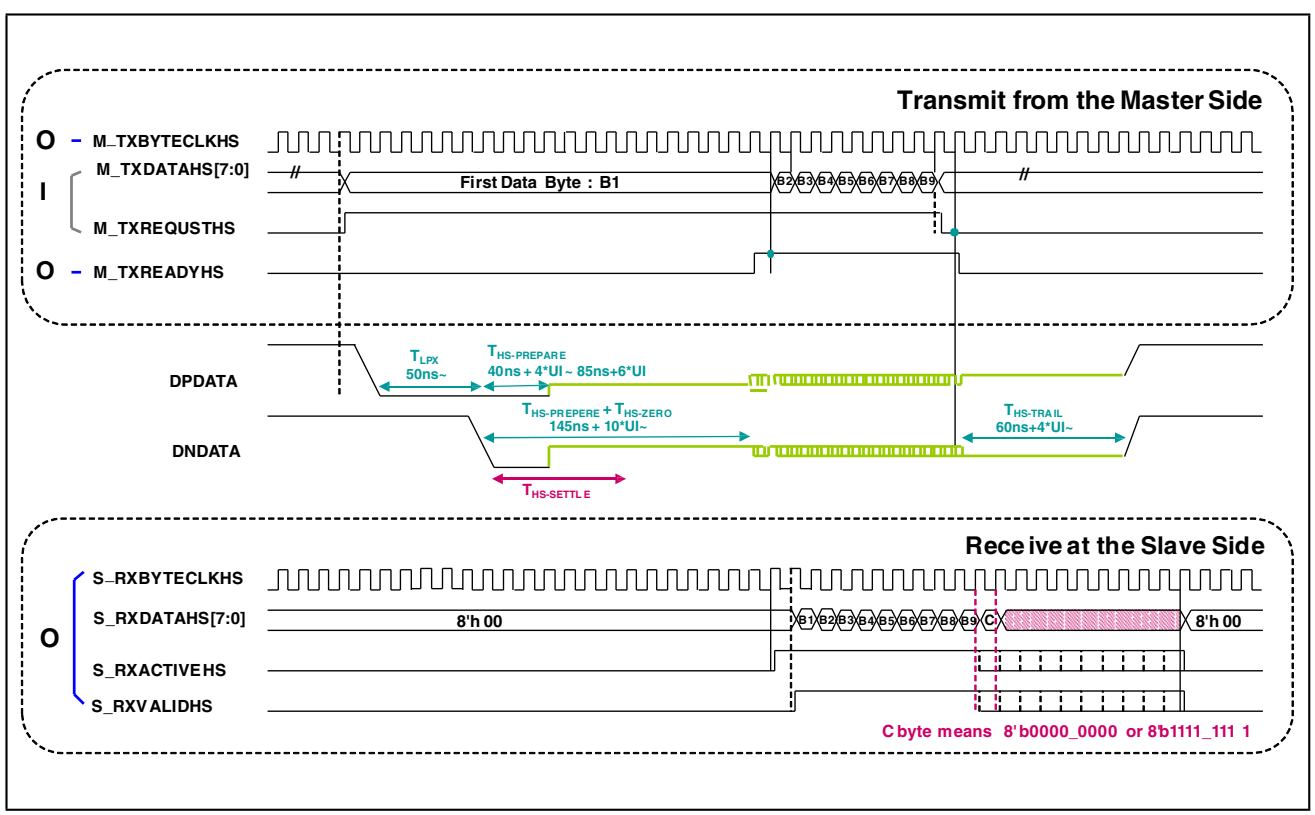


Figure 13-77. Data Lane: HS-TX and HS-RX Function

13.6.8.17.3 Clock Lane: ULPS Function

The following figure illustrates the ULPS function of the Clock Lane.

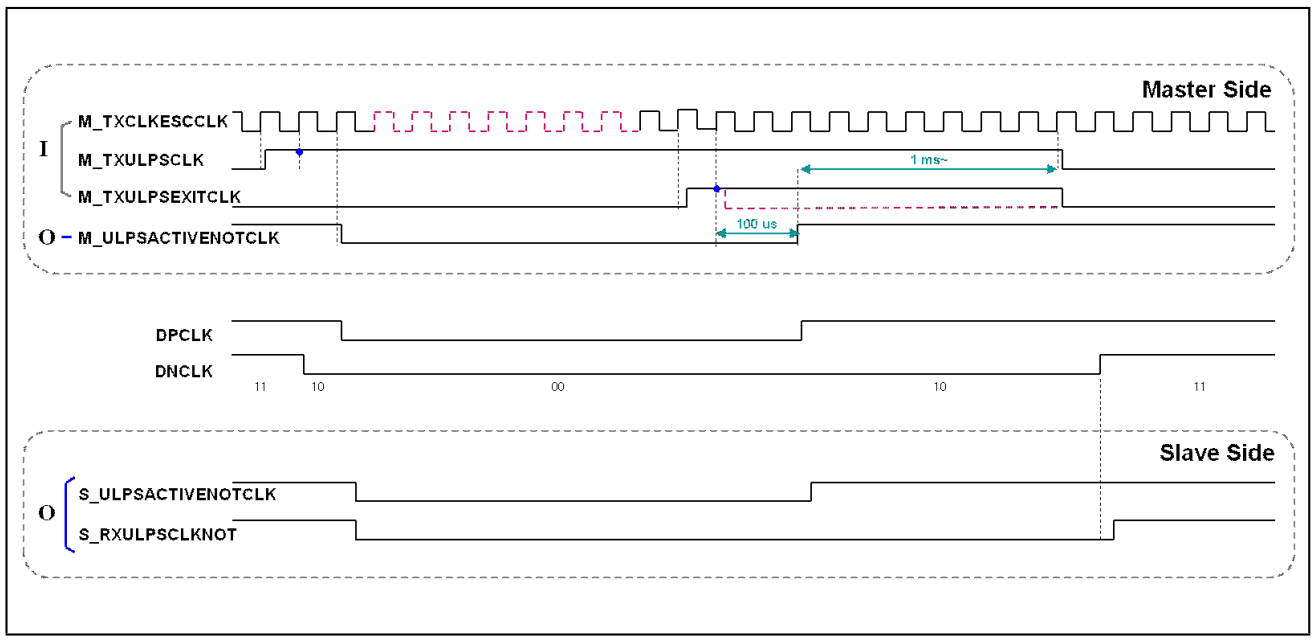


Figure 13-78. Clock Lane: ULPS Function

13.6.8.17.4 Data Lane: ULPS Function

The following figure illustrates the ULPS function of the Data Lane.

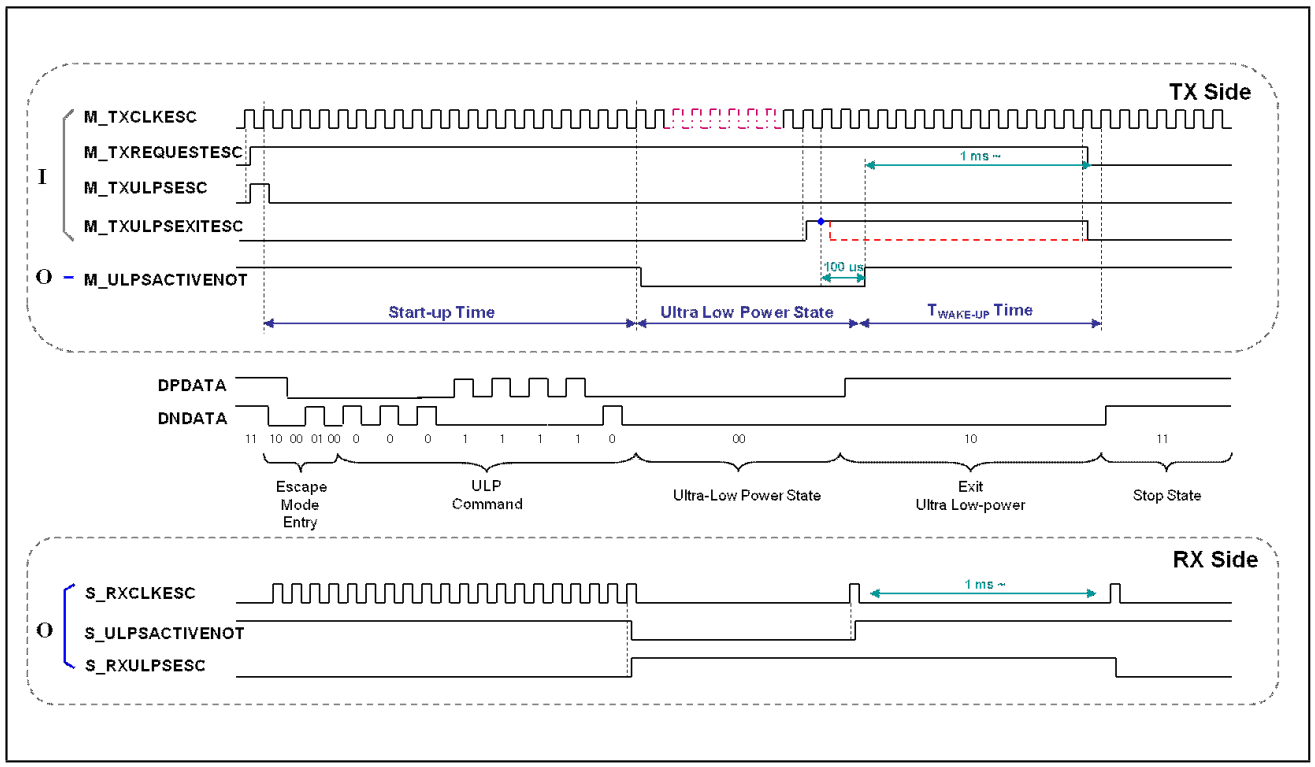


Figure 13-79. Data Lane: ULPS Function

13.6.8.17.5 Data Lane: LP-TX and LP-RX Function

The following figure illustrates the LP-TX and LP-RX function of the Data Lane.

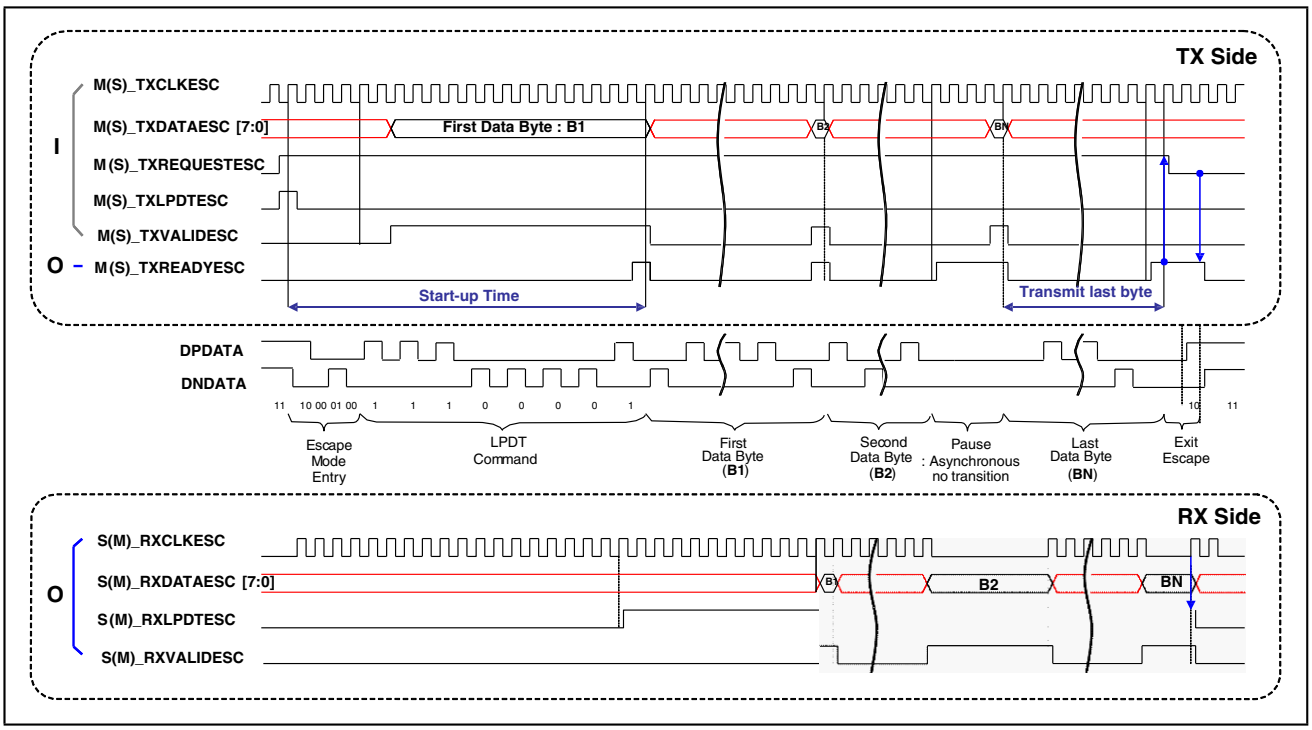


Figure 13-80. Data Lane: LP-TX and LP-RX Function

13.6.8.17.6 Data Lane: Remote Trigger Reset

The following figure illustrates the remote trigger reset of the Data Lane.

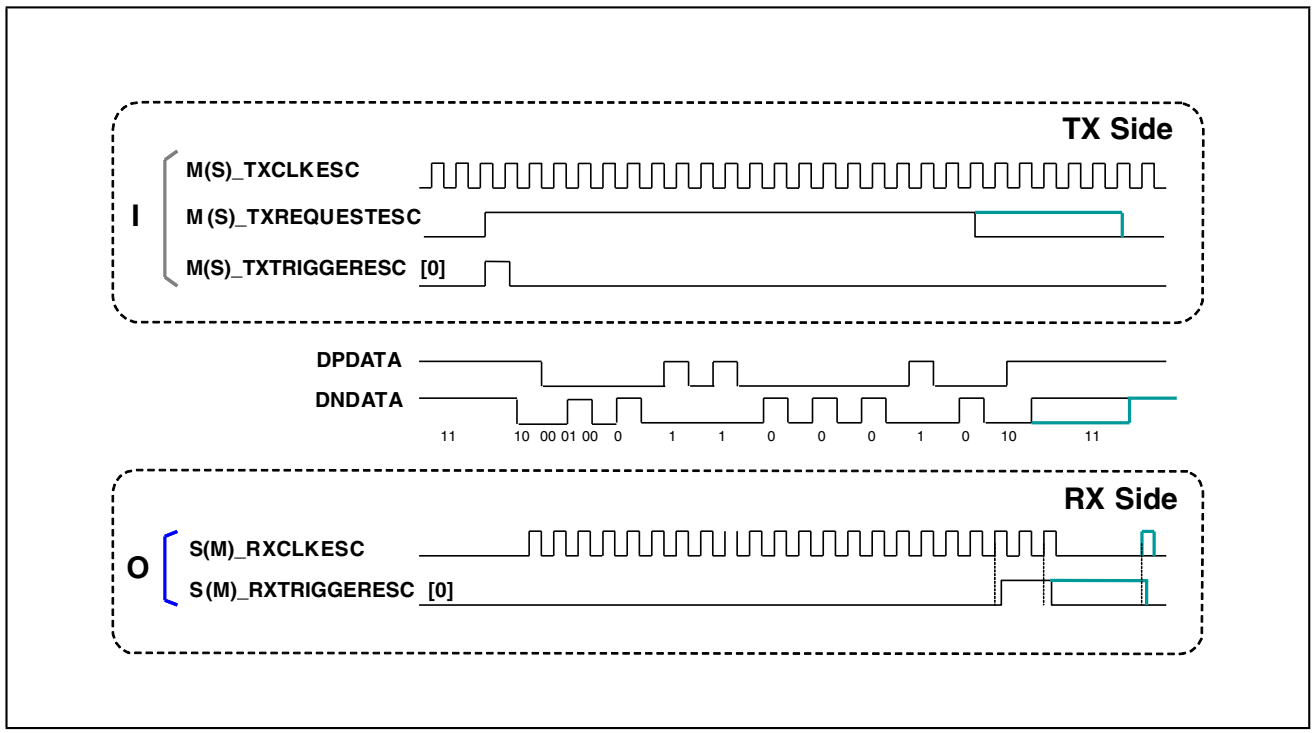


Figure 13-81. Data Lane: Remote Trigger Reset

13.6.8.17.7 Data Lane: Turn Around

The following figure illustrates the turnaround of the Data Lane.

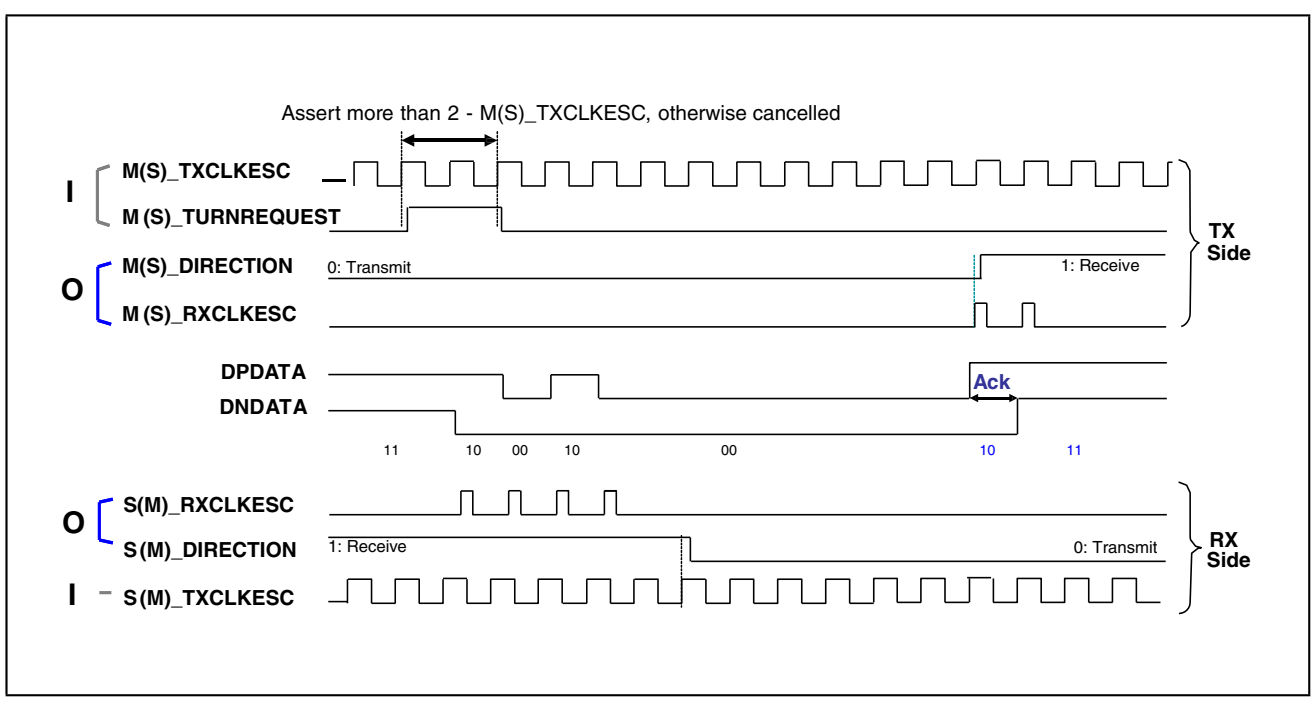


Figure 13-82. Data Lane: Turn Around

13.6.8.17.8 Skew Calibration

The following figure illustrates the Skew Calibration sequence.

The protocol should assert the M_TXSKEWCALHS synchronized M_TXBYTECLKHS for start Skew Calibration. PHY-Protocol Interface output signals of the Slave are 1'b0 in the Skew Calibration mode.

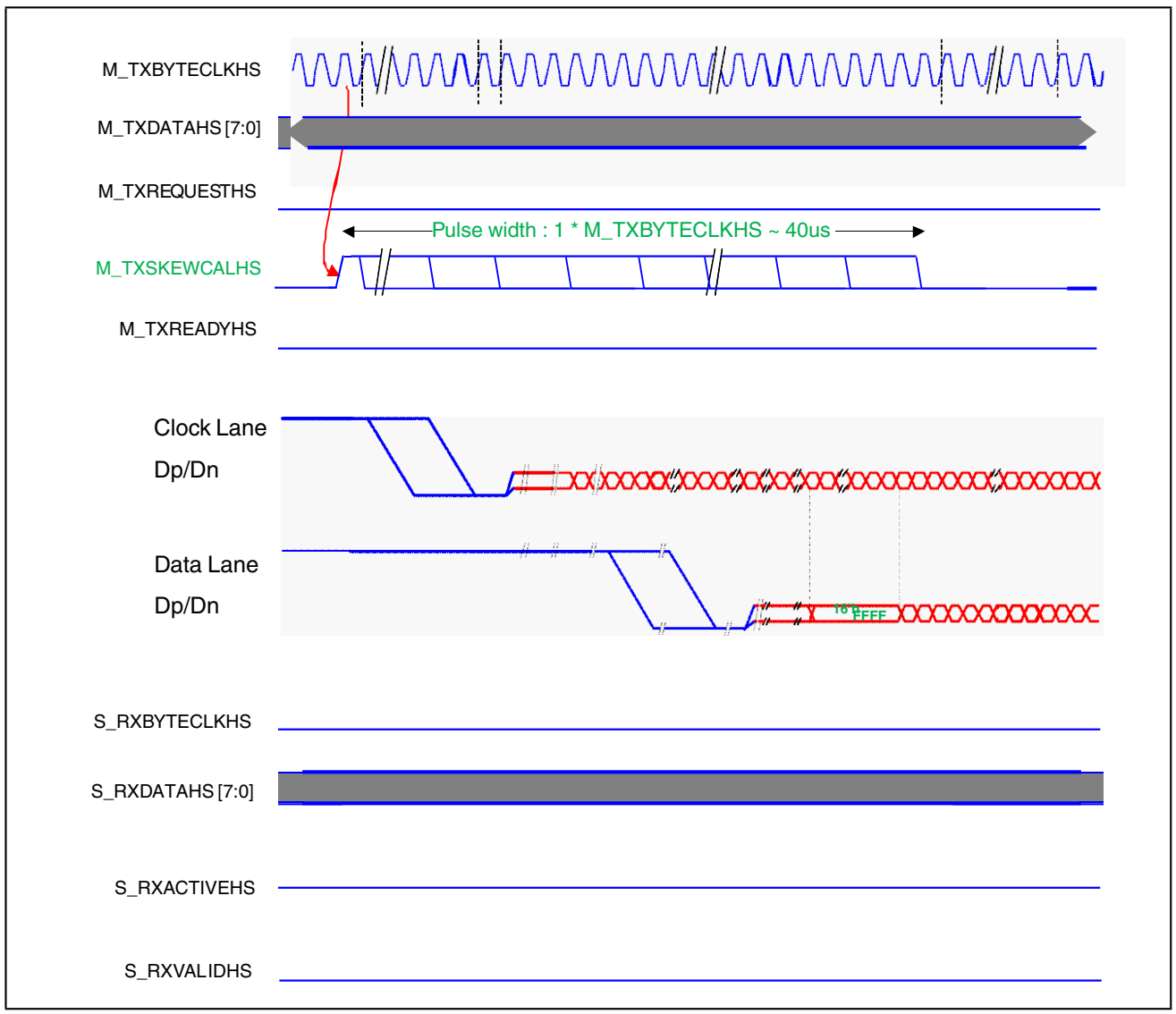


Figure 13-83. Skew Calibration

13.6.8.18 Physical Layer (PHY)

The Samsung D-PHY consists of:

- One Common block
- Master data four lane blocks
- Master clock lane block
- Slave data four lane blocks
- Slave clock lane block

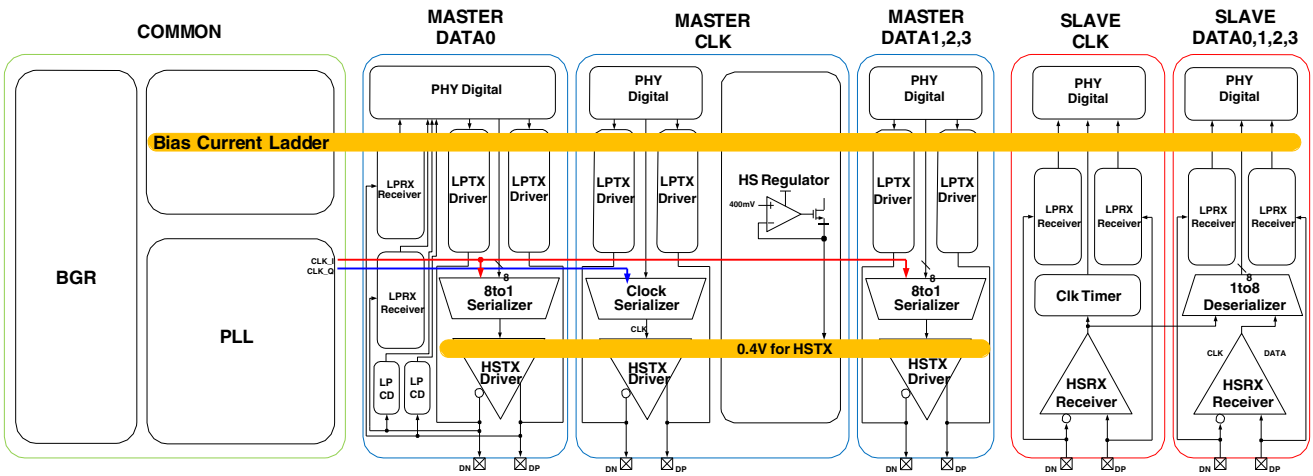


Figure 13-84. Physical Layer

The MIPI D-PHY requires one off-chip 2.2nF capacitor for Master 0.4V regulator. The LANE block consists of serializer, de-serializer, output driver block, LP-TX, and LP-RX block. The data is serialized and transmitted through DP/DN pads by voltage-mode driver. Detail operation and features of each block are described in following sections.

13.6.8.18.1 Common

The Common block consists of a band-gap reference (BGR), bias generator, and PLL. Detailed information on each block is described in the following sections.

13.6.8.18.2 BGR and bias generator

The BGR circuit generates high-accuracy reference voltage of 820mV across PVT variation. A bias generator makes internal-resistor (RMRES)-referred currents, and these currents are distributed to each block in Common and Lanes. The internal resistor-referred current is generated using internally integrated replica resistor to compensate resistance error across PVT variation.

13.6.8.18.3 Phase Locked Loop (PLL)

The PLL in the Common synthesizes high-speed clock, which is used for TX serializer from a reference clock. The nominal reference frequency is 26MHz.

In the PHY, the high frequency clock from PLL is used in the serialization of TX data. The following are the main features of the PLL:

- Supports output frequency from 80MHz to 2.1GHz clock
- Supports pre-defined and programmable divider setting for MIPI D-PHY specification
- Supports spread spectrum clocking (SSC) with sigma-delta modulated fractional divider.

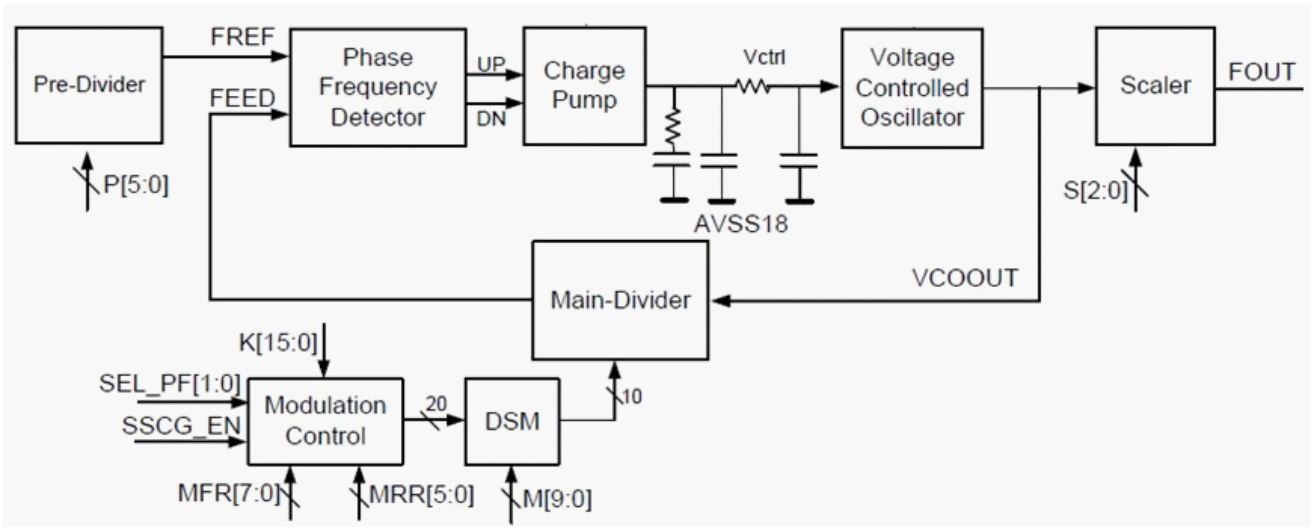


Figure 13-85. Phase Locked Loop (PLL)

13.6.8.18.4 DPHY PLL

The MIPI DSI DPHY PLL is a high performance PLL based frequency synthesizer that incorporates a independent output divider. The PLL block diagram is shown below.

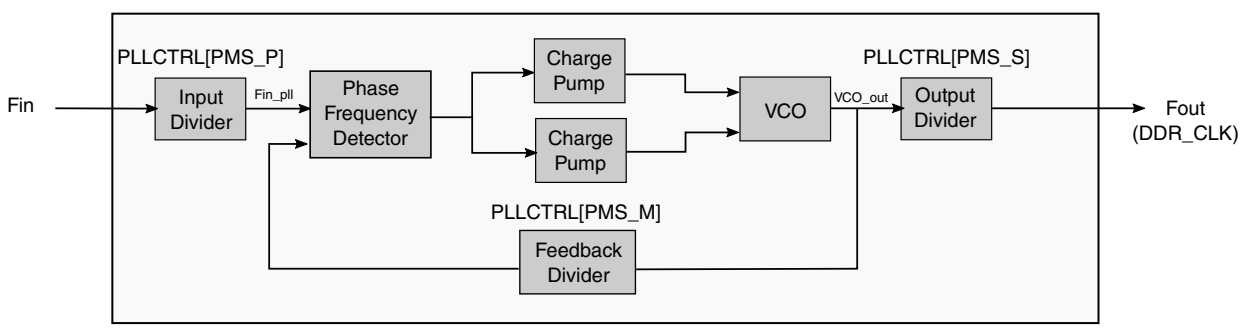


Figure 13-86. MIPI DSI DPHY PLL Block Diagram

The DPHY PLL input clock is Fin. The PLL output (Fout) multiplies the input frequency (Fin) by `(M / (P * 2^S))`.

NOTE

Setting P or M divider values as zero can cause malfunction of the PLL.

NOTE

The P and M divider values must be programmed to maintatin VCO stability. The VCO output frequency (VCO_out) range of the MIPI PLL is from 350 MHz to 750 MHz.

Table 13-51. DPHY PLL Parameters

Parameter	Description	Frequency Range (MHz)
Fin	Input Frequency	6-200
Fin_pll	P divider frequency	6-12
VCO_out	(M*Fin) / P	350-750
Fout	(M*Fin)/(P*2^S)	37.5-750
P	Input Divider	1-33
M	Feedback Divider	25-125
S	Output Divider (2^S)	1,2,4,8

13.6.8.18.5 Serializer

The serializer accepts 8-bit data from the PHY digital which is synchronized to TX byte clock. The parallel data is converted to the high-speed serial data. Finally, serializer transmits the serial data through the driver.

13.6.8.18.6 HS TX Driver

The driver generates baud-rate voltage waveform on TX output pads. The driver comprises of pre-driver and main-driver. The output data of serializer are delivered to the main-driver via the pre-driver. The main driver is based on voltage-mode driver to achieve a small power consumption and voltage swing.

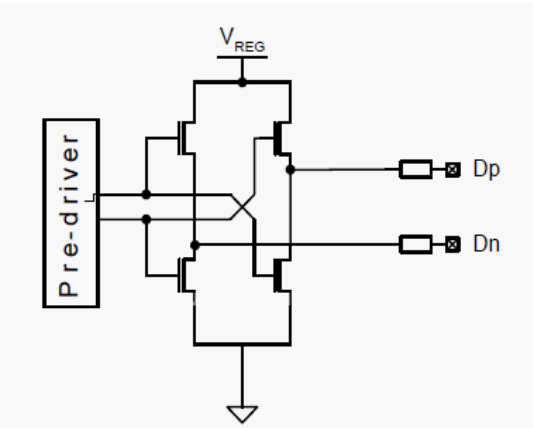


Figure 13-87. HS TX Driver

13.6.8.18.7 LP TX Driver

In the low power mode, LP TX driver sends the low frequency data under 10Mbps. Generally LP-11 stop state and HS enter sequence (LP-11 ◇ LP-01 ◇ LP-00) are used mostly.

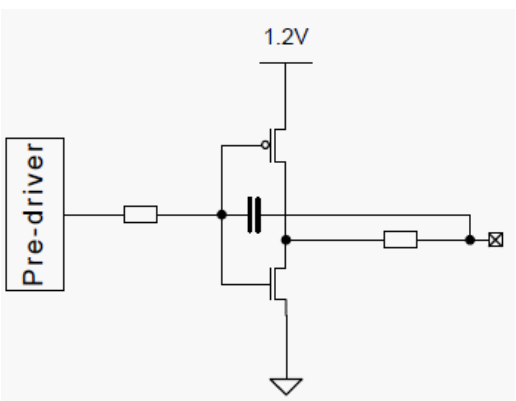


Figure 13-88. LP TX Driver

13.6.8.18.8 De-Serializer

In the BIST (Built-In Self-Test) mode, the serial data from the serializer is fed back to the de-serializer. And in the real mode, the real data from HS-RX receiver is sent to the de-serializer. De-serializer converted serial data to 8- bit parallel data and transferred them into PHY digital circuit for sending data to link or checking bit error at BIST mode.

13.6.8.18.9 HS RX Receiver

The high speed data from the MIPI D-PHY master has the MIPI D-PHY electrical characteristics. HS RX receiver convert it to the TTL CMOS level and send it to de-serializer for making the low parallel data.

13.6.8.18.10 LP RX Receiver

In the low power mode, LP RX receiver make esc clock from LP signal and converts the voltage level from 1.2V level to TTL CMOS logic level. And it sends them to PHY digital for command interpretation. And LP RX receiver always watches the DP/DN signal level. If DP/DN goes to LP-11, PHY recognizes Stop-state, and enters to low power mode.

13.6.9 Unused pin and power integration guide when MIPI IP into chip is not used

13.6.9.1 Unused signal pins

On chip guide:

- The unused input pins should be connected to the ground (incase of the active high operation) or the power (incase of the active low operation).
- The unused output pins should be connected nothing.

Off chip guide:

- The unused input pins (Slave DP/DN, Slave CLKP/CLKN) are tied to ground
- The unused output pins (Master DP/DN, Master CLKP/CLKN) are connected to nothing.
- M_VREG_0P4V pin into the unused Master block are connected to nothing. The external capacitor does not need to be connected.

13.6.9.2 Power into unused block

When the entire MIPI block is unused:

- Power pins are connected to nothing.
- GND pins are tied to ground.

When power is shared, some MIPI block is unused and some MIPI block is used:

- Power pins are tied to power.
- GND pins are tied to ground.

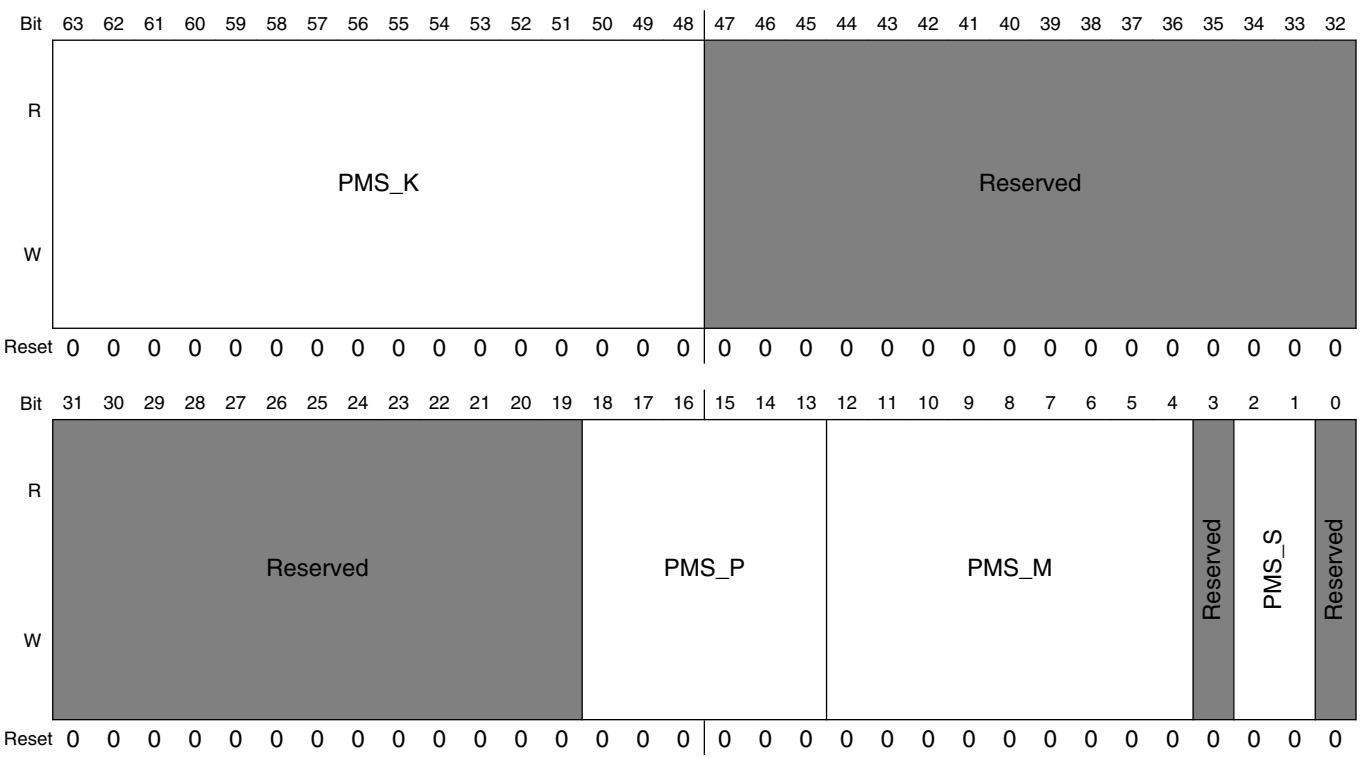
13.6.10 MIPI PHY Memory Map/Register Definition

MIPI_DPHY memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	Master PLL PMS Value setting Register (MIPI_DPHY_M_PLLPMS)	64	R/W	0000_0000 _0000_0000h	13.6.10.1/ 4151
8	Master PLL Control Register (MIPI_DPHY_M_PLLCTL)	64	R/W	0000_0000 _0000_0000h	13.6.10.2/ 4153
10	Master and Slave DPHY Control Register (MIPI_DPHY_B_DPHYCTL)	64	R/W	0000_0000 _0000_0000h	13.6.10.3/ 4154
18	Master and Slave DPHY Control Register (MIPI_DPHY_M_DPHYCTL)	64	R/W	See section	13.6.10.4/ 4158
20	Master and Slave DPHY Control Register (MIPI_DPHY_S_DPHYCTL)	64	R/W	0000_0000 _0000_0000h	13.6.10.5/ 4162

13.6.10.1 Master PLL PMS Value setting Register (MIPI_DPHY_M_PLLPMS)

Address: 0h base + 0h offset = 0h



MIPI_DPHY_M_PLLPMS field descriptions

Field	Description
63–48 PMS_K	Specifies the PLL PMS value for the K divider Please refer to the topic DPHY PLL for more information.
47–19 -	This field is reserved. -
18–13 PMS_P	Specifies the PLL PMS value for the P divider **NOTE:** The programmable divider range should be within 1 to 33 to ensure PLL stability. The P divider value should also ensure the input frequency (Fin) is divided down between the range of 6 MHz to 12 MHz. **NOTE:** The M and P divider values should be considered together to ensure VCO ouput frequency (VCO_out) range is between 350 MHz to 750 MHz. Please refer to the topic DPHY PLL for more information. 000000 Do not program, can cause malfunction 000001 Divide by 1 000010 Divide by 2 000011 Divide by 3 000100 Divide by 4 000101 Divide by 5 : 011110 Divide by 30 011111 Divide by 31 100000 Divide by 32 100001 Divide by 33
12–4 PMS_M	Specifies the PLL PMS value for the M divider **NOTE:** The programmable divider range should be within 25 to 125 to ensure PLL stability. **NOTE:** The M and P divider values should be considered together to ensure VCO ouput frequency (VCO_out) range is between 350 MHz to 750 MHz. Please refer to the topic DPHY PLL for more information. 000000000 Do not program, can cause malfunction 000011001 Divide by 25 000011010 Divide by 26 000011011 Divide by 27 000011100 Divide by 28 000011101 Divide by 29 : 001111010 Divide by 122 001111011 Divide by 123 001111100 Divide by 124 001111101 Divide by 125
3 -	This field is reserved.
2–1 PMS_S	Specifies the PLL PMS value for the S divider Please refer to the topic DPHY PLL for more information.

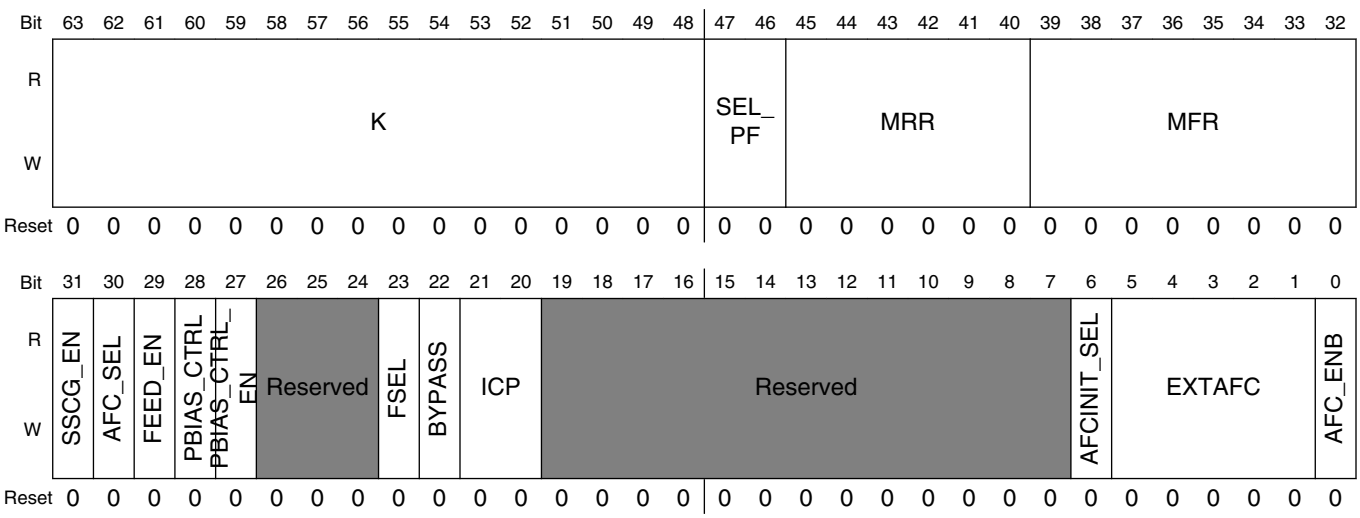
Table continues on the next page...

MIPI_DPHY_M_PLLPMS field descriptions (continued)

Field	Description
	00 Divide by 1 01 Divide by 2 10 Divide by 4 11 Divide by 8
0 -	This field is reserved.

13.6.10.2 Master PLL Control Register (MIPI_DPHY_M_PLLCTL)

Address: 0h base + 8h offset = 8h



MIPI_DPHY_M_PLLCTL field descriptions

Field	Description
63–48 K	
47–46 SEL_PF	
45–40 MRR	
39–32 MFR	
31 SSCG_EN	
30 AFC_SEL	
29 FEED_EN	

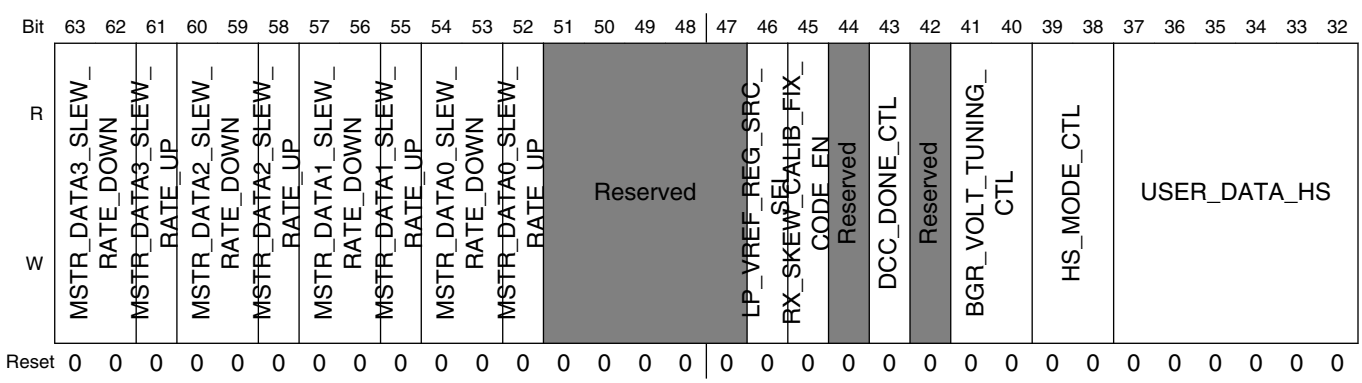
Table continues on the next page...

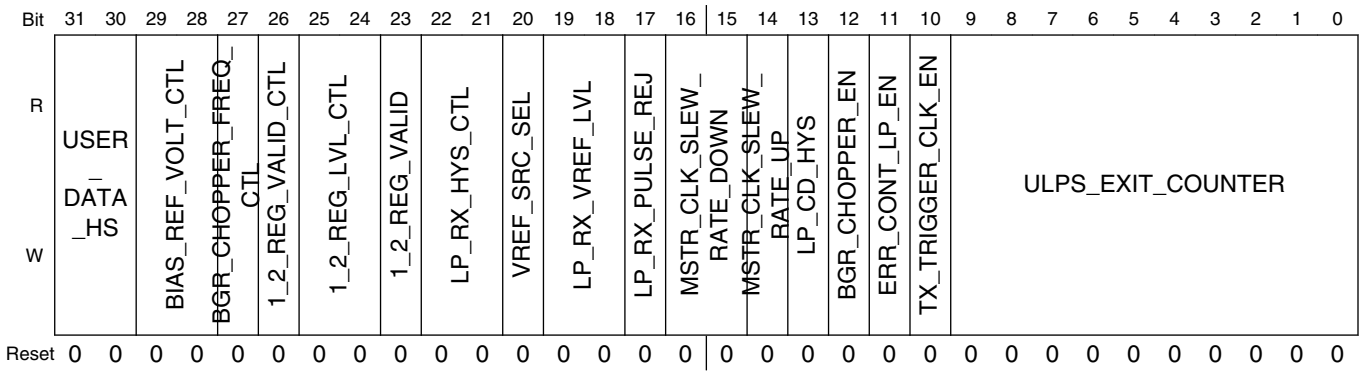
MIPI_DPHY_M_PLLCTL field descriptions (continued)

Field	Description
28 PBIAS_CTRL	
27 PBIAS_CTRL_EN	
26–24 -	This field is reserved. -
23 FSEL	
22 BYPASS	
21–20 ICP	
19–7 -	This field is reserved. -
6 AFCINIT_SEL	
5–1 EXTAFC	
0 AFC_ENB	Automatic Frequency Control Enable/Disable 0 AFC is enabled 1 AFC is disabled

13.6.10.3 Master and Slave DPHY Control Register (MIPI_DPHY_B_DPHYCTL)

Address: 0h base + 10h offset = 10h





MIPI_DPHY_B_DPHYCTL field descriptions

Field	Description
63–62 MSTR_DATA3_SLEW_RATE_DOWN	Master Data3 Lane's LP-TX Driver Slew Rate Down Control 00 No change 01 Decrease the slew rate by about 15% 10 Decrease the slew rate by about 15% 11 Decrease the slew rate by about 30%
61 MSTR_DATA3_SLEW_RATE_UP	Master Data3 Lane's LP-TX Driver Slew Rate Up Control 0 No change 1 Slew Rate UP
60–59 MSTR_DATA2_SLEW_RATE_DOWN	Master Data2 Lane's LP-TX Driver Slew Rate Down Control 00 No change 01 Decrease the slew rate by about 15% 10 Decrease the slew rate by about 15% 11 Decrease the slew rate by about 30%
58 MSTR_DATA2_SLEW_RATE_UP	Master Data2 Lane's LP-TX Driver Slew Rate Up Control 0 No change 1 Slew Rate UP
57–56 MSTR_DATA1_SLEW_RATE_DOWN	Master Data1 Lane's LP-TX Driver Slew Rate Down Control 00 No change 01 Decrease the slew rate by about 15% 10 Decrease the slew rate by about 15% 11 Decrease the slew rate by about 30%
55 MSTR_DATA1_SLEW_RATE_UP	Master Data1 Lane's LP-TX Driver Slew Rate Up Control 0 No change 1 Slew Rate UP
54–53 MSTR_DATA0_SLEW_RATE_DOWN	Master Data0 Lane's LP-TX Driver Slew Rate Down Control 00 No change 01 Decrease the slew rate by about 15% 10 Decrease the slew rate by about 15% 11 Decrease the slew rate by about 30%

Table continues on the next page...

MIPI_DPHY_B_DPHYCTL field descriptions (continued)

Field	Description
52 MSTR_DATA0_ SLEW_RATE_ UP	Master Data0 Lane's LP-TX Driver Slew Rate Up Control 0 No change 1 Slew Rate UP
51–47 -	This field is reserved. -
46 LP_VREF_REG_ SRC_SEL	LP Regulator Vref Source Selection 0 Generated from BGR 1 Generated from Current Mirror
45 RX_SKEW_ CALIB_FIX_ CODE_EN	RX Skew Calibration Fixing Code Enable/Disable Control 0 Disable 1 Enable
44 -	This field is reserved. -
43 DCC_DONE_ CTL	DCC "DONE" Signal Control 0 "DONE" from DCC block 1 U"DONE" is always 1
42 -	This field is reserved. -
41–40 BGR_VOLT_ TUNING_CTL	BGR Voltage Tuning Control 00 820mV 01 760mV 10 800mV 11 840mV
39–38 HS_MODE_CTL	HS Loopback Mode Control 00 Designated Pattern 01 PRBS7 10 All zero 11 User Data Pattern
37–30 USER_DATA_ HS	User Data Pattern for HS Loopback mode
29–28 BIAS_REF_ VOLT_CTL	Bias Reference Voltage 710m Control 00 712mV 01 724mV 10 733mV 11 706mV
27 BGR_ CHOPPER_ FREQ_CTL	BGR Chopper Frequency Control 0 3MHz 1 1.5MHz

Table continues on the next page...

MIPI_DPHY_B_DPHYCTL field descriptions (continued)

Field	Description
26 1_2_REG_ VALID_CTL	VREG12_EXTPWR Enable Control 0 Internal 1.2V regulator 1 External 1.2V power
25–24 1_2_REG_LVL_ CTL	1.2 Regulator Level Control 00 1.2V 01 1.23V 10 1.27V 11 1.26V
23 1_2_REG_VALID	1.2 Regulator Valid Level Section 0 Use "ulps_en" signal 1 Use valid signal from 1.2V regulator
22–21 LP_RX_HYS_ CTL	LP-RX Hysteresis Level Control 00 80mV 01 100mV 10 120mV 11 140mV
20 VREF_SRC_SEL	VREF Source Selection 0 Generated from the BGR 1 Generated from the current mirror
19–18 LP_RX_VREF_ LVL	LP-RX VREF Level Control 00 715mV 01 743mV 10 650mV 11 682mV
17 LP_RX_PULSE_ REJ	LP-RX Pulse Rejection Control 0 Enable 1 Disable
16–15 MSTR_CLK_ SLEW_RATE_ DOWN	Master Clock Lane's LP-TX Driver Slew Rate Down Control 00 No change 01 Decrease the slew rate by about 15% 10 Decrease the slew rate by about 15% 11 Decrease the slew rate by about 30%
14 MSTR_CLK_ SLEW_RATE_ UP	Master Clock Lane's LP-TX Driver Slew Rate Up Control 0 No change 1 Slew Rate UP
13 LP_CD_HYS	LP-CD Hysteresis Level Control 0 60mV 1 70mV

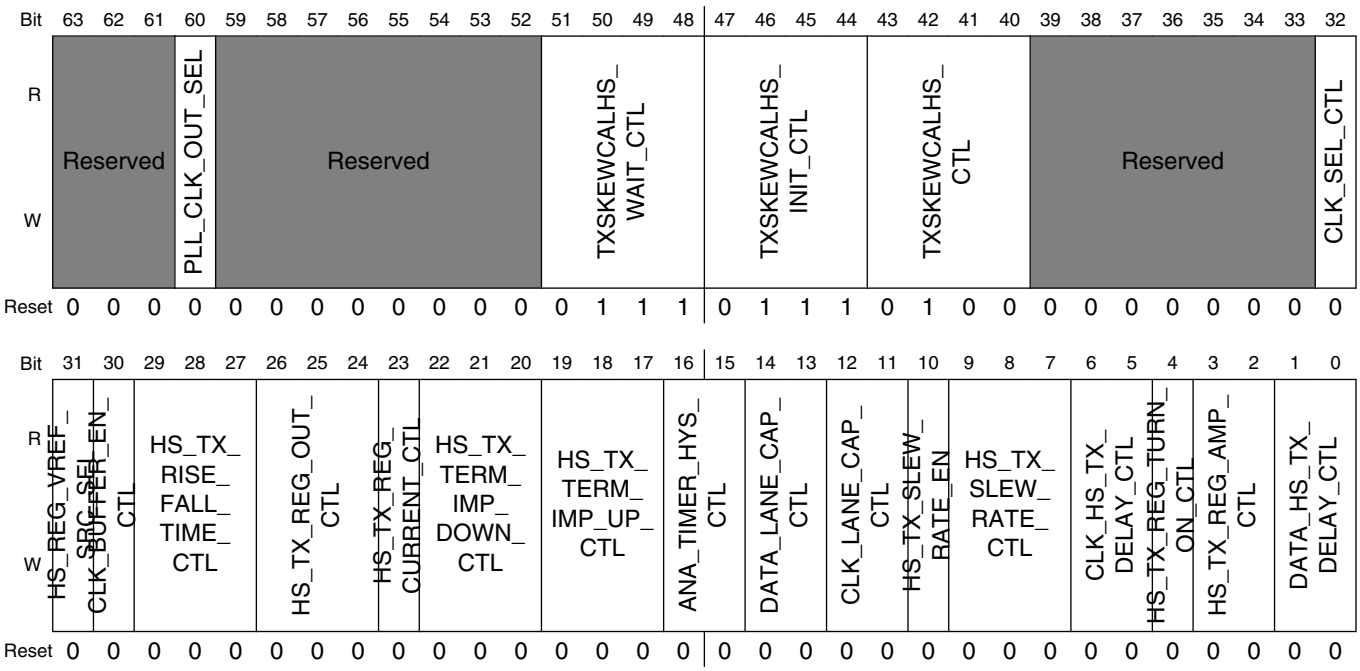
Table continues on the next page...

MIPI_DPHY_B_DPHYCTL field descriptions (continued)

Field	Description
12 BGR_ CHOPPER_EN	BGR Chopper Function Enable/Disable Control 0 Enable 1 Disable
11 ERR_CONT_LP_ EN	Error Contention LP Enable/Disable Control 0 Enable 1 Disable
10 TX_TRIGGER_ CLK_EN	TX Trigger Clk Enable/Disable 0 Enable 1 Disable
ULPS_EXIT_ COUNTER	ULPS EXIT Counter value Control You should set B_DPHYCTL[9:0] during initial or power-up sequence. 0x000 0.01 MHz 0x003 0.10 MHz 0x019 1.00 MHz 0x032 2.00 MHz 0x04B 3.00 MHz 0x064 4.00 MHz 0x07D 5.00 MHz 0x096 6.00 MHz 0x0AF 7.00 MHz 0x0C8 8.00 MHz 0x0E1 9.00 MHz 0x0FA 10.00 MHz 0x113 11.00 MHz 0x12C 12.00 MHz 0x145 13.00 MHz 0x15E 14.00 MHz 0x177 15.00 MHz 0x190 16.00 MHz 0x1A9 17.00 MHz 0x1C2 18.00 MHz 0x1DB 19.00 MHz 0x1F4 20.00 MHz

13.6.10.4 Master and Slave DPHY Control Register (MIPI_DPHY_M_DPHYCTL)

Address: 0h base + 18h offset = 18h



MIPI_DPHY_M_DPHYCTL field descriptions

Field	Description
63–61 -	This field is reserved. Reserved
60 PLL_CLK_OUT_SEL	PLL Clock Selection 0 Disable OUT to Other lane 1 Enable OUT to Other lane
59–52 -	This field is reserved. Reserved
51–48 TXSKEWCALHS_ WAIT_CTL	TxSkewCalHS Initial Wait Time Control Timing count for the waiting initial skew calibration function after power on. If M_DPHYCTL[51:48] = 0, the initial skew calibration function does not work.
47–44 TXSKEWCALHS_ INIT_CTL	TxSkewCalHS Initial Run Time Control Timing count for the initial skew calibration function's running time. If M_DPHYCTL[47:44] = 0, timing is same as count 7.
43–40 TXSKEWCALHS_ CTL	TxSkewCalHS Run Time Control Timing count for the skew calibration function's running time when M_TXSKEWCALHS is enabled. If M_DPHYCTL[43:40] = 0, timing is same as count 4.
39–33 -	This field is reserved. Reserved

Table continues on the next page...

MIPI_DPHY_M_DPHYCTL field descriptions (continued)

Field	Description
32 CLK_SEL_CTL	Clock Signal Control Selection (Use Only in case of M4S0, M1S0) 0 Generated from Internal PLL 1 Generated from External PLL
31 HS_REG_VREF_SRC_SEL	HS regulator Vref Source Selection 0 Generated from BGR 1 Generated from Current Mirror
30 CLK_BUFFER_EN_CTL	Clock Buffer Enable Select Control 0 HS_TX enable 1 PLL lock
29–27 HS_TX_RISE_FALL_TIME_CTL	HS-TX Rise and Fall Time Control 000 135mV 001 130mV 010 125mV 011 120mV 100 230mV 101 225mV 110 220mV 111 215mV
26–24 HS_TX_REG_OUT_CTL	HS-TX Regulator Output Level Control 000 400mV 001 410mV 010 420mV 011 440mV 100 200mV 101 360mV 110 380mV 111 390mV
23 HS_TX_REG_CURRENT_CTL	HS-TX Regulator Reference Current Control 0 No change 1 2.5 uA
22–20 HS_TX_TERM_IMP_DOWN_CTL	HS-TX Driver Termination Impedance Control (Down) 000 50 ohm 001 52 ohm 010 54 ohm 011 56 ohm 100 44 ohm 101 46 ohm 110 47 ohm 111 48 ohm

Table continues on the next page...

MIPI_DPHY_M_DPHYCTL field descriptions (continued)

Field	Description
19–17 HS_TX_TERM_IMP_UP_CTL	HS-TX Driver Termination Impedance Control (Up) 000 50 ohm 001 52 ohm 010 54 ohm 011 56 ohm 100 44 ohm 101 46 ohm 110 47 ohm 111 48 ohm
16–15 ANA_TIMER_HYS_CTL	Analog Timer Hysteresis Control 00 70mV 01 95mV 10 95mV 11 110mV
14–13 DATA_LANE_CAP_CTL	Data Lane Cap. Value Control for TCLK_PREPARE 00 No change 01 -6.6% 10 13.3% 11 6.6%
12–11 CLK_LANE_CAP_CTL	Clock Lane Cap. Value Control for TCLK_PREPARE 00 No change 01 -6.6% 10 13.3% 11 6.6%
10 HS_TX_SLEW_RATE_EN	HS-TX Driver Slew Rate Enable 0 SRC Disable 1 SRC Enable
9–7 HS_TX_SLEW_RATE_CTL	HS-TX Driver Slew Rate Control(SRC) Value
6–5 CLK_HS_TX_DELAY_CTL	Clock Lane HS-TX Driver Delay Control 00 No change 01 25 ps 10 55 ps 11 90 ps
4 HS_TX_REG_TURN_ON_CTL	HS-TX Regulator Turn-on Control. 0 No Change 1 Always Turn-On the HS Regulator
3–2 HS_TX_REG_AMP_CTL	HS-TX Regulator Amp current Control 00 No Change 01 12.5 uA

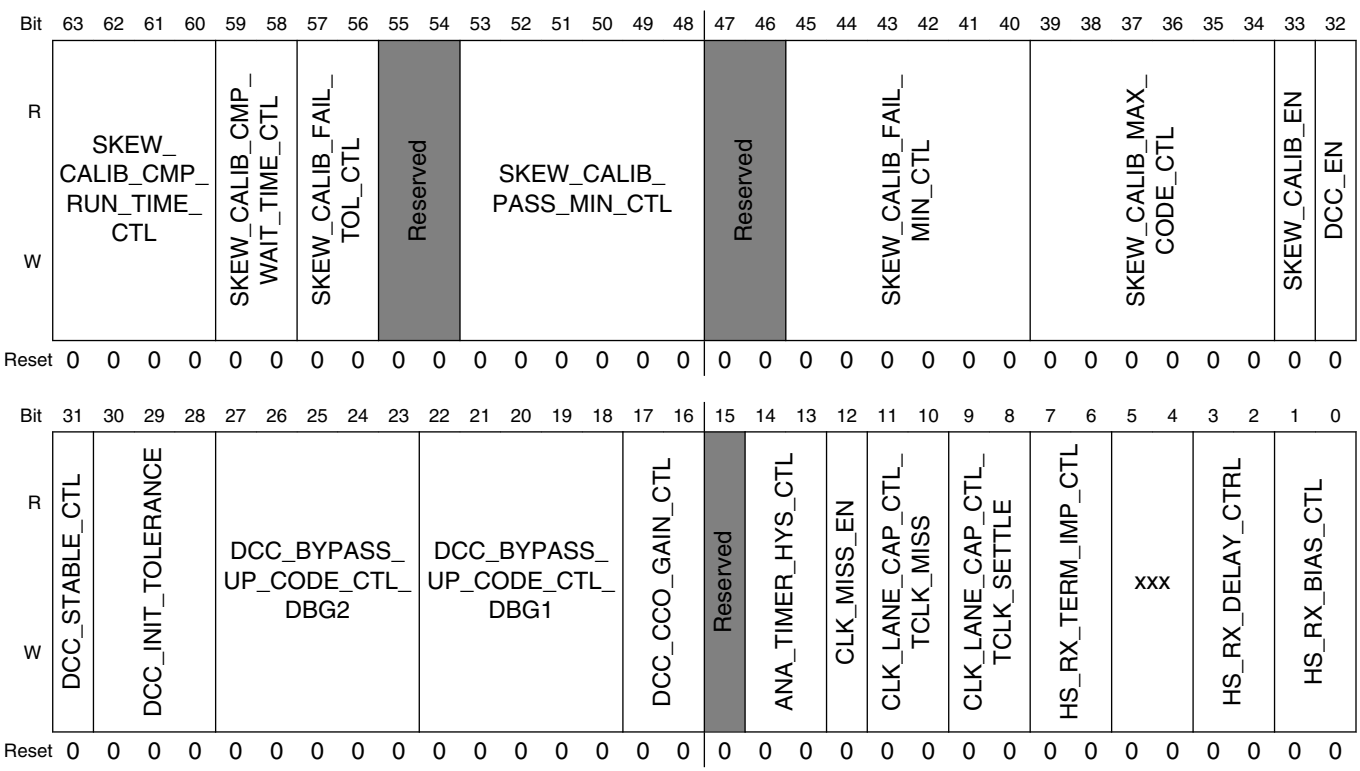
Table continues on the next page...

MIPI_DPHY_M_DPHYCTL field descriptions (continued)

Field	Description
	10 50 uA 11 16.6 uA
DATA_HS_TX_DELAY_CTL	Data Lane HS-TX Driver Delay Control 00 No change 01 25 ps 10 55 ps 11 90 ps

13.6.10.5 Master and Slave DPHY Control Register (MIPI_DPHY_S_DPHYCTL)

Address: 0h base + 20h offset = 20h



MIPI_DPHY_S_DPHYCTL field descriptions

Field	Description
63–60 SKEW_CALIB_CMP_RUN_TIME_CTL	RX Skew Calibration Compare-run time Control

Table continues on the next page...

MIPI_DPHY_S_DPHYCTL field descriptions (continued)

Field	Description
59–58 SKEW_CALIB_CMP_WAIT_TIME_CTL	RX Skew Calibration Compare-wait time Control Control waiting time at changing delay code 00 Min (Fast) 01 Reserved 10 Reserved 11 Mx (Slow)
57–56 SKEW_CALIB_FAIL_TOL_CTL	RX Skew Calibration Fail Tolerance Control 00 Recognizes the pass although fail appears 3 times 01 Recognizes the pass although fail appears 2 times 10 Recognizes the pass although fail appears 1 time 11 Reserved
55–54 -	This field is reserved. Reserved
53–48 SKEW_CALIB_PASS_MIN_CTL	RX Skew Calibration Pass-min Control
47–46 -	This field is reserved. Reserved
45–40 SKEW_CALIB_FAIL_MIN_CTL	RX Skew Calibration Fail-min Control
39–34 SKEW_CALIB_MAX_CODE_CTL	RX Skew Calibration Max Code Control
33 SKEW_CALIB_EN	Skew Calibration Function Enable/Disable Control 0 Skew Calibration Disable 1 Skew Calibration Enable
32 DCC_EN	DCC Function Enable/Disable 0 DCC Disable 1 DCC Enable
31 DCC_STABLE_CTL	DCC Stable Control 0 1 number counter running 1 2 number counter running
30–28 DCC_INIT_TOLERANCE	DCC Initial Tolerance 000 4 001 5 010 6 011 7 100 0 101 1

Table continues on the next page...

MIPI_DPHY_S_DPHYCTL field descriptions (continued)

Field	Description
	110 2 111 3
27–23 DCC_BYPASS_UP_CODE_CTL_DBG2	DCC Bypass Up Code Control for debugging
22–18 DCC_BYPASS_UP_CODE_CTL_DBG1	DCC Bypass Up Code Control for debugging
17–16 DCC_CCO_GAIN_CTL	DCC CCO Gain Control 00 1/1 01 1/4 10 1/0.66 11 1/1.33
15 -	This field is reserved. Reserved
14–13 ANA_TIMER_HYS_CTL	Analog Timer Hysteresis Control 00 70mV 01 95mV 10 95mV 11 110mV
12 CLK_MISS_EN	Clock Miss Function Enable/Disable Control 0 Enable 1 Disable
11–10 CLK_LANE_CAP_CTL_TCLK_MISS	Clock Lane Cap. Value Control for TCLK-MISS 00 No change 01 6.6% 10 13.2% 11 19.8%
9–8 CLK_LANE_CAP_CTL_TCLK_SETTLE	Clock Lane Cap. Value Control for TCLK-SETTLE 00 No change 01 3.45% 10 6.9% 11 10.35%
7–6 HS_RX_TERM_IMP_CTL	S-RX Termination Impedance Control 00 98 ohm 01 106 ohm 10 85 ohm 11 91 ohm
5–4 xxx	00 - 01 30 ps

Table continues on the next page...

MIPI_DPHY_S_DPHYCTL field descriptions (continued)

Field	Description
	10 60 ps 11 90 ps
3–2 HS_RX_DELAY_CTRL	Clock Lane HS RX Delay Control 00 0 ps 01 30 ps 10 60 ps 11 90 ps
HS_RX_BIAS_CTL	HS RX Bias Control 00 25 uA 01 30 uA 10 37.5 uA 11 50 uA

13.7 Sony/Philips Digital Interface (SPDIF)

13.7.1 Overview

The Sony/Philips Digital Interface (SPDIF) audio block is a stereo transceiver that allows the processor to receive and transmit digital audio.

The SPDIF transceiver allows the handling of both SPDIF channel status (CS) and User (U) data and includes a frequency measurement block that allows the precise measurement of an incoming sampling frequency.

A recovered clock is provided to drive both internal components in the system, such as ESAI ports, and external components, such as A/Ds or D/As, with clocking control provided via related registers.

As the SPDIF internal data width is 24-bit, the eight most-significant bits of all registers return zeros.

The figure below shows a block diagram of the SPDIF transceiver data paths (receiver and transmitter) and its interface.

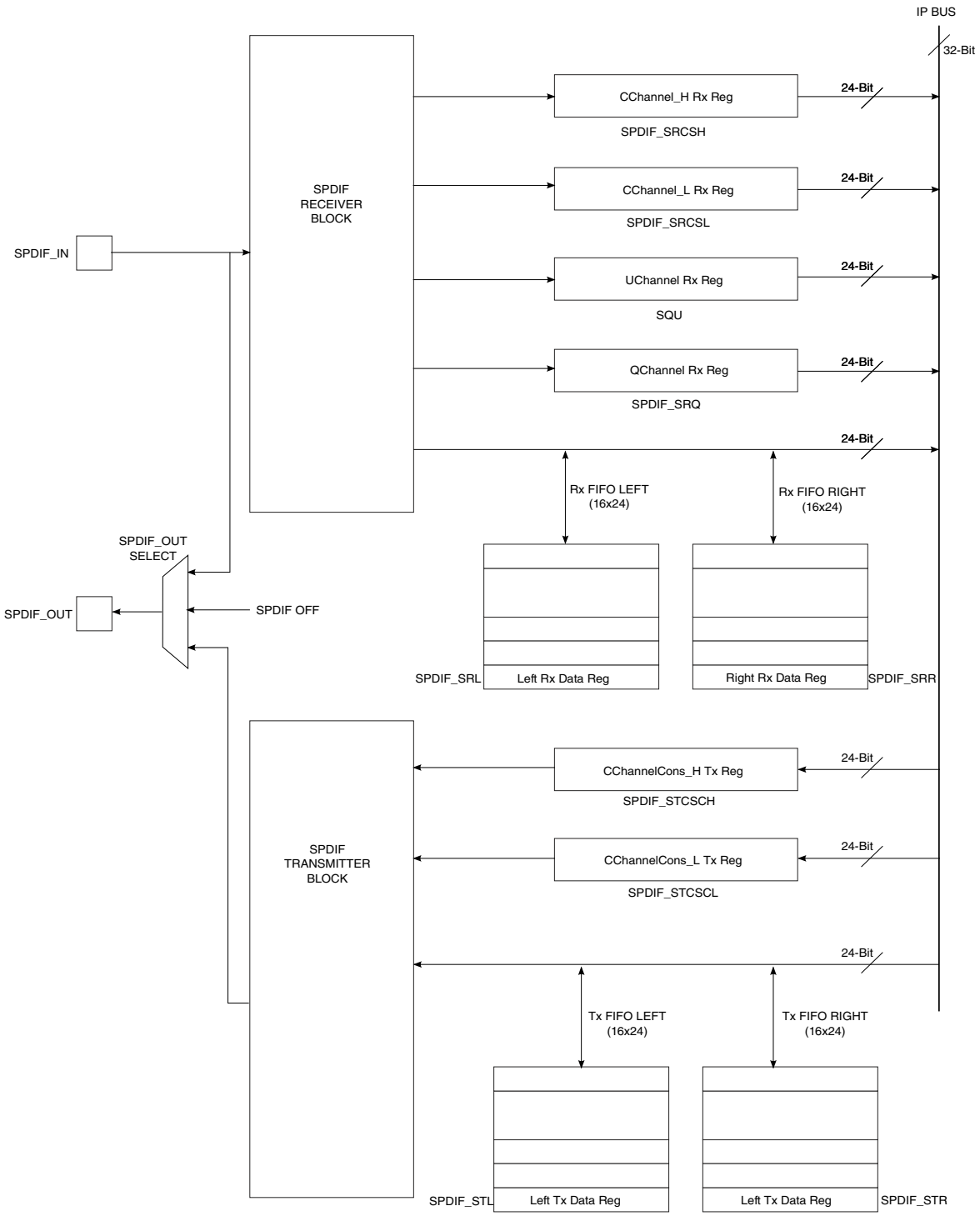


Figure 13-89. SPDIF Transceiver Data Interface Block Diagram

13.7.2 External Signals

The following table describes the external signals of SPDIF:

Table 13-52. SPDIF External Signals

Signal	Description	Direction
SPDIF_EXT_CLK	External clock signal	I
SPDIF_IN	Input line	I
SPDIF_LOCK	Lock signal	O
SPDIF_OUT	Output line signal	O
SPDIF_SR_CLK	SR clock signal	O

13.7.3 Clocks

The table found here describes the clock sources for SPDIF.

Please see clock control block for clock setting, configuration and gating information.

Table 13-53. SPDIF Clocks

Clock name	Description
gclkw_t0	Global clock
ipg_clk_s	Peripheral access clock
tx_clk	Module Tx clock

13.7.4 Functional Description

The SPDIF is composed of two parts: SPDIF Receiver and SPDIF Transmitter.

The SPDIF receiver extracts the audio data from each SPDIF frame and places the data in the SPDIF Rx left and right FIFOs. The Channel Status and User Bits are also extracted from each frame and placed in the corresponding registers. The SPDIF receiver also provides a bypass option for direct transfer of the SPDIF input signal to the SPDIF transmitter.

For the SPDIF transmitter, the audio data is provided by the processor via the SPDIFTxLeft and SPDIFTxRight registers. The Channel Status bits are also provided via the corresponding registers. The SPDIF transmitter generates a SPDIF output bitstream in the biphase mark format (IEC60958), which consists of audio data, channel status and user bits.

In the SPDIF transmitter, the IEC60958 biphase bit stream is generated on both edges of the SPDIF Transmit clock. The SPDIF Transmit clock is generated by the SPDIF internal clock generate block and the sources are from outside of the SPDIF block. For the SPDIF receiver, it can recover the SPDIF Rx clock.

The Rx clock is sent to the ASRC.

Figure 13-90 shows the clock structure of the SPDIF transceiver.

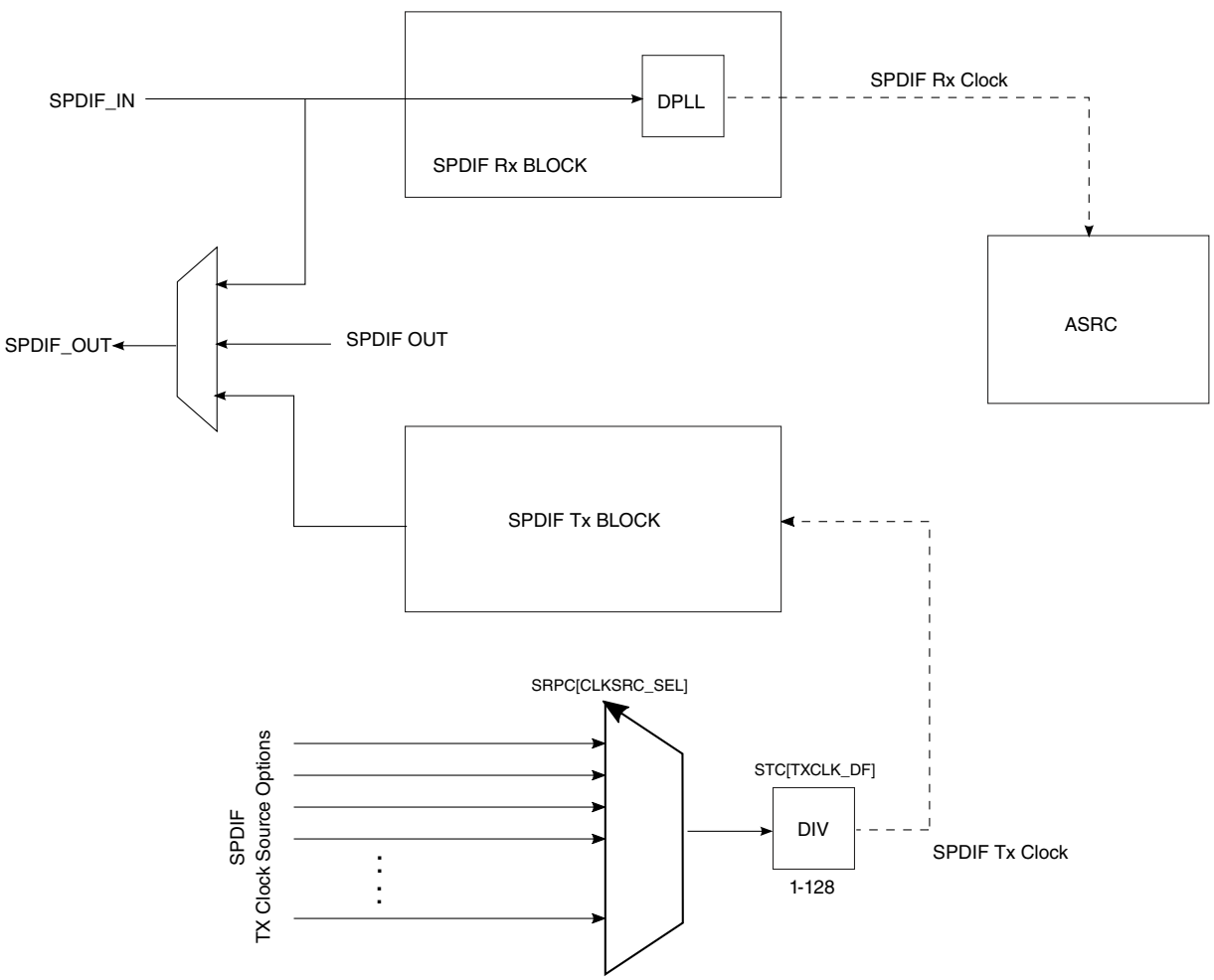


Figure 13-90. SPDIF Transceiver Clock Diagram

13.7.4.1 SPDIF Receiver

The SPDIF receiver extracts the audio data from each SPDIF frame and places the data in Rx left and right FIFOs.

The Tx left and right FIFOs are 16-deep and 24-bit-wide (equal to the audio data width). The Channel Status and User Bits are also extracted from each frame and placed in corresponding registers. The SPDIF receiver also provides a bypass option for direct transfer of the SPDIF input signal to the SPDIF transmitter.

The SPDIF receiver handles the main data audio stream and recovers the bit clock from the SPDIF input signal. The sample rate can be determined from the frequency measuring block. Additionally, the receiver supports the SPDIF C and U channels. The SPDIF C and U channel data is interfaced directly to memory-mapped registers.

All the data registers are controlled by the Interrupt Control Block and transferred to the memory-mapped IP bus.

The following functions are performed by the SPDIF receiver:

- Audio Data Reception see Audio Data Reception
- Channel Status bits Reception see Channel Status Reception
- U Channel bits Reception see User Bit Reception
- Validity Flag Reception see Validity Flag Reception
- SPDIF Receiver Exception support see SPDIF Receiver
- SPDIF Lock Detection

13.7.4.1.1 Audio Data Reception

The SPDIF Receiver block extracts the audio data from the IEC60958 stream, and outputs this via Rx left and right FIFOs to the memory-mapped registers SPDIFRxLeft and SPDIFRxRight.

Data from the SPDIF receiver is buffered in receive FIFO, and can be read by the processor from the memory-mapped registers.

- **SPDIF receiver data registers - Behavior on overrun, underrun**

The SPDIF Data Receive registers (SPDIFRxLeft and SPDIFRxRight) have individual FIFOs for left and right channel. As a result, there is always the possibility that left and right FIFOs may go out of sync due to FIFO underruns and FIFO overruns that affect only one part (left or right) of any FIFOs. To prevent this from happening, hardware has been added to the device. Two mechanisms to prevent mismatch between the FIFOs are available.

If a SPDIF Data Rx FIFO overrun occurs on e.g. the right half of the FIFO, the sample that caused the overrun is not written to the right half (due to overrun). Special hardware will make sure the next sample is not written to the left half of the FIFO. If the overrun occurs on the left half of the FIFO, the next sample is not written to the right half of the FIFO.

- **SPDIF receiver data registers - Automatic resynchronization of FIFOs**

An automatic FIFO resynchronization feature is available. It can be enabled and disabled separately for every FIFO. If it is enabled, the hardware will check to see if the left and right FIFOs are in sync. If that is not the case, it will set the filling pointer of the right FIFO to be equal to the filling pointer of the left FIFO.

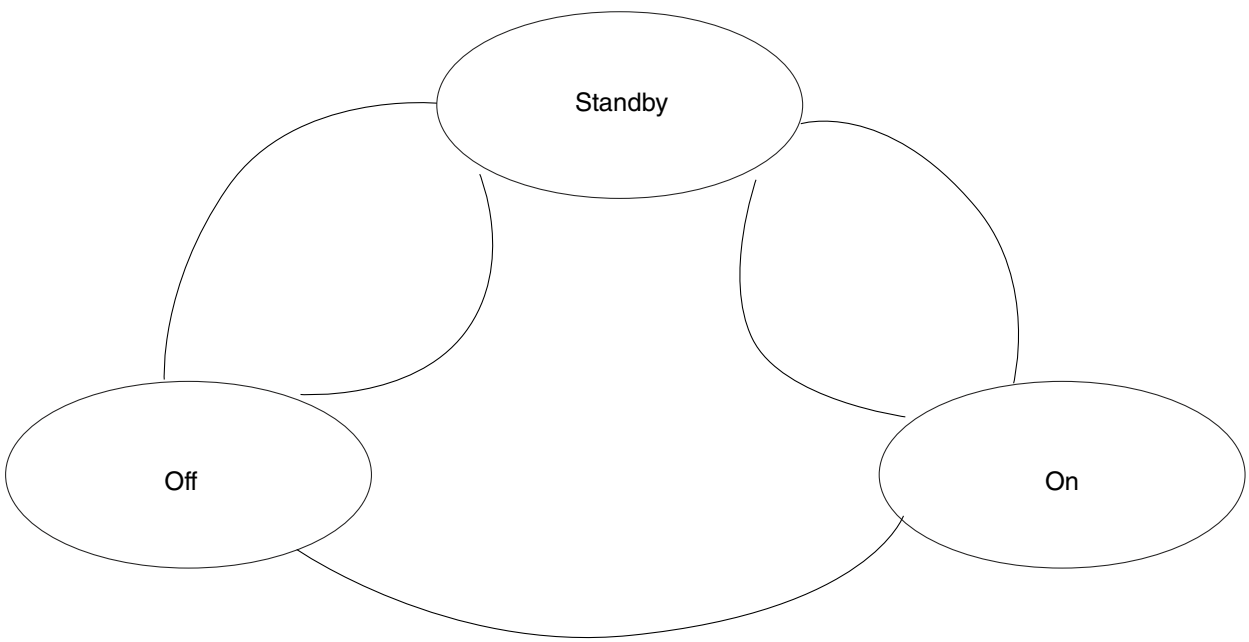


Figure 13-91. FIFO Auto-resync Controller State Machine

The operation is explained from the state diagram shown above. Every FIFO auto-resync controller has a state machine with 3 states: Off, StandBy and On. In the On state, the filling of the left FIFO is compared with the filling of right, and if they are not equal, right is made equal to left, and an interrupt is generated.

The controller will stay in Off state when the feature is disabled. When not disabled, the state machine will go to Off state on any processor read or write to the FIFO. It will go from On or Off to Standby on any left sample read from SPDIF Tx FIFOs, or on any left sample write to SPDIF Rx FIFOs. The controller will go from Standby to On on any right

sample read from SPDIF Tx FIFO, or on any right sample write to SPDIF Rx FIFO. There is a control bit in the SPDIFConfig register to enable/disable the feature for the SPDIF Rx FIFO and SPDIF Tx FIFO.

13.7.4.1.1.1 Application Note

The automatic FIFO resynchronization can be switched on, and will avoid all mismatches between left and right FIFOs, if the software obeys the following rules:

1. When the left data is read or written to the left FIFO, in the same place of the program, data must be read or written to the right FIFO. Maximum time difference between left and right is 1/2 sample clock. (E.g. if sample frequency is 44 Khz, approximately 10 micro-seconds. For 88 Khz, approximately 5 micro-seconds.)
2. Write/read data to FIFOs at least 2 samples at the time. If there is a mismatch Left-Right, the resync logic may go on only 1 sample clock after last data is read/written to the FIFO. Also acceptable is polling the FIFO, if at least part of the time 2 samples will be read/written to it.
3. The first 192 frames (after DPLL lock) must be discarded.

- **SPDIF receiver - Additional features**

There are three exceptions associated with the SPDIF Receivers FIFOs

- full
- under/overrun
- resync

When the "full" condition is set for processor data input registers, the processor should read data from the FIFO, before overrun occurs. When "full" is set, and the FIFO contains e.g. 6 samples, it is acceptable for the software to read first 6 samples from the LEFT address, followed by 6 samples from the RIGHT address, or 6 samples from the RIGHT address, followed by 6 samples from the LEFT address, or 1 sample LEFT, followed by 1 sample RIGHT repeated 6 times. There is no order specified.

The implementation for SPDIF Rx is a double FIFO, one for left and one for right. "full" is set when both FIFOs are full. "underrun, overrun" are set when one of the FIFOs do underrun or do overrun. The resync interrupt means hardware took special action to resynchronize left and right FIFOs.

The FIFO level at which the "full" interrupt is generated, is programmable via the Full Select field in the SPDIFConfigReg register.

Rx FIFO on and Rx FIFO reset.

Two additional control fields of the SPDIF Rx FIFO are the on/off select and FIFO reset fields.

If on/off select is set to off, all-zero will be read from the FIFO, irrespective of the data received over the SPDIF interface.

If FIFO reset is set, the FIFO is blocked at "1 sample in FIFO". In this, the full interrupt will be on if FullSelect is set to "00". If FullSelect is set to any other value, interrupt will be off. The other interrupts are always off.

13.7.4.1.2 Channel Status Reception

A total of 48 channel status bits are received in two registers. No interpretation is performed by the SPDIF receiver block.

Channel Status Bits are ordered first bit left. CS-channel MSB bit "0" is located in bit position 23 in the memory-mapped register SPDIFRxCChannel_h. CS-channel bit "23" is considered the LSB bit 0 in the register. C-channel bit 24 to 47 is seen as [23:0] bits of register SPDIFRxCChannel_l.

13.7.4.1.2.1 Channel Status Interrupt

When the value of a new SPDIF "CS" channel status frame is loaded in the register, an interrupt is generated. The interrupt is cleared when the processor writes the corresponding bit in the InterruptStat register.

13.7.4.1.3 User Bit Reception

There are two modes for U Channel reception, CD and non-CD. As is decided by USyncMode (bit 1 of CDText_Control register).

- **Behavior of U Channel receive interface on incoming CD U Channel Sub-code in SPDIF receiver**.

This mode is selected if UsyncMode, bit 1 in register CD Text control is set "1".

The CD sub-code stream embedded into the SPDIF U channel consists of a sequence of packets. Every packet is made up 98 "symbols". The first two symbols of every packet are "sync symbols", the other 96 symbols are "data symbols".

Any sequence found in the SPDIF U channel stream starting with a leading one, followed by 7 information bits, is recognized as a "data symbol". Subsequent data symbols are separated by "pauses". During the "pause", "zero bits" are seen on the SPDIF U channel.

Data symbols are coming in MSB first. The MSB is the leading one.

When a "long pause" is seen between 2 subsequent "data symbols", the SPDIF receiver will assume the reception of one or more "sync symbols". Table below gives details.

Table 13-54. Sync Control Bits

Number of U Channel zero bits	Corresponding number of sync symbols
0-1	Unpredictable, not allowed
2-10	0
11-22	1
23-34	2
35-46	3
>45	Unpredictable, not allowed

The recognition of the number of sync symbols derives from the fact that the U channel transmitter in the CD channel decoder will transmit one symbol on average every 12 SPDIF channel bits. On this average rate, there is a maximum tolerance of 5%.

The SPDIF receiver is tolerant of symbol errors. Due to the physical nature of the transmission of the data over the CD disc, not more than 1 out of any 5 consecutive user channel symbols may be in error. The error may cause a change in data value, which is not detected by this interface, or it may cause a data symbol to be seen as a sync symbol, or a sync symbol to be seen as a data symbol. However, not more than 1 out of any 5 consecutive user channel symbols should be affected in this way.

The SPDIF U channel circuitry recognizes the 98-symbol packet structure, and sends the 96 symbol payload to the processor application. The 96 symbol payload is transmitted to the processor via 2 registers:

- The SPDIFRxUChannel register. In this register, data is presented 3 symbols at the time to the processor. Every time 3 new valid symbols, received on the SPDIF U Channel are present, the UChannelRxFull interrupt is asserted. For one 98-symbol packet, 96 symbols are carried across SPDIFRxUChannel. To transfer all this data, 32 UChannelRxFull interrupts are generated.
- The QChannelReceive register. In this register, only the Q bit of the packet is accumulated.Operation is similar to UChannelReceive. Because only Q-bit is transferred, only 96 Q-bits are transferred for any 98-symbol packet. To transfer this data, 4 QChannelRxFull interrupts are generated. When QChannelRxFull occurs, it is coincident with UChannelRxFull. There is only one QChannelRxFull for every 8 UChannelRxFull. The convention is that most significant data is transmitted first, and is left-aligned in the registers.
- Timing regarding packet boundary is extracted by hardware. The last UChannelRxFull corresponding to a given packet should be coincident with the last QChannelRxFull. In this last U, Q channel interrupt, symbols 95-98 are received, Q channel bits 67-98. The interrupts are coincident with UQSyncFound, flagging last symbols of the current frame.

- When the start of the new packet is found before the current packet is complete (less than 98 symbols in the packet), the UQFrameError interrupt is set. The application software should read out UChannelReceive and QchannelReceive registers, discard the value, and assume the start of a new packet.
- As already said, packet sync extraction is tolerant for single-symbol errors. Packet sync detection is based on the recognition of the sequence data-sync-sync-data in the symbol stream, because this is the only syncing sequence that is not affected by single errors. If the sync symbols are not found 98 symbols after the previous occurrence, it is assumed to be destroyed by channel error, and a new sync symbols is interpolated.
- Normally, only data bytes are passed to the application software. Every databyte will have its most significant bit set. If sync symbols are passed to the application software, they are seen as all-zero symbols. Sync symbols can only end up in the data stream due to channel error.

- **Behavior of U Channel receive interface on incoming non-CD data**.

This mode is selected if UsyncMode, bit 1 in register CD Text control is set'0'.

In non-CD mode, the SPDIF U channel stream is recognized as a sequence of "data symbols". No packet recognition is done.

Any sequence found in the SPDIF U channel stream starting with a leading one, followed by 7 information bits, is recognized as a "data symbol". Subsequent data symbols are separated by "pauses". During the "pause", "zero bits" are seen on the SPDIF U channel.

3 consecutive data symbols seen in the SPDIF U Channel stream are grouped together into the SPDIFRxUChannel register. First symbol is left, last symbol is right aligned. When SPDIFRxUChannel contains 3 new data symbols, UChannelRxFull is asserted.

In this mode, the operation of QchannelRx and associated interrupt QchannelRxFull is reserved, undefined. And the operation of UQFrameError and UQSyncFound is also reserved, undefined.

The U channel is extracted, and output by the SPDIF Rx on SPDIFRxUChannel-Stream.

When incoming SPDIF data parity error or bit error is detected, and if the next SPDIF word for that channel is error-free, the SPDIF word in error is replaced with the average of the previous word and next word. When incoming SPDIF data parity error or bit error is detected, and the next SPDIF word is in error, the previous SPDIF word is repeated.

13.7.4.1.4 Validity Flag Reception

An interrupt is associated with the Validity flag. (interrupt 16 - SPDIFValNoGood). This interrupt is set every time a frame is seen on the SPDIF interface with the validity bit set to "invalid".

13.7.4.1.5 SPDIF Receiver Interrupt Exception Definition

Several SPDIF exceptions can trigger an interrupt. They are:

- Control Status channel change. Set when SPDIFRxCChannel_l register is updated. The register is updated for every new C-Channel received. The exception is reset on write to InterruptClear register.
- SPDIF Illegal Symbol. Set on reception of illegal symbol during SPDIF receive. Reset by writing register InterruptClear.[1]
- SPDIF bit error. Set on reception of bit error. (Parity bit does not match). Reset on write to InterruptClear register.
- Receive data FIFO full. Set when SPDIF receive data FIFO is full.
- Receive data FIFO underrun/overrun. Set when there is a underrun/overrun on the SPDIF receive data FIFO.
- Receive data FIFO resynchronization. Set when a resynchronization event occurs on the SPDIF receive data FIFO.
- Receive U Channel buffer full. Set when next 24 bits of U channel code are available.
- Receive Q Channel buffer overrun. Set when Q channel buffer overrun.
- Receive U Channel buffer overrun. Set on U channel buffer overrun.
- Receive Q Channel buffer full. Set when next 24 bits of Q channel code are available.
- Receive UQ sync found. Set when UQ channel sync found.
- Receive UQ frame error. Set when UQ frame error found.

13.7.4.1.6 Standards Compliance

The SPDIF interface is compatible with the Tech 3250-E standard of the European Broadcasting Union, except clause 6.3.3 and the IEC60958-3 Ed2 for relevant topics.

1. The SPDIF input is a biphase/mark modulated signal. The time between any two successive transitions of the SPDIF signal is always 1, 2 or 3 SPDIF symbol periods long. The SPDIF receiver will parse the stream, and split it in so-called symbols. It recognizes s1, s2 and s3 symbols, depending on the length of the symbols. Not all sequences of these symbols are allowed. To give an example, a sequence s2-s1-s1-s1-s2 cannot occur in a no-error SPDIF signal. If the receiver finds such an illegal sequence, the illegal symbol interrupt is set. No corrective action is undertaken. When the interrupt occurs, this means that(a) The SPDIF signal is destroyed by noise (b) The SPDIF frequency changed.

Supported input frequency range is 12 Khz up to 96 Khz. (fully compliant) and 96 Khz up to 176 Khz (Can interface with compliant SPDIF transmitter within same cabinet, making reasonable assumptions on jitter added due to interconnecting wire.)

Tolerated jitter on SPDIF input signals are 0.25 bit peak-peak for high frequencies. There is no jitter limit for low frequencies. The user channel extraction in CD mode is capable of coping with single-symbol errors, and still retrieve U channel frames on correct boundaries. This capability is required for reliable reception of CD-Text from some Philips CD channel decoders. This capability was deemed more important than compliance with the IEC60958 annex A.3 standard, and for this reason user channel reception is not compliant with IEC60958 annex A.3. However, the interface is capable to receive U channel inserted by a typical CD channel decoder. Also, in this case, it is more robust and tolerant for channel error than what is required by IEC60958 annex A.3.

13.7.4.1.7 SPDIF PLOCK Detection and Rxclk Output

Using the high speed system clock, the internal DPLL can extract the bit clock (advanced pulse) from the input bitstream. When this internal DPLL is locked, the LOCK bit of PhaseConfig Register will be set, and the SPDIF Lock output pin SPDIF_LOCK will be asserted.

After DPLL has locked, the pulses are generated, and the average pulse rate is 128 x the sampling frequency.(For a 44.1 Khz input sampling frequency, the average pulse rate = 128 x 44.1 Khz.) The pulse signal is used in the FreqMeas circuit to generate the frequency measurement result.

13.7.4.1.8 Measuring Frequency of SPDIF_RxClk

The internal DPLL can extract the bit clock (advanced plus) from the input bitstream. To do that, it is necessary to measure the frequency of the incoming signal in relationship with the system clock (BUS_CLK).

Associated with it are two registers, PhaseConfig and FreqMeas. The circuit will measure the frequency of the incoming clock as a function of the BUS_CLK. The circuit is a second-order filter. The output is a value represented by an unsigned number stored in the 24-bit FreqMeas register, giving the frequency of the source as a function of the BUS_CLK.

FreqMeas[23:0] = FreqMeas_CLK / BUS_CLK * 2^{10} * GAIN.

For example, if the GAIN is selected as 8*(2**10) (PhaseConfig[5:3] = 3'b011), the actual result

FreqMeas_CLK / BUS_CLK is equal to FreqMeas[23:0] / 2^{23}.

13.7.4.2 SPDIF Transmitter

Audio data for the SPDIF transmitter is provided by processor via the SPDIFTxLeft and SPDIFTxRight registers.

Clocking for SPDIF transmitter is selected through a multiplexer from several clock sources (see register bitfield description STC[TxClk_Source] for transmit clock source inputs). The SPDIF transmitter clock source can be divided down as needed using Txclk_DF. The SPDIF transmitter output can be chosen from either the SPDIF transmitter block, directly from the SPDIF receiver (via the output multiplexer), or disabled.

The SPDIF transmitter generates a SPDIF output bitstream in IEC60958 biphase mark format, consisting of audio data, channel status.

13.7.4.2.1 Audio Data Transmission

Audio data for the SPDIF transmitter is provided by the processor via SPDIFTxLeft and SPDIFTxRight registers. They send audio data to Tx left and right FIFOs. The Tx left and right FIFOs are also 16-deep and 24-width (equal to the audio data width).

- **SPDIF transmitter data registers - Behavior on overrun, underrun**

The SPDIF Data Transmit registers (SPDIFTxLeft and SPDIFTxRight) have individual FIFOs for left and right channel. As a result, there is always the possibility that left and right FIFOs may go out of sync due to FIFO underruns and FIFO overruns that affect only one part (left or right) of any FIFO. To prevent this from happening, hardware has been added on the device. Two mechanisms to prevent mismatch between the FIFOs are available.

If SPDIF Tx FIFO underruns on the right half of the FIFO, no sample leaves that FIFO (because it was already empty). Special hardware will make sure that the next sample read from the left FIFO will not leave the FIFO (no read strobe is generated). If the underrun occurs on the left half of the FIFO, next read strobe to the right FIFO is blocked.

- **SPDIF transmitter data registers - Automatic resynchronization of FIFOs**

See Audio Data Reception.

- **SPDIFTxLeft, SPDIFTxRight details**

With SPDIF Tx FIFOs three exceptions are associated.

- empty

- under/overrun
- resync

When the empty condition is set for processor data output registers, the processor should write data to the FIFO, before underrun occurs. When empty is set and, for instance, 6 samples need to be written, it is acceptable for the software to write first 6 samples from the LEFT address, followed by 6 samples from the RIGHT address, or 1 sample LEFT, followed by 1 sample RIGHT repeated 6 times. Left should be written before right. The implementation of all data out FIFOs is a double FIFO, one for left and one for right. Empty is set when both FIFOs are empty. Underrun, overrun are set when one of the FIFOs do underrun or do overrun. Resync is set when the hardware resynchronizes left and right FIFOs.

On receiving underrun, overrun interrupt, synchronization between Left and Right words in the FIFOs may be lost. Synchronization will not be lost when the underrun or overrun comes from the IEC60958 side of the FIFO. If the processor reads or writes more data from, for example, left than from right, synchronization will be lost. If automatic resynchronization is enabled, and if the software obeys the rules to let this work, resynchronization will be automatic.

13.7.4.2.2 Channel Status Transmission

A total of 48 Consumer channel status bits are transmitted from two registers. Channel Status Bits are ordered first bit left.

CS-channel MSB bit "0" is located in bit position 23 in the memory-mapped register SPDIFTxCChannelCons_h. CS-channel bit "23" is considered bit 0 in the register. C-channel bits 24-47 are seen as MSB-LSB bits of register SPDIFTxCChannelCons_l.

13.7.4.2.3 Validity Flag Transmission

The validity bit setting is performed via bit 5 of the SPDIF_SCR register.

13.7.5 SPDIF Memory Map/Register Definition

13.7.5.1 SPDIF register descriptions

13.7.5.1.1 SPDIF Memory map

SPDIF1 base address: 3009_0000h

SPDIF2 base address: 300A_0000h

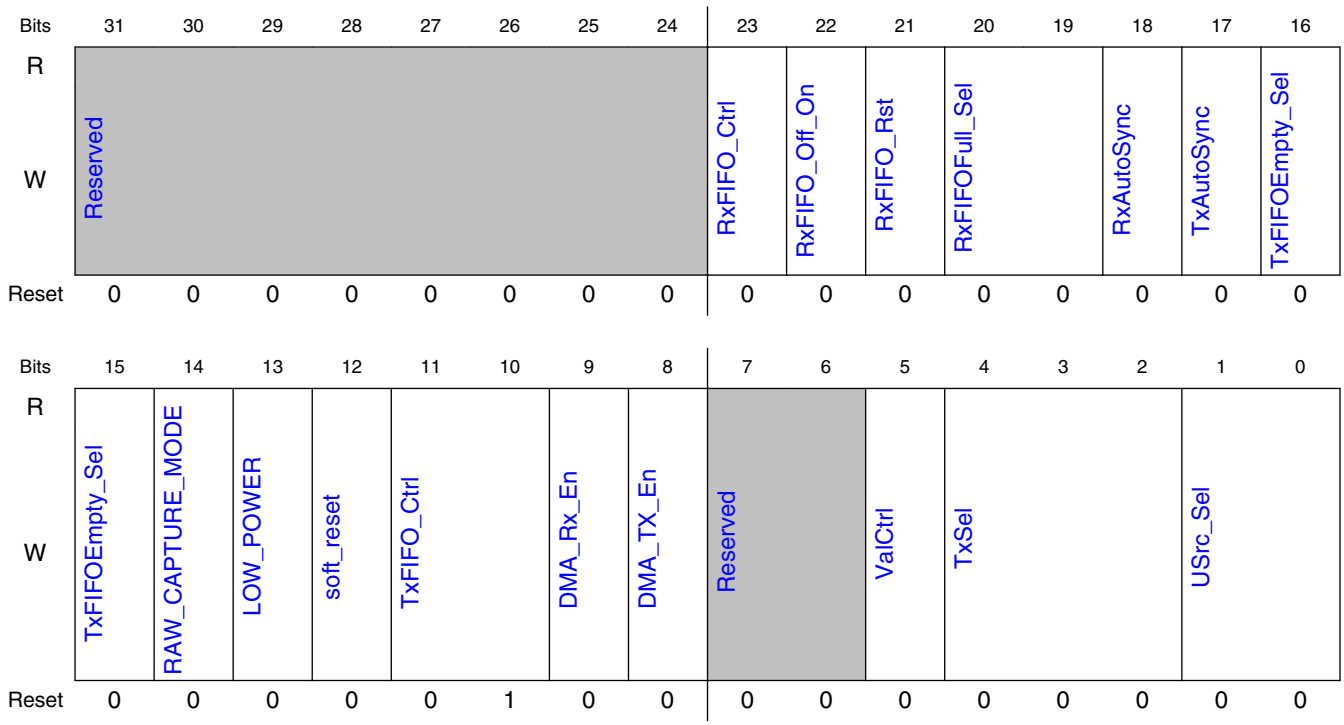
Offset	Register	Width (In bits)	Access	Reset value
0h	SPDIF Configuration Register (SCR)	32	RW	0000_0400h
4h	CDText Control Register (SRCD)	32	RW	0000_0000h
8h	PhaseConfig Register (SRPC)	32	RW	0000_0000h
Ch	InterruptEn Register (SIE)	32	RW	0000_0000h
10h	InterruptClear Register (SIC)	32	WO	0000_0000h
10h	InterruptStat Register (SIS)	32	RO	0000_0002h
14h	SPDIFRxLeft Register (SRL)	32	RO	0000_0000h
18h	SPDIFRxRight Register (SRR)	32	RO	0000_0000h
1Ch	SPDIFRxCChannel_h Register (SRCSH)	32	RO	0000_0000h
20h	SPDIFRxCChannel_l Register (SRCSL)	32	RO	0000_0000h
24h	UchannelRx Register (SRU)	32	RO	0000_0000h
28h	QchannelRx Register (SRQ)	32	RO	0000_0000h
2Ch	SPDIFTxLeft Register (STL)	32	WO	0000_0000h
30h	SPDIFTxRight Register (STR)	32	WO	0000_0000h
34h	SPDIFTxCChannelCons_h Register (STCSCH)	32	RW	0000_0000h
38h	SPDIFTxCChannelCons_l Register (STCSCL)	32	RW	0000_0000h
44h	FreqMeas Register (SRFM)	32	RO	0000_0000h
50h	SPDIFTxClk Register (STC)	32	RW	0002_0F00h

13.7.5.1.2 SPDIF Configuration Register (SCR)

13.7.5.1.2.1 Offset

Register	Offset
SCR	0h

13.7.5.1.2.2 Diagram



13.7.5.1.2.3 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23 RxFIFO_Ctrl	RxFIFO_Ctrl 0b - Normal operation 1b - Always read zero from Rx data register
22 RxFIFO_Off_On	RxFIFO_Off_On 0b - SPDIF Rx FIFO is on 1b - SPDIF Rx FIFO is off. Does not accept data from interface
21 RxFIFO_Rst	RxFIFO_Rst 0b - Normal operation 1b - Reset register to 1 sample remaining
20-19 RxFIFOFull_Sel	RxFIFOFull_Sel 00b - Full interrupt if at least 1 sample in Rx left and right FIFOs 01b - Full interrupt if at least 4 sample in Rx left and right FIFOs 10b - Full interrupt if at least 8 sample in Rx left and right FIFOs 11b - Full interrupt if at least 16 sample in Rx left and right FIFO
18 RxAutoSync	RxAutoSync 0b - Rx FIFO auto sync off 1b - RxFIFO auto sync on
17 TxAutoSync	TxAutoSync 0b - Tx FIFO auto sync off 1b - Tx FIFO auto sync on

Table continues on the next page...

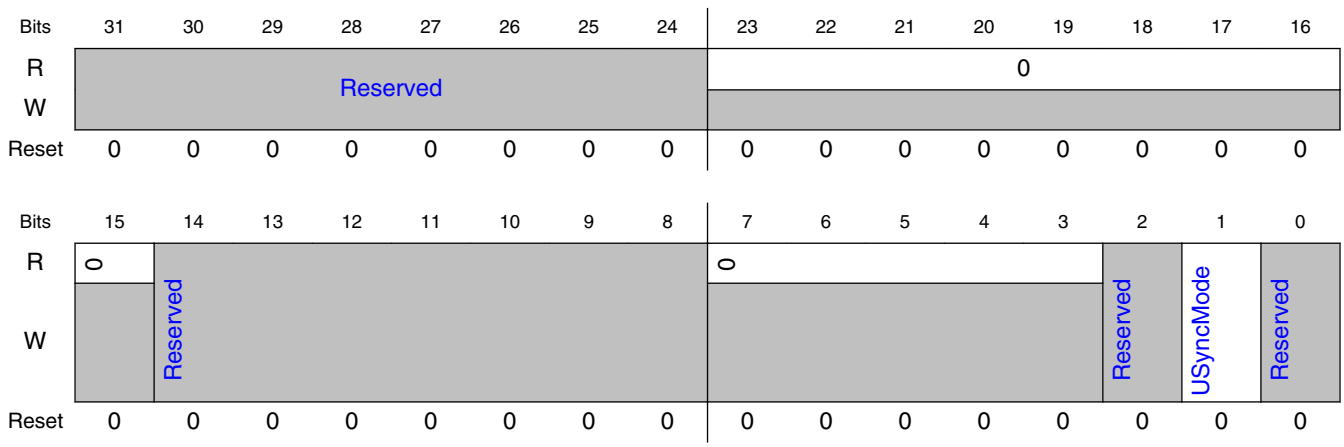
Field	Function
16-15 TxFIFOEmpty_Sel	TxFIFOEmpty_Sel 00b - Empty interrupt if 0 sample in Tx left and right FIFOs 01b - Empty interrupt if at most 4 sample in Tx left and right FIFOs 10b - Empty interrupt if at most 8 sample in Tx left and right FIFOs 11b - Empty interrupt if at most 12 sample in Tx left and right FIFOs
14 RAW_CAPTURE_MODE	RAW_CAPTURE_MODE 1'b0, the SPDIF RXwill work in normal mode. 1'b1, the SPDIF RX will work in raw capture mode. the SPDIF will put rx data to RX FIFO following the bleow format. In this mode the RX FIFO will be 32bits. X: {P, C, U, V, original_Data[23:0],4'b0100}; Y: {P, C, U, V, original_Data[23:0],4'b1000}; Z: {P, C, U, V, original_Data[23:0],4'b1100};
13 LOW_POWER	LOW_POWER When write 1 to this bit, it will cause SPDIF enter low-power mode. return 1 when SPDIF in Low-Power mode.
12 soft_reset	soft_reset When write 1 to this bit, it will cause SPDIF software reset. The software reset will last 8 cycles. When in the reset process, return 1 when read. else return 0 when read.
11-10 TxFIFO_Ctrl	TxFIFO_Ctrl 00b - Send out digital zero on SPDIF Tx 01b - Tx Normal operation 10b - Reset to 1 sample remaining 11b - Reserved
9 DMA_Rx_En	DMA_Rx_En DMA Receive Request Enable (RX FIFO full)
8 DMA_TX_En	DMA_TX_En DMA Transmit Request Enable (Tx FIFO empty)
7-6 —	- Reserved
5 ValCtrl	ValCtrl 0b - Outgoing Validity always set 1b - Outgoing Validity always clear
4-2 TxSel	TxSel 000b - Off and output 0 001b - Feed-through SPDIFIN 101b - Tx Normal operation
1-0 USrc_Sel	USrc_Sel 00b - No embedded U channel 01b - U channel from SPDIF receive block (CD mode) 10b - Reserved 11b - U channel from on chip transmitter

13.7.5.1.3 CDText Control Register (SRCD)

13.7.5.1.3.1 Offset

Register	Offset
SRCD	4h

13.7.5.1.3.2 Diagram



13.7.5.1.3.3 Fields

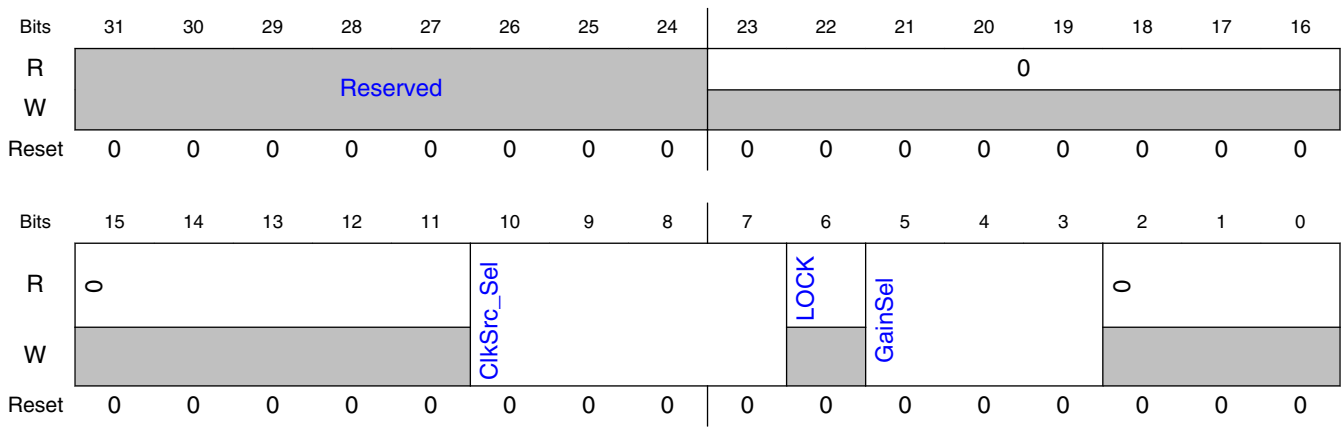
Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-15 —	- Return zeros when read
14-8 —	- Reserved. set to zero.
7-3 —	- Return zeros when read
2 —	- Reserved.
1 USyncMode	USyncMode 0b - Non-CD data 1b - CD user channel subcode
0 —	- Reserved.

13.7.5.1.4 PhaseConfig Register (SRPC)

13.7.5.1.4.1 Offset

Register	Offset
SRPC	8h

13.7.5.1.4.2 Diagram



13.7.5.1.4.3 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-11 —	- Reserved, return zeros when read
10-7 ClkSrc_Sel	ClkSrc_Sel Clock source selection, all other settings not shown are reserved: 0000b - Clock Selection from Audio Clock Mux (ACM)
6 LOCK	LOCK LOCK bit to show that the internal DPLL is locked, read only
5-3 GainSel	GainSel Gain selection: 000b - 24*(2**10) 001b - 16*(2**10) 010b - 12*(2**10) 011b - 8*(2**10)

Table continues on the next page...

Field	Function
	100b - 6*(2**10) 101b - 4*(2**10) 110b - 3*(2**10)
2-0 —	- Reserved, return zeros when read

13.7.5.1.5 InterruptEn Register (SIE)

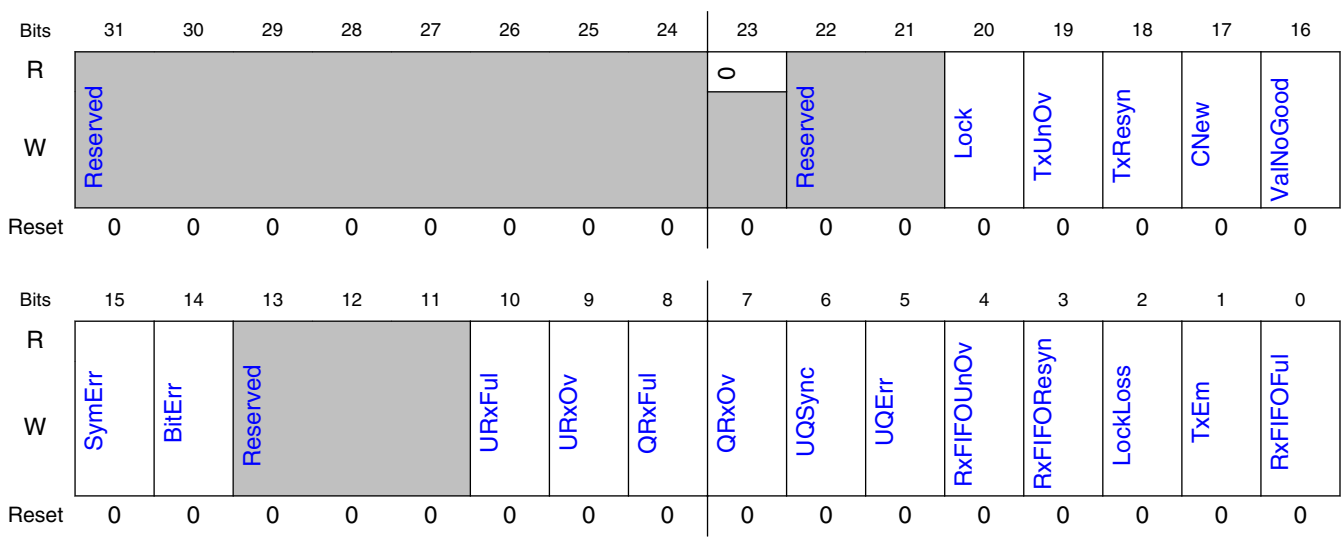
13.7.5.1.5.1 Offset

Register	Offset
SIE	Ch

13.7.5.1.5.2 Function

The InterruptEn register (SPDIF_SIE) provides control over the enabling of interrupts.

13.7.5.1.5.3 Diagram



13.7.5.1.5.4 Fields

Field	Function
31-24	This is a 24-bit register the upper byte is unimplemented.

Table continues on the next page...

Field	Function
—	
23 —	- Reserved
22-21 —	- Reserved. set to zero.
20 Lock	Lock SPDIF receiver's DPLL is locked
19 TxUnOv	TxUnOv SPDIF Tx FIFO under/overrun
18 TxResyn	TxResyn SPDIF Tx FIFO resync
17 CNew	CNew SPDIF receive change in value of control channel
16 ValNoGood	ValNoGood SPDIF validity flag no good
15 SymErr	SymErr SPDIF receiver found illegal symbol
14 BitErr	BitErr SPDIF receiver found parity bit error
13-11 —	- Reserved. set to zero.
10 URxFul	URxFul U Channel receive register full, can't be cleared with reg. IntClear. To clear it, read from U Rx reg.
9 URxOv	URxOv U Channel receive register overrun
8 QRxFul	QRxFul Q Channel receive register full, can't be cleared with reg. IntClear. To clear it, read from Q Rx reg.
7 QRxOv	QRxOv Q Channel receive register overrun
6 UQSync	UQSync U/Q Channel sync found
5 UQErr	UQErr U/Q Channel framing error
4 RxFIFOUnOv	RxFIFOUnOv Rx FIFO underrun/overrun
3 RxFIFOResyn	RxFIFOResyn Rx FIFO resync
2 LockLoss	LockLoss SPDIF receiver loss of lock
1	TxEm

Table continues on the next page...

Field	Function
TxEm	SPDIF Tx FIFO empty, can't be cleared with reg. IntClear. To clear it, write toTx FIFO.
0 RxFIFOFul	RxFIFOFul SPDIF Rx FIFO full, can't be cleared with reg. IntClear. To clear it, read from Rx FIFO.

13.7.5.1.6 InterruptClear Register (SIC)

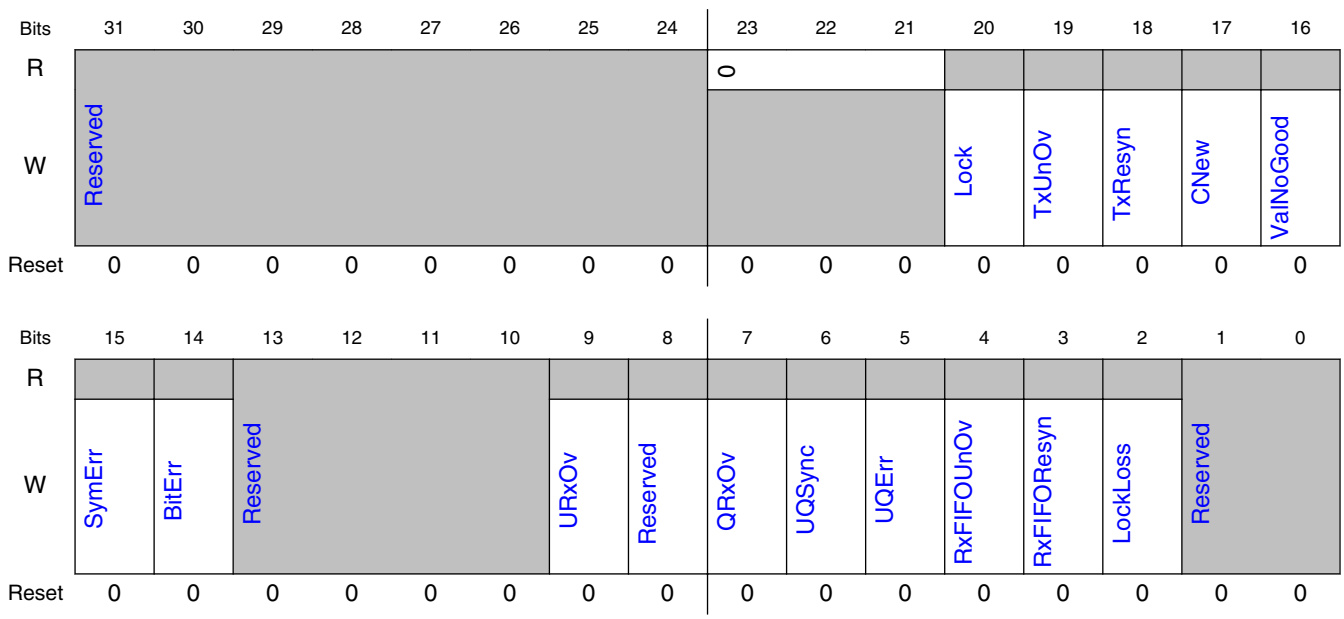
13.7.5.1.6.1 Offset

Register	Offset
SIC	10h

13.7.5.1.6.2 Function

The InterruptClear (SPDIF_SIC) register is a write only register and is used to clear interrupts.

13.7.5.1.6.3 Diagram



13.7.5.1.6.4 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-21 —	- Reserved, for InterruptStat/Clear return zeros when read.
20 Lock	Lock SPDIF receiver's DPLL is locked
19 TxUnOv	TxUnOv SPDIF Tx FIFO under/overrun
18 TxResyn	TxResyn SPDIF Tx FIFO resync
17 CNew	CNew SPDIF receive change in value of control channel
16 ValNoGood	ValNoGood SPDIF validity flag no good
15 SymErr	SymErr SPDIF receiver found illegal symbol
14 BitErr	BitErr SPDIF receiver found parity bit error
13-10 —	- Reserved.
9 URxOv	URxOv U Channel receive register overrun
8 —	- Reserved
7 QRxOv	QRxOv Q Channel receive register overrun
6 UQSync	UQSync U/Q Channel sync found
5 UQErr	UQErr U/Q Channel framing error
4 RxFIFOUnOv	RxFIFOUnOv Rx FIFO underrun/overrun
3 RxFIFOResyn	RxFIFOResyn Rx FIFO resync
2 LockLoss	LockLoss SPDIF receiver loss of lock
1-0 —	- Reserved.

13.7.5.1.7 InterruptStat Register (SIS)

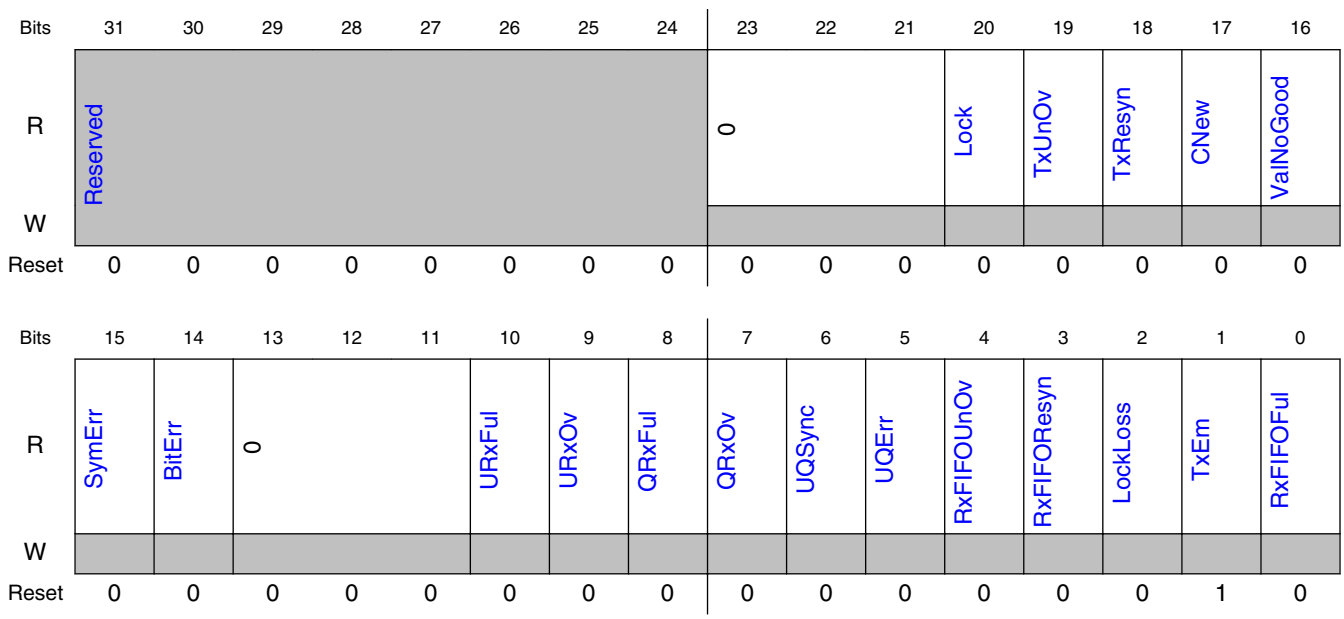
13.7.5.1.7.1 Offset

Register	Offset
SIS	10h

13.7.5.1.7.2 Function

The InterruptStat (SPDIF_SIS) register is a read only register that provides the status on interrupt operations.

13.7.5.1.7.3 Diagram



13.7.5.1.7.4 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-21 —	- Reserved, for InterruptStat/Clear return zeros when read.
20	Lock

Table continues on the next page...

Field	Function
Lock	SPDIF receiver's DPLL is locked
19	TxUnOv
TxUnOv	SPDIF Tx FIFO under/overrun
18	TxResyn
TxResyn	SPDIF Tx FIFO resync
17	CNew
CNew	SPDIF receive change in value of control channel
16	ValNoGood
ValNoGood	SPDIF validity flag no good
15	SymErr
SymErr	SPDIF receiver found illegal symbol
14	BitErr
BitErr	SPDIF receiver found parity bit error
13-11	-
—	Reserved. Return zeros when read
10	URxFul
URxFul	U Channel receive register full, can't be cleared with reg. IntClear. To clear it, read from U Rx reg.
9	URxOv
URxOv	U Channel receive register overrun
8	QRxFul
QRxFul	Q Channel receive register full, can't be cleared with reg. IntClear. To clear it, read from Q Rx reg.
7	QRxOv
QRxOv	Q Channel receive register overrun
6	UQSync
UQSync	U/Q Channel sync found
5	UQErr
UQErr	U/Q Channel framing error
4	RxFIFOUnOv
RxFIFOUnOv	Rx FIFO underrun/overrun
3	RxFIFOResyn
RxFIFOResyn	Rx FIFO resync
2	LockLoss
LockLoss	SPDIF receiver loss of lock
1	TxEm
TxEm	SPDIF Tx FIFO empty, can't be cleared with reg. IntClear. To clear it, write toTx FIFO.
0	RxFIFOFul
RxFIFOFul	SPDIF Rx FIFO full, can't be cleared with reg. IntClear. To clear it, read from Rx FIFO.

13.7.5.1.8 SPDIFRxLeft Register (SRL)

13.7.5.1.8.1 Offset

Register	Offset
SRL	14h

13.7.5.1.8.2 Function

SPDIFRxLeft register is an audio data reception register.

13.7.5.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								RxDataLeft							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RxDataLeft															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.7.5.1.8.4 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 RxDataLeft	RxDataLeft Processor receive SPDIF data left

13.7.5.1.9 SPDIFRxRight Register (SRR)

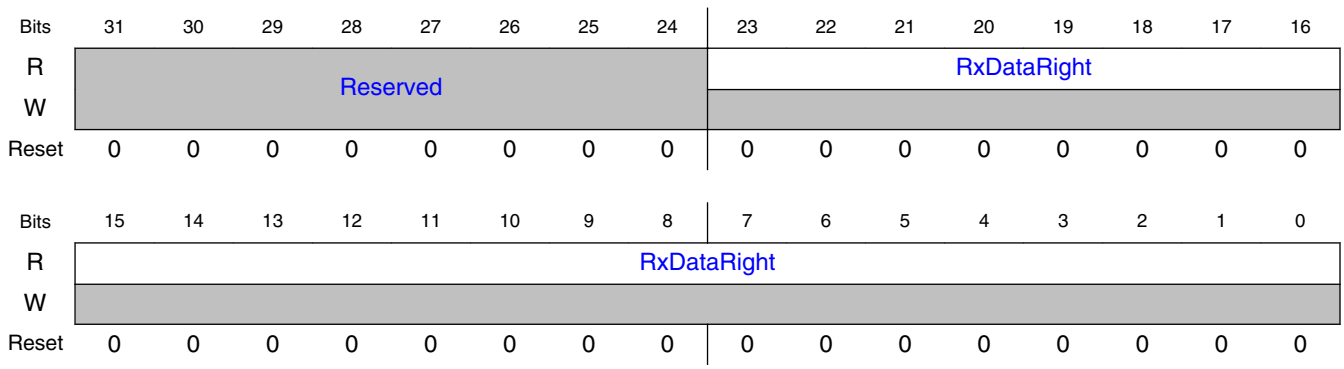
13.7.5.1.9.1 Offset

Register	Offset
SRR	18h

13.7.5.1.9.2 Function

SPDIFRxRight register is an audio data reception register.

13.7.5.1.9.3 Diagram



13.7.5.1.9.4 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 RxDataRight	RxDataRight Processor receive SPDIF data right

13.7.5.1.10 SPDIFRxCChannel_h Register (SRCSH)

13.7.5.1.10.1 Offset

Register	Offset
SRCSH	1Ch

13.7.5.1.10.2 Function

SPDIFRxCChannel_h register is a channel status reception register.

13.7.5.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								RxCChannel_h							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RxCChannel_h															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.7.5.1.10.4 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 RxCChannel_h	RxCChannel_h SPDIF receive C channel register, contains first 24 bits of C channel without interpretation

13.7.5.1.11 SPDIFRxCChannel_l Register (SRCSL)

13.7.5.1.11.1 Offset

Register	Offset
SRCSL	20h

13.7.5.1.11.2 Function

SPDIFRxCChannel_l register is a channel status reception register.

13.7.5.1.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								RxCChannel_l							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RxCChannel_l															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.7.5.1.11.4 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 RxCChannel_l	RxCChannel_l SPDIF receive C channel register, contains next 24 bits of C channel without interpretation

13.7.5.1.12 UchannelRx Register (SRU)

13.7.5.1.12.1 Offset

Register	Offset
SRU	24h

13.7.5.1.12.2 Function

UChannelRx register is a user bits reception register.

13.7.5.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								RxUChannel							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RxUChannel															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.7.5.1.12.4 Fields

Field	Function
31-24 —	[unimplemented] This is a 24-bit register the upper byte is unimplemented.
23-0 RxUChannel	RxUChannel SPDIF receive U channel register, contains next 3 U channel bytes

13.7.5.1.13 QchannelRx Register (SRQ)

13.7.5.1.13.1 Offset

Register	Offset
SRQ	28h

13.7.5.1.13.2 Function

QChannelRx register is a user bits reception register.

13.7.5.1.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								RxQChannel							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RxQChannel															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.7.5.1.13.4 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 RxQChannel	RxQChannel SPDIF receive Q channel register, contains next 3 Q channel bytes

13.7.5.1.14 SPDIFTxLeft Register (STL)

13.7.5.1.14.1 Offset

Register	Offset
STL	2Ch

13.7.5.1.14.2 Function

SPDIFTxLeft register is an audio data transmission register.

13.7.5.1.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								TxDataLeft							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TxDataLeft															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.7.5.1.14.4 Fields

Field	Function
31-24 —	[unimplemented] This is a 24-bit register the upper byte is unimplemented.
23-0 TxDataLeft	TxDataLeft SPDIF transmit left channel data. It is write-only, and always returns zeros when read

13.7.5.1.15 SPDIFTxRight Register (STR)

13.7.5.1.15.1 Offset

Register	Offset
STR	30h

13.7.5.1.15.2 Function

SPDIFTxRight register is an audio data transmission register.

13.7.5.1.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W									TxDataRight							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	TxDataRight															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.7.5.1.15.4 Fields

Field	Function
31-24 —	[unimplemented] This is a 24-bit register the upper byte is unimplemented.
23-0 TxDataRight	TxDataRight SPDIF transmit right channel data. It is write-only, and always returns zeros when read

13.7.5.1.16 SPDIFTxCChannelCons_h Register (STCSCH)

13.7.5.1.16.1 Offset

Register	Offset
STCSCH	34h

13.7.5.1.16.2 Function

SPDIFTxCChannelCons_h register is a channel status transmission register.

13.7.5.1.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TxCChannelCons_h							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TxCChannelCons_h															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.7.5.1.16.4 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 TxCChannelCons_h	TxCChannelCons_h SPDIF transmit Cons. C channel data, contains first 24 bits without interpretation. When read, it returns the latest data written by the processor

13.7.5.1.17 SPDIFTxCChannelCons_l Register (STCSCL)

13.7.5.1.17.1 Offset

Register	Offset
STCSCL	38h

13.7.5.1.17.2 Function

SPDIFTxCChannelCons_l register is a channel status transmission register.

13.7.5.1.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								TxCChannelCons_l							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TxCChannelCons_l															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.7.5.1.17.4 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 TxCChannelCons_l	TxCChannelCons_l SPDIF transmit Cons. C channel data, contains next 24 bits without interpretation. When read, it returns the latest data written by the processor

13.7.5.1.18 FreqMeas Register (SRFM)

13.7.5.1.18.1 Offset

Register	Offset
SRFM	44h

13.7.5.1.18.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								FreqMeas							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FreqMeas															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.7.5.1.18.3 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 FreqMeas	FreqMeas Frequency measurement data

13.7.5.1.19 SPDIFTxClk Register (STC)

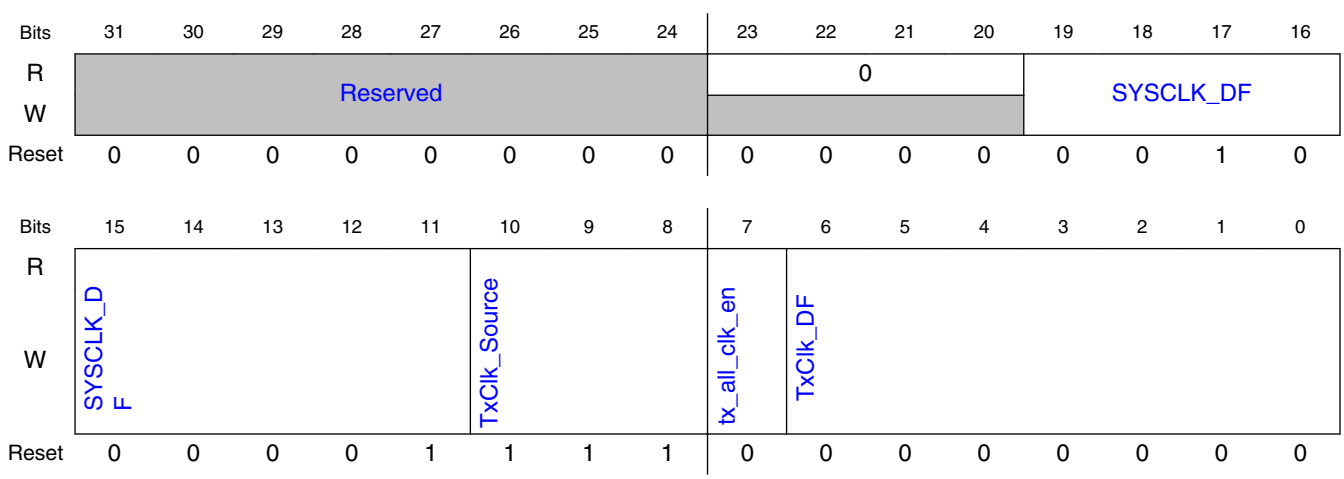
13.7.5.1.19.1 Offset

Register	Offset
STC	50h

13.7.5.1.19.2 Function

The SPDIFTxClk Control register includes the means to select the transmit clock and frequency division.

13.7.5.1.19.3 Diagram



13.7.5.1.19.4 Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-20 —	- Reserved, return zeros when read
19-11 SYSCLK_DF	SYSCLK_DF system clock divider factor, 2~512. 000000000b - no clock signal 000000001b - divider factor is 2 111111111b - divider factor is 512
10-8 TxClk_Source	TxClk_Source 000b - Clock Selection from Audio Clock Mux (ACM)
7 tx_all_clk_en	tx_all_clk_en Spdif transfer clock enable.When data is going to be transfered, this bit should be set to1. 0b - disable transfer clock. 1b - enable transfer clock.
6-0 TxClk_DF	TxClk_DF Divider factor (1-128) 0000000b - divider factor is 1 0000001b - divider factor is 2 1111111b - divider factor is 128

13.8 PDM Microphone Interface (MICFIL)

13.8.1 Preface

The purpose of this document is to present the PDM Microphone Interface block specification.

13.8.1.1 Acronyms and Abbreviations

Table 13-55 contains sample acronyms and abbreviations used in this document.

Table 13-55. Acronyms and Abbreviated Terms

Term	Meaning
CPU	Central Processing Unit (core processor)
DMA	Direct Memory Access

Table continues on the next page...

Table 13-55. Acronyms and Abbreviated Terms (continued)

Term	Meaning
CIC	Cascaded Integrator Comb filter
dB	Decibels
FIFO	First In First Out storage block
FIR	Finite Impulse Response digital filter type
IIR	Infinite Impulse Response digital filter type
IP	Intellectual Property
LSB	Least Significant Bit
MAC Unit	Multiply And Accumulate Unit
MCU	Microcontroller Unit
PDM	Pulse-density Modulation
PCM	Pulse-code Modulation
HWVAD	Hardware Voice Activity Detector

13.8.2 Introduction

This document describes the requirements and architecture for the implementation of a PDM Microphone Interface block.

13.8.2.1 Overview

PDM is a popular way to deliver audio from microphones to the processor in several applications, such as mobile telephones. However, current digital-audio systems use multibit audio signal (also known as multibit PCM) to represent the signal. For this purpose a set of FIR, CIC or/and Half Band filters are usually implemented on DSPs or software. This block implements the required digital interface to provide a 16-bit audio signal from a PDM microphone bitstream in a configurable output sampling rate.

The implementation of this digital interface is based on the application of digital signal processing techniques in hardware. The PDM Microphone Interface architecture was designed to gate saving and minimal power consumption. It implements a bunch of filters to transform a 1-bit PDM bitstream to a 16-bit PCM signal in the audio band. To avoid aliasing frequencies in passband, the overall filter has 80 dB stopband attenuation and passband ripple less than 0.2dB.

The whole module is implemented to work in a multichannel mode. All channels have the same configuration but each input channel could be turned on/off independently.

Figure 13-92 shows the block diagram for the PDM Microphone Interface module. It is composed by:

- an input interface for each pair of PDM microphones.
- a decimation filter by channel.
- a FIFO by channel.
- a time generation unit.
- shared interfaces to DMA, interrupts and SoC.
- One or more Hardware Voice Activity Detectors (HWVAD).

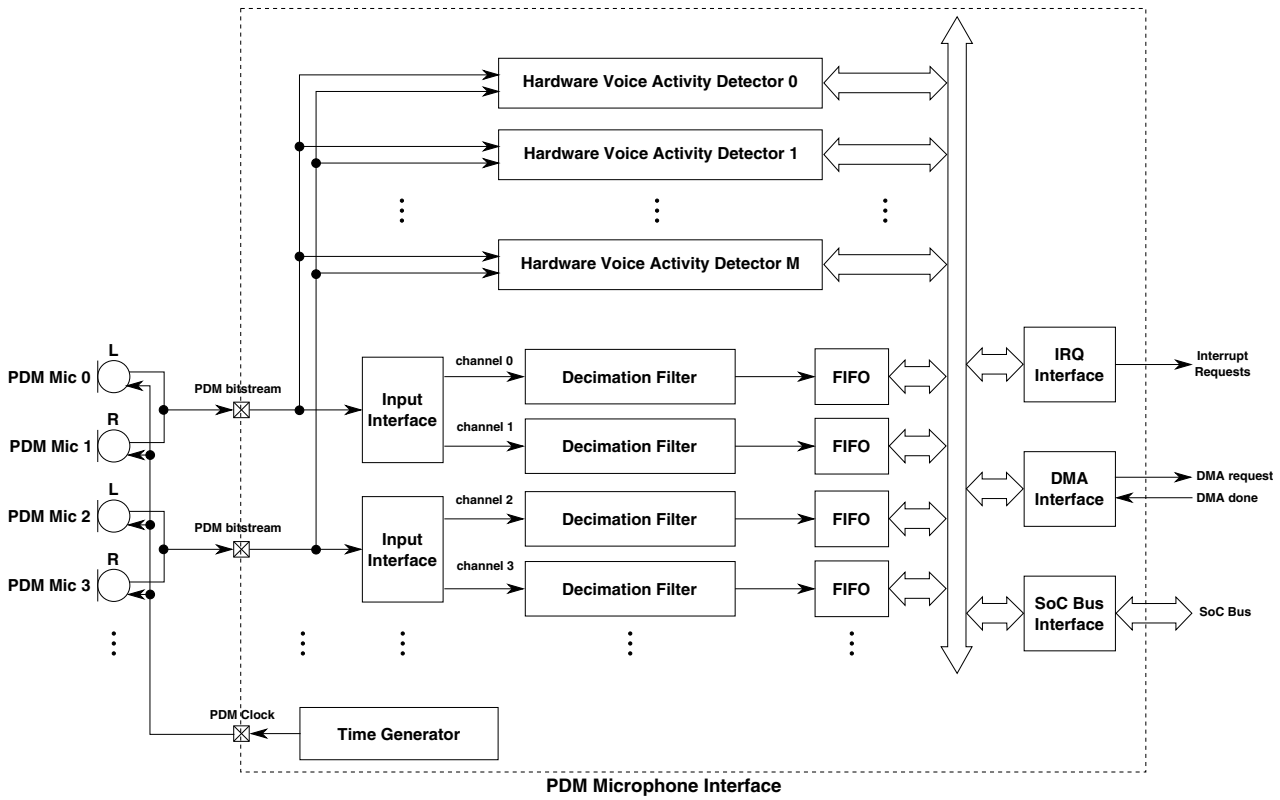


Figure 13-92. PDM Microphone Interface Block Diagram

The Time Generator unit generates a PDM clock to all microphones. All microphones must operate at the same clock frequency. Each input interface receives a time multiplexed PDM bitstream from two PDM microphones and it separates the two audio information in two channels: left (0) and right (1). Each channel is processed separately.

The Decimation Filter is composed internally by a CIC filter, a DC remover, a FIR and a Half Band filter. The Half Band and FIR filters are implemented in the traditional way . The filtering results are stored in individual FIFOs; a FIFO by channel. The FIFOs have overflow and underflow detectors to deliver an error interrupt request.

An interrupt request or DMA request could be delivered by the MICFIL. Then the filters results stored in FIFOs could be accessed by the SoC or DMA respectively via the internal bus interface .

The HWVAD takes data from the input of a selected PDM microphone to detect if there is any voice activity. When a voice activity is detected, an interrupt could be delivered to the system.

13.8.2.2 Features

This PDM Microphone Interface (MICFIL) block includes these main features:

- Fixed filtering characteristics for audio application.
- Full or partial set of channels operation with individual enable control.
- Programmable PDM clock generator.
- Programmable decimation rate.
- 16-bit signed output result.
- Overall stopband attenuation more than 80dB.
- Overall passband ripple less than 0.2dB.
- CIC filter
  - 5th order.
  - Programmable decimation rate.
- DC remover
  - First order IIR filter.
  - Programmable cut-off frequency.
- FIFOs with DMA capability.
  - Each FIFO is 8 entries length.
- Hardware Voice Activity Detector (HWVAD).
  - Interrupt capability.
  - Zero-Crossing Detection (ZCD) option.

13.8.2.3 Modes of operation

This section describes the operation modes of the MICFIL. These depends on the CPU power mode , the SoC Doze mode , the SoC Debug mode and MDIS, DBGE, DBG and DOZEN bitfields. The mode selection is summarized in Table 13-56.

Table 13-56. Operation Mode Selection

Mode	MDIS	SoC Doze & DOZEN=1	CPU Power Mode	DOZEN	SoC Debug | DBG	DBGE	CPU clock	MICFIL internal clock	Decimation Filter state	HWVAD[n] state
Normal	0	0	HSRUN / RUN / VLPR	X	0	X	Enabled	Enabled	Run (Depends on PDMIEN and CHENn)	Run (Depends on PDMIEN and VADEN)
Debug					1	1				
						0			Stop	Stop
Low Power			STOP/ VLPS	0	X	X	Gated (Externally)		Run (Depends on PDMIEN and CHENn)	Run (Depends on PDMIEN and VADEN)[1]
			LLS/VLLS						Stop	
Disable/Low Leakage			STOP/ VLPS/LLS/VLLS	1				Gated		Stop
		1	Any	X			Enabled			
	1	X					Gated (Internally)			

1. HWVAD[n] will RUN in VLLS/LLS only if it was placed on an always-on power domain in the SoC.

13.8.2.3.1 Normal mode

This is the default operational mode of the MICFIL block. In this mode the channels can be enabled and perform filtering operation. The input data are supplied by the PDM bitstreams and the PDM Clock is delivered to the PDM microphones.

13.8.2.3.2 Low Power mode

The Low Power mode is requested by the SoC only. In this mode the CPU clock is gated externally and the MICFIL is kept running. The MICFIL is in Low Power mode when the SoC is in STOP, VLPS, LLS or VLLS mode and DOZEN =0. For more information see Low Power Mode.

13.8.2.3.3 Debug mode

This mode is used to check the status of the IP when requested. In this mode the MICFIL can be stopped after the remaining data processing has finished depending on the DBGE bit value. This mode can be actived by a SoC debug request or asserting the DBG bitfield. For more information see Debug Mode.

13.8.2.3.4 Disable/Low Leakage mode

The Disable/Low Leakage (DLL) mode can be requested by the SoC or by the MICFIL. In this mode the MICFIL is stopped after the remaining data processing has finished. The MICFIL is in Disable/Low Leakage (DLL) mode when:

- The SoC is in STOP, VLPS, LLS or VLLS mode and DOZEN=1.
- The SoC is in Doze and DOZEN bit is set.
- MDIS is set.

Refer to Disable/Low Leakage mode section for more details.

13.8.2.4 Quality Modes

Independently of the power mode selected, if the decimation filter is running, the MICFIL can operate in 6 different quality modes, those can be selected by the QSEL bitfield in the CTRL_2 register. The decimation and interpolation rates in internal blocks are shown in Table 13-57. The quality selected can be configured dinamically when the Decimation Filter is running, the output rate is mantained in every quality selected.

The passband of the Decimation Filter depends on the quality mode, see Overall Bandpass for more information.

Table 13-57. Quality Modes

Quality Mode	QSEL bitfield	Sampler Interpolation	CIC Filter Decimation	Half Band Filter Decimation	FIR Filter Decimation	PDM Clock Rate	Passband	Relative Gain
High Quality	001	-	:(2OSR)	:2	:2	Output Rate x 8 x OSR	To ~0.5 x Output Rate	2^(OUTGAIN+5)
Medium Quality	000	-	:OSR	:2	:2	Output Rate x 4 x OSR	To ~0.5 x Output Rate	2^(OUTGAIN)
Low Quality	111	x2	:OSR	:2	:2	Output Rate x 2 x OSR	To ~0.5 x Output Rate	2^(OUTGAIN)
Very Low Quality 0	110	-	:(2OSR)	:2	-	Output Rate x 4 x OSR	To ~0.25 x Output Rate	2^(OUTGAIN+7)
Very Low Quality 1	101	-	:OSR	:2	-	Output Rate x 2 x OSR	To ~0.25 x Output Rate	2^(OUTGAIN+2)
Very Low Quality 2	100	x2	:OSR	:2	-	Output Rate x OSR	To ~0.25 x Output Rate	2^(OUTGAIN+2)

In order to mantain the same output rate in the Decimation Filter when the quality is changed, the PDM Clock rate changes depending on the quality selected. For more details about the PDM Clock see Time Generator.

The Sampler and the FIR filters are whether enabled or not depending on the quality selected, achieving a low power consumption when they are disabled. The CIC filter decimation rate is also whether doubled or not depending on the quality selected. For more information about see Decimation Filter.

13.8.3 External signal description

Note

The PDM Microphone Interface module does not provide metastability protection or filtering for input data. The data must be generated using the provided PDM clock.

13.8.3.1 PDM data bitstream bus

The PDM bitstream bus is the set of data bitstreams received from each PDM microphone. For more details, see Input Interface.

13.8.3.2 PDM generated clock

The PDM generated clock signal is the clocking signal generated to be delivered to the PDM microphones. The generated clock frequency is configurable. For more details, see Time Generator.

13.8.4 MICFIL register descriptions

This section provides the memory map and detailed description of all MICFIL registers. Any access to reserved areas can issue a slave bus error indication.

The registers allocated in this memory map are implemented for up to 4 pairs of microphones, that means 8 channels are available.

13.8.4.1 MICFIL Memory map

PDM_INTFC base address: 3008_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	MICFIL Control register 1 (CTRL_1)	32	RW	0000_0000h
4h	MICFIL Control register 2 (CTRL_2)	32	RW	0000_0000h
8h	MICFIL Status register (STAT)	32	W1C	0000_0000h
10h	MICFIL FIFO Control register (FIFO_CTRL)	32	RW	0000_0007h
14h	MICFIL FIFO Status register (FIFO_STAT)	32	W1C	0000_0000h
24h - 40h	MICFIL Output Result Register (DATACH0 - DATACH7)	32	RO	0000_0000h
64h	MICFIL DC Remover Control register (DC_CTRL)	32	RW	0000_0000h
74h	MICFIL Output Control register (OUT_CTRL)	32	RW	0000_0000h
7Ch	MICFIL Output Status register (OUT_STAT)	32	W1C	0000_0000h
90h	Voice Activity Detector 0 Control register (VAD0_CTRL_1)	32	RW	0000_0000h
94h	Voice Activity Detector 0 Control register (VAD0_CTRL_2)	32	RW	000A_0000h
98h	Voice Activity Detector 0 Status register (VAD0_STAT)	32	W1C	8000_0000h
9Ch	Voice Activity Detector 0 Signal Configuration (VAD0_SCONFIG)	32	RW	0000_0000h
A0h	Voice Activity Detector 0 Noise Configuration (VAD0_NCONFIG)	32	RW	8000_0000h
A4h	Voice Activity Detector 0 Noise Data (VAD0_NDATA)	32	RO	0000_0000h
A8h	Voice Activity Detector 0 Zero-Crossing Detector (VAD0_ZCD)	32	RW	0000_0004h

13.8.4.2 MICFIL Control register 1 (CTRL_1)

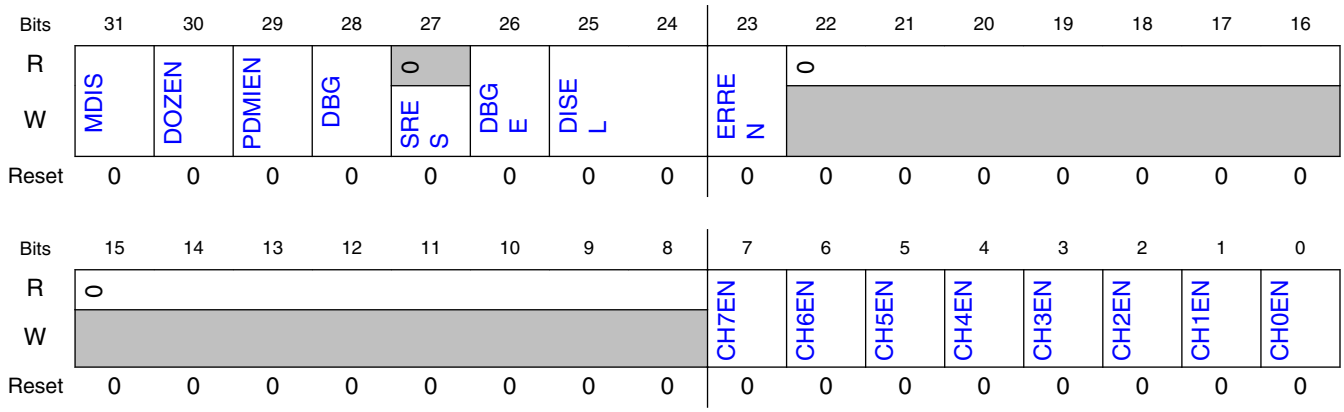
13.8.4.2.1 Offset

Register	Offset
CTRL_1	0h

13.8.4.2.2 Function

This MICFIL module configuration register provides configuration control bits for all sub-blocks and internal logic.

13.8.4.2.3 Diagram



13.8.4.2.4 Fields

Field	Function
31 MDIS	Module Disable The MDIS bit puts the whole PDM Interface in Disable/Low Leakage mode. The generated PDM Clock is stopped and consequently no data is generated by the microphones. Please see more details on Disable/Low Leakage mode. **NOTE:** When MDIS=1, writes to CTRL_1 and CTRL_2 only are allowed with the exception of writes to the DBG and SRES bits. A slave bus error indication will be issued when any register except CTRL_1 and CTRL_2 is written when MDIS=1. 0b - Normal Mode 1b - Disable/Low Leakage Mode
30 DOZEN	DOZE enable When SoC is in Doze mode and DOZEN is active the module enters in Disable/Low Leakage Mode. 0b - DOZE enable bit is not asserted 1b - DOZE enable bit is asserted
29 PDMIEN	PDM Inteface Enable The PDMIEN bit enables the operation of the filters in the module. Only the channels enabled by CHnEN starts to operate. It starts the clock generation and the overall filters operation. When it is negated, both the enabled channels and the clock generation will be stopped after finish the current filtering and only the pending data in the buffer could be processed by DMA or interrupt requests. **NOTE:** This field must be the latest to be enabled, for more information see the Initialization Procedure without HWVAD. 0b - PDM Interface disabled 1b - PDM Interface enabled.
28 DBG	Debug Mode This bit enables the Debug mode of the PDM interface. The module could be enabled or disabled in Debug mode depending on the DBGE bit value. See Debug Mode for more details. This bit is writable only when MDIS is cleared. 0b - PDM Interface is in Normal Mode. 1b - PDM Interface is in Debug Mode.
27 SRES	Software-reset bit

Table continues on the next page...

Field	Function
	The SRES is a self-negated bit which provides the CPU with the capability to initialize the PDM Interface module through the slave-bus interface. This bit always reads as zero, and this bit is only effective when MDIS is cleared. Refer to Soft-reset 0b - No action 1b - Software reset
26 DBGE	Module Enable in Debug Enables/disables the PDM interface operation in Debug mode. See Debug Mode for more details. 0b - PDM Interface is disabled in debug mode, after completing the current frame. 1b - PDM Interface is enabled in debug mode.
25-24 DISEL	DMA Interrupt Selection The DISEL field selects if the module performs DMA requests or filter interrupt requests when the number of FIFO elements surpasses the configured watermark. Note that requests are only generated from the channels whose CHnEN bits are asserted. **NOTE:** This bit mustn't change when BSY_FIL bit is asserted 00b - DMA and interrupt requests disabled 01b - DMA requests enabled 10b - Interrupt requests enabled 11b - Reserved
23 ERREN	Error Interruption Enable The ERREN bit enables Error Interrupts in the PDM Interface. This error could be caused by an overflow or underflow in the FIFOs, an overflow or underflow in the Decimation Filter output or when the MICFIL internal clock frequency is not enough to filter data of all enabled channels. The specific error type could be checked on the FIFOOVF or FIFOUND bits, the OUTOVF or OUTUNF bits or in the LOWFREQF bit. **NOTE:** This bit mustn't change when BSY_FIL bit is asserted 0b - Error Interrupts disabled 1b - Error Interrupts enabled
22-16 —	Reserved
15-8 —	Reserved
7 CH7EN	Channel 7 Enable The CH7EN bit enables the Channel 7 processing. Overall channel filters are enabled, DMA or interrupt request could be enabled depending of the DISEL bit. When is negated the overall channel filters and respective FIFO are disabled. It must be changed only when PDMIEN and BSY_FIL are negated. See Initialization Procedure without HWVAD for more information. **NOTE:** This bit mustn't change when BSY_FIL bit is asserted. 0b - Channel 7 disabled 1b - Channel 7 enabled
6 CH6EN	Channel 6 Enable The CH6EN bit enables the Channel 6 processing. Overall channel filters are enabled, DMA or interrupt request could be enabled depending of the DISEL bit. When is negated the overall channel filters and respective FIFO are disabled. It must be changed only when PDMIEN and BSY_FIL are negated. See Initialization Procedure without HWVAD for more information. **NOTE:** This bit mustn't change when BSY_FIL bit is asserted. 0b - Channel 6 disabled 1b - Channel 6 enabled
5	Channel 5 Enable

Table continues on the next page...

Field	Function
CH5EN	The CH5EN bit enables the Channel 5 processing. Overall channel filters are enabled, DMA or interrupt request could be enabled depending of the DISEL bit. When is negated the overall channel filters and respective FIFO are disabled. It must be changed only when PDMIEN and BSY_FIL are negated. See Initialization Procedure without HWVAD for more information. **NOTE:** This bit mustn't change when BSY_FIL bit is asserted. 0b - Channel 5 disabled 1b - Channel 5 enabled
4 CH4EN	Channel 4 Enable The CH4EN bit enables the Channel 4 processing. Overall channel filters are enabled, DMA or interrupt request could be enabled depending of the DISEL bit. When is negated the overall channel filters and respective FIFO are disabled. It must be changed only when PDMIEN and BSY_FIL are negated. See Initialization Procedure without HWVAD for more information. **NOTE:** This bit mustn't change when BSY_FIL bit is asserted. 0b - Channel 4 disabled 1b - Channel 4 enabled
3 CH3EN	Channel 3 Enable The CH3EN bit enables the Channel 3 processing. Overall channel filters are enabled, DMA or interrupt request could be enabled depending of the DISEL bit. When is negated the overall channel filters and respective FIFO are disabled. It must be changed only when PDMIEN and BSY_FIL are negated. See Initialization Procedure without HWVAD for more information. **NOTE:** This bit mustn't change when BSY_FIL bit is asserted. 0b - Channel 3 disabled 1b - Channel 3 enabled
2 CH2EN	Channel 2 Enable The CH2EN bit enables the Channel 2 processing. Overall channel filters are enabled, DMA or interrupt request could be enabled depending of the DISEL bit. When is negated the overall channel filters and respective FIFO are disabled. It must be changed only when PDMIEN and BSY_FIL are negated. See Initialization Procedure without HWVAD for more information. **NOTE:** This bit mustn't change when BSY_FIL bit is asserted. 0b - Channel 2 enabled 1b - Channel 2 disabled
1 CH1EN	Channel 1 Enable The CH1EN bit enables the Channel 1 processing. Overall channel filters are enabled, DMA or interrupt request could be enabled depending of the DISEL bit. When is negated the overall channel filters and respective FIFO are disabled. It must be changed only when PDMIEN and BSY_FIL are negated. See Initialization Procedure without HWVAD for more information. **NOTE:** This bit mustn't change when BSY_FIL bit is asserted. 0b - Channel 1 disabled 1b - Channel 1 enabled
0 CH0EN	Channel 0 Enable The CH0EN bit enables the Channel 0 processing. Overall channel filters are enabled, DMA or interrupt request could be enabled depending of the DISEL bit. When is negated the overall channel filters and respective FIFO are disabled. It must be changed only when PDMIEN and BSY_FIL are negated. See Initialization Procedure without HWVAD for more information. **NOTE:** This bit mustn't change when BSY_FIL bit is asserted. 0b - Channel 0 disabled 1b - Channel 0 enabled

13.8.4.3 MICFIL Control register 2 (CTRL_2)

13.8.4.3.1 Offset

Register	Offset
CTRL_2	4h

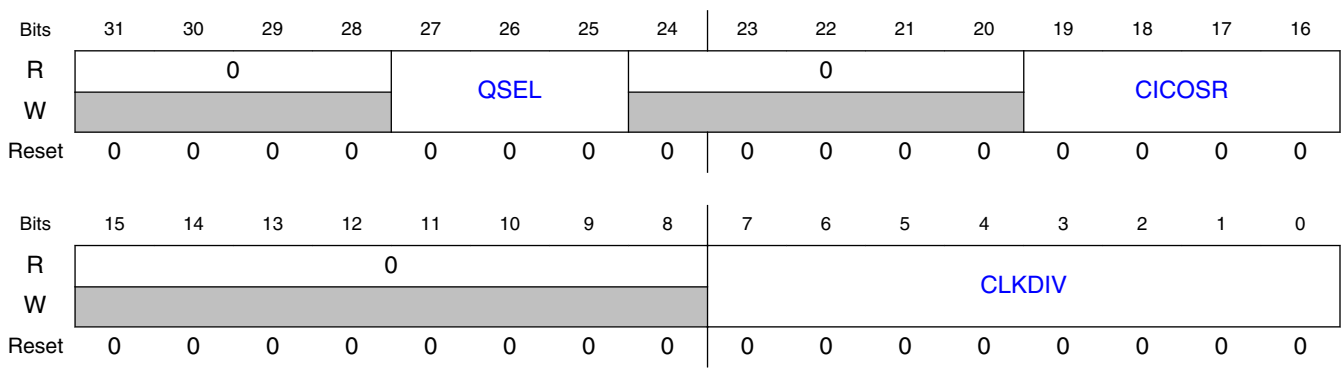
13.8.4.3.2 Function

This MICFIL module configuration register provides configuration control bits for all sub-blocks and internal logic.

NOTE

CLKDIV=0 has the same effect than CLKDIV=1. CLKDIV=0 or 1 has the same effect than CLKDIV=2 during Medium Quality mode. CLKDIV=0 or 1 is not allowed during High Quality mode.

13.8.4.3.3 Diagram



13.8.4.3.4 Fields

Field	Function
31-28 —	Reserved
27-25 QSEL	Quality Select This bit defines the actual quality mode of the Decimation Filter. See more information in Quality Modes.

Table continues on the next page...

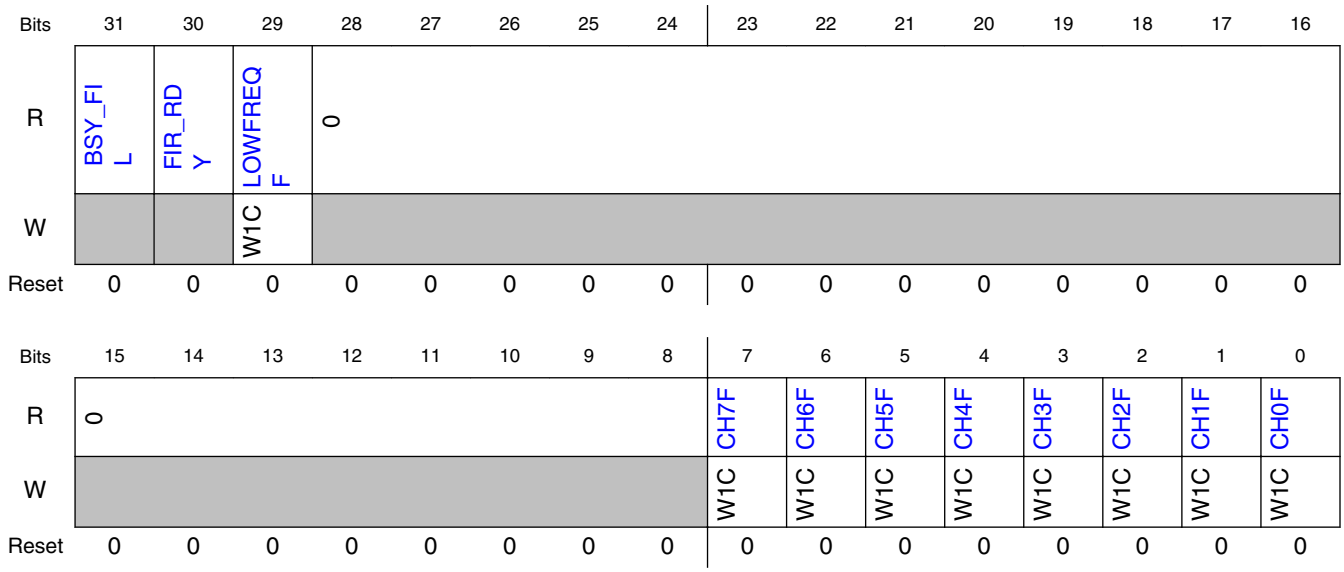
Field	Function
	NOTE: This bitfield mustn't change when BSY_FIL bit is asserted. 000b - Medium quality mode. 001b - High quality mode. 100b - Very low quality 2 mode. 101b - Very low quality 1 mode. 110b - Very low quality 0 mode. 111b - Low quality mode.
24-20 —	Reserved
19-16 CICOSR	CIC Oversampling Rate This bitfield defines the oversampling rate of the CIC filter. In case the Decimation Filter is in High or Very Low Quality 0 mode the oversampling rate is 2(16-CICOSR), otherwise it is (16-CICOSR). See CIC Filter for more information. **NOTE:** This bitfield mustn't change when BSY_FIL bit is asserted.
15-8 —	Reserved
7-0 CLKDIV	Clock Divider The CLKDIV field sets the divisor for the MICFIL internal clock. The division generates the PDM microphones clock and determines the output sampling rate of the overall filtering. See Time Generator for more information. **NOTE:** This bitfield mustn't change when BSY_FIL bit is asserted.

13.8.4.4 MICFIL Status register (STAT)

13.8.4.4.1 Offset

Register	Offset
STAT	8h

13.8.4.4.2 Diagram



13.8.4.4.3 Fields

Field	Function
31 BSY_FIL	Decimation Filter Busy Flag This flag signalizes when at least a Decimation Filter channel or at least a HWVAD is running and the PDM clock is being generated. See Busy Flag for more information. 0b - All Decimation Filters are stopped. 1b - At least one Decimation Filter channel is running.
30 FIR_RDY	FIR Filter Data Ready This bit indicates that all the TAPs from FIR Filter have been updated, and all data provided from now on will be reliable. After PDMIEN is set this bit field remains low for a time interval correspondent to about 68 FIR results. **NOTE:** When FIR is bypassed, it depends on Half Band filter. 0b - FIR Filter data not reliable. 1b - FIR Filter data reliable.
29 LOWFREQF	Low Frequency Flag This bit indicates that the CLKDIV bitfield value is not high enough to filter all enabled channels, and the FIR or Half Band filters didn't finish their operations before the next sample arrived. For more information see Minimum Required CLKDIV bitfield value. Write a logic 1 to this field to clear this flag. **NOTE:** The filtering results in this condition are unpredictable. 0b - CLKDIV value is OK. 1b - CLKDIV value is too low.
28-16 —	Reserved
15-8 —	Reserved
7	Channel 7 Output Data Flag

Table continues on the next page...

Field	Function
CH7F	The CH7F bit flag indicates that the FIFO from this channel has surpassed its watermark value and the data is available at the DATACH7 register. This flag contributes to generate an Interrupt or DMA request if enabled by DISEL in the CTRL_1. Write a logic 1 to this field to clear this flag. If DMA request is set (DISEL = 01) the CH7F flag will be set according to watermark level and will be cleared , in the other cases (IRQ included) the CH7F flag will be set according to the watermark level and will be cleared with the w1c functionality of the bit. **NOTE:** If DMA is enabled, this flag is self-cleared when the DMA transaction is finished. 0b - Channel's FIFO did not reach the number of elements configured in watermark bit-field. 1b - Channel's FIFO reached the number of elements configured in watermark bit-field.
6 CH6F	Channel 6 Output Data Flag The CH6F bit flag indicates that the FIFO from this channel has surpassed its watermark value and the data is available at the DATACH6 register. This flag contributes to generate an Interrupt or DMA request if enabled by DISEL in the CTRL_1. Write a logic 1 to this field to clear this flag. If DMA request is set (DISEL = 01) the CH6F flag will be set according to watermark level and will be cleared , in the other cases (IRQ included) the CH6F flag will be set according to the watermark level and will be cleared with the w1c functionality of the bit. **NOTE:** If DMA is enabled, this flag is self-cleared when the DMA transaction is finished. 0b - Channel's FIFO did not reach the number of elements configured in watermark bit-field. 1b - Channel's FIFO reached the number of elements configured in watermark bit-field.
5 CH5F	Channel 5 Output Data Flag The CH5F bit flag indicates that the FIFO from this channel has surpassed its watermark value and the data is available at the DATACH5 register. This flag contributes to generate an Interrupt or DMA request if enabled by DISEL in the CTRL_1. Write a logic 1 to this field to clear this flag. If DMA request is set (DISEL = 01) the CH5F flag will be set according to watermark level and will be cleared , in the other cases (IRQ included) the CH5F flag will be set according to the watermark level and will be cleared with the w1c functionality of the bit. **NOTE:** If DMA is enabled, this flag is self-cleared when the DMA transaction is finished. 0b - Channel's FIFO did not reach the number of elements configured in watermark bit-field. 1b - Channel's FIFO reached the number of elements configured in watermark bit-field.
4 CH4F	Channel 4 Output Data Flag The CH4F bit flag indicates that the FIFO from this channel has surpassed its watermark value and the data is available at the DATACH4 register. This flag contributes to generate an Interrupt or DMA request if enabled by DISEL in the CTRL_1. Write a logic 1 to this field to clear this flag. If DMA request is set (DISEL = 01) the CH4F flag will be set according to watermark level and will be cleared , in the other cases (IRQ included) the CH4F flag will be set according to the watermark level and will be cleared with the w1c functionality of the bit. **NOTE:** If DMA is enabled, this flag is self-cleared when the DMA transaction is finished. 0b - Channel's FIFO did not reach the number of elements configured in watermark bit-field. 1b - Channel's FIFO reached the number of elements configured in watermark bit-field.
3 CH3F	Channel 3 Output Data Flag The CH3F bit flag indicates that the FIFO from this channel has surpassed its watermark value and the data is available at the DATACH3 register. This flag contributes to generate an Interrupt or DMA request if enabled by DISEL in the CTRL_1. Write a logic 1 to this field to clear this flag. If DMA request is set (DISEL = 01) the CH3F flag will be set according to watermark level and will be cleared , in the other cases (IRQ included) the CH3F flag will be set according to the watermark level and will be cleared with the w1c functionality of the bit. **NOTE:** If DMA is enabled, this flag is self-cleared when the DMA transaction is finished. 0b - Channel's FIFO did not reach the number of elements configured in watermark bit-field. 1b - Channel's FIFO reached the number of elements configured in watermark bit-field.
2	Channel 2 Output Data Flag

Table continues on the next page...

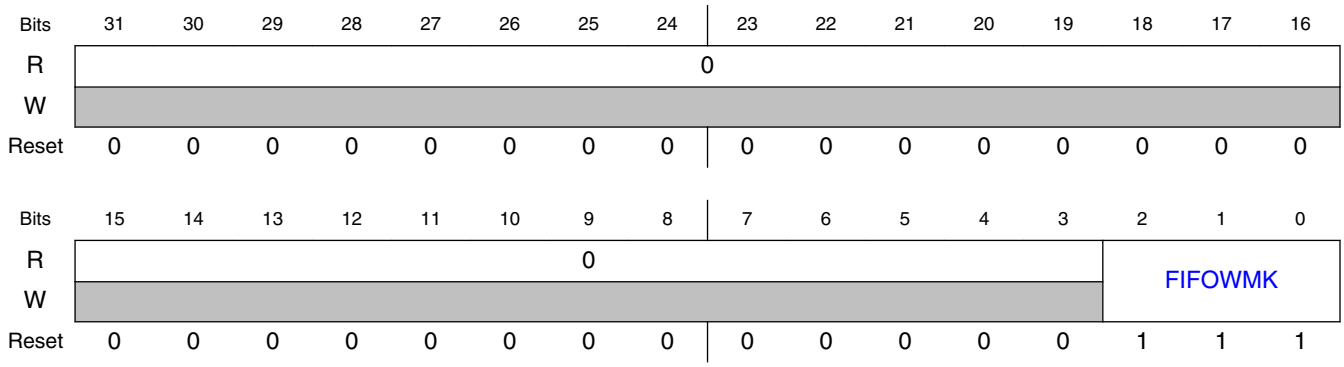
Field	Function
CH2F	The CH2F bit flag indicates that the FIFO from this channel has surpassed its watermark value and the data is available at the DATACH2 register. This flag contributes to generate an Interrupt or DMA request if enabled by DISEL in the CTRL_1. Write a logic 1 to this field to clear this flag. If DMA request is set (DISEL = 01) the CH2F flag will be set according to watermark level and will be cleared , in the other cases (IRQ included) the CH2F flag will be set according to the watermark level and will be cleared with the w1c functionality of the bit. **NOTE:** If DMA is enabled, this flag is self-cleared when the DMA transaction is finished. 0b - Channel's FIFO did not reach the number of elements configured in watermark bit-field. 1b - Channel's FIFO reached the number of elements configured in watermark bit-field.
1 CH1F	Channel 1 Output Data Flag The CH1F bit flag indicates that the FIFO from this channel has surpassed its watermark value and the data is available at the DATACH1 register. This flag contributes to generate an Interrupt or DMA request if enabled by DISEL in the CTRL_1. Write a logic 1 to this field to clear this flag. If DMA request is set (DISEL = 01) the CH1F flag will be set according to watermark level and will be cleared , in the other cases (IRQ included) the CH1F flag will be set according to the watermark level and will be cleared with the w1c functionality of the bit. **NOTE:** If DMA is enabled, this flag is self-cleared when the DMA transaction is finished. 0b - Channel's FIFO did not reach the number of elements configured in watermark bit-field. 1b - Channel's FIFO reached the number of elements configured in watermark bit-field.
0 CH0F	Channel 0 Output Data Flag The CH0F bit flag indicates that the FIFO from this channel has surpassed its watermark value and the data is available at the DATACH0 register. This flag contributes to generate an Interrupt or DMA request if enabled by DISEL in the CTRL_1. Write a logic 1 to this field to clear this flag. If DMA request is set (DISEL = 01) the CH0F flag will be set according to watermark level and will be cleared , in the other cases (IRQ included) the CH0F flag will be set according to the watermark level and will be cleared with the w1c functionality of the bit. **NOTE:** If DMA is enabled, this flag is self-cleared when the DMA transaction is finished. 0b - Chanel's FIFO did not reach the number of elements configured in watermark bit-field. 1b - Channel's FIFO reached the number of elements configured in watermark bit-field.

13.8.4.5 MICFIL FIFO Control register (FIFO_CTRL)

13.8.4.5.1 Offset

Register	Offset
FIFO_CTRL	10h

13.8.4.5.2 Diagram



13.8.4.5.3 Fields

Field	Function
31-3 —	Reserved
2-0 FIFOWMK	FIFO Watermark Control The FIFOWMK controls the watermark of the FIFO used to set CHnF. When the number of results in the FIFO is greater than FIFOWMK value then the CHnF flag is set. A DMA request or interrupt can be also generated according to DISEL bit-field. **NOTE:** This mustn't change when BSY_FIL is asserted. Before changing the watermark value service all filtered data and clear channel flags.

13.8.4.6 MICFIL FIFO Status register (FIFO_STAT)

13.8.4.6.1 Offset

Register	Offset
FIFO_STAT	14h

13.8.4.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FIFOUND7	FIFOUND6	FIFOUND5	FIFOUND4	FIFOUND3	FIFOUND2	FIFOUND1	FIFOUND0	FIFOOVF7	FIFOOVF6	FIFOOVF5	FIFOOVF4	FIFOOVF3	FIFOOVF2	FIFOOVF1	FIFOOVF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.8.4.6.3 Fields

Field	Function
31-16 —	Reserved
15 FIFOUND7	FIFO Underflow Exception flag for channel 7 The FIFOUND bit indicates an exceptional underflow condition in the FIFO. If the FIFO is empty and a read operation is requested then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO Underflow 1b - Exception by FIFO underflow
14 FIFOUND6	FIFO Underflow Exception flag for channel 6 The FIFOUND bit indicates an exceptional underflow condition in the FIFO. If the FIFO is empty and a read operation is requested then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO Underflow 1b - Exception by FIFO underflow
13 FIFOUND5	FIFO Underflow Exception flag for channel 5 The FIFOUND bit indicates an exceptional underflow condition in the FIFO. If the FIFO is empty and a read operation is requested then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO Underflow 1b - Exception by FIFO underflow
12 FIFOUND4	FIFO Underflow Exception flag for channel 4 The FIFOUND bit indicates an exceptional underflow condition in the FIFO. If the FIFO is empty and a read operation is requested then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO Underflow 1b - Exception by FIFO underflow
11 FIFOUND3	FIFO Underflow Exception flag for channel 3 The FIFOUND bit indicates an exceptional underflow condition in the FIFO. If the FIFO is empty and a read operation is requested then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO Underflow 1b - Exception by FIFO underflow

Table continues on the next page...

Field	Function
10 FIFOUND2	FIFO Underflow Exception flag for channel 2 The FIFOUND bit indicates an exceptional underflow condition in the FIFO. If the FIFO is empty and a read operation is requested then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO Underflow 1b - Exception by FIFO underflow
9 FIFOUND1	FIFO Underflow Exception flag for channel 1 The FIFOUND bit indicates an exceptional underflow condition in the FIFO. If the FIFO is empty and a read operation is requested then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO Underflow 1b - Exception by FIFO underflow
8 FIFOUND0	FIFO Underflow Exception flag for channel 0 The FIFOUND bit indicates an exceptional underflow condition in the FIFO. If the FIFO is empty and a read operation is requested then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO Underflow 1b - Exception by FIFO underflow
7 FIFOOVF7	FIFO Overflow Exception flag for channel 7 The FIFOOVF bit indicates an exceptional overflow condition in the FIFO. If the FIFO is full and an additional result is received to be written then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO overflow 1b - Exception by FIFO overflow
6 FIFOOVF6	FIFO Overflow Exception flag for channel 6 The FIFOOVF bit indicates an exceptional overflow condition in the FIFO. If the FIFO is full and an additional result is received to be written then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO overflow 1b - Exception by FIFO overflow
5 FIFOOVF5	FIFO Overflow Exception flag for channel 5 The FIFOOVF bit indicates an exceptional overflow condition in the FIFO. If the FIFO is full and an additional result is received to be written then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO overflow 1b - Exception by FIFO overflow
4 FIFOOVF4	FIFO Overflow Exception flag for channel 4 The FIFOOVF bit indicates an exceptional overflow condition in the FIFO. If the FIFO is full and an additional result is received to be written then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO overflow 1b - Exception by FIFO overflow
3 FIFOOVF3	FIFO Overflow Exception flag for channel 3 The FIFOOVF bit indicates an exceptional overflow condition in the FIFO. If the FIFO is full and an additional result is received to be written then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO overflow 1b - Exception by FIFO overflow
2 FIFOOVF2	FIFO Overflow Exception flag for channel 2 The FIFOOVF bit indicates an exceptional overflow condition in the FIFO. If the FIFO is full and an additional result is received to be written then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO overflow 1b - Exception by FIFO overflow
1 FIFOOVF1	FIFO Overflow Exception flag for channel 1 The FIFOOVF bit indicates an exceptional overflow condition in the FIFO. If the FIFO is full and an additional result is received to be written then this flag is set. Write a logic 1 to this field to clear this flag.

Table continues on the next page...

Field	Function
	0b - No exception by FIFO overflow 1b - Exception by FIFO overflow
0 FIFOOVF0	FIFO Overflow Exception flag for channel 0 The FIFOOVF bit indicates an exceptional overflow condition in the FIFO. If the FIFO is full and an additional result is received to be written then this flag is set. Write a logic 1 to this field to clear this flag. 0b - No exception by FIFO overflow 1b - Exception by FIFO overflow

13.8.4.7 MICFIL Output Result Register (DATACH0 - DATACH7)

13.8.4.7.1 Offset

For a = 0 to 7:

Register	Offset
DATACHa	24h + (a × 4h)

13.8.4.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.8.4.7.3 Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Channel n Data The DATA field is the result sample at the Channel n FIFO. The samples are integer signed 16-bit values in two's complement format.

13.8.4.8 MICFIL DC Remover Control register (DC_CTRL)

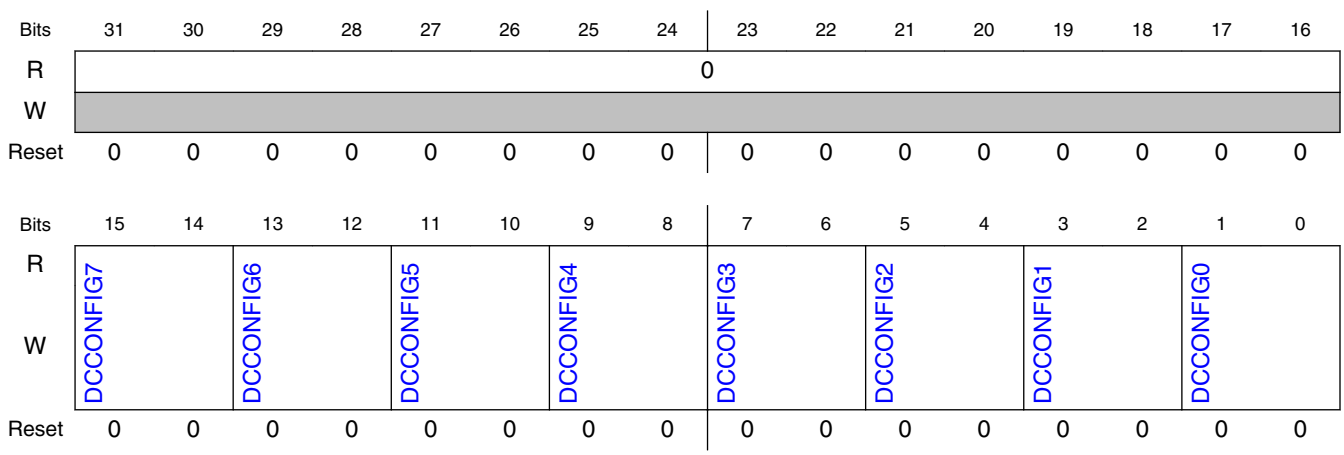
13.8.4.8.1 Offset

Register	Offset
DC_CTRL	64h

13.8.4.8.2 Function

This MICFIL module configuration register provides configuration control bits for all sub-blocks and internal logic.

13.8.4.8.3 Diagram



13.8.4.8.4 Fields

Field	Function
31-16 —	Reserved
15-14 DCCONFIG7	Channel 7 DC Remover Configuration This bitfield defines the value of the cut-off frequency of the DC Remover. 00b - DC Remover cut-off at 21Hz. 01b - DC Remover cut-off at 83Hz. 10b - DC Remover cut-off at 152Hz. 11b - DC Remover is bypassed.
13-12	Channel 6 DC Remover Configuration

Table continues on the next page...

Field	Function
DCCONFIG6	This bitfield defines the value of the cut-off frequency of the DC Remover. 00b - DC Remover cut-off at 21Hz. 01b - DC Remover cut-off at 83Hz. 10b - DC Remover cut-off at 152Hz. 11b - DC Remover is bypassed.
11-10	Channel 5 DC Remover Configuration
DCCONFIG5	This bitfield defines the value of the cut-off frequency of the DC Remover. 00b - DC Remover cut-off at 21Hz. 01b - DC Remover cut-off at 83Hz. 10b - DC Remover cut-off at 152Hz. 11b - DC Remover is bypassed.
9-8	Channel 4 DC Remover Configuration
DCCONFIG4	This bitfield defines the value of the cut-off frequency of the DC Remover. 00b - DC Remover cut-off at 21Hz. 01b - DC Remover cut-off at 83Hz. 10b - DC Remover cut-off at 152Hz. 11b - DC Remover is bypassed.
7-6	Channel 3 DC Remover Configuration
DCCONFIG3	This bitfield defines the value of the cut-off frequency of the DC Remover. 00b - DC Remover cut-off at 21Hz. 01b - DC Remover cut-off at 83Hz. 10b - DC Remover cut-off at 152Hz. 11b - DC Remover is bypassed.
5-4	Channel 2 DC Remover Configuration
DCCONFIG2	This bitfield defines the value of the cut-off frequency of the DC Remover. 00b - DC Remover cut-off at 21Hz. 01b - DC Remover cut-off at 83Hz. 10b - DC Remover cut-off at 152Hz. 11b - DC Remover is bypassed.
3-2	Channel 1 DC Remover Configuration
DCCONFIG1	This bitfield defines the value of the cut-off frequency of the DC Remover. 00b - DC Remover cut-off at 21Hz. 01b - DC Remover cut-off at 83Hz. 10b - DC Remover cut-off at 152Hz. 11b - DC Remover is bypassed.
1-0	Channel 0 DC Remover Configuration
DCCONFIG0	This bitfield defines the value of the cut-off frequency of the DC Remover. 00b - DC Remover cut-off at 21Hz. 01b - DC Remover cut-off at 83Hz. 10b - DC Remover cut-off at 152Hz. 11b - DC Remover is bypassed.

13.8.4.9 MICFIL Output Control register (OUT_CTRL)

13.8.4.9.1 Offset

Register	Offset
OUT_CTRL	74h

13.8.4.9.2 Function

This MICFIL module configuration register provides configuration control bits for all sub-blocks and internal logic.

13.8.4.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	OUTGAIN7				OUTGAIN6				OUTGAIN5				OUTGAIN4			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OUTGAIN3				OUTGAIN2				OUTGAIN1				OUTGAIN0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.8.4.9.4 Fields

Field	Function
31-28 OUTGAIN7	Channel 7 Decimation Filter Output Gain This bitfield is the gain value in the output of the decimation filter as a positive or negative (two's complement) number of bits to shift.
27-24 OUTGAIN6	Channel 6 Decimation Filter Output Gain This bitfield is the gain value in the output of the decimation filter as a positive or negative (two's complement) number of bits to shift.
23-20 OUTGAIN5	Channel 5 Decimation Filter Output Gain This bitfield is the gain value in the output of the decimation filter as a positive or negative (two's complement) number of bits to shift.
19-16 OUTGAIN4	Channel 4 Decimation Filter Output Gain This bitfield is the gain value in the output of the decimation filter as a positive or negative (two's complement) number of bits to shift.
15-12 OUTGAIN3	Channel 3 Decimation Filter Output Gain This bitfield is the gain value in the output of the decimation filter as a positive or negative (two's complement) number of bits to shift.
11-8	Channel 2 Decimation Filter Output Gain

Table continues on the next page...

Field	Function
OUTGAIN2	This bitfield is the gain value in the output of the decimation filter as a positive or negative (two's complement) number of bits to shift.
7-4 OUTGAIN1	Channel 1 Decimation Filter Output Gain This bitfield is the gain value in the output of the decimation filter as a positive or negative (two's complement) number of bits to shift.
3-0 OUTGAIN0	Channel 0 Decimation Filter Output Gain This bitfield is the gain value in the output of the decimation filter as a positive or negative (two's complement) number of bits to shift.

13.8.4.10 MICFIL Output Status register (OUT_STAT)

13.8.4.10.1 Offset

Register	Offset
OUT_STAT	7Ch

13.8.4.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								OUTUNF7	OUTUNF6	OUTUNF5	OUTUNF4	OUTUNF3	OUTUNF2	OUTUNF1	OUTUNF0
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								OUTOVF7	OUTOVF6	OUTOVF5	OUTOVF4	OUTOVF3	OUTOVF2	OUTOVF1	OUTOVF0
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.8.4.10.3 Fields

Field	Function
31-24 —	Reserved
23 OUTUNF7	Channel 7 Output Underflow Flag The OUTUNF bit indicates an exceptional underflow condition in the output. 0b - No exception by output underflow. 1b - Exception by output underflow.
22 OUTUNF6	Channel 6 Output Underflow Flag The OUTUNF bit indicates an exceptional underflow condition in the output. 0b - No exception by output underflow. 1b - Exception by output underflow.
21 OUTUNF5	Channel 5 Output Underflow Flag The OUTUNF bit indicates an exceptional underflow condition in the output. 0b - No exception by output underflow. 1b - Exception by output underflow.
20 OUTUNF4	Channel 4 Output Underflow Flag The OUTUNF bit indicates an exceptional underflow condition in the output. 0b - No exception by output underflow. 1b - Exception by output underflow.
19 OUTUNF3	Channel 3 Output Underflow Flag The OUTUNF bit indicates an exceptional underflow condition in the output. 0b - No exception by output underflow. 1b - Exception by output underflow.
18 OUTUNF2	Channel 2 Output Underflow Flag The OUTUNF bit indicates an exceptional underflow condition in the output. 0b - No exception by output underflow. 1b - Exception by output underflow.
17 OUTUNF1	Channel 1 Output Underflow Flag The OUTUNF bit indicates an exceptional underflow condition in the output. 0b - No exception by output underflow. 1b - Exception by output underflow.
16 OUTUNF0	Channel 0 Output Underflow Flag The OUTUNF bit indicates an exceptional underflow condition in the output. 0b - No exception by output underflow. 1b - Exception by output underflow.
15-8 —	Reserved
7 OUTOVF7	Channel 7 Output Overflow Flag The OUTOVF bit indicates an exceptional overflow condition in the output. 0b - No exception by output overflow. 1b - Exception by output overflow.
6 OUTOVF6	Channel 6 Output Overflow Flag The OUTOVF bit indicates an exceptional overflow condition in the output. 0b - No exception by output overflow. 1b - Exception by output overflow.

Table continues on the next page...

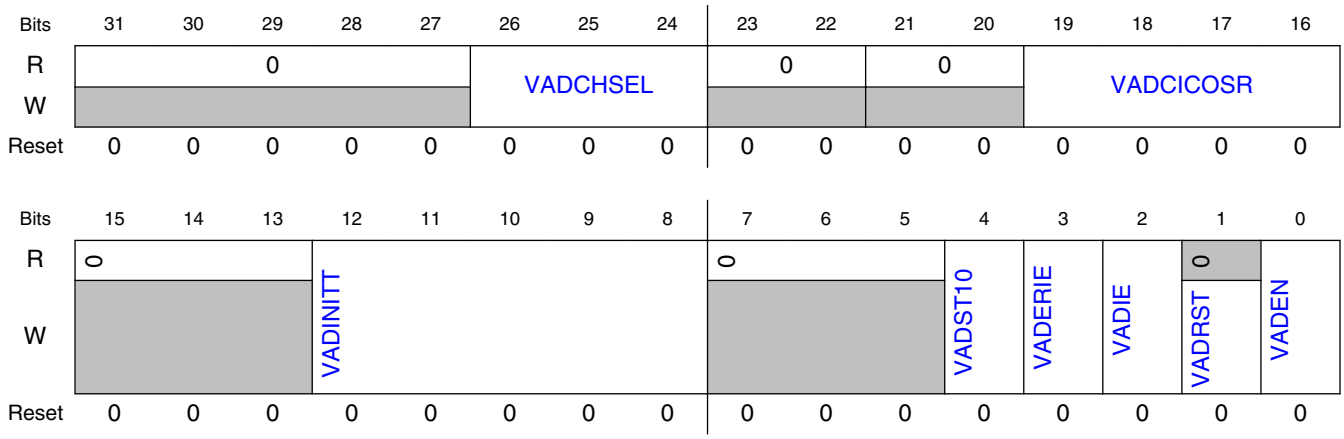
Field	Function
5 OUTOVF5	Channel 5 Output Overflow Flag The OUTOVF bit indicates an exceptional overflow condition in the output. 0b - No exception by output overflow. 1b - Exception by output overflow.
4 OUTOVF4	Channel 4 Output Overflow Flag The OUTOVF bit indicates an exceptional overflow condition in the output. 0b - No exception by output overflow. 1b - Exception by output overflow.
3 OUTOVF3	Channel 3 Output Overflow Flag The OUTOVF bit indicates an exceptional overflow condition in the output. 0b - No exception by output overflow. 1b - Exception by output overflow.
2 OUTOVF2	Channel 2 Output Overflow Flag The OUTOVF bit indicates an exceptional overflow condition in the output. 0b - No exception by output overflow. 1b - Exception by output overflow.
1 OUTOVF1	Channel 1 Output Overflow Flag The OUTOVF bit indicates an exceptional overflow condition in the output. 0b - No exception by output overflow. 1b - Exception by output overflow.
0 OUTOVF0	Channel 0 Output Overflow Flag The OUTOVF bit indicates an exceptional overflow condition in the output. 0b - No exception by output overflow. 1b - Exception by output overflow.

13.8.4.11 Voice Activity Detector 0 Control register (VAD0_CTRL_1)

13.8.4.11.1 Offset

Register	Offset
VAD0_CTRL_1	90h

13.8.4.11.2 Diagram



13.8.4.11.3 Fields

Field	Function
31-27 —	Reserved
26-24 VADCHSEL	Voice Activity Detector Channel Selector Selects the number of channel which the hardware voice activity detector will work. **NOTE:** This bitfield mustn't change when BSY_FIL is asserted.
23-22 —	Reserved
21-20 —	Reserved
19-16 VADCICOSR	Voice Activity Detector CIC Oversampling Rate This bitfield defines the oversampling rate of the CIC filter. The oversampling rate is (16-CICOSR). **NOTE:** This bitfield mustn't change when BSY_FIL is asserted.
15-13 —	Reserved
12-8 VADINITT	Voice Activity Detector Initialization Time Selects the number of frames to be used to initialize the voice detection. During this period the output of the voice activity detector is forced to inactive. For more information see Initialization Time. **NOTE:** The initialization time will be the time taken by [VADINITT+1] frames. **NOTE:** This bitfield mustn't change when BSY_FIL is asserted.
7-5 —	Reserved
4 VADST10	Voice Activity Detector Internal Filters Initialization Initializes the signal and noise filter with pre-filter value.

Table continues on the next page...

Field	Function
	NOTE: If the use of this bit is required, it should be stay pulsed by more than 2 cycles of system clock. **NOTE:** There is not voice activity detection when VADST10 is asserted. 0b - Normal operation. 1b - Filters are initialized.
3 VADERIE	Voice Activity Detector Error Interruption Enable The VADERIE bit enables Error Interrupts in the PDM Interface. This error could be caused by an overflow in the input of the HWVAD and it is signalized by the VADINSATF flag. 0b - HWVAD Error Interrupts disabled 1b - HWVAD Error Interrupts enabled
2 VADIE	Voice Activity Detector Interruption Enable This bit enables interrupts in the PDM Interface when voice activity event has been detected by the Hardware Voice Activity Detector (HWVAD). See the VADIF for more information. 0b - HWVAD Interrupts disabled 1b - HWVAD Interrupts enabled
1 VADRST	Voice Activity Detector Reset The VADRST is a self-negated bit which provides the CPU with the capability to initialize the VAD through the slave-bus interface. This bit always reads as zero. Refer to VAD Soft-reset.
0 VADEN	Voice Activity Detector Enable Enables the use of the Hardware Voice Activity Detector. **NOTE:** This bitfield mustn't change when BSY_FIL is asserted. 0b - The HWVAD is disabled. 1b - The HWVAD is enabled.

13.8.4.12 Voice Activity Detector 0 Control register (VAD0_CTRL_2)

13.8.4.12.1 Offset

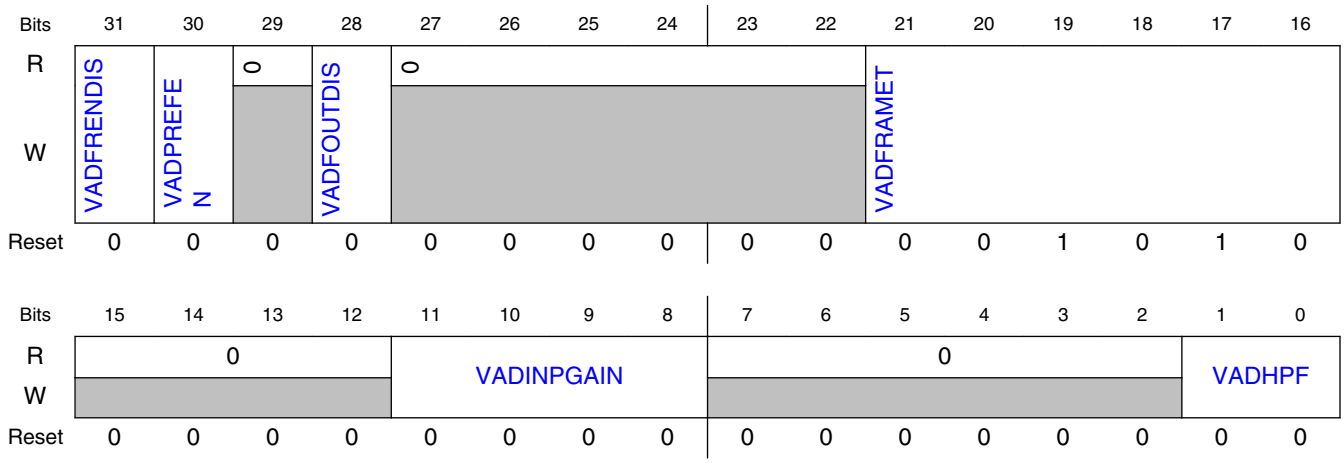
Register	Offset
VAD0_CTRL_2	94h

13.8.4.12.2 Function

NOTE

This register mustn't change when BSY_FIL is asserted.

13.8.4.12.3 Diagram



13.8.4.12.4 Fields

Field	Function
31 VADFRENDIS	Voice Activity Detector Frame Energy Disable Disables the calculus of energy in a frame. For more information see Noise Filter. 0b - Frame energy calculus enabled. 1b - Frame energy calculus disabled.
30 VADPREFEN	Voice Activity Detector Pre Filter Enable Enabled a pre-filter in the input of Noise and Signal filters. For more information see Pre-Filter. 0b - Pre-filter is bypassed. 1b - Pre-filter is enabled.
29 —	Reserved
28 VADFOUTDIS	Voice Activity Detector Force Output Disable Disables the output of the Energy/Envelope-based Detector. 0b - Output is enabled. 1b - Output is disabled.
27-22 —	Reserved
21-16 VADFRAMET	Voice Activity Detector Frame Time Selects the scale value to control the frame time duration as is described in Frame Time Setting. **NOTE:** VADFRAMET=0 is not allowed.
15-12 —	Reserved
11-8 VADINPGAIN	Voice Activity Detector Input Gain This bitfield is the gain value in the HWVAD input as a positive or negative (two's complement) number of bits to shift.
7-2	Reserved

Table continues on the next page...

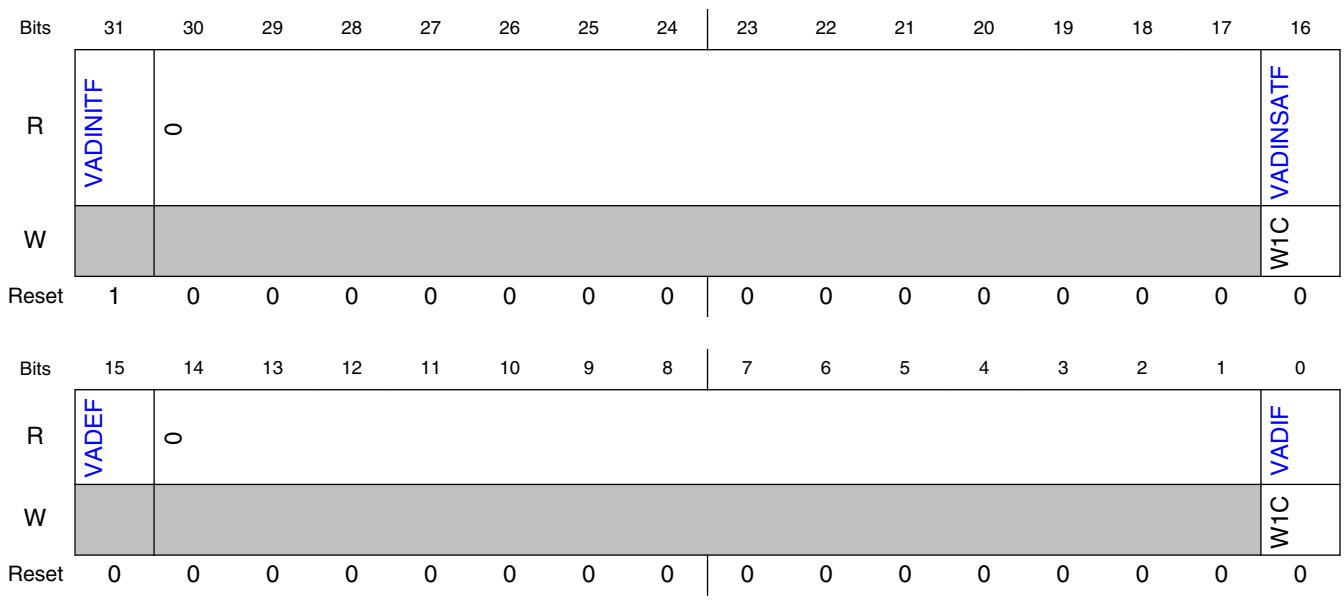
Field	Function
—	
1-0 VADHPF	Voice Activity Detector High-Pass Filter Configures the internal high-pass filter. 00b - Filter bypassed. 01b - Cut-off frequency at 1750Hz. 10b - Cut-off frequency at 215Hz. 11b - Cut-off frequency at 102Hz.

13.8.4.13 Voice Activity Detector 0 Status register (VAD0_STAT)

13.8.4.13.1 Offset

Register	Offset
VAD0_STAT	98h

13.8.4.13.2 Diagram



13.8.4.13.3 Fields

Field	Function
31 VADINITF	Voice Activity Detector Initialization Flag This flag signalizes when the HWVAD is being initialized according to the VADINITT bitfield value.

Table continues on the next page...

Field	Function
	0b - HWVAD is not being initialized. 1b - HWVAD is being initialized.
30-17 —	Reserved
16 VADINSATF	Voice Activity Detector Input Saturation Flag The VADINSATF bit indicates an exceptional saturation condition by an overflow or underflow in the HWVAD input. 0b - No exception by HWVAD input saturation. 1b - Exception by HWVAD input saturation.
15 VADEF	Voice Activity Detector Event Flag This bit indicates when voice activity event has been detected. 0b - Voice activity has not been detected by the HWVAD. 1b - Voice activity has been detected by the HWVAD.
14-1 —	Reserved
0 VADIF	Voice Activity Detector Interrupt Flag This bit indicates when voice activity has been detected by the HWVAD. Write a logic 1 to this field to clear this flag. 0b - Voice activity has not been detected by the HWVAD. 1b - Voice activity has been detected by the HWVAD.

13.8.4.14 Voice Activity Detector 0 Signal Configuration (VAD0_SCONFIG)

13.8.4.14.1 Offset

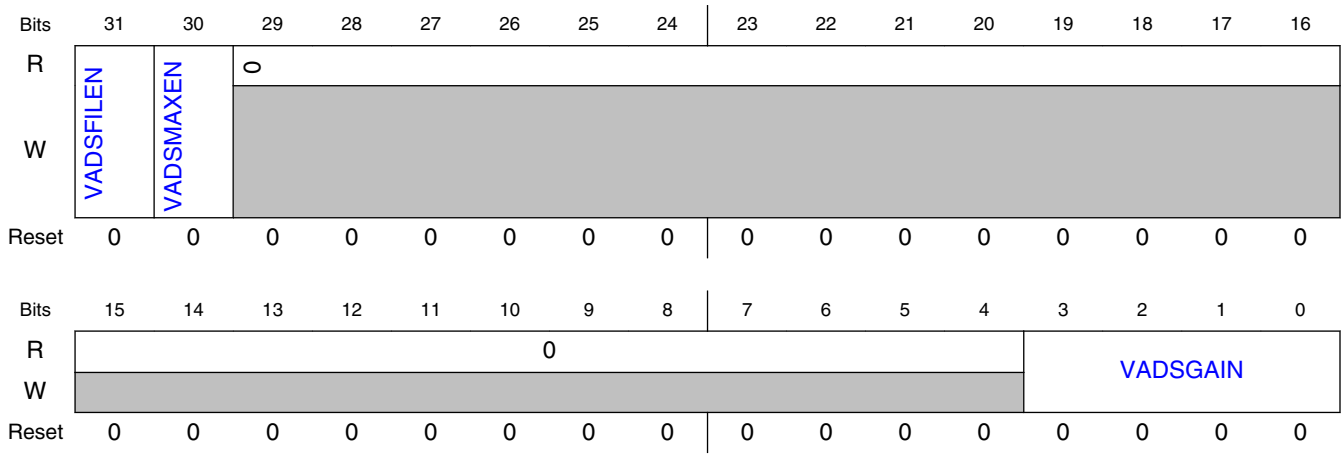
Register	Offset
VAD0_SCONFIG	9Ch

13.8.4.14.2 Function

NOTE

These bit-fields must be configured when the MICFIL is not running.

13.8.4.14.3 Diagram



13.8.4.14.4 Fields

Field	Function
31 VADSFILEN	Voice Activity Detector Signal Filter Enable Enables the signal filter. For more information see Signal Filter. 0b - Signal filter is disabled. 1b - Signal filter is enabled.
30 VADSMAXEN	Voice Activity Detector Signal Maximum Enable Selects whether or not a block to calculate the maximum value between the input and output of signal filter is used as signal estimation. For more information see Signal Filter. 0b - Maximum block is bypassed. 1b - Maximum block is enabled.
29-4 —	Reserved
3-0 VADSGAIN	Voice Activity Detector Signal Gain Gain value for the signal energy or envelope estimated. It is a unsigned value which is multiplied with the estimated signal data. **NOTE:** VADSGAIN=0 gain corresponds to 1.

13.8.4.15 Voice Activity Detector 0 Noise Configuration (VAD0_NCONFIG)

13.8.4.15.1 Offset

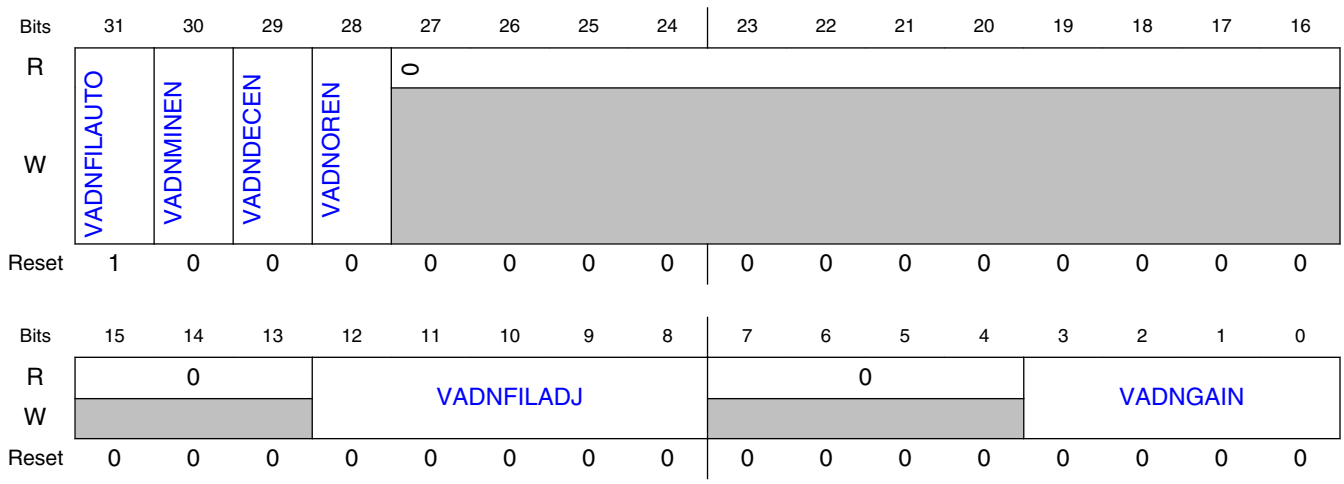
Register	Offset
VAD0_NCONFIG	A0h

13.8.4.15.2 Function

NOTE

These bit-fields must be configured when the MICFIL is not running.

13.8.4.15.3 Diagram



13.8.4.15.4 Fields

Field	Function
31 VADNFILAUTO	Voice Activity Detector Noise Filter Auto Selects whether or not the noise filter is activated automatically based in voice activity information. For more information see Noise Filter. 0b - Noise filter is always enabled. 1b - Noise filter is enabled/disabled based on voice activity information.
30 VADNMINEN	Voice Activity Detector Noise Minimum Enable Selects whether or not a block to calculate the minimum value between the input and output of noise filter is used as noise estimation. For more information see Noise Filter. 0b - Minimum block is bypassed. 1b - Minimum block is enabled.
29 VADNDECEN	Voice Activity Detector Noise Decimation Enable Selects whether or not the input of the noise filter is decimated. For more information see Noise Filter. 0b - Noise input is not decimated. 1b - Noise input is decimated.
28 VADNOREN	Voice Activity Detector Noise OR Enable Enables a OR logic in the output of minimum noise estimator block. For more information see Filter Result Gain Setting. 0b - Noise input is not decimated.

Table continues on the next page...

Field	Function
	1b - Noise input is decimated.
27-13 —	Reserved
12-8 VADNFILADJ	Voice Activity Detector Noise Filter Adjustment Selects the adjustment value of the noise filter. For more information see Noise Filter.
7-4 —	Reserved
3-0 VADNGAIN	Voice Activity Detector Noise Gain Gain value for the noise energy or envelope estimated. It is an unsigned value which is multiplied with the estimated noise data. **NOTE:** VADNGAIN=0 gain corresponds to 1.

13.8.4.16 Voice Activity Detector 0 Noise Data (VAD0_NDATA)

13.8.4.16.1 Offset

Register	Offset
VAD0_NDATA	A4h

13.8.4.16.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VADNDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.8.4.16.3 Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Field	Function
—	
15-0 VADNDATA	Voice Activity Detector Noise Data This bit-field is the noise energy or noise envelope calculated by the HWVAD. It can be used by software for a further voice activity detection. **NOTE:** This field value should be ignored when VADFOUTDIS=1 or VADST10=1.

13.8.4.17 Voice Activity Detector 0 Zero-Crossing Detector (VAD0_ZCD)

13.8.4.17.1 Offset

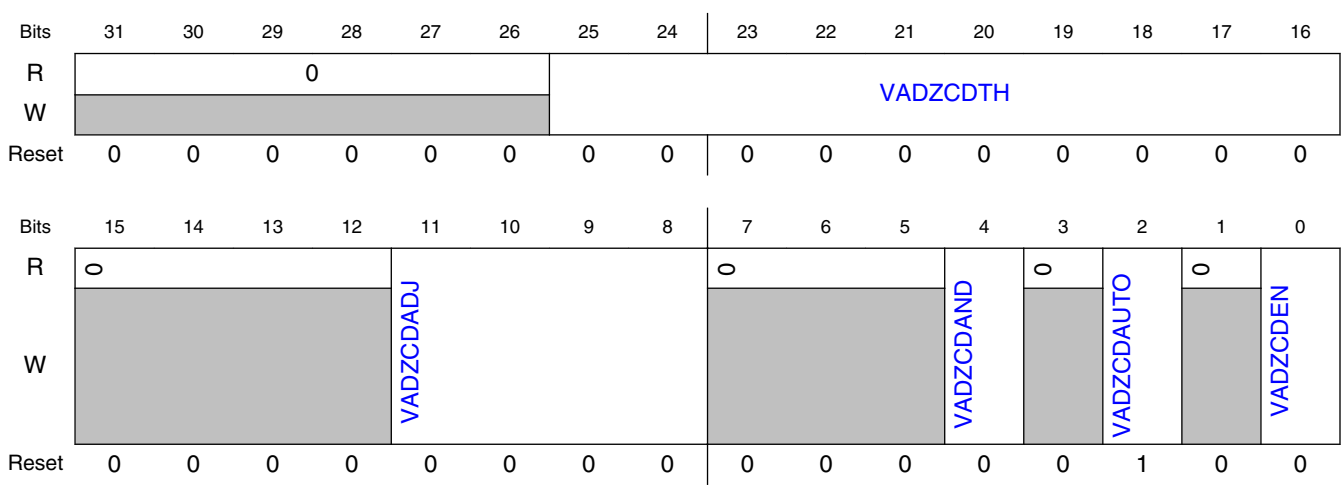
Register	Offset
VAD0_ZCD	A8h

13.8.4.17.2 Function

NOTE

These bit-fields must be configured when the MICFIL is not running.

13.8.4.17.3 Diagram



13.8.4.17.4 Fields

Field	Function
31-26 —	Reserved
25-16 VADZCDTH	Zero-Crossing Detector Threshold This bit-field is the initial value taken to estimate the ZCD threshold when it is estimated automatically (VASDZCDAUTO = 1), otherwise (VADZCDAUTO = 0) this bit-field is the static value of the ZCD threshold.
15-12 —	Reserved
11-8 VADZCDADJ	Zero-Crossing Detector Adjustment Selects the adjustment value of the threshold estimator. For more information see Zero-Crossing Detector.
7-5 —	Reserved
4 VADZCDAND	Zero-Crossing Detector AND Behavior Selects whether the Zero-Crossing detector results will be AND'ed or OR'ed with the energy-based detection to form the voice activity detection event. 0b - The ZCD result is OR'ed with the energy-based detection. 1b - The ZCD result is AND'ed with the energy-based detection.
3 —	Reserved
2 VADZCDAUTO	Zero-Crossing Detector Automatic Threshold Enables the automatic estimation of the ZCD threshold. 0b - The ZCD threshold is not estimated automatically, 1b - The ZCD threshold is estimated automatically.
1 —	Reserved
0 VADZCDEN	Zero-Crossing Detector Enable Enables the Zero-Crossing Detector (ZCD) in the Hardware Voice Activity Detector (HWVAD). 0b - The ZCD is disabled. 1b - The ZCD is enabled.

13.8.5 Functional Description

13.8.5.1 Overview

Figure 13-92 shows the block diagram of the PDM Microphone Interface (MICFIL) block. Basically this module is composed by a Decimation Filter and a FIFO by channel, an Input Interface for each pair of microphones, a shared Time Generator unit, a shared bus interface, a shared DMA interface and a shared interrupt interface.

The Decimation Filter implements a low-pass filter in the audio band (20Hz-22.5KHz @48KHz output sampling rate by default) with a configurable decimation rate. It is implemented using an arrange of a CIC, a Half Band, a FIR and a DC remover filter.

The Time Generator unit generates the PDM clock to the microphones. This clock is the same for all the PDM microphones and it is active for all the microphones, i.e. there is not possiblity to turn off the PDM clock for one microphone only.

Finally, the output of each Decimation Filter is stored in a FIFO buffer. Each FIFO is mapped in the DATACHn registers. It is possible to generate either an interrupt or a DMA request when, in each FIFO of all enabled channels, the number of data stored surpasses a configured watermark.

13.8.5.2 Time Generator

The Time Generator is the responsible to generate the PDM clock to the external PDM microphones.

The PDM Clock rate depends on the desired output rate (fout), the quality mode (defined by QSEL bitfield) and the CIC's OSR (defined by CICOSR bitfield) as is shown in Table 13-57.

To the MICFIL work properly, the CLKDIV bitfield must be configured depending on the desired Output rate, the internal MICFIL clock frequency and the OSR value (see CIC Filter) as shown in Equation 1 on page 4237. Once calculated the CLKDIV value, the generated PDM clock rate can be estimated as shown in Equation 2 on page 4237, where the K factor value depends on the quality mode (see Table 13-59).

$$CLKDIV = \frac{\text{MICFIL Clock rate}}{8(\text{OSR})(\text{Output Rate})}$$

Equation 1. CLKDIV bitfield value

$$\text{PDM clock rate} = \frac{\text{MICFIL Clock rate}}{2 \cdot \text{floor}(K \cdot CLKDIV)}$$

Equation 2. PDM clock rate

NOTE

The Time Generator is affected by soft-reset. The PDM Clock starts with a rising edge after a soft reset is issued.

13.8.5.2.1 Busy Flag

If a least one Decimation Filter channel is running and HWVAD is stopped (in Normal mode, Debug mode when DBGE is 1 or Low Power mode) and then all channels are stopped (by clearing PDMIEN bit, changing to Debug when DBGE=0 or changing to Disable/Low Leakage modes), the MICFIL waits for filter completion, then BSY_FIL is cleared.

If only HWVAD are running and then are stopped, BSY_FIL is cleared immediately.

This bit can also be used as a confirmation that all Decimation Filters are stopped in a transition to Debug (when DBGE=0) or Disable/Low Leakage modes.

13.8.5.3 Input Interface

The Input Interface block splits up the 1-bit bitstream incoming from the PDM microphone in two signals, one by channel, as is shown in Figure 13-92. The left microphone (L) will be directed to the lower channel and the right microphone (R) will be directed to the upper channel.

13.8.5.3.1 Input Interface Signals

The timing diagram of the input interface signals is shown in Figure 13-93. The data generated in the first half (Right microphone) of the PDM clock is directed to the Channel 1 and the data generated during the last half (Left microphone) is directed to the Channel 0.

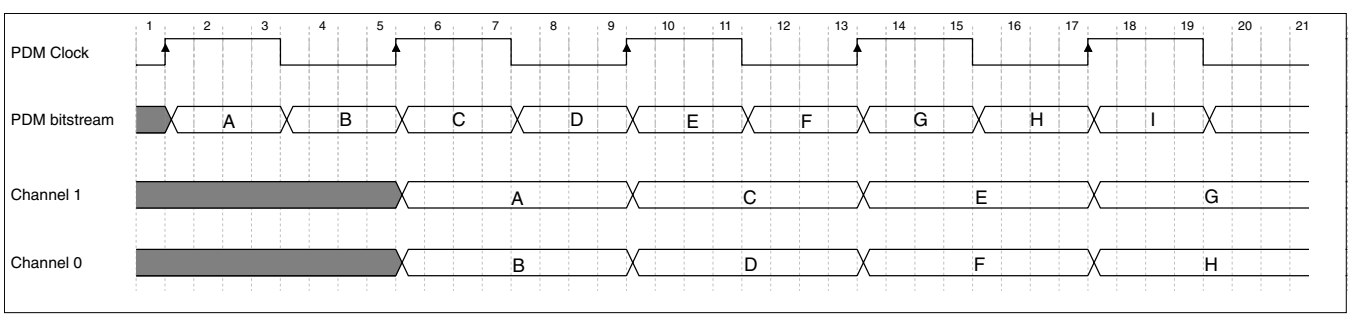


Figure 13-93. Input Interface signals

13.8.5.3.2 Input/Output Timing Requirements

The PDM microphones must meet the setup and hold timing requirements shown in Table 13-58 and Figure 13-94. The "k" factor value in Table 13-58 depends on the selected quality mode as shown in Table 13-59.

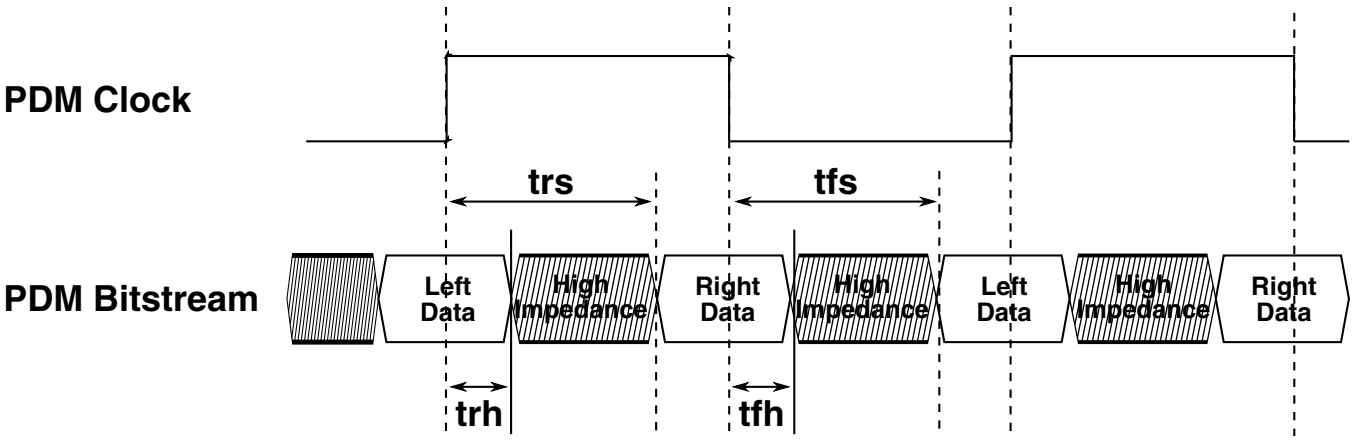


Figure 13-94. Input/Output TIming Requirements

Table 13-58. Timing parameters

Parameter	Value
trs, tfs	$<= \frac{\text{floor}(K \times CLKDIV) - 1}{\text{MICFIL Clock rate}}$ [1]
trh. tfh	>=0

1. Depending on K value, the user must make sure floor(K x CLKDIV) > 1 to avoid timing problems.

Table 13-59. K factor value

Quality mode	K factor
High Quality	1/2
Medium Quality, Very Low Quality 0	1
Low Quality, Very Low Quality 1	2
Very Low Quality 2	4

13.8.5.4 Decimation Filter

Figure 13-95 shows the block diagram of the Decimation Filter. This block is composed by a CIC filter, a DC remover, a FIR filter and a Half Band filter for each channel.

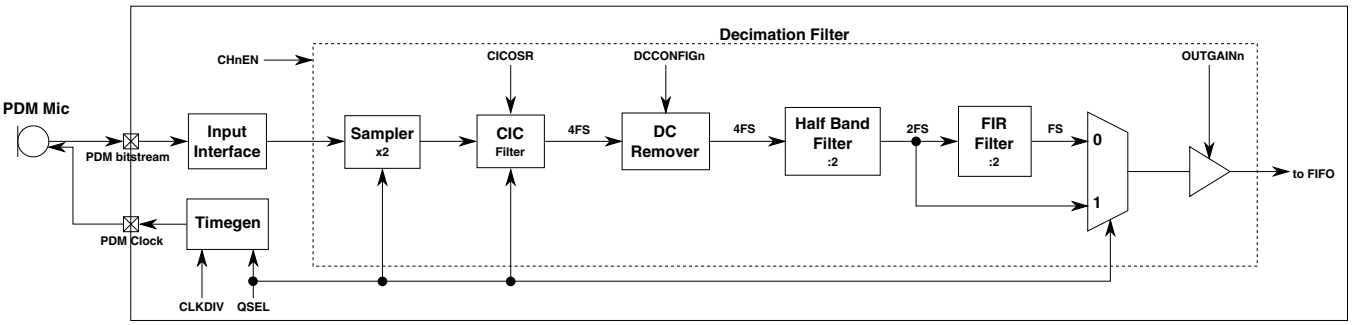


Figure 13-95. Decimation Filter Block Diagram

The Sampler block is activated depending on the quality mode selected (QSEL bitfield). This one samples twice the PDM data incoming from the Input Interface.

The CIC filter (Cascaded Integrator Comb filter) is a low pass filter commonly used to filter the sigma-delta modulator output signal. Its implementation is very simple without multipliers. It is composed by 5 cascadable unit sections of Integrator plus Differentiator units. Each section implements a low pass Sinc filter to improve the attenuation in the high frequencies range. It has a configurable decimation rate. This filter does not introduce phase distortions to the input signal, however the frequency amplitude response is not flat.

The DC remover is a high pass filter used to remove the DC component of the processed signal. It is a first order IIR filter of Chevichev type-I and it's implemented in the Direct Form-II. Its cut-off frequency is configurable.

The Half Band filter block implements a 19-tap low pass digital FIR filter with decimation by 2. It's used to compensate the high CIC drop in passband and helps the FIR filter block to flat the overall passband response.

The FIR filter block implements a 125-tap low pass digital FIR filter with decimation by 2 designed to achieve an overall filter passband ripple less than 0.2dB and a stopband attenuation more than 80dB. Such as the Half Band filter, it is also used to compensate the high CIC drop and to achieve a little passband ripple.

The Decimation Filter output depends on the quality mode selected by the QSEL bitfield, so it can come from the Half Band filter output directly or the FIR filter output.

Each Decimation Filter output can be adjusted by shifting right or left the filtered result. In case the adjusted output overflows or underflows the signed 16-bits, an error interrupt can be generated and the value is saturated to the maximum or minimum value respectively. Finally, each Decimation Filter result is sent to a FIFO.

13.8.5.4.1 CIC Filter

The CIC filter itself is an optimized implementation of the low pass Sinc filter with multiple stages. It is widely used because it requires just adders and subtractors to be implemented. The CIC filter is composed by sub-blocks called: integrator and comb. The architecture of the CIC filter is presented in Figure 13-96 and it consists of 5 integrator sections, and then followed by 5 comb sections downsampled.

The decimation rate of the CIC filters is defined by the CICOSR bitfield and by the quality selected (see Quality Modes). When the MICFIL is running in High and Very Low Quality 0 modes the decimation rate is 2OSR (where OSR = 16-CICOSR), otherwise the decimation rate is OSR only.

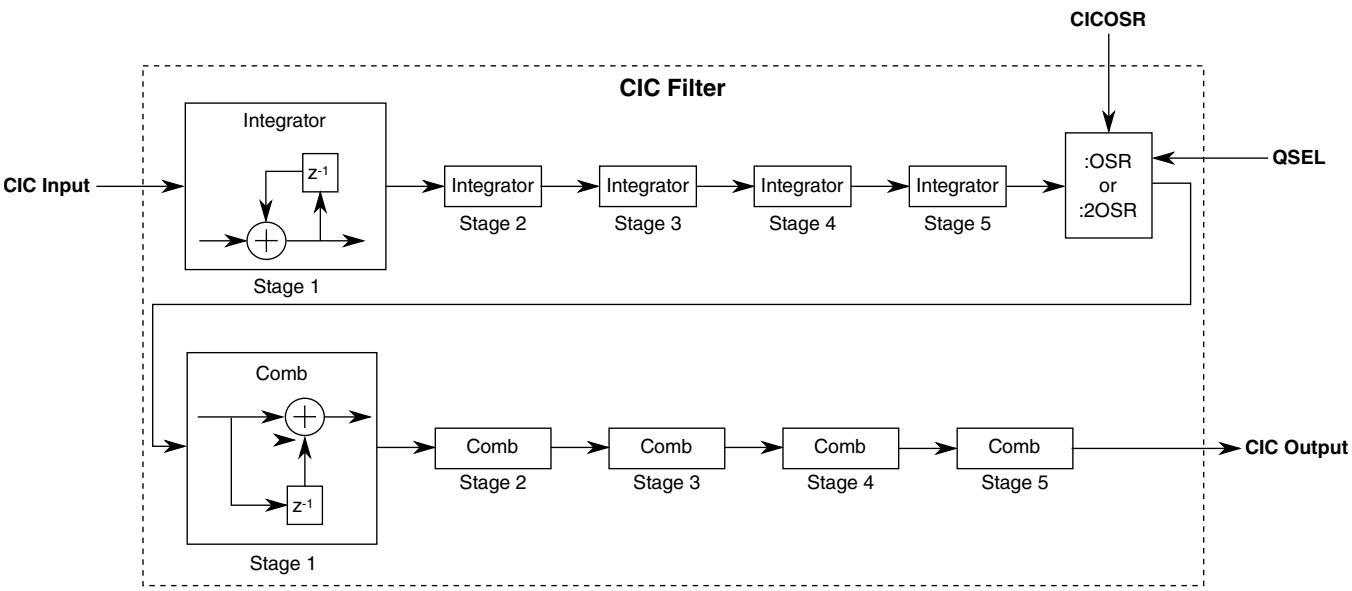


Figure 13-96. CIC Filter Block Diagram

13.8.5.4.2 DC Remover

The purpose of the DC Remover block is to remove the DC level component of the signal using a high-pass filter with a very low cut-off frequency selected by the DCCONFIGn bitfield. This filter can be bypassed when DCCONFIGn=11.

13.8.5.4.3 Half Band Filter

The use of half band filters is very attractive because it divides the base-band of the signal into two equal subbands and the half of the coefficients are zero-valued. The calculated filter has 19 coefficients (8 zero-valued) and its structure is shown in Figure 13-97.

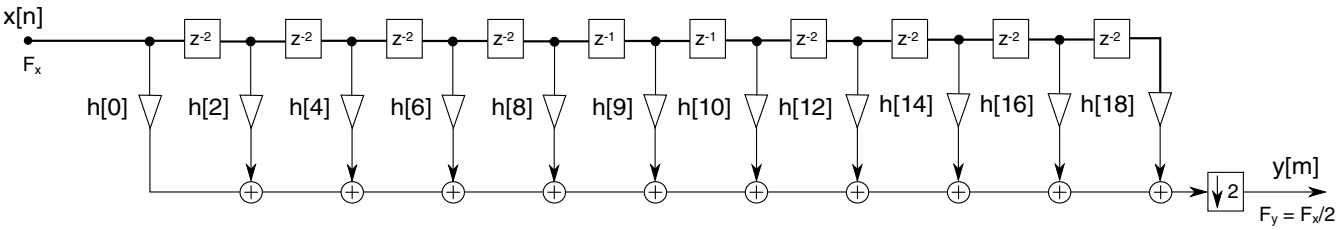


Figure 13-97. Half band filter structure

13.8.5.4.4 FIR Filter

The FIR block implements a fixed low-pass digital FIR filter. The calculated filter has 125 coefficients and its classical structure is shown in Figure 13-98.

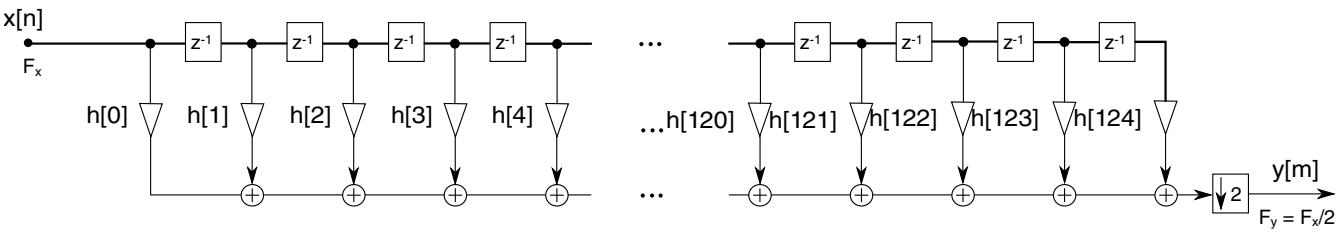


Figure 13-98. FIR filter structure

13.8.5.4.5 Decimated Output

The Decimation Filter's output depends on the quality mode selected by QSEL bitfield. It can be selected between the incoming signal from whether the Half Band filter or the FIR filter.

At the end, the selected output can be adjusted by a gain determined by the OUTGAINn bitfield for each channel (Figure 13-99), then the adjusted output is sent to its respective FIFO. The adjusted output is satured when it is greater than the maximum 16-bits signed positive value or lower than the 16-bits signed negative value. In both cases overflow (OUTOVFn) or underflow (OUTUNFn) flags are set respectively and an error interrupt is generated if ERREN bitfield value was previously enabled.

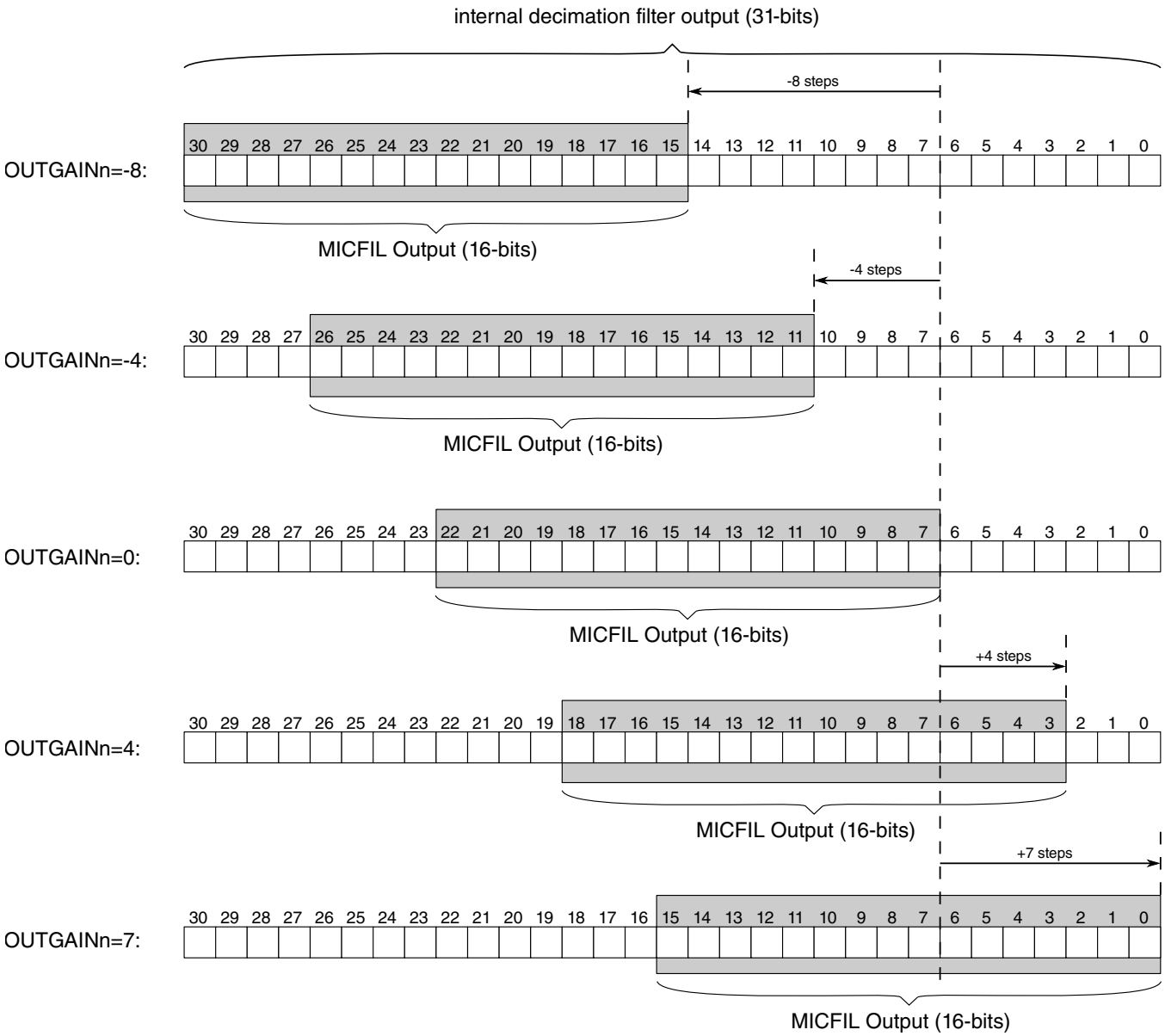


Figure 13-99. Output Gain

13.8.5.4.6 Minimum Required CLKDIV bitfield value

To work properly the CLKDIV bitfield value must have a minimum value according to the quality mode configured (see Quality Modes), the number of enabled channels (EC), OSR value (see CIC Filter) and K factor (see Input/Output Timing Requirements). When this condition is not met the LOWFREQF bit is asserted and an error interrupt can be issued (in case ERREN=1).

The equation to estimate the minimum required CLKDIV bitfield value in Medium, High and Low Quality modes is shown in Equation 3 on page 4243.

$$\text{floor}(K \times CLKDIV) >= K \times \frac{10 + 125EC}{8 \times OSR}$$

Equation 3. Minimum required CLKDIV value in Medium, High and Low Quality modes

The equation to estimate the minimum required CLKDIV bitfield value in Very Low Quality modes is shown in Equation 4 on page 4244.

$$\text{floor}(K \times CLKDIV) >= K \times \frac{10 + 19EC}{8 \times OSR}$$

Equation 4. Minimum required CLKDIV value in Very Low Quality modes

13.8.5.4.7 Overall Bandpass

The bandpass of the Decimation Filter depends on the quality mode and the output rate. For instance, the overall filter passband for different output data rates when the MICFIL is in High, Medium and Low Quality modes is shown in Table 13-60. The passband is about the half of the output rate because the FIR filter is enabled.

Table 13-60. Filter Passband in High, Medlum and Low Quality modes (DCCONFIGn=00)

Output Data Rate (KHz)	Overall Filter Passband
16	7Hz- 7.5KHz
24	10.5Hz - 11.25KHz
32	14Hz - 15KHz
48	21Hz - 22.5KHz

The overall filter passband for different output data rates when the MICFIL is in Very Low Quality modes is shown in Table 13-61.

Table 13-61. Filter Passband in Very Low Quality modes (DCCONFIGn=00)

Output Data Rate (KHz)	Overall Filter Passband
16	3.5Hz- 3.75KHz
24	5.25Hz - 5.63KHz
32	7Hz - 7.5KHz
48	10.5Hz - 11.25KHz
96	21Hz - 22.5KHz

13.8.5.5 Hardware Voice Activity Detector

The Hardware Voice Activity Detector (HWVAD) is a block responsible for detect voice activity in a channel selected by the user as shown in the Figure 13-100 and Figure 13-101. It can be configured in Envelope-based or Energy-based mode and it should be configured as described in Procedures.

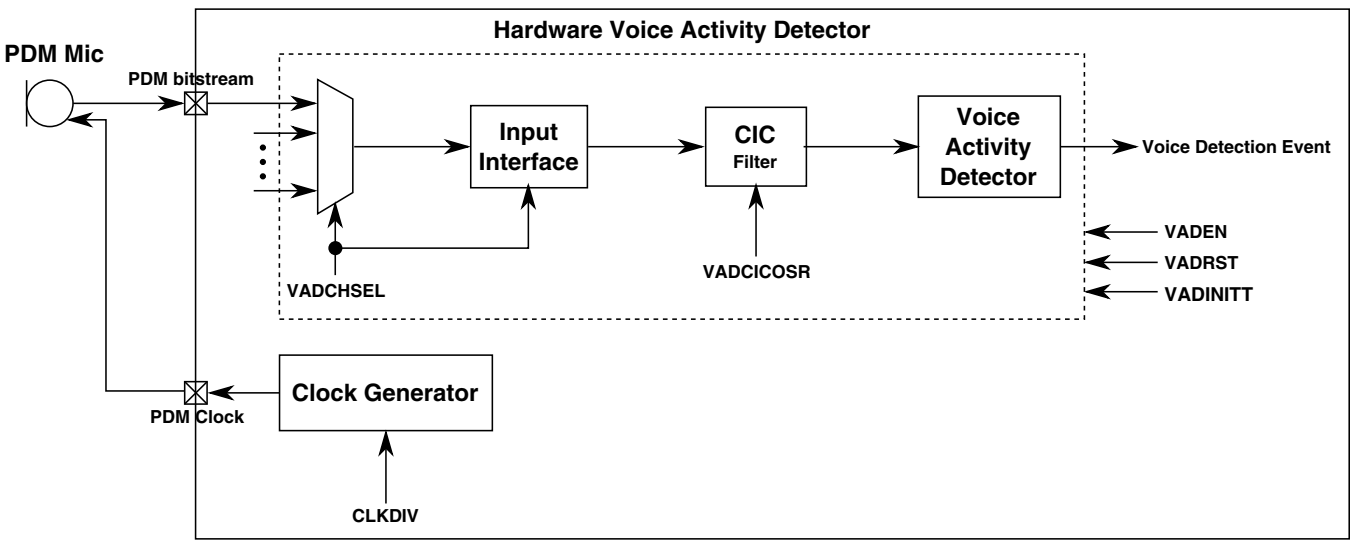


Figure 13-100. HWVAD Block Diagram

The HWVAD can divide the incoming PCM signal from its internal CIC in frames, the frame energy is used to estimate the background noise. Both frame energy and noise energy are examinated, a voice activity is detected when a specific delta between them is detected, then a interrupt will be issued. Optionally a Zero-Crossing Detection block (ZCD) could be enabled to avoid low energy voiced speech be missed, improving the voice detection performance.

The HWVAD detector can be enabled by the VADEN bit (VAD_CTRL_1 register) and could be reset by software through the VADRST bit (VAD_CTRL_1 register) . It is important to remark that the HWVAD detector could be enabled and/or reset only when the MICFIL isn't running i.e. when the BSY_FIL bit in STAT register is cleared.

To perform voice detection a Noise Filter and Signal Filter are used as shown Figure 13-101. In addition, a minimum block in the output of the Noise Filter and a maximum block in the ouput of the Signal Filter can be used. The result of both blocks can be scaled by gain blocks. Finally, the signal and noise estimated are compared, voice is detected when signal is greater than noise.

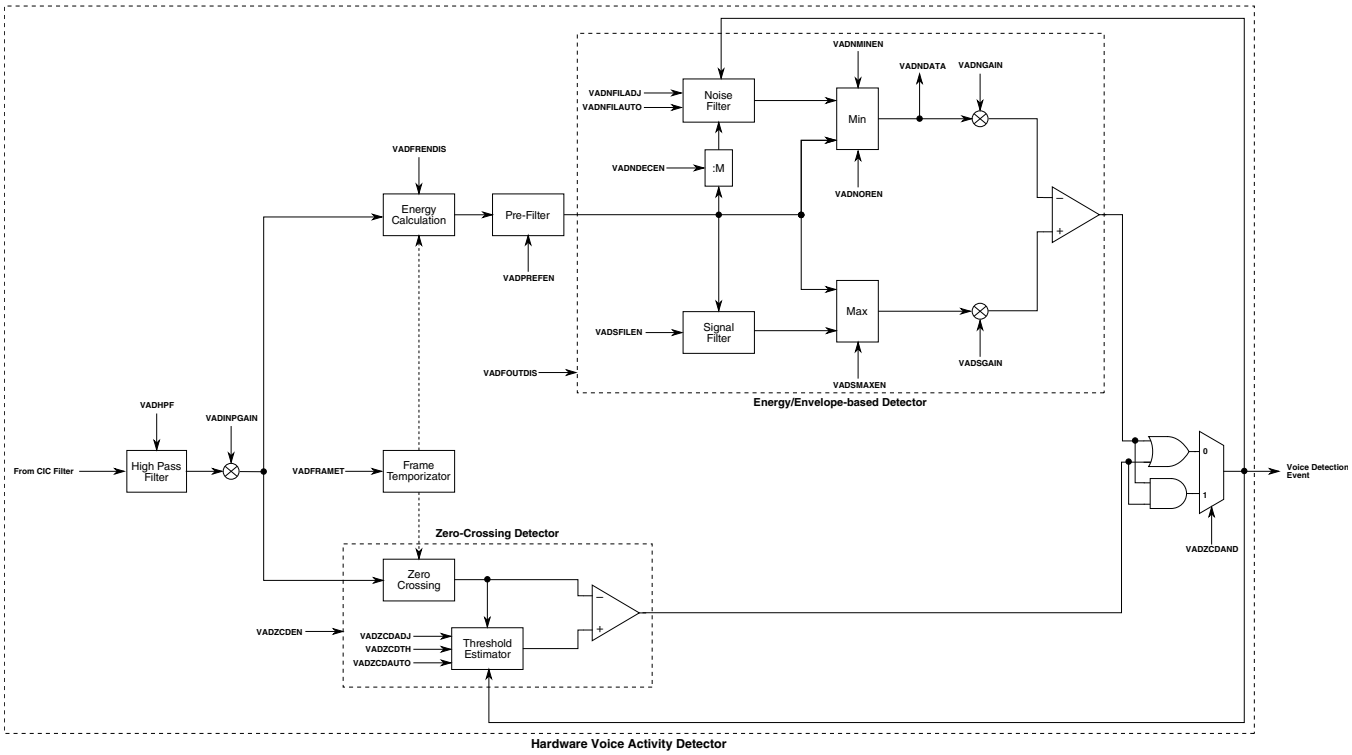


Figure 13-101. HWVAD Block Diagram (cont.)

13.8.5.5.1 High-Pass Filter

The purpose of the High-Pass Filter is to remove the low frequency components. The cut-off frequency is selected by the VADHPF bitfield. This filter is bypassed when VADHPF=00.

13.8.5.5.2 Input Setting

The HWVAD input has internally a user configurable High-Pass filter (see VADHPF bit description) and a user configurable gain (see VADINPGAIN). Both controls could be used to configure the module based on external noise conditions.

13.8.5.5.3 Frame Time Setting

In order to perform voice detection the signal frames incoming from the CIC filter can be divided in frames whose duration can be configured. The frame time will depend on the CIC output rate, the value of the VADFRAMET bitfield and a constant as is defined in Equation 5 on page 4246.

$$frame\ \ time = \frac{VADFRAMET \times 192}{CIC\ \ output\ \ rate}$$

Equation 5. Frame Time

For instance, in the typical case of the CIC filter output rate would be 192KHz, the frame time would be 10ms when VADFRAMET is 10.

13.8.5.5.4 Initialization Time

An initialization time can be configured by the VADINITT bitfield. During this initialization time no voice detection event is issued, Zero-Crossing and Energy/ Envelope-based detectors are kept disabled.

$$initialization\ \ time = (\text{VADINITT}+1) \times frame\ \ time$$

Equation 6. Initialization Time

13.8.5.5.5 Energy Calculation

In case VADFRENDIS = 0, the sum of the absolute values of all the samples of each frame are used to detect voice. Otherwise, if VADFRENDIS=1, the absolute value of the samples are used to detect voice i.e. the data incoming from CIC is not divided in frames.

13.8.5.5.6 Pre-Filter

The Pre-FIlter block is a low-pass filter which can be enabled when VADPREFEN=1.

13.8.5.5.7 Energy/Envelope-based Detector

To perform voice detection a Noise Filter and Signal Filter are used as shown in the HWVAD Block Diagram in the HWVAD Overview. In addition, a minimum block in the output of the Noise Filter and a maximum block in the output of the Signal Filter can be used. The result of both blocks can be scaled by gain blocks. Finally, the signal and noise estimated are compared, voice is detected when signal is greater than noise.

13.8.5.5.8 Noise Filter

The Noise Filter is a low-pass filter . This filter could be adjusted by the VADNFILADJ bitfield . In case VADNFILAUTO=1 the noise filter is enabled only when voice is not detected, when voice is detected the filter output does not change. In case VADNFILAUTO=0 the filter is always enabled.

A decimation stage in the input of the Noise Filter can be enabled by the VADNDECEN bitfield.

13.8.5.5.9 Minimum Noise

In order to estimate the noise, a block which calculates the minimum value between the input and output of the noise filter can be used. This block can be enabled by VADNMINEN, otherwise the output of the noise filter is passed directly.

The estimated noise of the output of this block is stored in the VADNDATA register for post-processing via software.

13.8.5.5.10 Signal Filter

The Signal Filter is a low-pass filter . This filter will be enabled when VADSFILEN=1, otherwise the filter will be bypassed.

13.8.5.5.11 Maximum Signal

In order to estimate the signal, a block which calculates the maximum value between the input and output of the signal filter can be used. This block can be enabled by VADSMAXEN, otherwise the output of the signal filter is passed directly.

13.8.5.5.12 Filter Result Gain Setting

There are two gain adjustment controls, one for the estimated noise (VADNGAIN) and another for signal estimated (VADSGAIN). Both gain adjustments outputs are compared, the comparison result determines whether or not there is voice in the channel.

13.8.5.5.13 Zero-Crossing Detector

Optionally, the HWVAD performance can be increased by the use of a Zero-Crossing Detection (ZCD) circuit. It is enabled by a VADZCDEN bit.

The threshold of the ZCD could be static or calculated dynamically depending on the VADZCDAUTO bit. When it is dynamically determined (VADZCDAUTO = 1) the threshold is calculated automatically taking the VADZCDTH bit-field as initial value, otherwise (VADZCDAUTO=0) the threshold is static in the VADZCDTH user configurable value.

The ZCD result can be AND'ed or OR'ed with the energy-based or envelope-based detection, it is selected by the VADZCDAND bit.The filter coefficient is equal to (VADZCDADJ+1)*2^(-4).

13.8.5.5.14 Interrupt

The HWVAD issues an interrupt when voice activity is detected and the VADIE bit in the VAD_CTRL_1 register is enabled. The voice activity event also will set the VADIF flag in the VAD_STAT register, this bit will be cleared when it is written 1.

13.8.5.5.15 Extended Operation

The VADIF bit could be used for a HWVAD extended operation i.e. using the VADNDATA bitfield value to determine a more accurate voice detection event by software. The procedure to follow will be:

1. An interrupt is issued by the HWVAD, the VADIF is set.
2. Leave the VADIF asserted while sofware is processing the VADNDATA bitfield content.
3. When software finishes to process the VADNDATA and detects voice activity:
   - If Envelope-based Detection is used, pulse VADST10 by more than 2-cycles and clear VADIF flag.
   - If Energy-based Detection is used, clear VADIF flag.

13.8.5.5.16 VAD Soft-reset

The VAD soft-reset is requested by writing 1 to the VADRST bit. It resets the following blocks and/or registers:

- VADIF, VADINSATF bitfields.
- All internal VAD buffers.

13.8.5.6 Bus Interface

The Bus Interface manages the Direct Memory Access (DMA) transactions, the interrupt requests and the user register interface with the SoC.

13.8.5.6.1 FIFO

The FIFO is a First-In, First-Out buffer that saves the overall filtering results. Each FIFO is 8 entries length. There is one FIFO for each channel. A push writes data into the FIFO, and a read pops data from FIFO. The push and pop operations from FIFO are in different clock domains, the push operation is made using the MICFIL clock while the pop operation uses the CPU clock . The FIFO has a configurable watermark FIFOWMK in the FIFO_CTRL register. When the amount of data in all the FIFOs of the enabled channels is greater than the watermark value (FIFO_CTRL[FIFOWMK]) a DMA/IRQ is requested depending on the DISEL field in the CTRL_1 register. After the filters are enabled by PDMIEN, the initial FIR results are discarded and not written into the FIFOs.

About 68 initial FIR results are not stored on FIFOs (FIR_RDY=0). After the STAT[FIR_RDY] register is set the data generated by the filter is valid and starts to push the results in FIFO.

A new filtered result is discarded (not stored) when the FIFO is full and overflow flag FIFOOVFn rises. Similarly, when trying a pop and FIFO is empty no valid data is read and underflow flag rises. In both cases, when overflow or underflow occurs they are flagged in the FIFO_STAT register to the respective channel, and an error interrupt is generated if the ERREN in the CTRL_1 register is asserted.

When the FIFO is reset by asserting the SRES soft-reset bit, zero values fills all FIFO positions, as well as push and pop pointers are set to initial position.

13.8.5.6.2 DMA

If DISEL=01 the FIFO stored samples will be read through DMA transactions. Then if there is at least one enabled channel and all FIFOs of all enabled channels reach their watermark the output interface makes a request for DMA transaction. When the PDMIEN is set and the filters start to operate, there is a period to wait the filter results to stabilize and no filter results are stored on the FIFOs. Therefore the first DMA request takes longer to occur than the following ones.

The CHnF flags in the STAT register are cleared when a DMA transaction is finished.

When the MICFIL is in Low Power mode and the watermark level is surpassed an asynchronous DMA request is delivered to the system, then the system wakes up the CPU clock, returning to Normal mode, to read FIFOs data.

13.8.5.6.3 Interrupt Requests

If DISEL=10 then an interrupt request is used to indicate that filtering results are stored in the FIFOs to be read by CPU. Also in this case the data can be read in the DATACHn registers and the flag CHnF of the STAT register are asserted for the enabled channels when the CHn FIFO surpasses the watermark level.

When the module is in Low Power mode the CPU clock is disabled, but the MICFIL clock remains enabled. In this case, an asynchronous interrupt is delivered to the system, it wakes up the CPU clock, returning to Normal mode. The system could read the FIFOs data and after should clear the CHnF flags to get a new interrupt request.

An error interrupt request can be generated also when an exceptional condition happens, such as a overflow or underflow in any channel output, an overflow or underflow of any FIFO, or the MICFIL clock rate is not the minimum value required to the MICFIL working (see Minimum Required CLKDIV bitfield value). This one is enabled by the

ERREN bit and the type of error is signalized by its respective status flag: OUTOVFn and OUTUNFn for channel outputs errors, FIFOOVF and FIFOUND flags for FIFO errors, and LOWFREQF for MICFIL clock rate.

13.8.5.7 Low Power Mode

In Low Power mode the MICFIL is able to filter the PDM microphone data and perform voice detection such as in Normal mode and can generate a wake up event through asynchronous DMA/IRQ requests depending on CHENn, VADEN and PDMIEN values.

In case the CPU is on a STOP/VLPS entry , PDMIEN is asserted and either CHENn or VADEN are asserted, then the CPU clock is gated.

In case the CPU is on a VLLS/LLS entry , VADEN is asserted and PDMIEN is asserted, then the CPU clock is gated.

In case the CPU is on a VLLS/LLS entry and either PDMIEN or VADEN are cleared, then the mode will be acknowledged when the last sample is processed.

The DMA and IRQ requests can be generated asynchronously to wake up the CPU. Then the system wakes up the CPU clock, and get back the MICFIL to Normal mode to read FIFOs data if required.

NOTE

The asynchronous DMA or interrupt requests are generated only in this mode.

13.8.5.8 Debug Mode

In case DBGE=0, if a debug request is received or DBG is asserted while the MICFIL is running with at least a channel enabled, the PDM generated clock and the MICFIL will be stopped after finish to process remaining data. Otherwise, in case DBGE=1, the module will continue processing, it will not stop.

The PDM Interface cannot enter Debug mode once in Disable/Low Leakage or Low Power mode.

The Debug mode request affects all channels in the MICFIL. Access to output buffers remain operational, as well as their related flags.

NOTE

Once a debug request is issued, it can not be aborted.

13.8.5.9 Disable/Low Leakage mode

When Disable/Low Leakage (DLL) mode is requested while the MICFIL is running with at least a channel enabled, the PDM generated clock and the MICFIL will be stopped after finish to process remaining data, then CPU clock and MICFIL clock are gated. In case this mode is requested via SoC Doze when DOZEN=1 the MICFIL clock is gated but the CPU clock remains running.

After the MICFIL clock is stopped, the BSY_FIL bit is self-cleared and the mode is acknowledged .

Writes only to CTRL_1 and CTRL_2 registers are allowed with the exception of writes to the DBG and SRES bits, which are ignored, but read is still available for these bits.

NOTE

It can take up to 1020 MICFIL clock cycles from the request to the stop of the processing.

13.8.5.10 Soft-reset

The soft-reset is used to clear all the data in the filters. It is requested by writing 1 to the SRES bit. It reset the following blocks and/or registers:

- All internal buffers in Decimation Filters and Voice Activity Detectors.
- All FIFOs.
- All CHnF flags and LOWFREQF flag on the STAT register, and the FIFO_STAT register.
- Time Generator.

13.8.6 Procedures

Following are some simple initialization steps to be done before using the MICFIL block.

13.8.6.1 Initialization Procedure without HWVAD

The sequence of steps for the block initialization without enabling voice activity detector, i.e. using decimation filters only, is as follows:

1. Program the bit-fields as desired for your application (CLKDIV, CHnEN, FIFOWMK, DISEL, ERREN etc.). For more details see Application Clock and CLKDIV calculation example.
2. Start the overall module asserting the PDMIEN bit.
3. The module is ready to receive data from the input ports of the enabled channels.

13.8.6.2 Initialization Procedure with HWVAD

In this section is described the initialization procedures to initialize MICFIL with the voice activity detector.

The voice activity detector can work in two modes: Energy-based mode or Envelope-based mode. Optionally, a Zero-Crossing Detector can be enabled to improve the voice detection.

13.8.6.2.1 HWVAD in Energy-based mode

The sequence of steps to initialize the voice activity detector in Energy-based mode is as follows:

1. Program the control bitfields as desired for your application (CLKDIV, CHnEN, FIFOWMK, DISEL, ERREN etc.). For more details see Application Clock and CLKDIV calculation example.
2. Configure general HWVAD parameters:
   a. CIC oversampling rate (VADCICOSR bitfield).
   b. Configure the voice source channel (VADCHSEL bitfield).
   c. Configure the internal initialization timer (VADINITT bitfield) if required. More details in Initialization Time.
   d. Assert VADEN bitfield.
3. Configure input, noise and signal gains according to the environmental noise conditions:
   a. Configure the input gain (VADINPGAIN bitfield).
   b. Configure the signal gain (VADSGAIN bitfield).
   c. Configure the noise gain (VADNGAIN bitfield).
4. Enable the high-pass filter if required (VADHPF bitfield).
5. Configure the VADNFILADJ bitfield according to environmental noise conditions (VADNFILADJ = 16 is recommended).
6. Put HWVAD in energy-based mode (default values):
   a. Keep the VADFRENDIS bitfield cleared.
   b. Keep the VADPREFEN bitfield cleared.
   c. Keep the VADSFILEN bitfield cleared.

d. Keep the VADSMAXEN bitfield cleared.
e. Keep the VADNFILAUTO bitfield asserted.
f. Keep the VADNMINEN bitfield cleared.
g. Keep the VADNDECEN bitfield cleared.
h. Keep the VADNOREN bitfield cleared.

7. Initialize the Zero-Crossing Detector (Zero-Crossing Detector Initialization) if required.
8. Enable interrupts if required (VADIE and VADERIE bitfields).
9. Start the overall module asserting the PDMIEN bit.
10. The module is ready to receive data from the input ports of the enabled channels and to trigger interrupts when voice is detected.

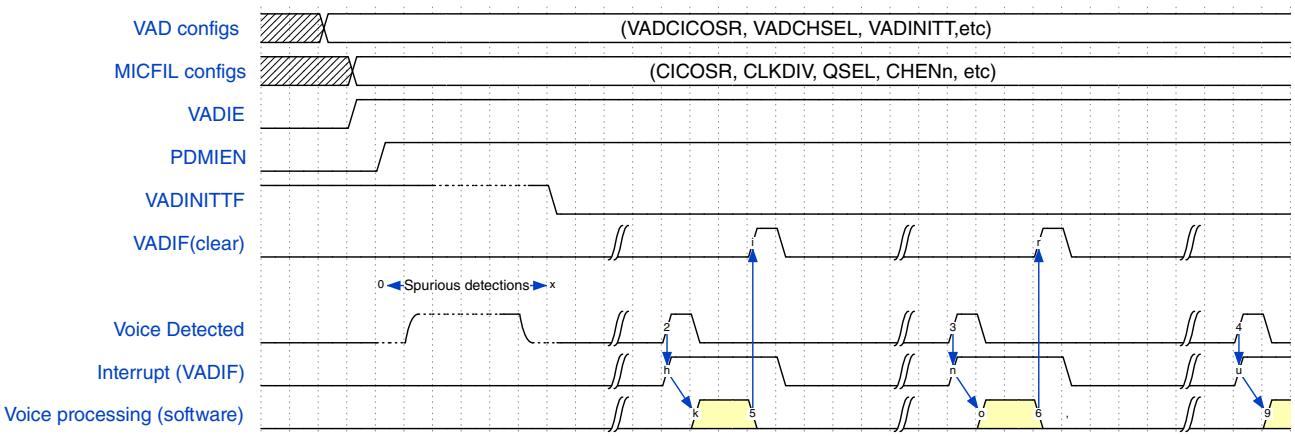


Figure 13-102. Energy-based initialization sequence

13.8.6.2.2 HWVAD in Envelope-based mode

The sequence of steps to initialize MICIL with the voice activity detector in Envelope-based mode is as follows:

1. Program the control bitfields as desired for your application (CLKDIV, CHnEN, FIFOWMK, DISEL, ERREN etc.). For more details see Application Clock and CLKDIV calculation example.
2. Configure general HWVAD parameters:
   a. CIC oversampling rate (VADCICOSR bitfield).
   b. Configure the voice source channel (VADCHSEL bitfield).
   c. Configure the internal initialization timer (VADINITT bitfield) if required. More details in Initialization Time.
   d. Assert VADEN bitfield.
3. Configure input, noise and signal gains according the environment noise conditions:
   a. Configure the input gain (VADINPGAIN bitfield).
   b. Configure the signal gain (VADSGAIN bitfield).
   c. Configure the noise gain (VADNGAIN bitfield).

4. Enable the high-pass filter if required (VADHPF bitfield).
5. Put HWVAD in envelope-based mode:
   a. Keep VADNFILADJ bitfield cleared.
   b. Assert the VADFRENDIS bitfield.
   c. Assert the VADPREFEN bitfield.
   d. Assert the VADSFILEN bitfield.
   e. Assert the VADSMAXEN bitfield.
   f. Clear the VADNFILAUTO bitfield.
   g. Assert the VADNMINEN bitfield.
   h. Assert the VADNDECEN bitfield.
   i. Assert the VADNOREN bitfield.
6. Initialize the Zero-Crossing Detector (Zero-Crossing Detector Initialization) if required.
7. Start the overall module asserting the PDMIEN bit.
8. Wait for VADINITF to be cleared.
9. Write VADST10 bitfield three times.
10. Enable interrupts if required (VADIE and VADERIE bitfields).
11. The module is ready to receive data from the input ports of the enabled channels and to trigger interrupts when voice is detected.

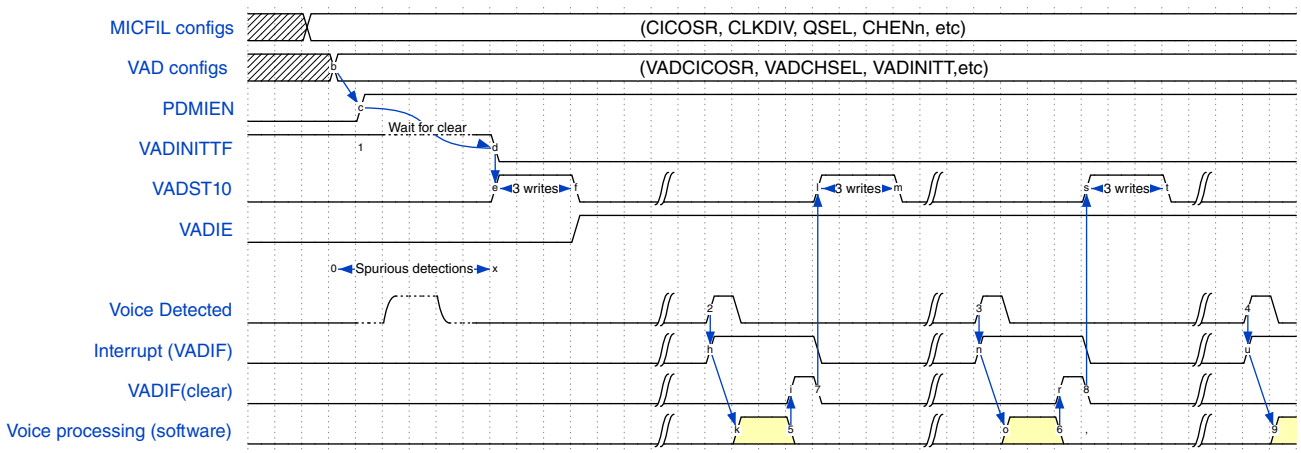


Figure 13-103. Envelope-based initialization sequence

13.8.6.2.3 Zero-Crossing Detector Initialization

The sequence of steps to initialize the Zero-Crossing Detector in Automatic mode is as follows:

1. Configure VADZCDADJ bitfield.
2. Configure an initial threshold in VADZCDTH according to the environmental noise conditions if required.
3. Configure VADZCDAND bitfield.

4. Keep VADZCDAUTO bitfield asserted.
5. Assert VADZCDEN.

The sequence of steps to initialize the Zero-Crossing Detector in Manual mode is as follows:

1. Configure a threshold in VADZCDTH according to the environmental noise conditions if required.
2. Configure VADZCDAND bitfield.
3. Clear VADZCDAUTO bitfield.
4. Assert VADZCDEN.

13.8.6.3 Reconfiguration Procedure

The sequence of steps for the block reconfiguration is as follows:

1. Deassert PDMIEN bit.
2. Wait to BSY_FIL to be cleared.
3. Run a soft-reset cycle.
4. Run a initilialization procedure as described in Initialization Procedure without HWVAD or Initialization Procedure with HWVAD.

13.8.6.4 Application Clock and CLKDIV calculation example

Assume that it is required to work at High Quality mode, CIC decimation by 16 (OSR = 16), 4 channels enabled (EC=4) and an output rate equal to 48KHz:

1. Check the PDM rate for High Quality mode in Table 13-57. PDM rate = 48KHz x 8 x 16 = 6.144MHz. (It is assumed that the PDM microphone part can work at this frequency; if not, quality mode need to be changed or CIC decimation need to be decreased.)
2. Check the K factor value for High Quality mode in Table 13-59. K factor is 1/2.
3. Calculate minimum CLKDIV value (see Equation 3 on page 4243): floor(K x CLKDIV) >= K x (10 + 125EC)/(8 x OSR) then floor(K x CLKDIV) >= 1.9922, Finally, CLKDIV >= 4.
4. Calculate minimum MICFIL application clock (see Equation 1 on page 4237). MICFIL clock rate = 8 x OSR x CLKDIV x (Output rate). From step 3: MICFIL clock rate >= 8 x16 x 4 x 48KHz. Finally, MICFIL clock rate >= 24.58MHz.

13.9 Synchronous Audio Interface (SAI)

13.9.1 Introduction

The I^2S (or I2S) module provides a synchronous audio interface (SAI) that supports full-duplex serial interfaces with frame synchronization such as I^2S, AC97, TDM, and codec/DSP interfaces.

13.9.1.1 Features

Note that some of the features are not supported across all SAI instances; see the chip-specific information in the first section of this chapter.

- Transmitter with independent bit clock and frame sync supporting 2 data lines
- Receiver with independent bit clock and frame sync supporting 2 data lines
- Each data line can support a maximum Frame size of 32 words
- Word size of between 8-bits and 32-bits
- Word size configured separately for first word and remaining words in frame
- Asynchronous 128 x 32-bit FIFO for each transmit and receive data line
- Supports graceful restart after FIFO error
- Supports automatic restart after FIFO error without software intervention
- Supports packing of 8-bit and 16-bit data into each 32-bit FIFO word
- Supports combining multiple data line FIFOs into single data line FIFO
- Independent 32-bit timestamp counters and bit counters for monitoring transmit and receive progress

13.9.1.2 Block diagram

The following block diagram also shows the module clocks.

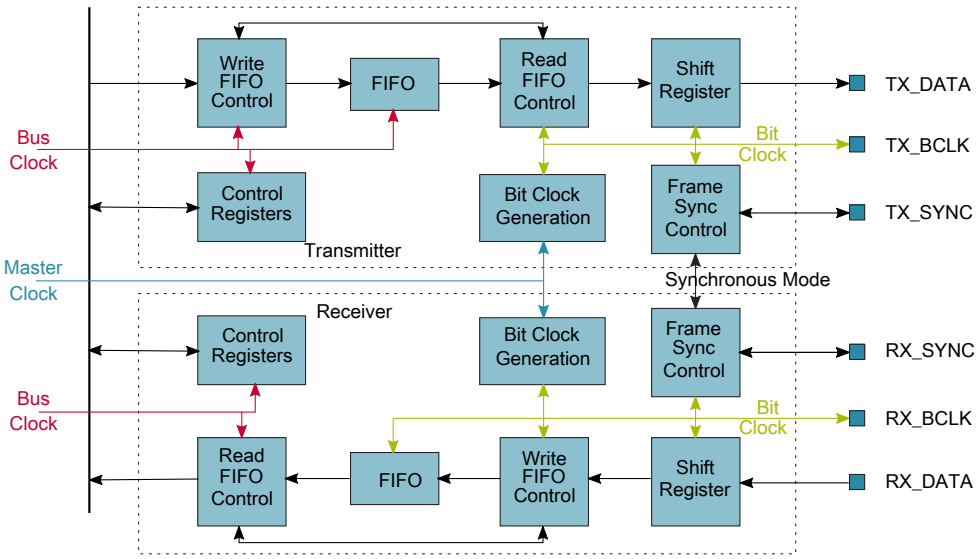


Figure 13-104. I²S/SAI block diagram

13.9.1.3 Modes of operation

Module power modes include Run mode, and Debug mode.

13.9.1.3.1 Run mode

In Run mode, the SAI transmitter and receiver operate normally.

13.9.1.3.2 Debug mode

In Debug mode, the SAI transmitter and/or receiver can continue operating provided the Debug Enable bit is set. When TCSR[DBGE] or RCSR[DBGE] bit is clear and Debug mode is entered, the SAI is disabled after completing the current transmit or receive frame. The transmitter and receiver bit clocks are not affected by Debug mode.

13.9.2 External signals

Name	Function	I/O
TX_BCLK	Transmit Bit Clock. The bit clock is an input when externally generated and an output when internally generated.	I/O

Table continues on the next page...

Name	Function	I/O
TX_SYNC	Transmit Frame Sync. The frame sync is an input sampled synchronously by the bit clock when externally generated and an output generated synchronously by the bit clock when internally generated.	I/O
TX_DATA[1:0]	Transmit Data. The transmit data is generated synchronously by the bit clock and is tristated whenever not transmitting a word.	O
RX_BCLK	Receive Bit Clock. The bit clock is an input when externally generated and an output when internally generated.	I/O
RX_SYNC	Receive Frame Sync. The frame sync is an input sampled synchronously by the bit clock when externally generated and an output generated synchronously by the bit clock when internally generated.	I/O
RX_DATA[1:0]	Receive Data. The receive data is sampled synchronously by the bit clock.	I
MCLK	Audio Master Clock. The audio master clock is an input when externally generated and an output when internally generated.	I/O

13.9.3 Functional description

This section provides a complete functional description of the block.

13.9.3.1 SAI clocking

The SAI clocks include:

- The audio master clock
- The bit clock
- The bus clock

13.9.3.1.1 Audio master clock

The audio master clock is used to generate the bit clock when the receiver or transmitter is configured for an internally generated bit clock. The transmitter and receiver can independently select between the bus clock and up to three audio master clocks to generate the bit clock.

The SAI peripheral controls the pin direction and post-divider of an audio master clock. The post-divide can be used to divide down the clock output on the MCLK signal pin, without affecting the audio master clock used by the transmitter and receiver.

The audio master clock generation and selection is chip-specific. Refer to chip-specific clocking information about how the audio master clocks are generated.

13.9.3.1.2 Bit clock

The SAI transmitter and receiver support asynchronous free-running bit clocks that can be generated internally from an audio master clock or supplied externally. There is also the option for synchronous bit clock and frame sync operation between the receiver and transmitter or between multiple SAI peripherals.

- If both transmitter and receiver are configured for asynchronous operation, then the transmitter and receiver will each use *their own* bit clock and frame sync.
- If the *transmitter* is configured for asynchronous mode and the receiver is configured for synchronous mode, then both transmitter and receiver will use the *transmitter* bit clock and frame sync.
- If the *receiver* is configured for asynchronous mode and the transmitter is configured for synchronous mode, then both transmitter and receiver will use the *receiver* bit clock and frame sync.

Note that the software configures synchronous or asynchronous mode, and that choice selects the bit clock/frame sync used.

Externally generated bit clocks must be:

- Enabled before the SAI transmitter or receiver is enabled
- Disabled after the SAI transmitter or receiver is disabled and completes its current frames

If the SAI transmitter or receiver is using an externally generated bit clock in asynchronous mode and that bit clock is generated by an SAI that is disabled in stop mode, then the transmitter or receiver should be disabled by software before entering stop mode. This issue does not apply when the transmitter or receiver is in a synchronous mode because all synchronous SAIs are enabled and disabled simultaneously.

13.9.3.1.3 Bus clock

The bus clock is used by the control and configuration registers and to generate synchronous interrupts and DMA requests.

NOTE

Although there is no specific minimum bus clock frequency specified, the bus clock frequency must be fast enough (relative

to the bit clock frequency) to ensure that the FIFOs can be serviced, without generating either a transmitter FIFO underrun or receiver FIFO overflow condition.

13.9.3.2 SAI resets

The SAI is asynchronously reset on system reset. The SAI has a software reset and a FIFO reset.

13.9.3.2.1 Software reset

The SAI transmitter includes a software reset that resets all transmitter internal logic, including the bit clock generation, status flags, and FIFO pointers. It does not reset the configuration registers. The software reset remains asserted until cleared by software.

The SAI receiver includes a software reset that resets all receiver internal logic, including the bit clock generation, status flags and FIFO pointers. It does not reset the configuration registers. The software reset remains asserted until cleared by software.

13.9.3.2.2 FIFO reset

The SAI transmitter includes a FIFO reset that synchronizes the FIFO write pointer to the same value as the FIFO read pointer. This empties the FIFO contents and is to be used after TCSR[FEF] is set, and before the FIFO is re-initialized and TCSR[FEF] is cleared. The FIFO reset is asserted for one cycle only.

The SAI transmitter can also reset the FIFO of individual data channels by setting the appropriate TCR3[CFR] bit. This should only be done when the corresponding TCR3[TCE] bit is clear.

The SAI receiver includes a FIFO reset that synchronizes the FIFO read pointer to the same value as the FIFO write pointer. This empties the FIFO contents and is to be used after the RCSR[FEF] is set and any remaining data has been read from the FIFO, and before the RCSR[FEF] is cleared. The FIFO reset is asserted for one cycle only.

The SAI receiver can also reset the FIFO of individual data channels by setting the appropriate RCR3[CFR] bit. This should only be done when the corresponding RCR3[RCE] bit is clear.

13.9.3.3 Synchronous modes

The SAI transmitter and receiver can operate synchronously to each other.

13.9.3.3.1 Synchronous mode

The SAI transmitter and receiver can be configured to operate with synchronous bit clock and frame sync.

If the transmitter bit clock and frame sync are to be used by both the transmitter and receiver:

- The transmitter must be configured for asynchronous operation and the receiver for synchronous operation.
- In synchronous mode, the receiver is enabled only when both the transmitter and receiver are enabled.
- It is recommended that the transmitter is the last enabled and the first disabled.

If the receiver bit clock and frame sync are to be used by both the transmitter and receiver:

- The receiver must be configured for asynchronous operation and the transmitter for synchronous operation.
- In synchronous mode, the transmitter is enabled only when both the receiver and transmitter are both enabled.
- It is recommended that the receiver is the last enabled and the first disabled.

When operating in synchronous mode, only the bit clock, frame sync, and transmitter/receiver enable are shared. The transmitter and receiver otherwise operate independently, although configuration registers must be configured consistently across both the transmitter and receiver.

13.9.3.4 Frame sync configuration

When enabled, the SAI continuously transmits and/or receives frames of data. Each frame consists of a fixed number of words and each word consists of a fixed number of bits. Within each frame, any given word can be masked causing the receiver to ignore that word and the transmitter to tri-state for the duration of that word.

The frame sync signal is used to indicate the start of each frame. A valid frame sync requires a rising edge (if active high) or falling edge (if active low) to be detected and the transmitter or receiver cannot be busy with a previous frame. A valid frame sync is also ignored (slave mode) or not generated (master mode) for the first four bit clock cycles after enabling the transmitter or receiver.

The transmitter and receiver frame sync can be configured independently with any of the following options:

- Externally generated or internally generated
- Active high or active low
- Assert with the first bit in frame or asserts one bit early
- Assert for a duration between 1 bit clock and the first word length
- Frame length from 1 to 32 words per frame
- Word length to support 8 to 32 bits per word
    - First word length and remaining word lengths can be configured separately
- Words can be configured to transmit/receive MSB first or LSB first

These configuration options cannot be changed after the SAI transmitter or receiver is enabled.

13.9.3.5 Data FIFO

Each transmit and receive channel includes a FIFO of size 128 x 32-bit. The FIFO data is accessed using the SAI Transmit/Receive Data Registers.

13.9.3.5.1 Data alignment

Data in the FIFO can be aligned anywhere within the 32-bit wide register through the use of the First Bit Shifted configuration field, which selects the bit index (between 31 and 0) of the first bit shifted.

Examples of supported data alignment and the required First Bit Shifted configuration are illustrated in Figure 13-105 for LSB First configurations and Figure 13-106 for MSB First configurations.

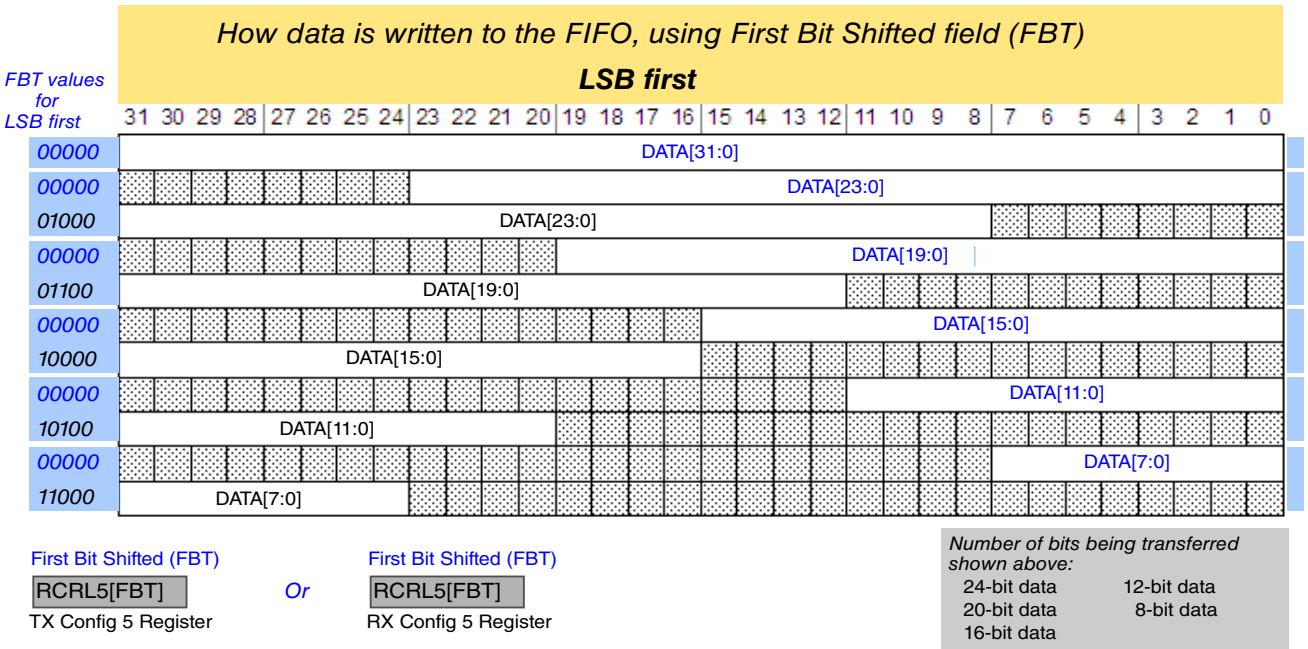


Figure 13-105. SAI first bit shifted, LSB first

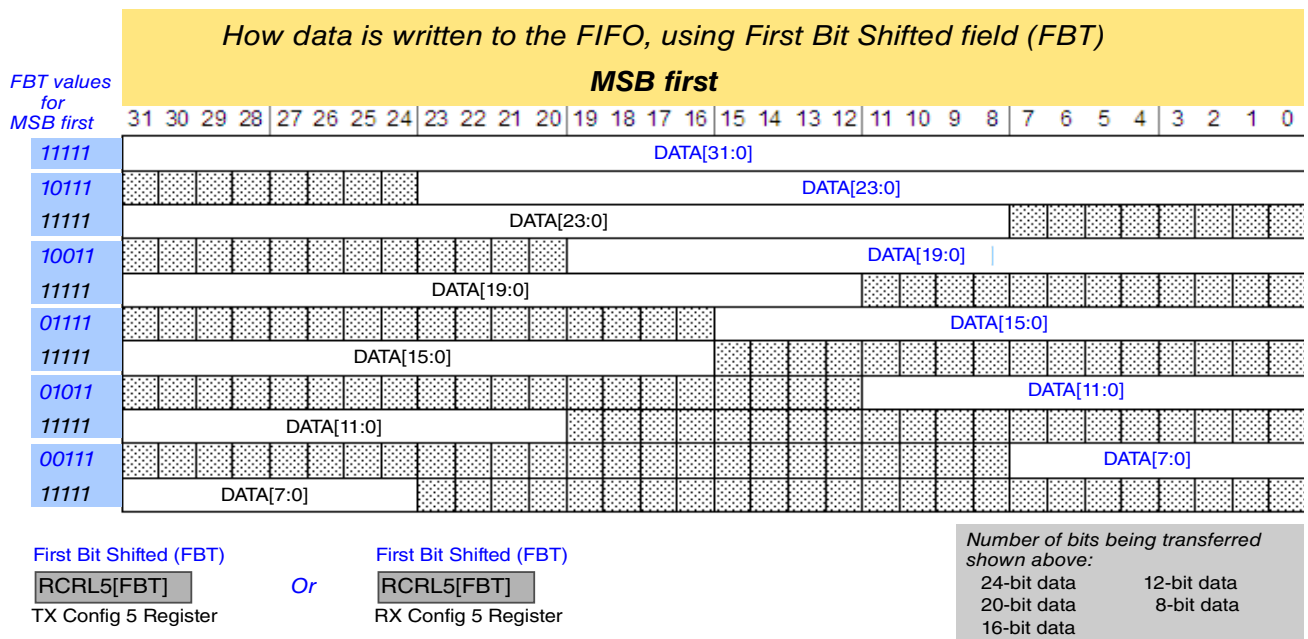


Figure 13-106. SAI first bit shifted, MSB first

13.9.3.5.2 FIFO pointers

When writing to a Transmit Data Register (TDR*n*), the Write FIFO Pointer (WFP) of the corresponding Transmit FIFO Register (TFR*n*) increments after each valid write. The SAI supports 8-bit, 16-bit and 32-bit writes to the Transmit Data Register and the FIFO pointer will increment after each individual write. Note that 8-bit writes should only be used when transmitting up to 8-bit data; 16-bit writes should only be used when transmitting up to 16-bit data.

- If the Transmit FIFO is full, then writes to a Transmit Data Register are ignored.
- If the Transmit FIFO is empty, then to avoid a FIFO underrun, the Transmit Data Register must be written at least 3 bit clocks before the start of the next unmasked word. Before enabling the transmitter, the Transmit FIFO should be initialized with data (since after the transmitter is enabled, the transmitter will start a new frame, and if no data is in the FIFO, then the transmitter will immediately give an error).

When reading a Receive Data Register (RDR*n*), the Read FIFO Pointer (RFP) of the corresponding Receive FIFO Register (RFR*n*) increments after each valid read. The SAI supports 8-bit, 16-bit and 32-bit reads from the RDR and the FIFO pointer will increment after each individual read. Note that 8-bit reads should only be used when receiving up to 8-bit data; 16-bit reads should only be used when receiving up to 16-bit data.

- If the Receive FIFO is empty, then reads from a Receive Data Register are ignored.
- If the Receive FIFO is full, then to avoid a FIFO overrun, the Receive Data Register must be read at least 3 bit clocks before the end of an unmasked word.

13.9.3.5.3 FIFO packing

FIFO packing supports storing multiple 8-bit or 16-bit data words in one 32-bit FIFO word for the transmitter and/or receiver. While this can be emulated by adjusting the number of bits per word and number of words per frame (for example, one 32-bit word per frame versus two 16-bit words per frame), FIFO packing does not require even multiples of words per frame and fully supports word masking. When FIFO packing is enabled, the FIFO pointers only increment when the full 32-bit FIFO word has been written (transmit) or read (receive) by software, supporting scenarios where different words within each frame are loaded/stored in different areas of memory.

When 16-bit FIFO packing is enabled for transmit, the transmit shift register is loaded at the start of each frame and after every second unmasked transmit word. The first word transmitted is taken from 16-bit word at byte offset $0 (first bit is selected by TCFG5[FBT] must be configured within this 16-bit word) and the second word transmitted is taken from the 16-bit word at byte offset $2 (first bit is selected by TCSR5[FBT][3:0]). The transmitter will transmit logic zero until the start of the next word once the 16-bit word has been transmitted.

When 16-bit FIFO packing is enabled for receive, the receive shift register is stored after every second unmasked received word, and at the end of each frame if there is an odd number of unmasked received words in each frame. The first word received is stored in the 16-bit word at byte offset $0 (first bit is selected by RCFG5[FBT] and must be configured within this 16-bit word) and the second word received is stored in the 16-bit word at byte offset $2 (first bit is selected by RCSR5[FBT][3:0]). The receiver will ignore received data until the start of the next word once the 16-bit word has been received.

The 8-bit FIFO packing is similar to 16-bit packing except four words are loaded or stored into each 32-bit FIFO word. The first word is loaded/stored in byte offset $0, second word in byte offset $1, third word in byte offset $2 and fourth word in byte offset $3. The TCFG5[FBT] and/or RCFG5[FBT] must be configured within byte offset $0.

13.9.3.5.4 FIFO Combine

FIFO combining mode allows the separate FIFOs for multiple data channels to be used as a single FIFO for either software accesses or a single data channel or both. Note that the enabled data channels must be contiguous and data channel 0 must be enabled when FIFO Combine mode is enabled.

Combining FIFOs for software access (writing transmit FIFO registers, reading receive FIFO registers) allows a DMA controller or software to read or write multiple FIFOs without incrementing the address that is accessed. Once enabled, the first software access to a FIFO register will access the first enabled channel FIFO, while the second access to a FIFO register will access the second enabled channel FIFO. This continues until software accesses the last enabled channel FIFO and the pointer resets back to the first enabled channel FIFO. To reset the pointer manually, software can reset the FIFOs or disable the FIFO combining on software accesses.

Combining FIFOs for transmit data channels allows one data channel to use the FIFOs of all enabled channel FIFOs, with identical data output on each enabled data channel. The transmit shift registers for all enabled data channels are loaded at the start of each frame and every N unmasked words (where N is the number of enabled data channels). The first word transmitted is loaded from the first enabled channel FIFO, while the second word transmitted is loaded from the second enabled channel FIFO, and so on until the end of the frame. Since the first word in each frame is always loaded from the first enabled data channel, it is recommended that the number of unmasked words in each frame is evenly divisible by the number of enabled data channels.

Combining FIFOs for receive data channels allows one data channel to use the FIFOs of all enabled channel FIFOs, with received data from channel 0 stored into each enabled data channel. The receive shift register for all enabled data channels are stored after every

N unmasked words (where N is the number of enabled data channels). The first word received is stored to the first enabled channel FIFO, while the second word received is stored to the second enabled channel FIFO, and so on until the end of the frame. Since the first word in each frame is always stored the first enabled data channel, it is recommended that the number of unmasked words in each frame is evenly divisible by the number of enabled data channels.

Note that combining FIFOs for data channels will load or store each channel FIFO at the same time. This means that FIFO error conditions are only checked every N words (where N is the number of enabled data channels) and that the FIFO warning and request flags will assert if any of the enabled data channel meets the warning flag or request flag conditions.

13.9.3.6 Word mask register

The SAI transmitter and receiver each contain a word mask register, namely TMR and RMR, that can be used to mask any word in the frame. Because the word mask register is double buffered, software can update it before the end of each frame to mask a particular word in the next frame.

The TMR causes the Transmit Data pin to be tri-stated for the length of each selected word and the transmit FIFO is not read for masked words.

The RMR causes the received data for each selected word to be discarded and not written to the receive FIFO.

13.9.3.7 Timestamp Counter

The transmitter and receiver implement separate timestamp counters and bit counters. The bit counter increments at the end of each bit in a frame whenever the transmitter or receiver is enabled. The bit counter continues to increment if a word is masked or a FIFO underrun or overrun condition occurs, but does not increment if an external frame sync is configured and no frame sync is detected. The bit counter can be reset by software, due to synchronization delays it can take up to 3 bit clock cycles for the bit clock counter to continue incrementing after a bit counter reset.

The timestamp counter increments on the bus interface clock whenever it is enabled, but can be configured to synchronize to the bit counter by waiting for the bit counter to increment first (if configured, this synchronization is performed whenever the time counter is reset). The current value of the timestamp counter is latched whenever the bit counter increments. Reading the bit counter register will cause the latched timestamp

value to be saved in the bit counter timestamp register. The timestamp counter can be reset by software, this also resets the latched timestamp value and the bit counter timestamp register.

The timestamp counter and bit counter can be used by software to track the progress of the transmitter and receiver. It can also be used to calculate the relative frequency of the bit clock against the bus interface clock.

13.9.3.8 Interrupts and DMA requests

The SAI transmitter and receiver generate separate interrupts and separate DMA requests, but support the same status flags.

13.9.3.8.1 FIFO request flag

The FIFO request flag is set based on the number of entries in the FIFO and the FIFO watermark configuration.

The transmit FIFO request flag is set when the number of entries in any of the enabled transmit FIFOs is less than or equal to the transmit FIFO watermark configuration and is cleared when the number of entries in each enabled transmit FIFO is greater than the transmit FIFO watermark configuration.

The receive FIFO request flag is set when the number of entries in any of the enabled receive FIFOs is greater than the receive FIFO watermark configuration and is cleared when the number of entries in each enabled receive FIFO is less than or equal to the receive FIFO watermark configuration.

The FIFO request flag can generate an interrupt or a DMA request.

13.9.3.8.2 FIFO warning flag

The FIFO warning flag is set based on the number of entries in the FIFO.

The transmit warning flag is set when the number of entries in any of the enabled transmit FIFOs is empty and is cleared when the number of entries in each enabled transmit FIFO is not empty.

The receive warning flag is set when the number of entries in any of the enabled receive FIFOs is full and is cleared when the number of entries in each enabled receive FIFO is not full.

The FIFO warning flag can generate an Interrupt or a DMA request.

13.9.3.8.3 FIFO error flag

The transmit FIFO error flag is set when the any of the enabled transmit FIFOs underflow. After it is set, all enabled transmit channels will transmit zero data before TCSR[FEF] is cleared.

When TCR4[FCONT] is set, the FIFO will continue transmitting data following an underflow without software intervention. To ensure that data is transmitted in the correct order, the transmitter will continue from the same word number in the frame that caused the FIFO to underflow, but only after new data has been written to the transmit FIFO. Software should still clear the TCSR[FEF] flag, but without reinitializing the transmit FIFOs.

RCSR[FEF] is set when the any of the enabled receive FIFOs overflow. After it is set, all enabled receive channels discard received data until RCSR[FEF] is cleared and the next next receive frame starts. All enabled receive FIFOs should be emptied before RCSR[FEF] is cleared.

When RCR4[FCONT] is set, the FIFO will continue receiving data following an overflow without software intervention. To ensure that data is received in the correct order, the receiver will continue from the same word number in the frame that caused the FIFO to overflow, but only after data has been read from the receive FIFO. Software should still clear the RCSR[FEF] flag, but without emptying the receive FIFOs.

The FIFO error flag can generate only an interrupt.

13.9.3.8.4 Sync error flag

The sync error flag, TCSR[SEF] or RCSR[SEF], is set when configured for an externally generated frame sync and the external frame sync asserts when the transmitter or receiver is busy with the previous frame. The external frame sync assertion is ignored and the sync error flag is set. When the sync error flag is set, the transmitter or receiver continues checking for frame sync assertion when idle or at the end of each frame.

The sync error flag can generate an interrupt only.

13.9.3.8.5 Word start flag

The word start flag is set at the start of the second bit clock for the selected word, as configured by the Word Flag register field.

The word start flag can generate an interrupt only.

13.9.4 Memory map and register definition

A read or write access to an address from offset 0x108 and above will result in a bus error.

13.9.4.1 I2S register descriptions

13.9.4.1.1 I2S memory map

SAI1 base address: 3001_0000h

SAI2 base address: 3002_0000h

SAI3 base address: 3003_0000h

SAI5 base address: 3005_0000h

SAI6 base address: 3006_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0301_0002h
4h	Parameter Register (PARAM)	32	RO	0005_0702h
8h	SAI Transmit Control Register (TCSR)	32	RW	0000_0000h
Ch	SAI Transmit Configuration 1 Register (TCR1)	32	RW	0000_0000h
10h	SAI Transmit Configuration 2 Register (TCR2)	32	RW	0000_0000h
14h	SAI Transmit Configuration 3 Register (TCR3)	32	RW	0000_0000h
18h	SAI Transmit Configuration 4 Register (TCR4)	32	RW	0000_0000h
1Ch	SAI Transmit Configuration 5 Register (TCR5)	32	RW	0000_0000h
20h - 24h	SAI Transmit Data Register (TDR0 - TDR1)	32	WORZ	0000_0000h
40h - 44h	SAI Transmit FIFO Register (TFR0 - TFR1)	32	RO	0000_0000h
60h	SAI Transmit Mask Register (TMR)	32	RW	0000_0000h
70h	SAI Transmit Timestamp Control Register (TTCR)	32	RW	0000_0000h
74h	SAI Transmit Timestamp Register (TTSR)	32	RO	0000_0000h
78h	SAI Transmit Bit Count Register (TBCR)	32	RO	0000_0000h
7Ch	SAI Transmit Bit Count Timestamp Register (TBCTR)	32	RO	0000_0000h
88h	SAI Receive Control Register (RCSR)	32	RW	0000_0000h
8Ch	SAI Receive Configuration 1 Register (RCR1)	32	RW	0000_0000h
90h	SAI Receive Configuration 2 Register (RCR2)	32	RW	0000_0000h
94h	SAI Receive Configuration 3 Register (RCR3)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
98h	SAI Receive Configuration 4 Register (RCR4)	32	RW	0000_0000h
9Ch	SAI Receive Configuration 5 Register (RCR5)	32	RW	0000_0000h
A0h - A4h	SAI Receive Data Register (RDR0 - RDR1)	32	RO	0000_0000h
C0h - C4h	SAI Receive FIFO Register (RFR0 - RFR1)	32	RO	0000_0000h
E0h	SAI Receive Mask Register (RMR)	32	RW	0000_0000h
F0h	SAI Receive Timestamp Control Register (RTCR)	32	RW	0000_0000h
F4h	SAI Receive Timestamp Register (RTSR)	32	RO	0000_0000h
F8h	SAI Receive Bit Count Register (RBCR)	32	RO	0000_0000h
FCh	SAI Receive Bit Count Timestamp Register (RBCTR)	32	RO	0000_0000h
100h	SAI MCLK Control Register (MCR)	32	RW	0000_0000h

13.9.4.1.2 Version ID Register (VERID)

13.9.4.1.2.1 Offset

Register	Offset
VERID	0h

13.9.4.1.2.2 Function

Contains version numbers for the module design and feature set.

13.9.4.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

13.9.4.1.2.4 Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the specification.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000000000000000b - Standard feature set. 0000000000000010b - Standard feature set with Timestamp Registers.

13.9.4.1.3 Parameter Register (PARAM)

13.9.4.1.3.1 Offset

Register	Offset
PARAM	4h

13.9.4.1.3.2 Function

Contains parameter values that were implemented in the module.

13.9.4.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												FRAME			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FIFO				0				DATALINE			
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	0

13.9.4.1.3.4 Fields

Field	Function
31-20	Reserved

Table continues on the next page...

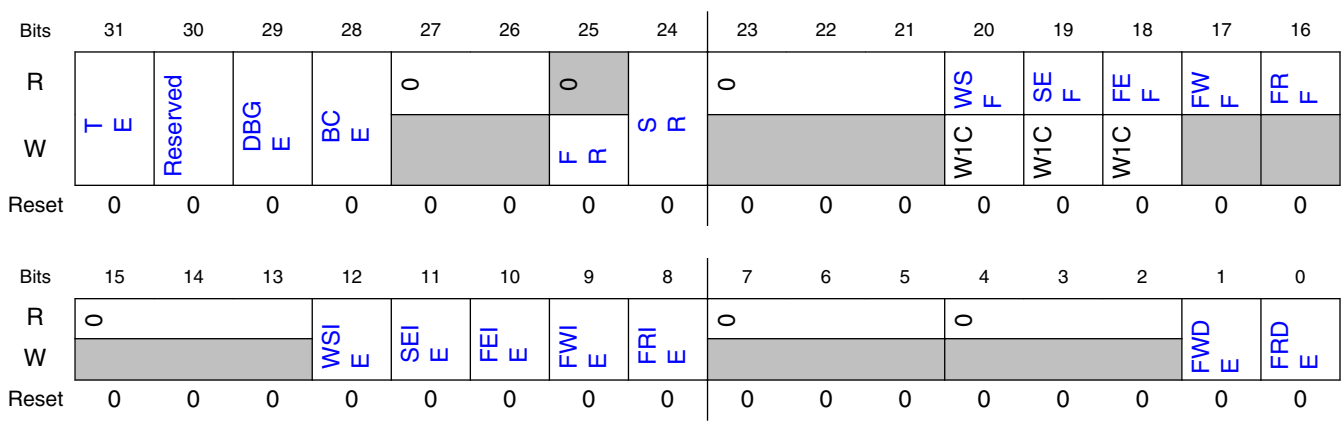
Field	Function
—	
19-16 FRAME	Frame Size The maximum number of slots per frame is 2^FRAME.
15-12 —	Reserved
11-8 FIFO	FIFO Size The number of words in each FIFO is 2^FIFO.
7-4 —	Reserved
3-0 DATALINE	Number of Datalines The number of datalines implemented.

13.9.4.1.4 SAI Transmit Control Register (TCSR)

13.9.4.1.4.1 Offset

Register	Offset
TCSR	8h

13.9.4.1.4.2 Diagram



13.9.4.1.4.3 Fields

Field	Function
31 TE	Transmitter Enable

Table continues on the next page...

Field	Function
	Enables/disables the transmitter. When software clears this field, the transmitter remains enabled, and this bit remains set, until the end of the current frame. 0b - Transmitter is disabled. 1b - Transmitter is enabled, or transmitter has been disabled and has not yet reached end of frame.
30 —	Reserved Software should only write zero to this reserved bit.
29 DBGE	Debug Enable Enables/disables transmitter operation in Debug mode. The transmit bit clock is not affected by debug mode. 0b - Transmitter is disabled in Debug mode, after completing the current frame. 1b - Transmitter is enabled in Debug mode.
28 BCE	Bit Clock Enable Enables the transmit bit clock, separately from the TE. This field is automatically set whenever TE is set. When software clears this field, the transmit bit clock remains enabled, and this bit remains set, until the end of the current frame. 0b - Transmit bit clock is disabled. 1b - Transmit bit clock is enabled.
27-26 —	Reserved
25 FR	FIFO Reset Empties the FIFO, and sets the FIFO read and write pointers to the same value, which may or may not be zero. Reading this field will always return zero. FIFO pointers should only be reset when the transmitter is disabled or the FIFO error flag is set. 0b - No effect. 1b - FIFO reset.
24 SR	Software Reset When set, resets the internal transmitter logic including the FIFO read and write pointers. Software-visible registers are not affected, except for the status registers. 0b - No effect. 1b - Software reset.
23-21 —	Reserved
20 WSF	Word Start Flag Indicates that the start of the configured word has been detected. Write a logic 1 to this field to clear this flag. 0b - Start of word not detected. 1b - Start of word detected.
19 SEF	Sync Error Flag Indicates that an error in the externally-generated frame sync has been detected. Write a logic 1 to this field to clear this flag. 0b - Sync error not detected. 1b - Frame sync error detected.
18 FEF	FIFO Error Flag Indicates that an enabled transmit FIFO has underrun. Write a logic 1 to this field to clear this flag. 0b - Transmit underrun not detected. 1b - Transmit underrun detected.
17	FIFO Warning Flag

Table continues on the next page...

Field	Function
FWF	Indicates that an enabled transmit FIFO is empty. 0b - No enabled transmit FIFO is empty. 1b - Enabled transmit FIFO is empty.
16 FRF	FIFO Request Flag Indicates that the number of words in an enabled transmit channel FIFO is less than or equal to the transmit FIFO watermark. 0b - Transmit FIFO watermark has not been reached. 1b - Transmit FIFO watermark has been reached.
15-13 —	Reserved
12 WSIE	Word Start Interrupt Enable Enables/disables word start interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
11 SEIE	Sync Error Interrupt Enable Enables/disables sync error interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
10 FEIE	FIFO Error Interrupt Enable Enables/disables FIFO error interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
9 FWIE	FIFO Warning Interrupt Enable Enables/disables FIFO warning interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
8 FRIE	FIFO Request Interrupt Enable Enables/disables FIFO request interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
7-5 —	Reserved
4-2 —	Reserved
1 FWDE	FIFO Warning DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.
0 FRDE	FIFO Request DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.

13.9.4.1.5 SAI Transmit Configuration 1 Register (TCR1)

13.9.4.1.5.1 Offset

Register	Offset
TCR1	Ch

13.9.4.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TFW							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.5.3 Fields

Field	Function
31-7 —	Reserved
6-0 TFW	Transmit FIFO Watermark Configures the watermark level for all enabled transmit channels.

13.9.4.1.6 SAI Transmit Configuration 2 Register (TCR2)

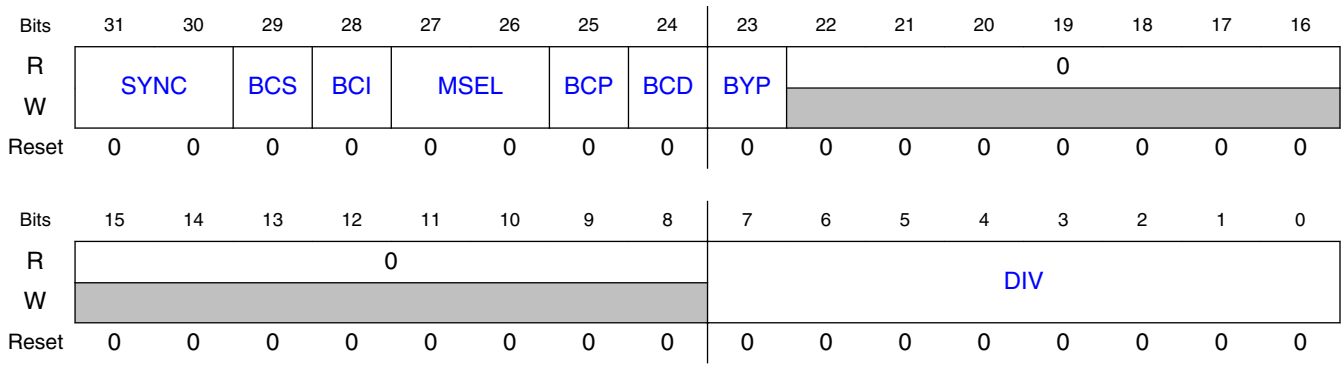
13.9.4.1.6.1 Offset

Register	Offset
TCR2	10h

13.9.4.1.6.2 Function

This register must not be altered when TCSR[TE] is set.

13.9.4.1.6.3 Diagram



13.9.4.1.6.4 Fields

Field	Function
31-30 SYNC	Synchronous Mode Configures between asynchronous and synchronous modes of operation. When configured for a synchronous mode of operation, the receiver must be configured for asynchronous operation. 00b - Asynchronous mode. 01b - Synchronous with receiver. 10b - Reserved. 11b - Reserved.
29 BCS	Bit Clock Swap This field swaps the bit clock used by the transmitter. When the transmitter is configured in asynchronous mode and this bit is set, the transmitter is clocked by the receiver bit clock (RX_BCLK). This allows the transmitter and receiver to share the same bit clock, but the transmitter continues to use the transmit frame sync (TX_SYNC). When the transmitter is configured in synchronous mode, the transmitter BCS field and receiver BCS field must be set to the same value. When both are set, the transmitter and receiver are both clocked by the transmitter bit clock (TX_BCLK) but use the receiver frame sync (RX_SYNC). 0b - Use the normal bit clock source. 1b - Swap the bit clock source.
28 BCI	Bit Clock Input When this field is set and using an internally generated bit clock in either synchronous or asynchronous mode, the bit clock actually used by the transmitter is delayed by the pad output delay (the transmitter is clocked by the pad input as if the clock was externally generated). This has the effect of decreasing the data input setup time, but increasing the data output valid time. The slave mode timing from the datasheet should be used for the transmitter when this bit is set. In synchronous mode, this bit allows the transmitter to use the slave mode timing from the datasheet, while the receiver uses the master mode timing. This field has no effect when configured for an externally generated bit clock . 0b - No effect. 1b - Internal logic is clocked as if bit clock was externally generated.
27-26 MSEL	MCLK Select

Table continues on the next page...

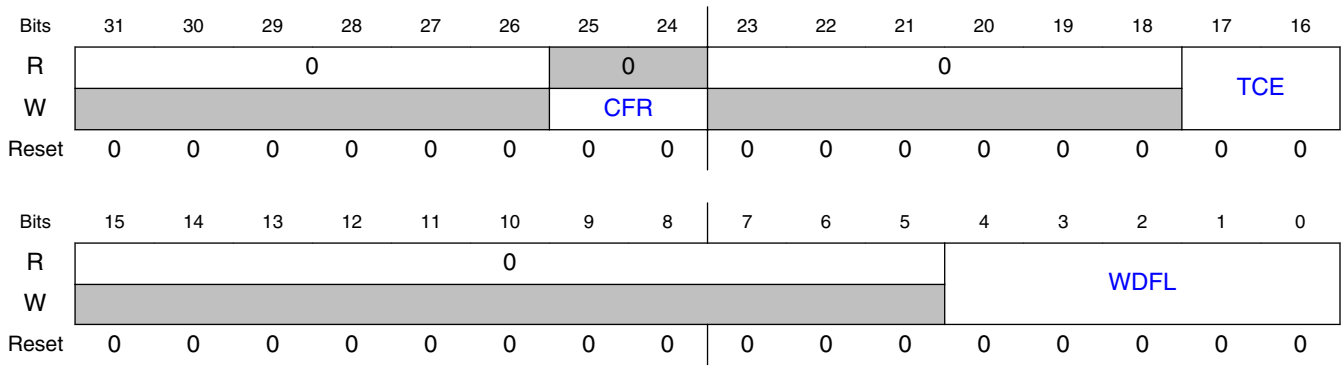
Field	Function
	Selects the audio Master Clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE:** Depending on the device, some Master Clock options might not be available. See the chip-specific information for the meaning of each option. 00b - Bus Clock selected. 01b - Master Clock (MCLK) 1 option selected. 10b - Master Clock (MCLK) 2 option selected. 11b - Master Clock (MCLK) 3 option selected.
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. 0b - Bit clock is active high with drive outputs on rising edge and sample inputs on falling edge. 1b - Bit clock is active low with drive outputs on falling edge and sample inputs on rising edge.
24 BCD	Bit Clock Direction Configures the direction of the bit clock. 0b - Bit clock is generated externally in Slave mode. 1b - Bit clock is generated internally in Master mode.
23 BYP	Bit Clock Bypass Bypasses the bit clock divider, internal bit clock is divide by 1 of the audio master clock. 0b - Internal bit clock is generated from bit clock divider. 1b - Internal bit clock is divide by one of the audio master clock.
22-8 —	Reserved
7-0 DIV	Bit Clock Divide Divides down the audio master clock to generate the bit clock when configured for an internal bit clock. The division value is (DIV + 1) * 2.

13.9.4.1.7 SAI Transmit Configuration 3 Register (TCR3)

13.9.4.1.7.1 Offset

Register	Offset
TCR3	14h

13.9.4.1.7.2 Diagram



13.9.4.1.7.3 Fields

Field	Function
31-26 —	Reserved
25-24 CFR	Channel FIFO Reset Resets the FIFO pointers for a specific channel. Reading this field will always return zero. FIFO pointers should only be reset when a channel is disabled or the FIFO error flag is set. The width of CFR field = the number of transmit channels (call it N). For example, if CFR is 2 bits wide, then bit position 24 refers to transmit channel 1 FIFO pointer and bit position 25 refers to transmit channel 2 FIFO pointer. Setting bit 24 resets transmit channel 1 FIFO pointer, and setting bit 25 enables transmit channel 2 FIFO pointer. Setting bit N will reset transmit channel N FIFO pointer. 0b - No effect. 1b - Transmit data channel N FIFO is reset.
23-18 —	Reserved
17-16 TCE	Transmit Channel Enable Enables the corresponding data channel for transmit operation. Changing TCE field will take effect immediately for generating the FIFO request and warning flags, but at the end of each frame for transmit operation. The width of TCE field = the number of transmit channels (call it N). For example, if TCE field is 2 bits wide, then bit position 16 refers to transmit channel 1 and bit position 17 refers to transmit channel 2. Setting bit 16 enables transmit channel 1, and setting bit 17 enables transmit channel 2. Setting bit N will enable transmit channel N. 0b - Transmit data channel N is disabled. 1b - Transmit data channel N is enabled.
15-5 —	Reserved
4-0 WDFL	Word Flag Configuration Configures which word sets the start of word flag. The value written must be one less than the word number. For example, writing 0 configures the first word in the frame. When configured to a value greater than TCR4[FRSZ], then the start of word flag is never set.

13.9.4.1.8 SAI Transmit Configuration 4 Register (TCR4)

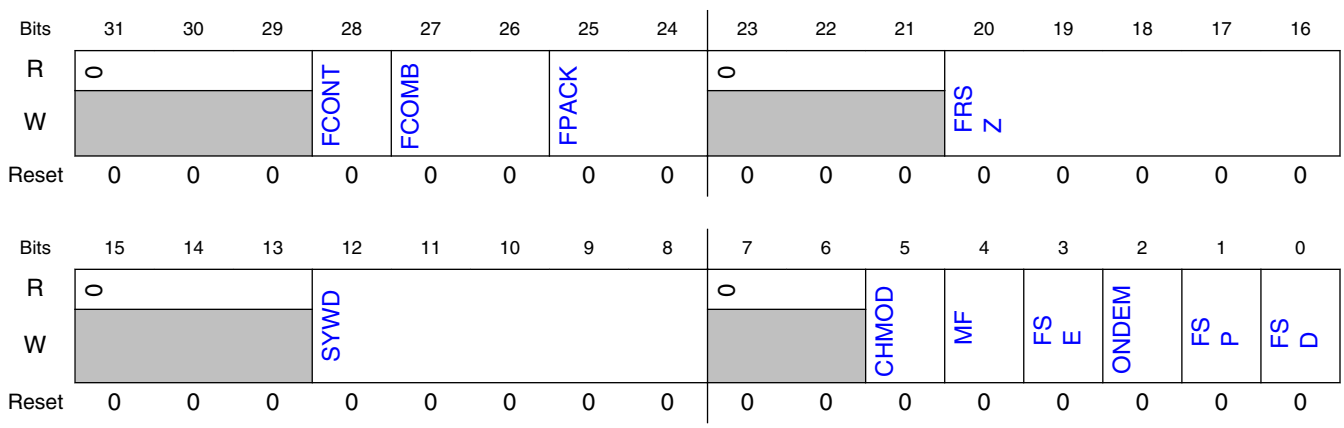
13.9.4.1.8.1 Offset

Register	Offset
TCR4	18h

13.9.4.1.8.2 Function

This register must not be altered when TCSR[TE] is set.

13.9.4.1.8.3 Diagram



13.9.4.1.8.4 Fields

Field	Function
31-29 —	Reserved
28 FCONT	FIFO Continue on Error Configures when the SAI will continue transmitting after a FIFO error has been detected. 0b - On FIFO error, the SAI will continue from the start of the next frame after the FIFO error flag has been cleared. 1b - On FIFO error, the SAI will continue from the same word that caused the FIFO error to set after the FIFO warning flag has been cleared.
27-26 FCOMB	FIFO Combine Mode

Table continues on the next page...

Field	Function
	When FIFO combine mode is enabled for FIFO writes, software writing to any FIFO data register will alternate the write among the enabled data channel FIFOs. For example, if two data channels are enabled then the first write will be performed to the first enabled data channel FIFO and the second write will be performed to the second enabled data channel FIFO. Reseting the FIFO or disabling FIFO combine mode for FIFO writes will reset the pointer back to the first enabled data channel. When FIFO combine mode is enabled for FIFO reads from the transmit shift registers, the transmit data channel output will alternate between the enabled data channel FIFOs. For example, if two data channels are enabled then the first unmasked word will be transmitted from the first enabled data channel FIFO and the second unmasked word will be transmitted from the second enabled data channel FIFO. Since the first word of the frame is always transmitted from the first enabled data channel FIFO, it is recommended that the number of unmasked words per frame is evenly divisible by the number of enabled data channels. 00b - FIFO combine mode disabled. 01b - FIFO combine mode enabled on FIFO reads (from transmit shift registers). 10b - FIFO combine mode enabled on FIFO writes (by software). 11b - FIFO combine mode enabled on FIFO reads (from transmit shift registers) and writes (by software).
25-24 FPACK	FIFO Packing Mode Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8-bit or 16-bit then only the first 8-bit or 16-bits are loaded from the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO write pointer will only increment when the full 32-bit FIFO word has been written by software. 00b - FIFO packing is disabled 01b - Reserved 10b - 8-bit FIFO packing is enabled 11b - 16-bit FIFO packing is enabled
23-21 —	Reserved
20-16 FRSZ	Frame size Configures the number of words in each frame. The value written must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 32 words.
15-13 —	Reserved
12-8 SYWD	Sync Width Configures the length of the frame sync in number of bit clocks. The value written must be one less than the number of bit clocks. For example, write 0 for the frame sync to assert for one bit clock only. The sync width cannot be configured longer than the first word of the frame.
7-6 —	Reserved
5 CHMOD	Channel Mode Configures if transmit data pins are configured for TDM mode or Output mode. 0b - TDM mode, transmit data pins are tri-stated when slots are masked or channels are disabled. 1b - Output mode, transmit data pins are never tri-stated and will output zero when slots are masked or channels are disabled.
4 MF	MSB First Configures whether the LSB or the MSB is transmitted first. 0b - LSB is transmitted first.

Table continues on the next page...

Field	Function
	1b - MSB is transmitted first.
3 FSE	Frame Sync Early 0b - Frame sync asserts with the first bit of the frame. 1b - Frame sync asserts one bit before the first bit of the frame.
2 ONDEM	On Demand Mode When set, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is clear. 0b - Internal frame sync is generated continuously. 1b - Internal frame sync is generated when the FIFO warning flag is clear.
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync. 0b - Frame sync is active high. 1b - Frame sync is active low.
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0b - Frame sync is generated externally in Slave mode. 1b - Frame sync is generated internally in Master mode.

13.9.4.1.9 SAI Transmit Configuration 5 Register (TCR5)

13.9.4.1.9.1 Offset

Register	Offset
TCR5	1Ch

13.9.4.1.9.2 Function

This register must not be altered when TCSR[TE] is set.

13.9.4.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			WNW					0			W0W				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			FBT					0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.9.4 Fields

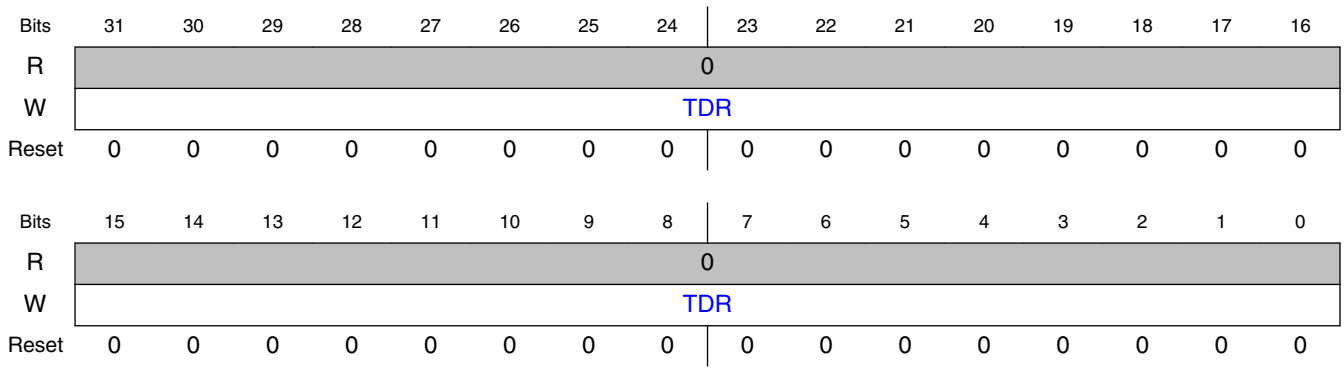
Field	Function
31-29 —	Reserved
28-24 WNW	Word N Width Configures the number of bits in each word, for each word except the first in the frame. The value written must be one less than the number of bits per word. Word width of less than 8 bits is not supported.
23-21 —	Reserved
20-16 W0W	Word 0 Width Configures the number of bits in the first word in each frame. The value written must be one less than the number of bits in the first word. Word width of less than 8 bits is not supported if there is only one word per frame.
15-13 —	Reserved
12-8 FBT	First Bit Shifted Configures the bit index for the first bit transmitted for each word in the frame. If configured for MSB First, the index of the next bit transmitted is one less than the current bit transmitted. If configured for LSB First, the index of the next bit transmitted is one more than the current bit transmitted. The value written must be greater than or equal to the word width when configured for MSB First. The value written must be less than or equal to 31-word width when configured for LSB First.
7-0 —	Reserved

13.9.4.1.10 SAI Transmit Data Register (TDR0 - TDR1)

13.9.4.1.10.1 Offset

Register	Offset
TDR0	20h
TDR1	24h

13.9.4.1.10.2 Diagram



13.9.4.1.10.3 Fields

Field	Function
31-0 TDR	Transmit Data Register Writes to this register when the transmit FIFO is not full will push the data written into the transmit data FIFO. Writes to this register when the transmit FIFO is full are ignored.

13.9.4.1.11 SAI Transmit FIFO Register (TFR0 - TFR1)

13.9.4.1.11.1 Offset

Register	Offset
TFR0	40h
TFR1	44h

13.9.4.1.11.2 Function

The MSB of the read and write pointers is used to distinguish between FIFO full and empty conditions. If the read and write pointers are identical, then the FIFO is empty. If the read and write pointers are identical except for the MSB, then the FIFO is full.

13.9.4.1.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WCP	0							WFP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RFP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.11.4 Fields

Field	Function
31 WCP	Write Channel Pointer When FIFO Combine mode is enabled for writes, indicates that this data channel is the next FIFO to be written. 0b - No effect. 1b - FIFO combine is enabled for FIFO writes and this FIFO will be written on the next FIFO write.
30-24 —	Reserved
23-16 WFP	Write FIFO Pointer FIFO write pointer for transmit data channel.
15-8 —	Reserved
7-0 RFP	Read FIFO Pointer FIFO read pointer for transmit data channel.

13.9.4.1.12 SAI Transmit Mask Register (TMR)

13.9.4.1.12.1 Offset

Register	Offset
TMR	60h

13.9.4.1.12.2 Function

This register is double-buffered and updates:

1. When TCSR[TE] is first set
2. At the end of each frame.

This allows the masked words in each frame to change from frame to frame.

13.9.4.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TWM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TWM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.12.4 Fields

Field	Function
31-0 TWM	Transmit Word Mask Configures whether the transmit word is masked (transmit data pins are tri-stated or drive zero and transmit data not read from FIFO) for the corresponding word in the frame. 00000000000000000000000000000000b - Word N is enabled. 00000000000000000000000000000001b - Word N is masked. The transmit data pins are tri-stated or drive zero when masked.

13.9.4.1.13 SAI Transmit Timestamp Control Register (TTCR)

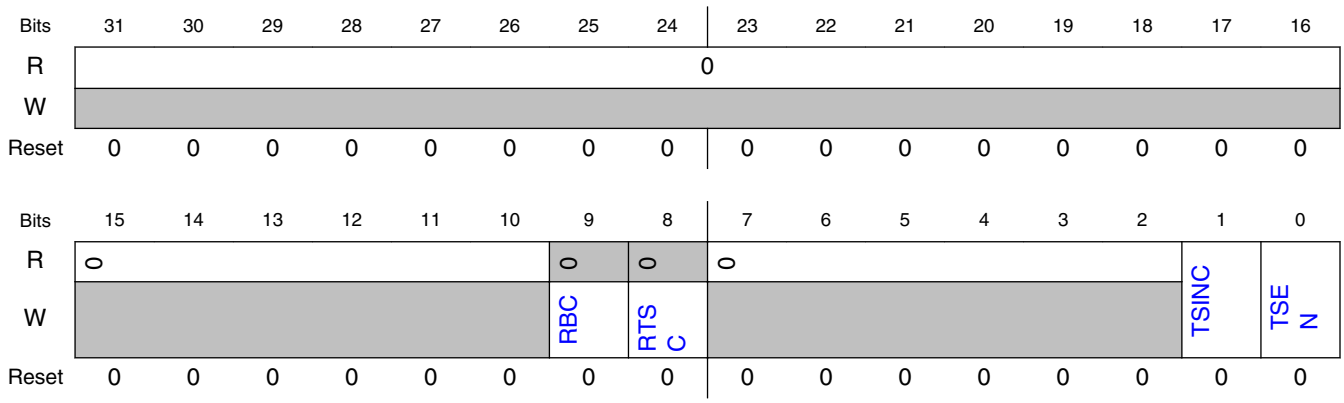
13.9.4.1.13.1 Offset

Register	Offset
TTCR	70h

13.9.4.1.13.2 Function

This register controls the transmit timestamp counter and bit clock counter.

13.9.4.1.13.3 Diagram



13.9.4.1.13.4 Fields

Field	Function
31-10 —	Reserved
9 RBC	Reset Bit Counter Reset the bit counter. 0b - Bit counter is not reset. 1b - Bit counter is reset.
8 RTSC	Reset Timestamp Counter Reset the timestamp counter. 0b - Timestamp counter is not reset. 1b - Timestamp counter is reset.
7-2 —	Reserved
1 TSINC	Timestamp Increment Configures when the timestamp counter starts to increment. 0b - Timestamp counter starts to increment when enabled and the bit counter has incremented. 1b - Timestamp counter starts to increment when enabled.
0 TSEN	Timestamp Enable Enables the timestamp counter to increment on the bus interface clock. 0b - Timestamp counter is disabled. 1b - Timestamp counter is enabled.

13.9.4.1.14 SAI Transmit Timestamp Register (TTSR)

13.9.4.1.14.1 Offset

Register	Offset
TTSR	74h

13.9.4.1.14.2 Function

This register returns the current value of the transmit timestamp counter.

13.9.4.1.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TSC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TSC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.14.4 Fields

Field	Function
31-0 TSC	Timestamp Counter Current value of the timestamp counter.

13.9.4.1.15 SAI Transmit Bit Count Register (TBCR)

13.9.4.1.15.1 Offset

Register	Offset
TBCR	78h

13.9.4.1.15.2 Function

This register returns the current value of the transmit bit counter. The bit counter increments for each bit in a frame when the SAI transmitter is enabled. Reading this

register causes the Bit Count Timestamp Regsiter to update with the value of the timestamp counter that correponds to the bit count register value.

13.9.4.1.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BCNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BCNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.15.4 Fields

Field	Function
31-0 BCNT	Bit Counter Returns the current value of the bit counter.

13.9.4.1.16 SAI Transmit Bit Count Timestamp Register (TBCTR)

13.9.4.1.16.1 Offset

Register	Offset
TBCTR	7Ch

13.9.4.1.16.2 Function

Returns the value of the timestamp register that corresponds to the value last read from the bit count register.

13.9.4.1.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BCTS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BCTS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.16.4 Fields

Field	Function
31-0 BCTS	Bit Timestamp Returns the timestamp counter value that matches the last read of the bit count register.

13.9.4.1.17 SAI Receive Control Register (RCSR)

13.9.4.1.17.1 Offset

Register	Offset
RCSR	88h

13.9.4.1.17.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RE	Reserved	DBGE	BCE	0		0	SR	0			WSF	SEF	FEF	FWF	FRF
W							FR					W1C	W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			WSIE	SEIE	FEIE	FWIE	FRIE	0			0			FWDE	FRDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.17.3 Fields

Field	Function
31 RE	Receiver Enable Enables/disables the receiver. When software clears this field, the receiver remains enabled, and this bit remains set, until the end of the current frame. 0b - Receiver is disabled. 1b - Receiver is enabled, or receiver has been disabled and has not yet reached end of frame.
30 —	Reserved Software should only write zero to this reserved bit.
29 DBGE	Debug Enable Enables/disables receiver operation in Debug mode. The receive bit clock is not affected by Debug mode. 0b - Receiver is disabled in Debug mode, after completing the current frame. 1b - Receiver is enabled in Debug mode.
28 BCE	Bit Clock Enable Enables the receive bit clock, separately from RE. This field is automatically set whenever RE is set. When software clears this field, the receive bit clock remains enabled, and this field remains set, until the end of the current frame. 0b - Receive bit clock is disabled. 1b - Receive bit clock is enabled.
27-26 —	Reserved
25 FR	FIFO Reset Empties the FIFO, and sets the FIFO read and write pointers to the same value, which may or may not be zero. Reading this field will always return zero. FIFO pointers should only be reset when the receiver is disabled or the FIFO error flag is set. 0b - No effect. 1b - FIFO reset.
24 SR	Software Reset Resets the internal receiver logic including the FIFO pointers. Software-visible registers are not affected, except for the status registers. 0b - No effect. 1b - Software reset.
23-21 —	Reserved
20 WSF	Word Start Flag Indicates that the start of the configured word has been detected. Write a logic 1 to this field to clear this flag. 0b - Start of word not detected. 1b - Start of word detected.
19 SEF	Sync Error Flag Indicates that an error in the externally-generated frame sync has been detected. Write a logic 1 to this field to clear this flag. 0b - Sync error not detected. 1b - Frame sync error detected.
18 FEF	FIFO Error Flag Indicates that an enabled receive FIFO has overflowed. Write a logic 1 to this field to clear this flag.

Table continues on the next page...

Field	Function
	0b - Receive overflow not detected. 1b - Receive overflow detected.
17 FWF	FIFO Warning Flag Indicates that an enabled receive FIFO is full. 0b - No enabled receive FIFO is full. 1b - Enabled receive FIFO is full.
16 FRF	FIFO Request Flag Indicates that the number of words in an enabled receive channel FIFO is greater than the receive FIFO watermark. 0b - Receive FIFO watermark not reached. 1b - Receive FIFO watermark has been reached.
15-13 —	Reserved
12 WSIE	Word Start Interrupt Enable Enables/disables word start interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
11 SEIE	Sync Error Interrupt Enable Enables/disables sync error interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
10 FEIE	FIFO Error Interrupt Enable Enables/disables FIFO error interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
9 FWIE	FIFO Warning Interrupt Enable Enables/disables FIFO warning interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
8 FRIE	FIFO Request Interrupt Enable Enables/disables FIFO request interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
7-5 —	Reserved
4-2 —	Reserved
1 FWDE	FIFO Warning DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.
0 FRDE	FIFO Request DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.

13.9.4.1.18 SAI Receive Configuration 1 Register (RCR1)

13.9.4.1.18.1 Offset

Register	Offset
RCR1	8Ch

13.9.4.1.18.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RFW							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.18.3 Fields

Field	Function
31-7 —	Reserved
6-0 RFW	Receive FIFO Watermark Configures the watermark level for all enabled receiver channels.

13.9.4.1.19 SAI Receive Configuration 2 Register (RCR2)

13.9.4.1.19.1 Offset

Register	Offset
RCR2	90h

13.9.4.1.19.2 Function

This register must not be altered when RCSR[RE] is set.

13.9.4.1.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SYNC		BCS	BCI	MSEL		BCP	BCD	BYP	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								DIV							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.19.4 Fields

Field	Function
31-30 SYNC	Synchronous Mode Configures between asynchronous and synchronous modes of operation. When configured for a synchronous mode of operation, the transmitter must be configured for asynchronous operation. 00b - Asynchronous mode. 01b - Synchronous with transmitter. 10b - Reserved. 11b - Reserved.
29 BCS	Bit Clock Swap This field swaps the bit clock used by the receiver. When the receiver is configured in asynchronous mode and this bit is set, the receiver is clocked by the transmitter bit clock (TX_BCLK). This allows the transmitter and receiver to share the same bit clock, but the receiver continues to use the receiver frame sync (RX_SYNC). When the receiver is configured in synchronous mode, the transmitter BCS field and receiver BCS field must be set to the same value. When both are set, the transmitter and receiver are both clocked by the receiver bit clock (RX_BCLK) but use the transmitter frame sync (TX_SYNC). 0b - Use the normal bit clock source. 1b - Swap the bit clock source.
28 BCI	Bit Clock Input When this field is set and using an internally generated bit clock in either synchronous or asynchronous mode, the bit clock actually used by the receiver is delayed by the pad output delay (the receiver is clocked by the pad input as if the clock was externally generated). This has the effect of decreasing the data input setup time, but increasing the data output valid time. The slave mode timing from the datasheet should be used for the receiver when this bit is set. In synchronous mode, this bit allows the receiver to use the slave mode timing from the datasheet, while the transmitter uses the master mode timing. This field has no effect when configured for an externally generated bit clock . 0b - No effect.

Table continues on the next page...

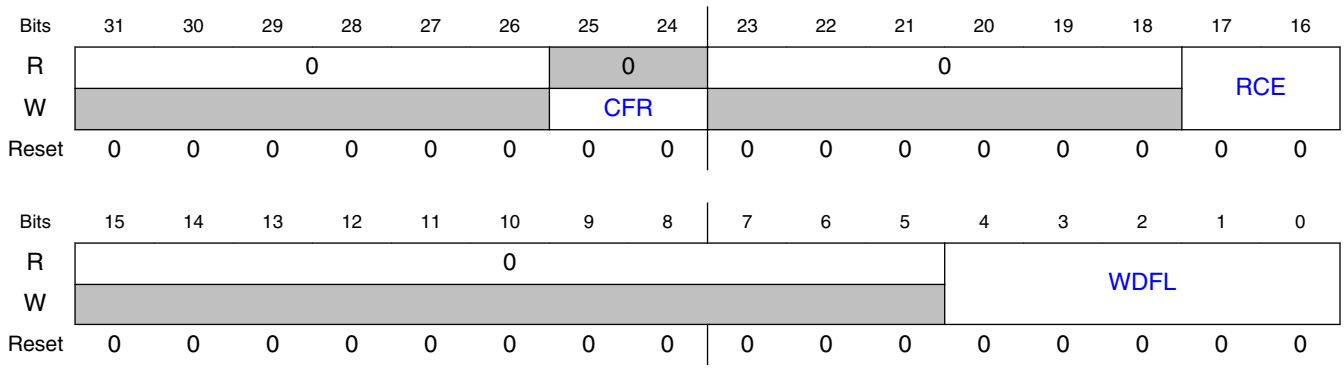
Field	Function
	1b - Internal logic is clocked as if bit clock was externally generated.
27-26 MSEL	MCLK Select Selects the audio Master Clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE:** Depending on the device, some Master Clock options might not be available. See the chip-specific information for the availability and chip-specific meaning of each option. 00b - Bus Clock selected. 01b - Master Clock (MCLK) 1 option selected. 10b - Master Clock (MCLK) 2 option selected. 11b - Master Clock (MCLK) 3 option selected.
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. 0b - Bit Clock is active high with drive outputs on rising edge and sample inputs on falling edge. 1b - Bit Clock is active low with drive outputs on falling edge and sample inputs on rising edge.
24 BCD	Bit Clock Direction Configures the direction of the bit clock. 0b - Bit clock is generated externally in Slave mode. 1b - Bit clock is generated internally in Master mode.
23 BYP	Bit Clock Bypass Bypasses the bit clock divider, internal bit clock is divide by 1 of the audio master clock. 0b - Internal bit clock is generated from bit clock divider. 1b - Internal bit clock is divide by one of the audio master clock.
22-8 —	Reserved
7-0 DIV	Bit Clock Divide Divides down the audio master clock to generate the bit clock when configured for an internal bit clock. The division value is (DIV + 1) * 2.

13.9.4.1.20 SAI Receive Configuration 3 Register (RCR3)

13.9.4.1.20.1 Offset

Register	Offset
RCR3	94h

13.9.4.1.20.2 Diagram



13.9.4.1.20.3 Fields

Field	Function
31-26 —	Reserved
25-24 CFR	Channel FIFO Reset Resets the FIFO pointers for a specific channel. Reading this field will always return zero. FIFO pointers should only be reset when a channel is disabled or the FIFO error flag is set. The width of CFR field = the number of receive channels (call it N). For example, if CFR is 2 bits wide, then bit position 24 refers to receive channel 1 FIFO pointer and bit position 25 refers to receive channel 2 FIFO pointer. Setting bit 24 resets receive channel 1 FIFO pointer, and setting bit 25 enables receive channel 2 FIFO pointer. Setting bit N will reset receive channel N FIFO pointer. 0b - No effect. 1b - Receive data channel N FIFO is reset.
23-18 —	Reserved
17-16 RCE	Receive Channel Enable Enables the corresponding data channel for receive operation. Changing this field will take effect immediately for generating the FIFO request and warning flags, but at the end of each frame for receive operation. The width of RCE field = the number of receive channels (call it N). For example, if RCE field is 2 bits wide, then bit position 16 refers to receive channel 1 and bit position 17 refers to receive channel 2. Setting bit 16 enables receive channel 1, and setting bit 17 enables receive channel 2. Setting bit N will enable receive channel N. 0b - Receive data channel N is disabled. 1b - Receive data channel N is enabled.
15-5 —	Reserved
4-0 WDFL	Word Flag Configuration Configures which word the start of word flag is set. The value written should be one less than the word number (for example, write zero to configure for the first word in the frame). When configured to a value greater than the Frame Size field, then the start of word flag is never set.

13.9.4.1.21 SAI Receive Configuration 4 Register (RCR4)

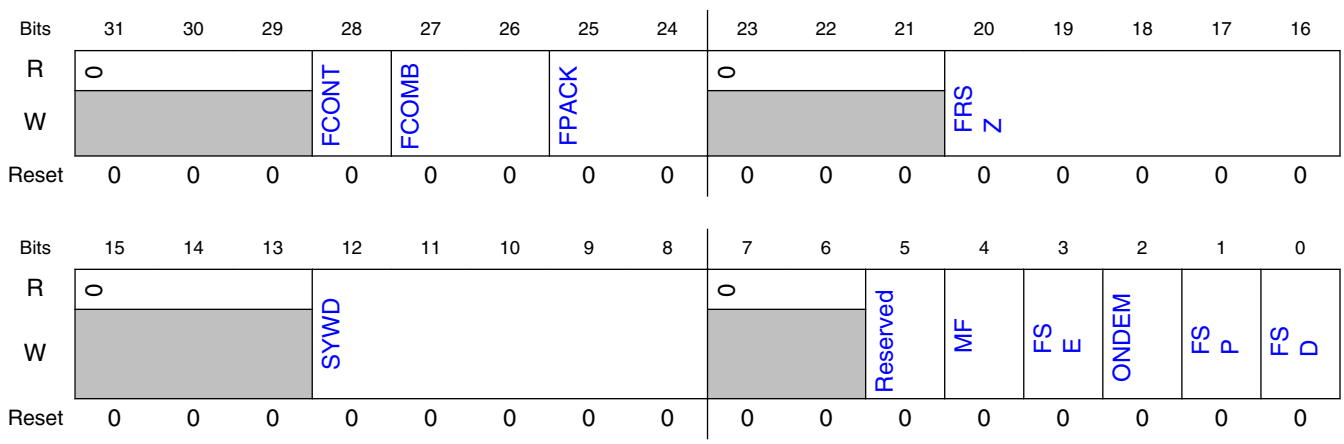
13.9.4.1.21.1 Offset

Register	Offset
RCR4	98h

13.9.4.1.21.2 Function

This register must not be altered when RCSR[RE] is set.

13.9.4.1.21.3 Diagram



13.9.4.1.21.4 Fields

Field	Function
31-29 —	Reserved
28 FCONT	FIFO Continue on Error Configures when the SAI will continue receiving after a FIFO error has been detected. 0b - On FIFO error, the SAI will continue from the start of the next frame after the FIFO error flag has been cleared. 1b - On FIFO error, the SAI will continue from the same word that caused the FIFO error to set after the FIFO warning flag has been cleared.
27-26 FCOMB	FIFO Combine Mode

Table continues on the next page...

Field	Function
	When FIFO combine mode is enabled for FIFO reads, software reading any FIFO data register will alternate the read among the enabled data channel FIFOs. For example, if two data channels are enabled then the first read will be performed to the first enabled data channel FIFO and the second read will be performed to the second enabled data channel FIFO. Reseting the FIFO or disabling FIFO combine mode for FIFO reads will reset the pointer back to the first enabled data channel. When FIFO combine mode is enabled for FIFO writes from the receive shift registers, the first enabled data channel input will alternate between the enabled data channel FIFOs. For example, if two data channels are enabled then the first unmasked received word will be stored in the first enabled data channel FIFO and the second unmasked received word will be stored in the second enabled data channel FIFO. Since the first word of the frame is always stored in the first enabled data channel FIFO, it is recommended that the number of unmasked words per frame is evenly divisible by the number of enabled data channels. 00b - FIFO combine mode disabled. 01b - FIFO combine mode enabled on FIFO writes (from receive shift registers). 10b - FIFO combine mode enabled on FIFO reads (by software). 11b - FIFO combine mode enabled on FIFO writes (from receive shift registers) and reads (by software).
25-24 FPACK	FIFO Packing Mode Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8-bit or 16-bit then only the first 8-bit or 16-bits are stored to the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO read pointer will only increment when the full 32-bit FIFO word has been read by software. 00b - FIFO packing is disabled 01b - Reserved. 10b - 8-bit FIFO packing is enabled 11b - 16-bit FIFO packing is enabled
23-21 —	Reserved
20-16 FRSZ	Frame Size Configures the number of words in each frame. The value written must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 32 words.
15-13 —	Reserved
12-8 SYWD	Sync Width Configures the length of the frame sync in number of bit clocks. The value written must be one less than the number of bit clocks. For example, write 0 for the frame sync to assert for one bit clock only. The sync width cannot be configured longer than the first word of the frame.
7-6 —	Reserved
5 —	Reserved Software should only write zero to this bit.
4 MF	MSB First Configures whether the LSB or the MSB is received first. 0b - LSB is received first. 1b - MSB is received first.
3	Frame Sync Early

Table continues on the next page...

Field	Function
FSE	0b - Frame sync asserts with the first bit of the frame. 1b - Frame sync asserts one bit before the first bit of the frame.
2 ONDEM	On Demand Mode When set, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is clear. 0b - Internal frame sync is generated continuously. 1b - Internal frame sync is generated when the FIFO warning flag is clear.
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync. 0b - Frame sync is active high. 1b - Frame sync is active low.
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0b - Frame Sync is generated externally in Slave mode. 1b - Frame Sync is generated internally in Master mode.

13.9.4.1.22 SAI Receive Configuration 5 Register (RCR5)

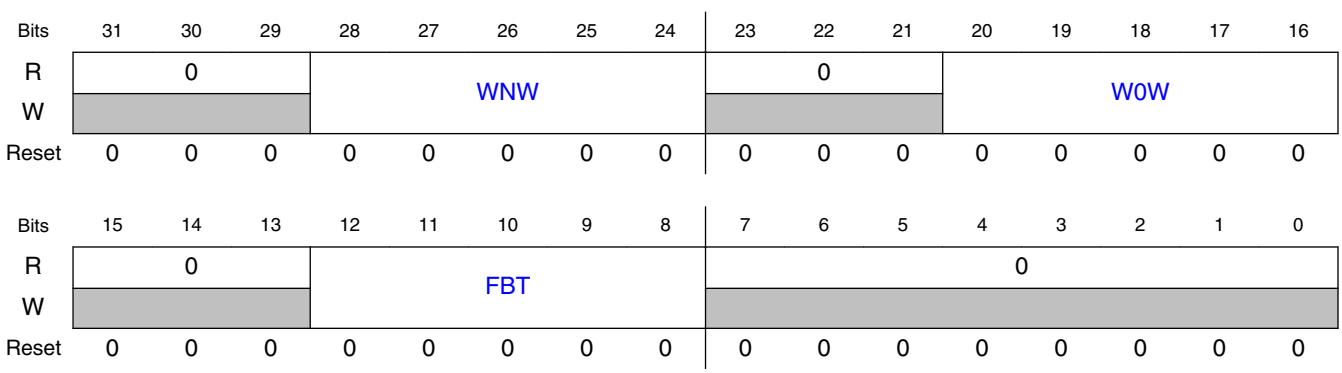
13.9.4.1.22.1 Offset

Register	Offset
RCR5	9Ch

13.9.4.1.22.2 Function

This register must not be altered when RCSR[RE] is set.

13.9.4.1.22.3 Diagram



13.9.4.1.22.4 Fields

Field	Function
31-29 —	Reserved
28-24 WNW	Word N Width Configures the number of bits in each word, for each word except the first in the frame. The value written must be one less than the number of bits per word. Word width of less than 8 bits is not supported.
23-21 —	Reserved
20-16 W0W	Word 0 Width Configures the number of bits in the first word in each frame. The value written must be one less than the number of bits in the first word. Word width of less than 8 bits is not supported if there is only one word per frame.
15-13 —	Reserved
12-8 FBT	First Bit Shifted Configures the bit index for the first bit received for each word in the frame. If configured for MSB First, the index of the next bit received is one less than the current bit received. If configured for LSB First, the index of the next bit received is one more than the current bit received. The value written must be greater than or equal to the word width when configured for MSB First. The value written must be less than or equal to 31-word width when configured for LSB First.
7-0 —	Reserved

13.9.4.1.23 SAI Receive Data Register (RDR0 - RDR1)

13.9.4.1.23.1 Offset

Register	Offset
RDR0	A0h
RDR1	A4h

13.9.4.1.23.2 Function

Reading this register introduces one additional peripheral clock wait state on each read.

13.9.4.1.23.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.23.4 Fields

Field	Function
31-0 RDR	Receive Data Register Reads from this register when the receive FIFO is not empty will return the data from the top of the receive FIFO. Reads from this register when the receive FIFO is empty are ignored.

13.9.4.1.24 SAI Receive FIFO Register (RFR0 - RFR1)

13.9.4.1.24.1 Offset

Register	Offset
RFR0	C0h
RFR1	C4h

13.9.4.1.24.2 Function

The MSB of the read and write pointers is used to distinguish between FIFO full and empty conditions. If the read and write pointers are identical, then the FIFO is empty. If the read and write pointers are identical except for the MSB, then the FIFO is full.

13.9.4.1.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								WFP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCP	0							RFP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.24.4 Fields

Field	Function
31-24 —	Reserved
23-16 WFP	Write FIFO Pointer FIFO write pointer for receive data channel.
15 RCP	Receive Channel Pointer When FIFO Combine mode is enabled for reads, indicates that this data channel is the next FIFO to be read. 0b - No effect. 1b - FIFO combine is enabled for FIFO reads and this FIFO will be read on the next FIFO read.
14-8 —	Reserved
7-0 RFP	Read FIFO Pointer FIFO read pointer for receive data channel.

13.9.4.1.25 SAI Receive Mask Register (RMR)

13.9.4.1.25.1 Offset

Register	Offset
RMR	E0h

13.9.4.1.25.2 Function

This register is double-buffered and updates:

1. When RCSR[RE] is first set
2. At the end of each frame

This allows the masked words in each frame to change from frame to frame.

13.9.4.1.25.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RWM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RWM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.25.4 Fields

Field	Function
31-0 RWM	Receive Word Mask Configures whether the receive word is masked (received data ignored and not written to receive FIFO) for the corresponding word in the frame. 00000000000000000000000000000000b - Word N is enabled. 00000000000000000000000000000001b - Word N is masked.

13.9.4.1.26 SAI Receive Timestamp Control Register (RTCR)

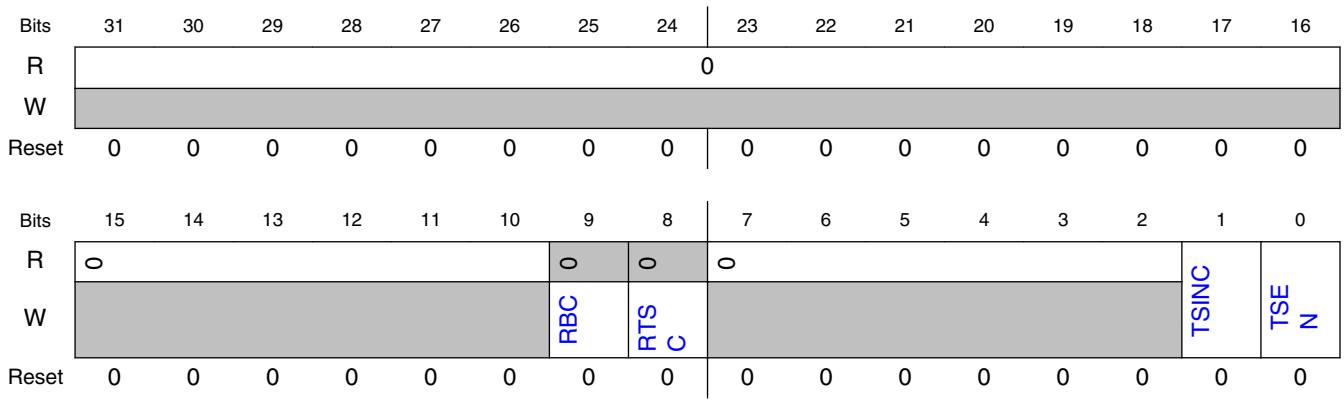
13.9.4.1.26.1 Offset

Register	Offset
RTCR	F0h

13.9.4.1.26.2 Function

This register controls the receive timestamp counter and bit clock counter.

13.9.4.1.26.3 Diagram



13.9.4.1.26.4 Fields

Field	Function
31-10 —	Reserved
9 RBC	Reset Bit Counter Reset the receive bit counter. 0b - Bit counter is not reset. 1b - Bit counter is reset.
8 RTSC	Reset Timestamp Counter Reset the receive timestamp counter. 0b - Timestamp counter is not reset. 1b - Timestamp counter is reset.
7-2 —	Reserved
1 TSINC	Timestamp Increment Configures when the timestamp counter starts to increment. 0b - Timestamp counter starts to increment when enabled and the bit counter has incremented. 1b - Timestamp counter starts to increment when enabled.
0 TSEN	Timestamp Enable Enables the timestamp counter to increment on the bus interface clock. 0b - Timestamp counter is disabled. 1b - Timestamp counter is enabled.

13.9.4.1.27 SAI Receive Timestamp Register (RTSR)

13.9.4.1.27.1 Offset

Register	Offset
RTSR	F4h

13.9.4.1.27.2 Function

This register returns the current value of the receive timestamp counter.

13.9.4.1.27.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TSC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TSC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.27.4 Fields

Field	Function
31-0 TSC	Timestamp Counter Current value of the timestamp counter.

13.9.4.1.28 SAI Receive Bit Count Register (RBCR)

13.9.4.1.28.1 Offset

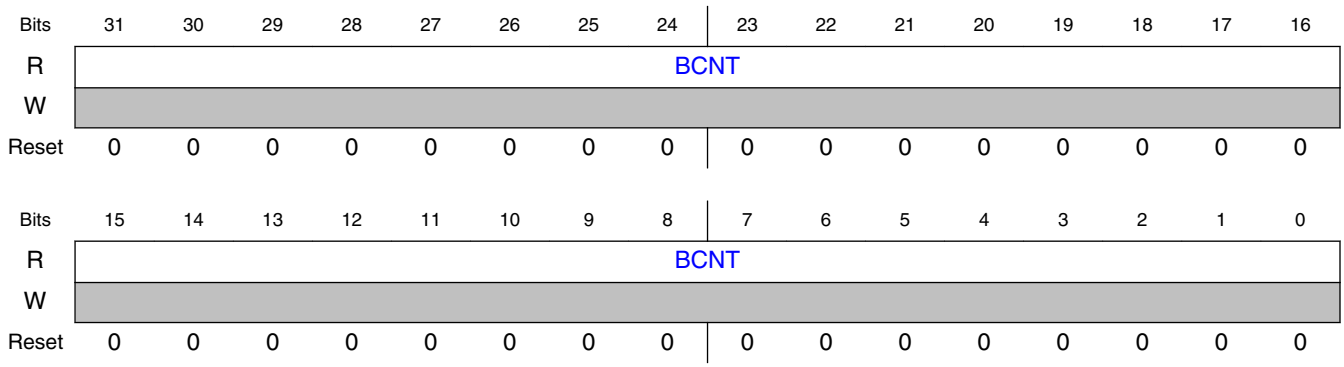
Register	Offset
RBCR	F8h

13.9.4.1.28.2 Function

This register returns the current value of the receive bit counter. The bit counter increments for each bit in a frame when the SAI receiver is enabled. Reading this register

causes the Bit Count Timestamp Regsiter to update with the value of the timestamp counter that correponds to the bit count register value.

13.9.4.1.28.3 Diagram



13.9.4.1.28.4 Fields

Field	Function
31-0 BCNT	Bit Counter Returns the current value of the bit counter.

13.9.4.1.29 SAI Receive Bit Count Timestamp Register (RBCTR)

13.9.4.1.29.1 Offset

Register	Offset
RBCTR	FCh

13.9.4.1.29.2 Function

Returns the value of the timestamp register that corresponds to the value last read from the bit count register.

13.9.4.1.29.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BCTS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BCTS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.9.4.1.29.4 Fields

Field	Function
31-0	Bit Timestamp
BCTS	Returns the timestamp counter value that matches the last read of the bit count register.

13.9.4.1.30 SAI MCLK Control Register (MCR)

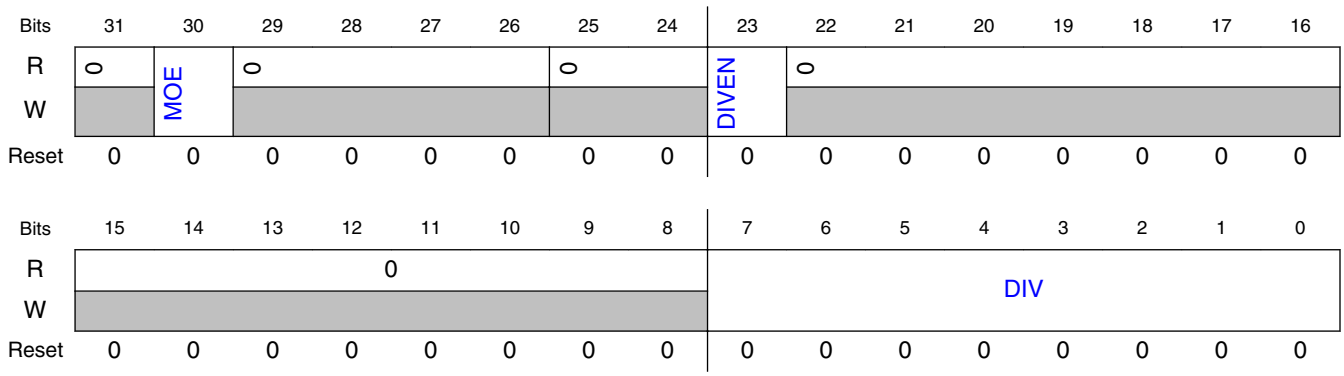
13.9.4.1.30.1 Offset

Register	Offset
MCR	100h

13.9.4.1.30.2 Function

The MCLK Control Register (MCR) controls the audio master clock.

13.9.4.1.30.3 Diagram



13.9.4.1.30.4 Fields

Field	Function
31 —	Reserved
30 MOE	MCLK Output Enable Configures MCLK signal pin as an input or an output. 0b - MCLK signal pin is an input. 1b - MCLK signal pin is an output.
29-26 —	Reserved
25-24 —	Reserved
23 DIVEN	MCLK Post Divide Enable Enables the post-divider that is output on the MCLK signal pin, the output divider does not impact the audio master clock used by the transmitter or receiver. 0b - Output on MCLK signal pin is the audio master clock. 1b - Output on MCLK signal pin is a post-divided version of audio master clock.
22-8 —	Reserved
7-0 DIV	MCLK Post Divide Post-divide for the audio master clock that only divides the output to the MCLK signal pin. The division value is (DIV + 1) * 2.

Chapter 14
Graphics Processing Unit (GPU)

14.1 GPU Overview

14.1.1 Overview

This section provides a high-level overview pf the Graphics Processing Unit (GPU) Subsystem in the chip. This chip uses two GPU cores, one for 3D processing and the other to provide 2D/3D acceleration.

14.1.1.1 Block Diagram

The high level block diagram is shown as below:

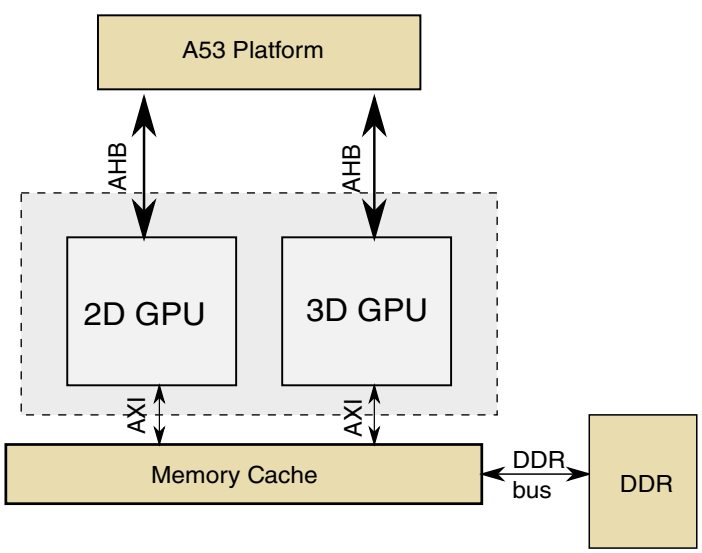


Figure 14-1. High Level Diagram

The 3D GPU and 2D GPU are designed in such a away that if only one core is required, the other can be gated off.

14.1.2 Features

The key features of the GPUs are described below:

- 3D GPU Core:
  - Support OpenGL ES 1.1, 2.0
  - Support OpenVG 1.1
  - TrustZone support using a local MMU to manage secure regions
- 2D GPU Core:
  - Support multi-source composition
  - Support one-pass filter

14.2 2D Graphics Processing Unit (GPU2D)

14.2.1 Overview

This Graphics Processing Unit defines a high-performance, multi-pipe 2D raster graphics core that accelerates the 2D graphics display on a variety of consumer devices and provides advanced compression capabilities.

It supports the following graphic APIs:

- DirectFB (on Linux / Linux Embedded)
- GDI/DirectDraw (on Windows Embedded Compact 7 / Embedded CE 6)
- Android

14.2.1.1 Block Diagram

The block diagram and a description of the functional blocks of the complete graphics pipeline is given below.

- Host Interface - Allows the core to communicate with external memory and the CPU through the AXI or the AHB bus. In this block, data crosses clock domain boundaries.
- Memory Controller - Internal memory management unit that controls the block-to-host memory requests.
- Graphics Pipeline Front End - Inserts high level primitives and commands into the graphics pipeline.

- 2D Drawing and Scaling Engine - Draws 2D graphics primitives and rasterizes 2D images.
- Pixel Engine - Pixel manipulation and filtering on rendered images. The GPU has four pixel pipes.

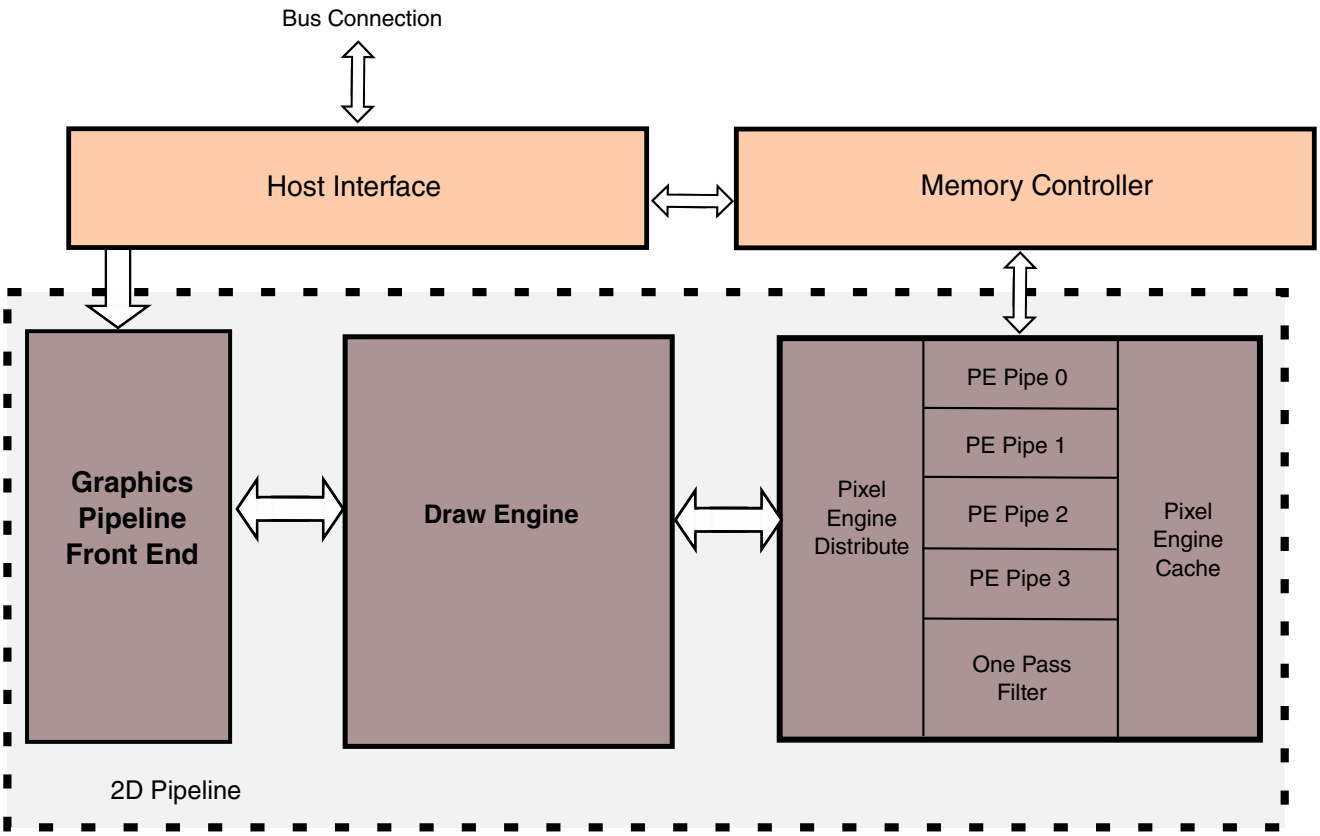


Figure 14-2. GPU Block Diagram

14.2.2 GPU Features

The features of the GPU include:

- 2D GPU Unit:
  - Bit Blit
  - Stretch Blit
  - Rectangle fill and clear
  - Line drawing
  - Filter Blit
  - Mono expansion for text rendering
  - ROP2, ROP3 and ROP4
  - Alpha blending, including Java 2 Porter-Duff compositing blending rules
  - 32K x 32K coordinate system
  - 90 / 180 / 270 degree rotation

    - Transparency by monochrome mask, chroma key, or pattern mask
    - Support 2x2 in 4x4 tile format
    - A8 output with rotation in filter Blit and bit Blit
    - Src/Dest color key full bypass support
- Multi Source Blending:
    - Full support for Multi source blending with variable block size to improve BW and reduce SW overhead
    - Up to 8 sources are supported
    - Programmable block size guarantees cache efficiency so each source is read once and each destination is written once
    - Supports 90, 180, 270 degree rotation with different block size for higher cache efficiency
- YUV Support:
    - Full Multi destination support for converting non-planar YUV formats to planar YUV. It is used in extracting various components from input color into different destination planes
    - YUV422 output with alpha blending supported
- Clock disabling:
    - The core clock enters the frequency scaling block, and a clock with missing pulses is output, buffered, and sent to clock gating logic. The clock gate output will be used by the 2D GPU blocks. The core clock can be shut down through software by setting the proper register bits
- AXI Bus:
    - AXI 4-bit ID, such that all Read and Write transactions have a unique AXI ID
    - AXI out of order read return supported
- Additional Enhancements:
    - Full functional MMU with variable page size support
- Rotation:
    - 90° / 180° / 270° / Mirror rotation is supported for all primitives
    - Independent source rotation
- Transparency Mode:
    - For Monochrome Expansion:
        - Opaque
        - Conditional transparency - Transparent if the current pixel matches the specified value
    - For blits:
        - Opaque
        - Masked transparency - Transparent if the mask for the current pixel or pattern is zero

    - Source Conditional transparency - Transparent if the source pixel is within the specified value range
    - Destination Conditional transparency - Transparent if the destination pixel is not within the specified value range
- Clipping:
    - One clipping rectangle is supported for all bit blit primitives

14.2.2.1 GPU Host and Memory Interface Features

Table 14-1. Host and memory interface features

Feature	GPU Support
AHB interface	32-bit
Interface for External memory access	one 128-bit AXI
Virtual memory support	Yes
Out of order return support	Yes, supports out of order return for different clients
Code and data memory location restrictions	Unrestricted; arbitrary memory reads and writes
System address space	256 Kbytes (64K 32-bit words); however, the range of addresses used is only 4 Kbytes
Physical address	32 bits
Resource locks with CPU	Semaphore lock
Read write request size support	Memory controller supports 16, 32, 64, 128 byte read and write requests
MMU with variable page size support	MMU supports variable page sizes (4KB, 64KB, 1MB, 16MB)
GPU Cache Flush	Auto-flush at programmable intervals

14.2.2.2 GPU Power Management Features

Table 14-2. Power management features

Feature	Support
Low power CMOS technology compatible	Yes
Dynamic clock scaling	Yes
Automatic clock gating of flip flops and rams	Yes
Global clock gating of unused macro blocks	Yes
Active(ON), Idle, Standby and Sleep(OFF) Programmable Power Modes	Yes

14.2.2.3 Composition and 2D Hardware Features

This topic highlights the hardware features of the dedicated 2D GPU.

Table 14-3. Composition and 2D Hardware Features

Feature	Support
Programmable OPs	ROP2, ROP3, ROP4 full alpha blending and transparency
Fixed function	Line draw,Rectangle fill, Rectangle Clear, Bit blit, Stretch blit, Filter blit
Blit support	Copy (Bit), Filter, Monochrome Mask, Stretch/Shrink
Source and Destination formats	The graphics engine supports the following source and destination formats for data, bit blits and filter blits. In addition to these source and destination RGB formats, their swizzle formats (ARGB, RGBA, ABGR, BGRA) are also supported. For YUV formats, the GPU supports their U/V swap formats.

Format	Bit Blit Input	Bit Blit Dest	Stretch Blit Input	Stretch Blit Dest	Filter Blit Input	Filter Blit Dest	OPF Input	OPF Dest	Multi-Source Input	Multi-Source Dest
A1R5G5B5	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
A4R4G4B4	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
A8R8G8B8	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
X1R5G5B5	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
X4R4G4B4	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
X8R8G8B8	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
R5G6B5	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
A8	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
YUY2 (packed	Y	Y	Y	N	Y	Yv	Y	Yv	Y	Y

Table continues on the next page...

Table 14-3. Composition and 2D Hardware Features

Feature	Support										
	Format	Bit Blit Input	Bit Blit Dest	Stretch Blit Input	Stretch Blit Dest	Filter Blit Input	Filter Blit Dest	OPF Input	OPF Dest	Multi-Source Input	Multi-Source Dest
	YUV422)										
	UYVY (packed YUV422)	Y	Y	Y	N	Y	N	Y	Y	Y	Y
	YV12 (planar YUV420)	Y	MD	Y	N	Y	N	Y	N	Y	N
	NV12 (semi-planar YUV420)	Y	MD	Y	N	Y	N	Y	N	Y	N
	NV16 (semi-planar YUV422)	Y	MD	Y	N	Y	N	Y	N	Y	N
	8-bit color index	N	N	Y	N	Y	N	N	N	Y	N
	1-bit monochrome	Y	N	N	N	N	N	N	N	N	N

Table continues on the next page...

Table 14-3. Composition and 2D Hardware Features (continued)

Feature	Support
Alpha blending modes	Java2 Porter-Duff, Chroma Key, Pattern Mask
Alpha blending during filter blit	Not supported
Transparency	Supported by monochrome mask, chroma key or pattern mask
Dithering	2D dither support
Image scaling	Filter Blit FIR resampling filter with up to 9 taps, One Pass Filter with 3/5 taps
Rotation	90 / 180 / 270 degrees with mirror
Clipping	one clipping rectangle supported for all bit blit primitives.
Text rendering	Monochrome expansion
Rendering size	32k x 32k raster 2D coordinate system
Compression	Lossless, tiled compression support

14.2.2.4 Composition and 2D Operations and Features

14.2.2.4.1 Line

The LINE operation draws a line. The coordinates for two points are given - start point and end point and the end point is not drawn.

Lines are rendered using the Bresenham algorithm. The Bresenham algorithm has the advantage of using integer arithmetic and has no accumulation of rounding errors.

In the case of line, only ROP2 and ROP4 are supported. It operates on pattern and destination. The pattern should have a transparency mask in order to use ROP4.

Clipping is supported for lines on a per pixel basis.

the figure below shows the operation in more detail.

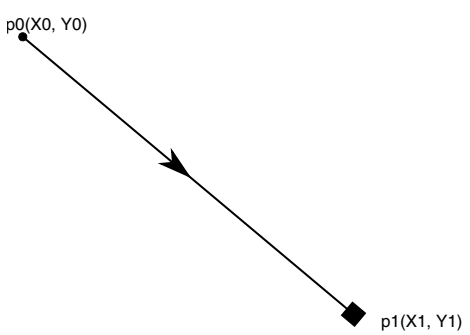


Figure 14-3. Line

14.2.2.4.2 Rectangle Fill and Clear

Rectangle fill suffuses a rectangle area with a given color. Essentially rectangle fill is a pattern fill, where an 8x8 pattern is initialized with the specified color. It supports ROP2 and ROP4 with the pattern and destination as its inputs. If ROP4 is used, the pattern should have a transparency mask.

Clear is similar to rectangle fill except that it does not use a pattern. A 32-bit clear value with 4-bit byte mask is used to fill the entire rectangle area.

Both rectangle fill and clear support clipping, which is performed on a per primitive basis.

14.2.2.4.3 Bit Blit

Bit blit transfers data from one area of a memory (source) to another area of the memory (destination). The source and destination can be from the same or from different memory locations. Both source and destination must be described by a rectangular area. The source and destination rectangles can be the same size (most bit blits are of this nature) or they can be different sizes, in which case the operation becomes a stretch or shrink blit.

Bit blit supports ROP2, ROP3, and ROP4 which includes source, destination and pattern, and an optional transparency color.

Clipping can be performed on a primitive basis.

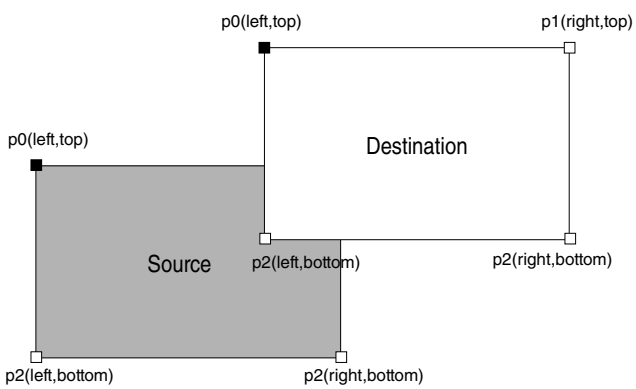


Figure 14-4. Bit Blit

14.2.2.4.4 Stretch Blit

The stretch blit primitive performs a bit blit operation with stretch or shrink. The modified Bresenham algorithm is used to generate corresponding coordinates for fast stretching. The stretch factor is specified in a 15.0 fixed-point format. Stretch blit is not

allowed to overlap, that is, no part of source and destination can share any piece of memory. Non-stretch blits can overlap. For stretch blit clipping is performed on a per pixel basis.

14.2.2.4.5 Monochrome Expansion and Mask Blit

Monochrome expansion and mask blit are different operations, although both use the bit stream from the command buffer. And both can be the source for ROP4 source selection. This means that each output pixel can be a combination of source, pattern, monochrome mask (for masked blits) and destination.

Monochrome expansion

For monochrome expansion, the bit from the stream is used to switch on/off a solid color that is defined in a register. This mechanism enables the use of just one bit per pixel to represent colors. In effect, the MONO EXPANSION primitive increases color representation from one bit per pixel to multiple bits per pixel. A typical application for mono color is font drawing.

Monochrome expansion does not support overlapping of the source and destination. It is the responsibility of the driver to make sure that the command will never be executed on overlapping source and destination.

Mask Blit

For mask blit, the bit from the stream is used to toggle on/off a color in the source frame buffer. Mask blit takes its color source from memory and its monochrome mask from the command stream. Clipping is supported and is performed on a per pixel basis.

14.2.2.4.6 YUV Output for 2D Operations

YUV output is available for packed or planar YUV input formats per the following table.

Table 14-4. YUV Output Support for Packed or Planar YUV Source per 2D Operation

2D Operation	Destination
Bit blit	Packed YUV
One Pass Filter	Packed YUV
Multi-source	Packed YUV (no scaling)
Multi-destination	Planar YUV
Stretch blit	NO
Two Pass Filter	NO

14.2.2.4.7 ARGB Data Conversion

The pixels read from source or destination will be expanded into A8R8G8B8 format to maintain lossless pixel operations. The resulting pixels will be converted into the destination format.

14.2.2.4.8 YUV to RGB Conversion

YUV data can be converted into 8-bit per component RGB format at the output of the cache only. Once converted, there is no way back to YUV format. GPU supports BT.601 and BT.709 YUV to RGB color conversion standards.

In BT.601, the YUV to RGB conversion is done using the following approximation:

```
16 ≤ Y ≤ 235
16 ≤ U ≤ 240
16 ≤ V ≤ 240

A = Y – 16 B = U – 128 C = V – 128
R = clip((298*A + 410*C + 128) >> 8)
G = clip((298*A – 101*B – 209*C + 128) >> 8)
B = clip((298*A + 519*B + 128) >> 8)
```

The Y, U and V components are clamped prior to the conversion. Y is clamped between 16 and 235, inclusively. U and V are clamped between 16 and 240, inclusively.

In BT.709, the R, G, B equations are slightly changed to:

```
R = clip((298*A + 461*C + 128) >> 8)
G = clip((298*A – 55*B – 137*C + 128) >> 8)
B = clip((298*A + 543*B + 128) >> 8)
```

14.2.2.4.9 Color Index Input Conversion Support

Color index is supported for source data only. A look-up table with 256 entries is provided for indexing the data. The table is fully programmable. The conversion is done when pixels are read out of the cache.

14.2.2.4.10 Source/Destination Pre-multiply Support

The GPU supports pre-multiply source alpha or global alpha, or source global color for global colorizing.

On destination, the GPU supports destination pre-multiply and destination alpha.

On destination, the GPU supports destination pre-multiply destination alpha, destination de-multiply alpha.

Table 14-5. Pre-multiplied modes

Microsoft Alpha Blending	Equation
S:Fix Alpha	D.C=A*S.C+(1-A)*D.C
	D.A=A*S.A+(1-A)*D.A
S:Per-Pixel	D.C=S.C+(1-S.A)*D.C
	D.A=S.A+(1-S.A)*D.A
S:Fix+Per-Pixel	D.C=A*S.C+(1-S.A)*D.C
	D.A=A*S.A+(1-S.A)*D.A
S:Per-Pixel	D.C=S.A*S.C+(1-S.A)*D.C
	D.A=S.A+(1-S.A)*D.A
S:Fix+Per-Pixel	D.C=A*S.A*S.C+(1-S.A)*D.C
	D.A=A*S.A+(1-S.A)*D.A

NOTE

D: Destination

S: Source

A: Alpha

C: Color

14.2.2.4.11 Source/Destination Pre-multiply Support

The GPU supports alpha blending together with ROP. Alpha blending function is performed on ROP function result source.

The general alpha blending equations are:

$$Cd = Fs * Cs' + Fd * Cd'$$
$$Ad = Fs * As'' + Fd * Ad''$$

Where, Cs' is the source color component (adjusted for NPM if necessary)

Cd' is the destination color component (adjusted for NPM if necessary)

As'' is the modified source alpha component

Ad'' is the modified destination alpha component

Fs is fraction of the source that contributes to the final value

Fd is fraction of the destination that contributes to the final value

The blending is done in four logical stages:

1. Transparent/opaque conversion - In this stage, the incoming alpha (source or destination independently) can be inverted if needed to match the internal alpha rule. Internally, an alpha of 0 means transparent, while an alpha of “0xFF” means opaque. External content might follow the opposite rule. The output of the block is either As (Ad for destination) or 1-As (1-Ad for destination).
2. Global value substitution - A global alpha value from a register can be used to substitute or scaled the incoming alpha. An incoming alpha As can pass-through, be directly substituted by Ags (global alpha) or scaled by the global alpha value (As * Ags). The source and destination have distinct global alpha values.
3. Blending factor generation - At this stages, the blending factors are generated (refer to table below). Each alpha can take the values 0, 1, A or 1-A depending on the blending mode.
4. Final blending - This is the final stage which implements the operations described by equations.
5.

The stages are shown in the figure below:

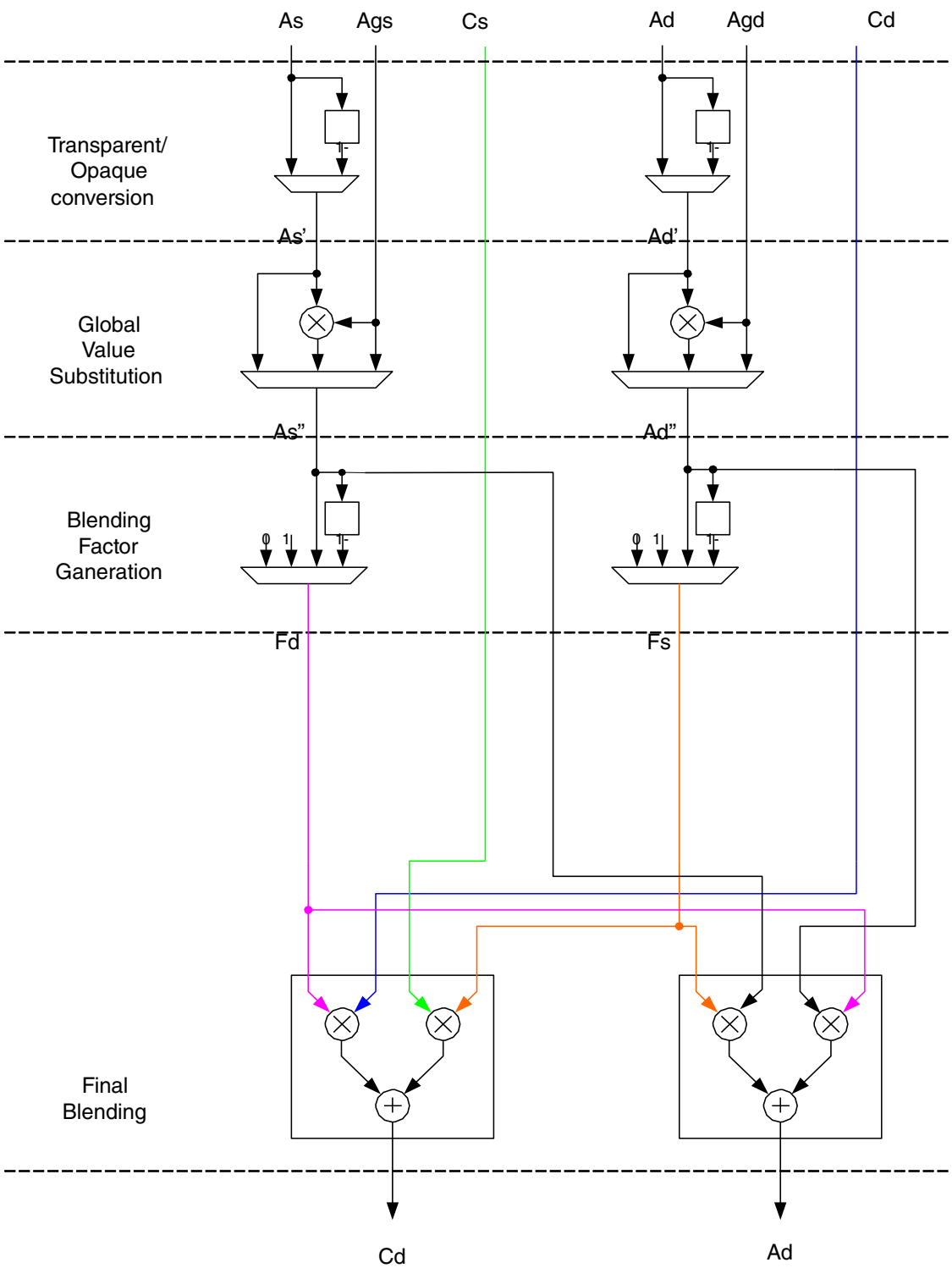


Figure 14-5. Alpha Blender for One Pixel

The fractions take the values described in the following table, depending on the blending mode:

Table 14-6. Blending Modes Fractions Description

Blending Mode	Fs	Fd
Clear	0	0
SRC	1	0
DST	0	1
SRC_OVER	1	1 – As”
DST_OVER	1 – Ad”	1
SRC_IN	Ad”	0
DST_IN	0	As”
SRC_OUT	1 – Ad”	0
DST_OUT	0	1 – As”
SRC_ATOP	Ad”	1 – As”
DST_ATOP	1 – Ad”	As”
XOR	1 – Ad”	1 – As”

14.2.2.4.12 Dither

Color intensity at any point of an image can be thought of as a real number between 0.0 and 1.0 at each location. However, in reality only a discrete number of intensity levels can be represented. This discretization leads to a total error in the color intensities of an image equal to the square root of the sum of the squares from each point.

This error results in contouring. In an image where intensities change slowly, this will cause noticeable jumps. Dithering can be used to diffuse the intensity across neighboring pixels.

In dithering mode pixels are scanned in order, and errors in calculating a pixel's intensity are distributed (i.e., diffused) to neighboring pixels to keep the overall intensity of the image closer to the input intensity.

Dithering is supported for 16-bit color destination formats. The dither table is programmable by software.

14.2.2.4.13 Multi-Source Blending

Up to 8 sources are supported. Programmable block size guarantees cache efficiency so each source is read once and each destination is written once. It supports 90, 180, 270 degree destination and source rotation and mirror with different block size to have high

cache efficiency. It also supports ROP2, ROP3, and ROP4 which includes source, destination and pattern, and an optional transparency color. Alpha blending between every source is supported.

14.2.2.4.14 GPU Cache Management

SW cache flush is supported to flush the GPU cache to memory. Auto-flush of GPU cache is also supported. SW sets up the auto-flush interval, and HW will do the cache flush automatically at programmable intervals.

14.3 3D Graphics Processing Unit (GPU3D)

14.3.1 Overview

The graphics processing unit (GPU) provides high performance, high quality graphics, low power consumption, and the smallest silicon footprint for its class. The dynamic power consumption is minimized by extensive use of multi-level hierarchical clock gating.

An optimized software stack, complete software development tools, and a growing application ecosystem are supported by a robust graphics pipeline designed for industry-standard APIs, and with full support for Android, Linux, and Windows embedded development platforms. The GPU supports the following graphics APIs:

- OpenGL ES 2.0/1.1
- EGL 1.4
- OpenGL 2.1
- OpenVG 1.1

14.3.1.1 Block Diagram

A block diagram and a description of the functional blocks of the complete graphics pipeline is given below.

- Host Interface - Allows GCCORE to communicate with external memory and the CPU through AXI or AHB bus. In this block data crosses clock domain boundaries.
- Memory Controller - Internal memory management unit that controls the block-to-host memory request interface.
- Graphics Pipeline Front End - Inserts high level primitives and commands into the graphics pipeline.

- Ultra-threaded Unified Shader - SIMD processor that performs as both vertex shader and fragment shader. When used as a vertex shader it performs geometry transformations and lighting computations. When used as a fragment shader it applies texture data and computes color values for each pixel. The GPU has one (1) such vec4 shader.
- 3D Rendering Engine - Converts triangles and lines into pixels. Computes slopes of color attributes and texture coordinates. Performs clipping.
- Texture Engine - Retrieves texture information from memory upon request by the fragment shader. Performs interpolation and filtering, and transfers the computed value to the fragment shader or the vertex shader. This GCCORE texture unit can process one pixel or one vertex /cycle.
- Pixel Engine/Resolve - Pixel engine does alpha blending and visible surface determination. Resolve does tiling and de-tiling. This GCCORE Pixel Engine can process one pixel/cycle.

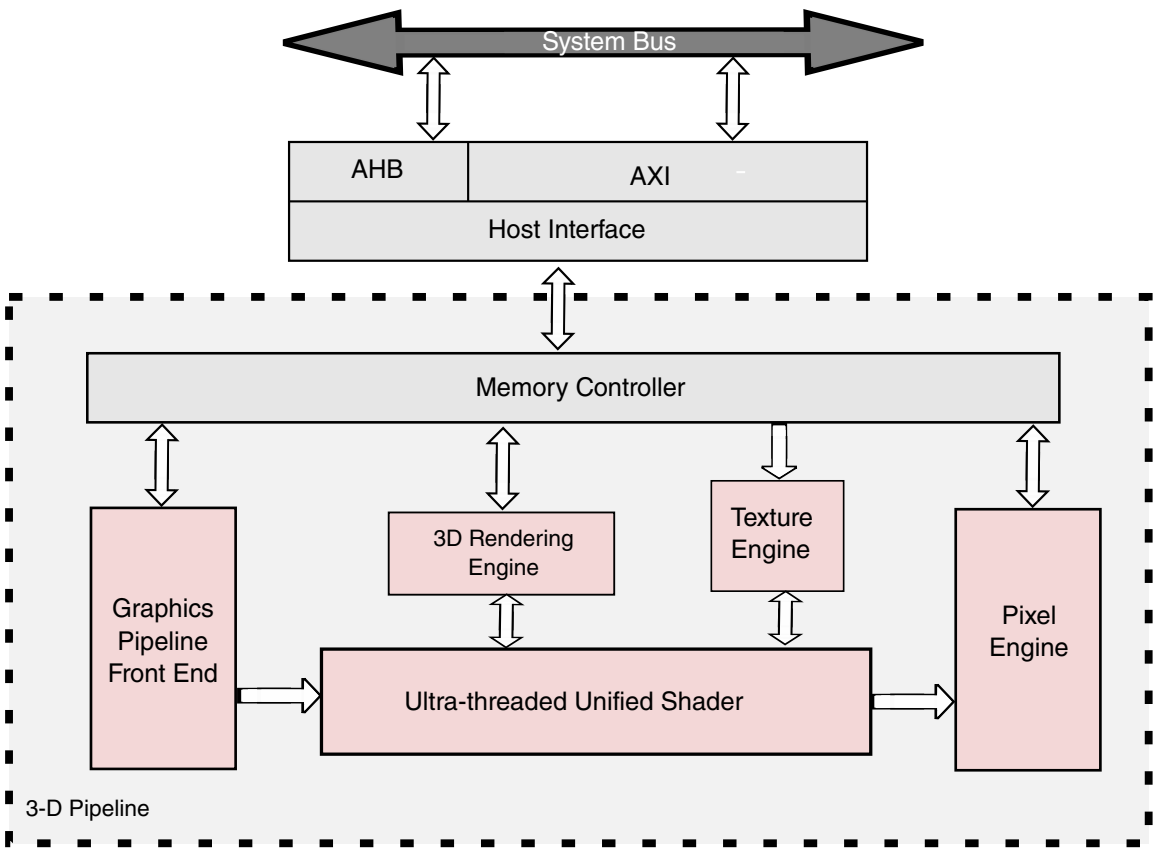


Figure 14-6. GPU Block Diagram

14.3.2 GPU Features

The features of the GPU 3D unit include:

- OpenGL ES 2.0 / 1.1 compliance, including extensions; OpenVG 1.1
- IEEE 32-bit floating-point pipeline
- Ultra-threaded, unified vertex and fragment (pixel) shaders
- Low bandwidth at both high and low data rates
- Low CPU loading
- Up to 12 programmable elements per vertex
- Dependent texture operation with high-performance
- Alpha blending
- Depth and stencil compare
- Point sampling, bi-linear sampling, tri-linear filtering, and cubic textures
- Resolve and fast clear
- 8k x 8k texture size and 8k x 8k rendering target
- 4 Vertex DMA streams
- MMU functionality supported

The following table describes API support and select architectural features of the GPU:

Table 14-7. GPU Architecture Features

Feature	GPU Support
Primary API	OpenGL ES 2.0 / 1.1
Additional APIs	OpenVG 1.1 DirectX 11 (9_3) OpenGL 2.1
Other graphics support	EGL 1.4
Drivers	OpenGL ES 2.0 / 1.1 OpenVG 1.1 EGL 1.4 DirectX 11 (9_3) OpenGL 2.1
Operating systems	Windows Embedded Compact 7, Embedded CE 6 Embedded Linux Android 5.x / 4.x / 3.x
Z (depth)	Early Z support included
Stencil	Early stencil support included
GLSL ES Shader languages	1.0
Shader model compatibility	SM3.0
Shader types and execution units	One programmable Scalable Ultra-threaded Unified Shader (SIMD4: ctl-flow,tx-load) one instruction issue per shader per clock; IEEE 32-bit floating-point pipeline supports long shader instructions
FSAA anti-aliasing mechanisms	High quality MSAA 4x

14.3.2.1 GPU Host and Memory Interface Features

Table 14-8. GPU host and memory interface features

Feature	GPU Support
AHB interface	32-bit
AXI interface	1 64-bit AXI interface for external memory access
Virtual memory support	Yes
Code and data memory location restrictions	Unrestricted; arbitrary memory reads and writes
Physical address	32 bits
Resource locks with CPU	Semaphore lock
TLB	4 cache lines per requestor

14.3.2.2 GPU Power Management Features

Table 14-9. Power management features

Feature	GPU Support
Low power CMOS technology compatible	Yes
Automatic clock gating of flip flops and rams	Yes
Global clock gating of unused macro blocks	Yes
Software controlled effective clock frequency without changing the PLL	Yes

14.3.2.3 GPU Command Processor Features

Table 14-10. GPU command processor features

Feature	GPU Support
Counters	Variety of hardware counters for performance profiling

14.3.2.4 Texture processing

Table 14-11. Texture Processing Features

Feature	GPU Support
Fixed-point input texture formats	A8, L8, I8, A8L8, ARGB4, XRGB4, ARGB8, XRGB8, ABGR8, XBGR8, R5G6B5, A1RGB5, X1RGB5, YUY2, UYVY, D16, D24X8, A8_OES, DXT1, DXT2, DXT3, DXT4, DXT5, ETC1. All fixed-point formats are filtered.

Table continues on the next page...

Table 14-11. Texture Processing Features (continued)

<table>
<tr><th>Feature</th><th>GPU Support</th></tr>
<tr><td></td><td>
<table>
<tr><th>Bits</th><th>Format</th><th>Alpha/X/s</th><th>R</th><th>B</th><th>G</th></tr>
<tr><td>16</td><td>ARGB4444</td><td>4</td><td>4</td><td>4</td><td>4</td></tr>
<tr><td>16</td><td>XRGB4444</td><td>4 don't care</td><td>4</td><td>4</td><td>4</td></tr>
<tr><td>16</td><td>ARGB1555</td><td>1</td><td>5</td><td>5</td><td>5</td></tr>
<tr><td>16</td><td>XRGB1555</td><td>1 don't care</td><td>5</td><td>5</td><td>5</td></tr>
<tr><td>16</td><td>RGB565</td><td>0</td><td>5</td><td>6</td><td>5</td></tr>
<tr><td>32</td><td>ARGB8888</td><td>8</td><td>8</td><td>8</td><td>8</td></tr>
<tr><td>32</td><td>XRGB8888</td><td>8 don't care</td><td>8</td><td>8</td><td>8</td></tr>
<tr><td>32</td><td>ABGR8888</td><td>8</td><td>8</td><td>8</td><td>8</td></tr>
<tr><td>32</td><td>XBGR8888</td><td>8 don't care</td><td>8</td><td>8</td><td>8</td></tr>
</table>
<table>
<tr><th>Planes</th><th>Format</th><th>Mode</th><th>Y</th><th>U</th><th>V</th><th>UV</th><th>YUYV</th><th>UYVY</th></tr>
<tr><td>1</td><td>YUY2</td><td>4:2:2</td><td></td><td></td><td></td><td></td><td>1</td><td></td></tr>
<tr><td>1</td><td>UYVY</td><td>4:2:2</td><td></td><td></td><td></td><td></td><td></td><td>1</td></tr>
</table>
</td></tr>
<tr><td>Additional texture formats supported through Resolve conversion</td><td>Resolve converts these planar formats to YUV4:2:2 packed:
<table>
<tr><th>Planes</th><th>Format</th><th>Mode</th><th>Y</th><th>U</th><th>V</th><th>UV</th><th>YUYV</th><th>UYVY</th></tr>
<tr><td>3</td><td>YV12</td><td>4:2:0</td><td>1</td><td>1</td><td>1</td><td></td><td></td><td></td></tr>
<tr><td>2</td><td>NV12</td><td>4:2:0</td><td>1</td><td></td><td></td><td>1</td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>Texture compression</td><td>4 bits and 8 bits per texel</td></tr>
<tr><td>Compressed texture formats</td><td>DXT1, DXT2, DXT3, DXT4, DXT5, ETC1. All compressed formats are filtered.</td></tr>
<tr><td>Compressed Texture Supertile</td><td>Supported</td></tr>
<tr><td>Texture size maximum</td><td>8k x 8k</td></tr>
<tr><td>Addressing modes</td><td>wrap, mirror, clamp</td></tr>
<tr><td>Mipmap support</td><td>14 mipmap levels; programmable LOD biasing and replacement</td></tr>
<tr><td>Shadow texture</td><td>Depth texture PCF filtering</td></tr>
<tr><td>Texture coordinate fraction bits</td><td>5 bits</td></tr>
<tr><td>Texture sampler units</td><td>12 samplers, indexable</td></tr>
<tr><td>Textures per fragment maximum</td><td>8 texture samplers</td></tr>
<tr><td>Dependent texture operation</td><td>High performance; unlimited dependent texture reads</td></tr>
<tr><td>Dependent tx per fragment max, relative sampling</td><td>No Limit</td></tr>
<tr><td>Texture repeat max</td><td>256</td></tr>
<tr><td>Texture types</td><td>2D cube map, 1D, projected, depth, bump map, displacement map</td></tr>
<tr><td>Texture filters</td><td>Point sample, bi-linear, tri-linear</td></tr>
</table>

Table continues on the next page...

Table 14-11. Texture Processing Features (continued)

Feature	GPU Support
Texture component mapping: D3D, OGL ES options	Supports both D3D and OES options
Texture size types	Power-of-2, Non-square texture support

14.3.2.5 Fragment Processing and Render Targets

Table 14-12. Fragment Processing and Render Target Features

<table>
<tr><th>Feature</th><th>GPU Support</th></tr>
<tr><td>FSAA mechanisms</td><td>MSAA 4x, SSAA 4x</td></tr>
<tr><td>Fragment color, alpha, Z, stencil precision</td><td>
<table>
<tr><th>Bits</th><th>Format</th><th>Alpha</th><th>R</th><th>B</th><th>G</th></tr>
<tr><td>16</td><td>ARGB4444</td><td>4</td><td>4</td><td>4</td><td>4</td></tr>
<tr><td>16</td><td>XRGB4444</td><td>4 don't care</td><td>4</td><td>4</td><td>4</td></tr>
<tr><td>16</td><td>ARGB1555</td><td>1</td><td>5</td><td>5</td><td>5</td></tr>
<tr><td>16</td><td>XRGB1555</td><td>1 don't care</td><td>5</td><td>5</td><td>5</td></tr>
<tr><td>16</td><td>RGB565</td><td>0</td><td>5</td><td>6</td><td>5</td></tr>
<tr><td>32</td><td>ARGB8888</td><td>8</td><td>8</td><td>8</td><td>8</td></tr>
<tr><td>32</td><td>XRGB8888</td><td>8 don't care</td><td>8</td><td>8</td><td>8</td></tr>
<tr><td>32</td><td>ABGR8888</td><td>8</td><td>8</td><td>8</td><td>8</td></tr>
<tr><td>32</td><td>XBGR8888</td><td>8 don't care</td><td>8</td><td>8</td><td>8</td></tr>
</table>
<table>
<tr><th>Bits</th><th>Format</th><th>Depth</th><th>Stencil</th></tr>
<tr><td>16</td><td>D16</td><td>16</td><td>0</td></tr>
<tr><td>32</td><td>D24S8</td><td>24</td><td>8</td></tr>
</table>
</td></tr>
<tr><td>Fragment storage</td><td>16-bit color and Z, 32-bit color and Z for each fragment. Lossless compression, no storage reduction.</td></tr>
<tr><td>Alpha support</td><td>Individual fragment alpha masking</td></tr>
</table>

14.3.2.6 Destination / Alpha Blending

Table 14-13. Destination / Alpha Blending Features

<table>
<tr><th>Feature</th><th>GPU Support</th></tr>
<tr><td>Destination color formats</td><td>
<table>
<tr><th>Bits</th><th>Format</th><th>Alpha</th><th>R</th><th>B</th><th>G</th></tr>
<tr><td>16</td><td>ARGB4444</td><td>4</td><td>4</td><td>4</td><td>4</td></tr>
<tr><td>16</td><td>XRGB4444</td><td>X</td><td>4</td><td>4</td><td>4</td></tr>
</table>
</td></tr>
</table>

Table 14-13. Destination / Alpha Blending Features

Feature	GPU Support					
	16	ARGB1555	1	5	5	5
	16	XRGB1555	X	5	5	5
	16	RGB565	0	5	6	5
	32	ARGB8888	8	8	8	8
	32	XRGB8888	X	8	8	8
	32	ABGR8888	8	8	8	8

14.3.3 Usage Mode

The GPU should be programmed through the NXP provided driver. NXP does not provide support for software that directly programs the GPU registers. APIs for programming the GPU through the software driver are described in separate driver documentation.

Chapter 15
Video Processing Unit (VPU)

15.1 VPU G1 (VPU_G1)

15.1.1 Overview

This block details the VPU G1 hardware based decoders and API. The decoders are able to decode H.264, VP8. The decoders conform to the H.264 Baseline, Main and High profile, and VP8 Main Profile

NOTE

The overall performance and capabilities of the decoders is system dependent. Formats and features that are presented depend on the chip configuration.

15.1.2 Features (G1)

The API is implemented on top of the VPU G1 decoder H.264 and VP8. The features of the decoders are noted in the following sections.

15.1.2.1 Decoder Features

The VPU G1 decoder has the following features:

- H.264 baseline profile, levels 1-4.1 decoding
  - Byte stream input
  - NAL unit input
- VP8 decoding
- Video post-processing features
  - Frame rotation 90 degrees left/right
  - Frame mirroring horizontally/vertically
  - Frame cropping

- Frame conversion from YCbCr formats to 16-bit or 32-bit RGB formats
- Frame scaling with maximum up-scaling factor of 3
- Two rectangular or alpha blending masks for output frame

The post-processing features can be used in pipeline with the decoder. The post-processing features can also be used as stand-alone, without performing any decoding.

- H.264 main and high profile, levels 1-4.1 decoding
- VP8 decoding

15.1.3 Video Frame Storage Format

This chapter describes the different input and output picture storage formats supported by the decoder and post-processor. Notice that some of the formats are only used for input or output.

15.1.3.1 YCbCr 4:2:0 Planar Format

In the planar format each video sample component forms one memory plane. The luminance and both chrominance planes must be stored in a linear and contiguous memory block as shown in Figure 3. The luminance samples are stored in raster-scan order (Y0Y1Y2Y3Y4….). The chrominance samples are stored in two planes also in raster scan order (Cb0Cb1Cb2Cb3… and Cr0Cr1Cr2Cr3…).

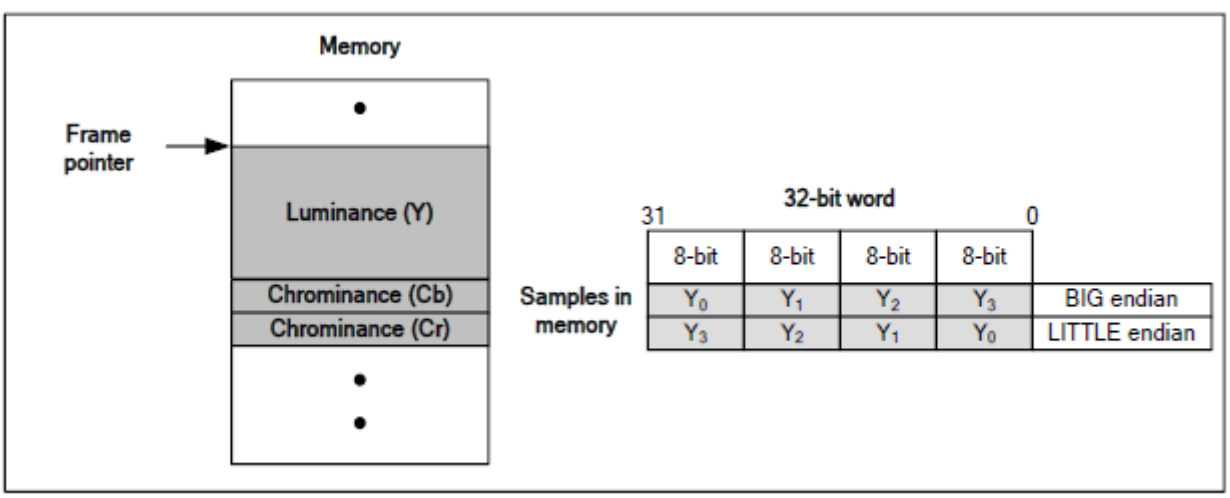


Figure 15-1. YCBCR 4:2:0 Planar Format External Memory Usage

The YCbCr 4:2:0 planar format is supported only as an input format for the postprocessor. In this format each pixel takes 12 bits of memory.

15.1.3.2 YCbCr 4:2:0 Semi-Planar Format

In semi-planar YCbCr 4:2:0 format the luminance samples form one plane in memory, and chrominance samples form another. The luminance and chrominance planes must be stored in a linear and contiguous memory block as presented in the following figure. The luminance pixels are stored in raster-scan order Y0Y1Y2Y3Y4…. The interleaved chrominance CbCr samples are stored in raster-scan order in memory as Cb0Cr0Cb1Cr1Cb2Cr2Cb3....

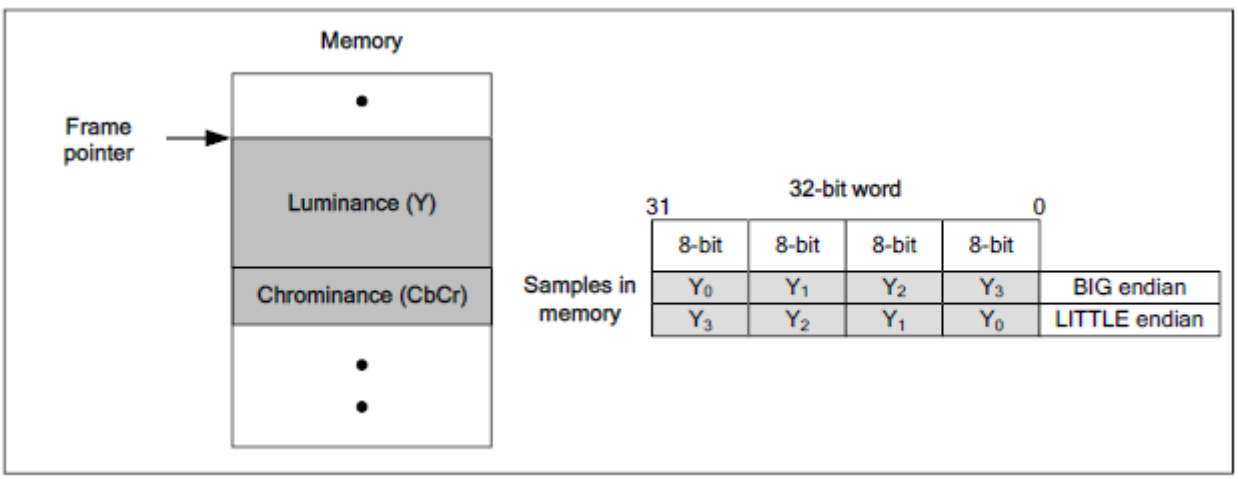


Figure 15-2. YCBCR 4:2:0 Planar Format External Memory Usage

The YCbCr 4:2:0 semi-planar format is supported both as an input and as an output format for the post-processor. In this format each pixel takes 12 bits of memory. As the chrominance components are interleaved, the bus load caused by this format can be slightly lower than with the planar format due to the reduced amount of non-sequential memory addressing.

15.1.3.3 YCbCr 4:2:2 Interleaved Format

In the interleaved YCbCr 4:2:2 format the pixel samples form a single plane in which the data has to be stored linearly and contiguously as shown in Figure 5. The pixel data is in raster scan order and the chrominance samples are interleaved between the luminance samples as Y0Cb0Y1Cr0Y2Cb1Y3Cr1….

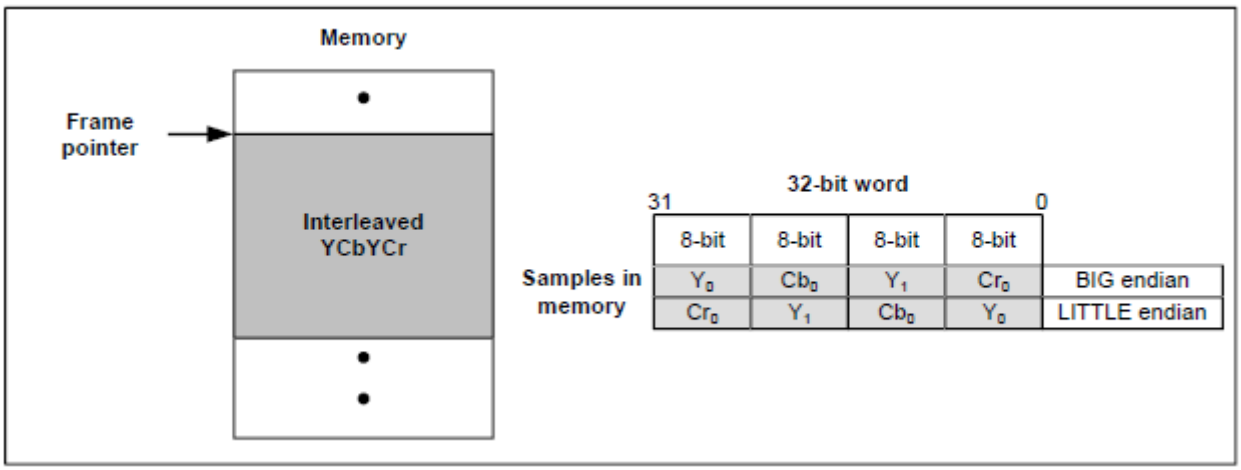


Figure 15-3. YCBCR 4:2:2 Interleaved Format External Memory Usage

The YCbCr 4:2:2 interleaved format is supported both as an input and as an output format for the post-processor. In this format each pixel takes 16 bits of memory. Although there is more data to be transferred than in the 4:2:0 formats, the interleaved format is bus effective as there are no non-sequential memory accesses caused by the plane changes.

15.1.3.4 G2 Decoder Tiled 4x4 Format

The output picture of the decoder is in semi-planar YCbCr 4:2:0 4x4 tiled format, i.e. luminance data forms one plane in memory, and chrominance data forms another. The distinction from raster scan format is that the output picture is grouped into tiles, which are then stored linearly and contiguously in the memory. The chrominance tiles have to be stored right after the luminance tiles in external memory as shown in Figure 6.

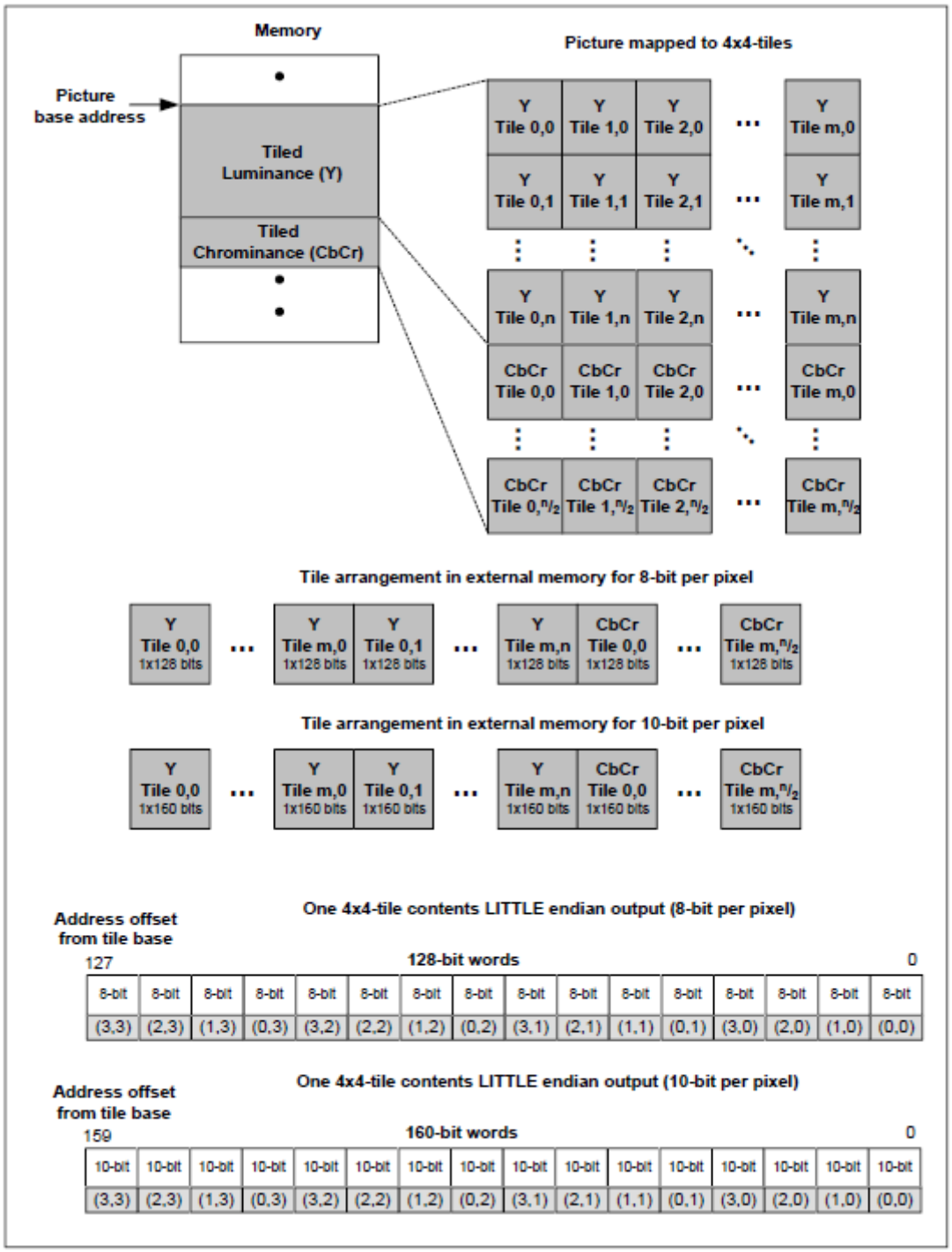


Figure 15-4. Tiled 4x4 Format External Memory Usage

15.1.3.5 RGB 16bpp Format

In this format each pixel is represented by 16 or less bits containing the red, blue and green samples. There are several 16bpp formats which use different number of bits for each sample. For example the RGB 5-5-5 format uses 5 bits for each sample and 1 bit is left unused or can represent a transparency flag, where RGB 5-6-5 uses 6 bits for the G sample and 5 bits for R and B samples. Common for all 16bpp types is that two pixels fit into one 32-bit space.

The data has to be stored linearly and contiguously in the memory as shown in Figure 7. The order of the sample bytes is always defined with a 16 bit word which will define the byte order in the memory. The pixel samples are stored in raster-scan order.

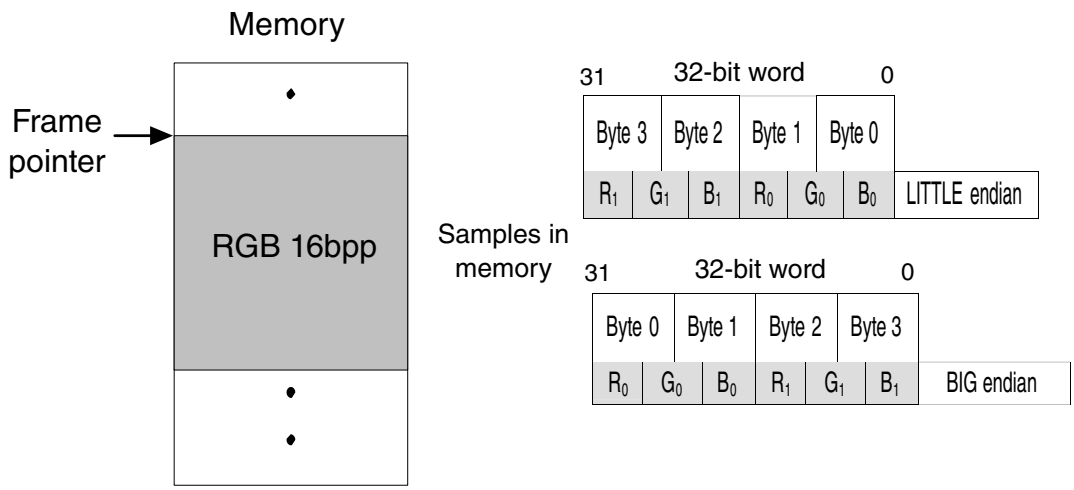


Figure 15-5. RGB 16bpp Format External Memory Usage

The RGB 16bpp formats are supported only as post-processor output formats.

15.1.3.6 RGB 32bpp Format

Any RGB format that has its pixels represented by more than 16 bits each is considered to be of 32bpp type. Typically in this format each pixel is represented by three bytes containing a red, blue and green sample and a 4th byte which can be empty or hold an alpha blending value. Common for all 32bpp types is that only one pixel fit into one 32-bit space.

The data has to be stored linearly and contiguously in the memory as shown in Figure 8. The order of the sample bytes is always defined on a 32 bit word and that will define the byte order in the memory (for example, ARGB order means that on a little endian system

the B sample will be stored in the lowest offset byte followed by the G, R and A samples. In a big endian system the A sample will be on the lowest offset byte). The pixel samples are stored in raster-scan order.

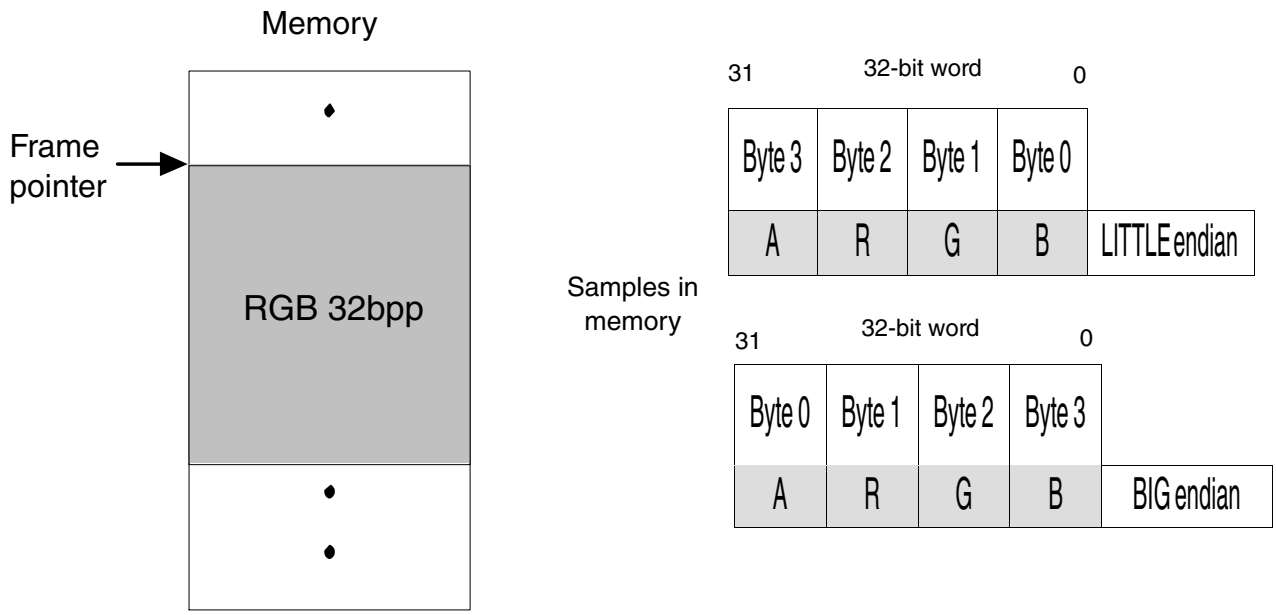


Figure 15-6. RGB 32bpp Format External Memory Usage

15.1.4 G1 Register Decoder

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
swreg0										
31:16	-	x	x	x	x	x	x	x	x	x
15:12	-	x	x	x	x	x	x	x	x	x
11:4	-	x	x	x	x	x	x	x	x	x
3	-	x	x	x	x	x	x	x	x	x
2:0	-	x	x	x	x	x	x	x	x	x
swreg1										
31:25	-									
24	sw_dec_pic_inf	x	x							
23:19	-									
18	sw_dec_timeout	x	x	x	x	x	x	x	x	x
17	sw_dec_slice_int			x						
16	sw_dec_error_int	x	x	x	x	x	x	x	x	x
15	sw_dec_aso_int	x							x	

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
14	sw_dec_buffer_int	x	x	x	x	x	x	x	x	x
13	sw_dec_bus_int	x	x	x	x	x	x	x	x	x
12	sw_dec_rdy_int	x	x	x	x	x	x	x	x	x
11	sw_dec_abort_int	x	x	x	x	x	x	x	x	x
10:9	-									
8	sw_dec_irq	x	x	x	x	x	x	x	x	x
7:6	-									
5	sw_dec_abort_e	x	x	x	x	x	x	x	x	x
4	sw_dec_irq_dis	x	x	x	x	x	x	x	x	x
3:1	-									
0	sw_dec_e	x	x	x	x	x	x	x	x	x
swreg2										
31:24	sw_dec_axi_rd_id	x	x	x	x	x	x	x	x	x
23	sw_dec_timeout_e	x	x	x	x	x	x	x	x	x
22	sw_dec_strswap32_e	x	x	x	x	x	x	x	x	x
21	sw_dec_strendian_e	x	x	x	x	x	x	x	x	x
20	sw_dec_inswap32_e	x	x	x	x	x	x	x	x	x
19	sw_dec_outswap32_e	x	x	x	x	x	x	x	x	x
18	sw_dec_data_disc_e	x	x	x	x	x	x	x	x	x
18	sw_dec_2chan_dis	x	x	x	x	x	x	x	x	x
17	sw_tiled_mode_msb	x	x		x	x	x	x	x	x
17	sw_dec_out_tiled_e	x	x	x	x	x				
16:11	sw_dec_latency	x	x	x	x	x	x	x	x	x
10	sw_dec_clk_gate_e	x	x	x	x	x	x	x	x	x
9	sw_dec_in_endian	x	x	x	x	x	x	x	x	x
8	sw_dec_out_endian	x	x	x	x	x	x	x	x	x
7:5	sw_priority_mode	x	x	x	x	x				
7	sw_tiled_mode_lsb	x	x	x	x	x	x	x	x	x

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
6	sw_dec_adv_pre_dis	x	x		x	x	x	x	x	x
5	sw_dec_scmd_dis	x	x	x	x	x	x	x	x	x
4:0	sw_dec_max_burst	x	x	x	x	x	x	x	x	x
swreg3										
31:28	sw_dec_mode	x	x	x	x	x	x	x	x	x
27	sw_rlc_mode_e	x	x							
26	sw_skip_mode		x						x	x
25	sw_divx3_e		x							
24	sw_pjpeg_e			x						
23	sw_pic_interlace_e	x	x		x	x		x		x
22	sw_pic_fieldmode_e	x	x		x	x				x
21	sw_pic_b_e		x		x	x		x		x
20	sw_pic_inter_e		x		x	x	x		x	x
19	sw_pic_topfield_e	x	x		x	x				x
18	sw_fwd_interlace_e		x		x	x				
17	sw_sorenson_e		x							
16	sw_ref_topfield_e				x					
15	sw_dec_out_dis	x	x	x	x	x	x	x	x	x
14	sw_filtering_dis	x	x		x	x	x	x		x
13	sw_webp_e								x	
13	sw_mvc_e	x								
13	sw_pic_fixed_quant				x					x
12	sw_write_mvs_e	x	x		x	x	x	x	x	x
11	sw_reftopfirst_e				x					
10	sw_seq_mbaff_e	x								
9	sw_picord_count_e	x								
8	sw_dec_ahb_hlock_e	x	x	x	x	x	x	x	x	x
7:0	sw_dec_axi_wr_id	x	x	x	x	x	x	x	x	x
swreg4										
31:23	sw_pic_mb_width	x	x	x	x	x	x	x	x	x
22:19	sw_mb_width_off				x					

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
18:11	sw_pic_mb_height_p	x	x	x	x	x	x	x	x	x
10:7	sw_mb_height_off				x					
6	sw_alt_scan_e		x			x				
5	sw_topfieldfirst_e		x		x					
4:0	sw_ref_frames	x			x					
5:3	sw_pic_mb_w_ext			x					x	
2:0	sw_pic_mb_h_ext		x	x		x	x	x	x	
0	sw_pic_refer_flag									x
swreg5										
31:26	sw_strm_start_bit	x	x	x	x	x	x	x	x	x
25	sw_sync_marker_e		x	x	x					
24	sw_type1_quant_e	x	x							
23:19	sw_ch_qp_offset	x	x							
18:14	sw_ch_qp_offset2	x								
13:1	-	x								
0	sw_fieldpic_flag_e	x								
18:16	sw_intradc_vlc_thr		x							
15:0	sw_vop_time_incr		x							
24	sw_dq_profile				x					
23	sw_dqbi_level				x					
22	sw_range_red_frm_e				x					
21	-				x					
20	sw_fast_uvmc_e				x					
19:18	-				x					
17	sw_transdctab		x		x					
16:15	sw_transacfrm		x		x					
14:13	sw_transacfrm2		x		x					
12:10	sw_mb_mode_tab				x					
9:7	sw_mvtab				x					
6:4	sw_cbptab				x					

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
3:2	sw_2mv_blk_pat_tab				x					
1:0	sw_4mv_blk_pat_tab				x					
24	sw_qscale_type					x				
23:5	-					x				
4	sw_con_mv_e					x				
3:2	sw_intra_dc_prec					x				
1	sw_intra_vlc_tab					x				
0	sw_frame_pred_dct					x				
12:11	sw_jpeg_qtables			x						
10:8	sw_jpeg_mode			x						
7	sw_jpeg_filright_e			x						
6	sw_jpeg_stream_all			x						
5	sw_cr_ac_vlctable			x						
4	sw_cb_ac_vlctable			x						
3	sw_cr_dc_vlctable			x						
2	sw_cb_dc_vlctable			x						
1	sw_cr_dc_vlctable3			x						
0	sw_cb_dc_vlctable3			x						
23:18	sw_strm1_start_bit						x		x	
17	sw_huffman_e						x			
16	sw_multistream_e						x			
15:8	sw_boolean_value						x		x	
7:0	sw_boolean_range						x		x	
9:5	sw_alpha_offset									x
4:0	sw_beta_offset									x
swreg6										
31	sw_start_code_e	x			x					
30:25	sw_init_qp	x	x		x	x	x			x

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
24	sw_ch_8pix_ileav_e	x								
31:24	sw_stream_len_ext								x	
23:0	sw_stream_len	x	x	x	x	x	x	x	x	x
swreg7										
31	sw_cabac_e	x								
30	sw_blackwhite_e	x								
29	sw_dir_8x8_infer_e	x								
28	sw_weight_pred_e	x								
27:26	sw_weight_bipr_idc	x								
25	sw_avs_h264_h_ext	x								x
24:21	-	x								
20:16	sw_framenum_len	x						x		
15:0	sw_framenum	x								
31	sw_bitplane0_e				x					
30	sw_bitplane1_e				x					
29	sw_bitplane2_e				x					
28:24	sw_alt_pquant				x					
23:20	sw_dq_edges				x					
19	sw_ttmbf				x					
18:14	sw_pqindex				x					
13	sw_vc1_height_ext				x					
12	sw_bilin_mc_e				x		x		x	
11	sw_uniqp_e				x					
10	sw_halfqp_e				x					
9:8	sw_ttfrm				x					
7	sw_2nd_byte_emul_e				x					
6	sw_dquant_e				x					
5	sw_vc1_adv_e				x					
31:26	sw_dct1_start_bit								x	
25:20	sw_dct2_start_bit								x	
13	sw_ch_mv_res								x	
11:9	sw_init_dc_match0								x	

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
8:6	sw_init_dc_match1								x	
5	sw_vp7_version								x	
swreg8										
31	sw_const_intra_e	x								
30	sw_filt_ctrl_pres	x								
29	sw_rdpic_cnt_pres	x								
28	sw_8x8trans_flag_e	x								
27:17	sw_refpic_mk_len	x								
16	sw_idr_pic_e	x								
15:0	sw_idr_pic_id	x								
31:24	sw_mv_scalefactor				x					
23:19	sw_ref_dist_fwd				x					
18:14	sw_ref_dist_bwd				x					
17:14	sw_loop_filt_limit						x			
13	sw_variance_test_e						x			
12:10	sw_mv_threshold						x			
9:0	sw_var_threshold						x			
8	sw_divx_idct_e		x							
7:0	sw_divx3_slice_size		x							
31:30	sw_rv_profile							x		
29:28	sw_rv_osv_quant							x		
27:14	sw_rv_fwd_scale							x		
13:0	sw_rv_bwd_scale							x		
31:16	sw_init_dc_comp0								x	
15:0	sw_init_dc_comp1								x	
swreg9										
31:24	sw_pps_id	x								
23:19	sw_refidx1_active	x								
18:14	sw_refidx0_active	x								
7:0	sw_poc_length	x								
24	sw_icomp0_e				x					

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
23:16	sw_iscale0				x					
15:0	sw_ishift0				x					
23:0	sw_stream1_len						x		x	
31:2	sw_mb_ctrl_base	x	x							
12:0	sw_pic_slice_am							x		
27:24	sw_coeffs_part_am								x	
swreg10										
31:2	sw_diff_mv_base	x	x							
29:25	sw_pinit_rlist_f9	x								
24:20	sw_pinit_rlist_f8	x								
19:15	sw_pinit_rlist_f7	x								
14:10	sw_pinit_rlist_f6	x								
9:5	sw_pinit_rlist_f5	x								
4:0	sw_pinit_rlist_f4	x								
24	sw_icomp1_e				x					
23:16	sw_iscale1				x					
15:0	sw_ishift1				x					
31:2	sw_segment_base								x	
1	sw_segment_upd_e								x	
0	sw_segment_e								x	
swreg11										
31:2	sw_i4x4_or_dc_base	x	x							
29:25	sw_pinit_rlist_f15	x								
24:20	sw_pinit_rlist_f14	x								
19:15	sw_pinit_rlist_f13	x								
14:10	sw_pinit_rlist_f12	x								
9:5	sw_pinit_rlist_f11	x								
4:0	sw_pinit_rlist_f10	x								
24	sw_icomp2_e				x					
23:16	sw_iscale2				x					
15:0	sw_ishift2				x					
29:24	sw_dct3_start_bit								x	
23:18	sw_dct4_start_bit								x	
17:12	sw_dct5_start_bit								x	
11:6	sw_dct6_start_bit								x	
5:0	sw_dct7_start_bit								x	

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
swreg12										
31:0	sw_rlc_vlc_base	x	x	x	x	x	x	x	x	x
swreg13										
31:2	sw_dec_out_base	x	x	x	x	x	x	x	x	x
1	sw_dpb_ilace_mode	x	x		x					x
swreg14										
31:2	sw_refer0_base	x	x		x	x	x	x	x	x
1	sw_refer0_field_e	x								x
0	sw_refer0_topc_e	x								x
31:2	sw_jpg_ch_out_base			x					x	
swreg15										
31:2	sw_refer1_base	x	x		x	x		x	x	x
1	sw_refer1_field_e	x								x
0	sw_refer1_topc_e	x								x
7:0	sw_jpeg_slice_h			x					x	
swreg16										
31:2	sw_refer2_base	x	x		x	x		x		x
1	sw_refer2_field_e	x								x
0	sw_refer2_topc_e	x								x
31	-									
30:24	sw_ac1_code6_cnt			x						
23:22	-									
21:16	sw_ac1_code5_cnt			x						
15:11	sw_ac1_code4_cnt			x						
10:7	sw_ac1_code3_cnt			x						
6	-									
5:3	sw_ac1_code2_cnt			x						
2	-									
1:0	sw_ac1_code1_cnt			x						
swreg17										

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
31:2	sw_refer3_base	x	x		x	x		x		x
1	sw_refer3_field_e	x								x
0	sw_refer3_topc_e	x								x
31:24	sw_ac1_code10_cnt			x						
23:16	sw_ac1_code9_cnt			x						
15:8	sw_ac1_code8_cnt			x						
7:0	sw_ac1_code7_cnt			x						
swreg18										
31:2	sw_refer4_base	x					x		x	
1	sw_refer4_field_e	x								
0	sw_refer4_topc_e	x								
31:16	sw_pic_header_len				x					
15:14	-				x					
13	sw_pic_4mv_e				x					
12	-				x					
11	sw_range_red_ref_e				x					
10:9	sw_vc1_difmv_range				x					
8	-				x					
7:6	sw_mv_range				x					
5	sw_overlap_e				x					
4:3	sw_overlap_method				x					
19	sw_alt_scan_flag_e		x			x				
18:15	sw_fcode_fwd_hor		x			x				
14:11	sw_fcode_fwd_ver					x				
10:7	sw_fcode_bwd_hor		x			x				
6:3	sw_fcode_bwd_ver					x				
2	sw_mv_accuracy_fwd		x		x	x				

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
1	sw_mv_accuracy_bwd					x				
1	sw_mpeg4_vc1_rc		x		x					
0	sw_prev_anc_type		x		x			x		x
31:24	sw_ac1_code14_cnt			x						
23:16	sw_ac1_code13_cnt			x						
15:8	sw_ac1_code12_cnt			x						
7:0	sw_ac1_code11_cnt			x						
0	sw_gref_sign_bias								x	
swreg19										
31:2	sw_refer5_base	x							x	
1	sw_refer5_field_e	x								
0	sw_refer5_topc_e	x								
26:0	sw_trb_per_trd_d0		x							
24	sw_icomp3_e				x					
23:16	sw_iscale3				x					
15:0	sw_ishift3				x					
31:27	sw_ac2_code4_cnt			x						
26:23	sw_ac2_code3_cnt			x						
22	-									
21:19	sw_ac2_code2_cnt			x						
18	-									
17:16	sw_ac2_code1_cnt			x						
15:8	sw_ac1_code16_cnt			x						
7:0	sw_ac1_code15_cnt			x						
29:24	sw_scan_map_1						x		x	
23:18	sw_scan_map_2						x		x	
17:12	sw_scan_map_3						x		x	
11:6	sw_scan_map_4						x		x	

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
5:0	sw_scan_map_5						x		x	
0	sw_aref_sign_bias								x	
swreg20										
31:2	sw_refer6_base	x								
31:2	sw_vp8_dec_ch_base	x								
1	sw_vp8_stride_e								x	
0	sw_vp8_ch_base_e								x	
1	sw_refer6_field_e	x								
0	sw_refer6_topc_e	x								
26:0	sw_trb_per_trd_dm1		x							
24	sw_icomp4_e				x					
23:16	sw_iscale4				x					
15:0	sw_ishift4				x					
31:24	sw_ac2_code8_cnt			x						
23:16	sw_ac2_code7_cnt			x						
15										
14:8	sw_ac2_code6_cnt			x						
7:6	-									
5:0	sw_ac2_code5_cnt			x						
29:24	sw_scan_map_6						x		x	
23:18	sw_scan_map_7						x		x	
17:12	sw_scan_map_8						x		x	
11:6	sw_scan_map_9						x		x	
5:0	sw_scan_map_10						x		x	
swreg21										
31:2	sw_refer7_base	x								
31:27	sw_y_stride_pow2								x	
26:22	sw_c_stride_pow2								x	
1	sw_refer7_field_e	x								
0	sw_refer7_topc_e	x								

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
26:0	sw_trb_per_trd_d1		x							
31:24	sw_ac2_code12_cnt			x						
23:16	sw_ac2_code11_cnt			x						
15:8	sw_ac2_code10_cnt			x						
7:0	sw_ac2_code9_cnt			x						
29:24	sw_scan_map_11						x		x	
23:18	sw_scan_map_12						x		x	
17:12	sw_scan_map_13						x		x	
11:6	sw_scan_map_14						x		x	
5:0	sw_scan_map_15						x		x	
swreg22										
31:2	sw_refer8_base	x								
31:2	sw_dct_strm1_base								x	
1	sw_refer8_field_e	x								
0	sw_refer8_topc_e	x								
31:24	sw_ac2_code16_cnt			x						
23:16	sw_ac2_code15_cnt			x						
15:8	sw_ac2_code14_cnt			x						
7:0	sw_ac2_code13_cnt			x						
29:24	sw_scan_map_16						x			
23:18	sw_scan_map_17						x			
17:12	sw_scan_map_18						x			
11:6	sw_scan_map_19						x			
5:0	sw_scan_map_20						x			

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
swreg23										
31:2	sw_refer9_base	x								
31:2	sw_dct_strm2_base								x	
1	sw_refer9_field_e	x								
0	sw_refer9_topc_e	x								
31:28	sw_dc1_code8_cnt			x						
27:24	sw_dc1_code7_cnt			x						
23:20	sw_dc1_code6_cnt			x						
19:16	sw_dc1_code5_cnt			x						
15:12	sw_dc1_code4_cnt			x						
11:8	sw_dc1_code3_cnt			x						
7	-									
6:4	sw_dc1_code2_cnt			x						
3:2	-									
1:0	sw_dc1_code1_cnt			x						
29:24	sw_scan_map_21						x			
23:18	sw_scan_map_22						x			
17:12	sw_scan_map_23						x			
11:6	sw_scan_map_24						x			
5:0	sw_scan_map_25						x			
swreg24										
31:2	sw_refer10_base	x								
31:2	sw_dct_strm3_base								x	
1	sw_refer10_field_e	x								
0	sw_refer10_topc_e	x								

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
31:28	sw_dc1_code16_cnt			x						
27:24	sw_dc1_code15_cnt			x						
23:20	sw_dc1_code14_cnt			x						
19:16	sw_dc1_code13_cnt			x						
15:12	sw_dc1_code12_cnt			x						
11:8	sw_dc1_code11_cnt			x						
7:4	sw_dc1_code10_cnt			x						
3:0	sw_dc1_code9_cnt			x						
29:24	sw_scan_map_26						x			
23:18	sw_scan_map_27						x			
17:12	sw_scan_map_28						x			
11:6	sw_scan_map_29						x			
5:0	sw_scan_map_30						x			
swreg25										
31:2	sw_refer11_base	x								
31:2	sw_dct_strm4_base								x	
1	sw_refer11_field_e	x								
0	sw_refer11_topc_e	x								
31:28	sw_dc2_code8_cnt			x						
27:24	sw_dc2_code7_cnt			x						
23:20	sw_dc2_code6_cnt			x						
19:16	sw_dc2_code5_cnt			x						
15:12	sw_dc2_code4_cnt			x						

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
11:8	sw_dc2_code3_cnt			x						
7	-									
6:4	sw_dc2_code2_cnt			x						
3:2	-									
1:0	sw_dc2_code1_cnt			x						
29:24	sw_scan_map_31						x			
23:18	sw_scan_map_32						x			
17:12	sw_scan_map_33						x			
11:6	sw_scan_map_34						x			
5:0	sw_scan_map_35						x			
swreg26										
31:2	sw_refer12_base	x								
31:2	sw_dct_strm5_base								x	
1	sw_refer12_field_e	x								
0	sw_refer12_topc_e	x								
31:28	sw_dc2_code16_cnt			x						
27:24	sw_dc2_code15_cnt			x						
23:20	sw_dc2_code14_cnt			x						
19:16	sw_dc2_code13_cnt			x						
15:12	sw_dc2_code12_cnt			x						
11:8	sw_dc2_code11_cnt			x						
7:4	sw_dc2_code10_cnt			x						
3:0	sw_dc2_code9_cnt			x						
29:24	sw_scan_map_36						x			

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
23:18	sw_scan_map_37						x			
17:12	sw_scan_map_38						x			
11:6	sw_scan_map_39						x			
5:0	sw_scan_map_40						x			
swreg27										
31:2	sw_refer13_base	x								
1	sw_refer13_field_e	x								
0	sw_refer13_topc_e	x								
31:28	sw_dc3_code8_cnt			x						
27:24	sw_dc3_code7_cnt			x						
23:20	sw_dc3_code6_cnt			x						
19:16	sw_dc3_code5_cnt			x						
15:12	sw_dc3_code4_cnt			x						
11:8	sw_dc3_code3_cnt			x						
7	-									
6:4	sw_dc3_code2_cnt			x						
3:2	-									
1:0	sw_dc3_code1_cnt			x						
31:2	sw_bitpl_ctrl_base				x		x		x	
swreg28										
31:2	sw_refer14_base	x								
31:2	sw_dct_strm6_base								x	
1	sw_refer14_field_e	x								
0	sw_refer14_topc_e	x								
31:16	sw_ref_invd_cur_1									x

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
15:0	sw_ref_invd_cur_0									x
31:28	sw_dc3_code16_cnt			x						
27:24	sw_dc3_code15_cnt			x						
23:20	sw_dc3_code14_cnt			x						
19:16	sw_dc3_code13_cnt			x						
15:12	sw_dc3_code12_cnt			x						
11:8	sw_dc3_code11_cnt			x						
7:4	sw_dc3_code10_cnt			x						
3:0	sw_dc3_code9_cnt			x						
29:24	sw_scan_map_41						x			
23:18	sw_scan_map_42						x			
17:12	sw_scan_map_43						x			
11:6	sw_scan_map_44						x			
5:0	sw_scan_map_45						x			
swreg29										
31:2	sw_refer15_base	x								
31:2	sw_dct_strm7_base								x	
1	sw_refer15_field_e	x								
0	sw_refer15_topc_e	x								
31:16	sw_ref_invd_cur_3									x
15:0	sw_ref_invd_cur_2									x
29:24	sw_scan_map_46						x			
23:18	sw_scan_map_47						x			

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
17:12	sw_scan_map_48						x			
11:6	sw_scan_map_49						x			
5:0	sw_scan_map_50						x			
swreg30										
31:16	sw_refer1_nbr	x								
15:0	sw_refer0_nbr	x								
31:16	sw_ref_dist_cur_1									x
15:0	sw_ref_dist_cur_0									x
31	sw_filt_type								x	
30:28	sw_filt_sharpness								x	
27:21	sw_filt_mb_adj_0								x	
20:14	sw_filt_mb_adj_1								x	
13:7	sw_filt_mb_adj_2								x	
6:0	sw_filt_mb_adj_3								x	
swreg31										
31:16	sw_refer3_nbr	x								
15:0	sw_refer2_nbr	x								
29:24	sw_scan_map_51						x			
23:18	sw_scan_map_52						x			
17:12	sw_scan_map_53						x			
11:6	sw_scan_map_54						x			
5:0	sw_scan_map_55						x			
31:16	sw_ref_dist_cur_3									x
15:0	sw_ref_dist_cur_2									x
27:21	sw_filt_ref_adj_0								x	
20:14	sw_filt_ref_adj_1								x	
13:7	sw_filt_ref_adj_2								x	
6:0	sw_filt_ref_adj_3								x	
swreg32										
31:16	sw_refer5_nbr	x								

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
15:0	sw_refer4_nbr	x								
29:24	sw_scan_map_56						x			
23:18	sw_scan_map_57						x			
17:12	sw_scan_map_58						x			
11:6	sw_scan_map_59						x			
5:0	sw_scan_map_60						x			
31:16	sw_ref_invd_col_1									x
15:0	sw_ref_invd_col_0									x
23:18	sw_filt_level_0								x	
17:12	sw_filt_level_1								x	
11:6	sw_filt_level_2								x	
5:0	sw_filt_level_3								x	
swreg33										
31:16	sw_refer7_nbr	x								
15:0	sw_refer6_nbr	x								
29:24	sw_scan_map_61						x			
23:18	sw_scan_map_62						x			
17:12	sw_scan_map_63						x			
31:16	sw_ref_invd_col_3									x
15:0	sw_ref_invd_col_2									x
31:27	sw_quant_delta_0								x	
26:22	sw_quant_delta_1								x	
21:11	sw_quant_0								x	
10:0	sw_quant_1								x	
swreg34										
31:16	sw_refer9_nbr	x								
15:0	sw_refer8_nbr	x								
31:22	sw_pred_bc_tap_0_3		x		x		x	x	x	x

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
21:12	sw_pred_bc_tap_1_0				x		x	x	x	x
11:2	sw_pred_bc_tap_1_1				x		x	x	x	x
swreg35										
31:16	sw_refer11_nbr	x								
15:0	sw_refer10_nbr	x								
31:22	sw_pred_bc_tap_1_2				x		x	x	x	x
21:12	sw_pred_bc_tap_1_3				x		x	x	x	x
11:2	sw_pred_bc_tap_2_0				x		x		x	x
swreg36										
31:16	sw_refer13_nbr	x								
15:0	sw_refer12_nbr	x								
31:22	sw_pred_bc_tap_2_1				x		x		x	x
21:12	sw_pred_bc_tap_2_2				x		x		x	x
11:2	sw_pred_bc_tap_2_3				x		x		x	x
swreg37										
31:16	sw_refer15_nbr	x								
15:0	sw_refer14_nbr	x								
31:22	sw_pred_bc_tap_3_0						x		x	
21:12	sw_pred_bc_tap_3_1						x		x	
11:2	sw_pred_bc_tap_3_2						x		x	
swreg38										
31:0	sw_refer_lterm_e	x								
31:22	sw_pred_bc_tap_3_3						x		x	
21:12	sw_pred_bc_tap_4_0						x		x	
11:2	sw_pred_bc_tap_4_1						x		x	
swreg39										
31:0	sw_refer_valid_e	x								
31:22	sw_pred_bc_tap_4_2						x		x	

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
21:12	sw_pred_bc_tap_4_3						x		x	
11:2	sw_pred_bc_tap_5_0						x		x	
swreg40										
31:2	sw_qtable_base	x	x	x		x	x	x	x	
swreg41										
31:2	sw_dir_mv_base	x	x	x	x	x		x	x	x
swreg42										
29:25	sw_binit_rlist_b2	x								
24:20	sw_binit_rlist_f2	x								
19:15	sw_binit_rlist_b1	x								
14:10	sw_binit_rlist_f1	x								
9:5	sw_binit_rlist_b0	x								
4:0	sw_binit_rlist_f0	x								
31:22	sw_pred_bc_tap_5_1						x		x	
21:12	sw_pred_bc_tap_5_2						x		x	
11:2	sw_pred_bc_tap_5_3						x		x	
31:24	sw_weight_qp_1									x
23:21	sw_ref_delta_col_0									x
20:18	sw_ref_delta_col_1									x
17:15	sw_ref_delta_col_2									x
14:12	sw_ref_delta_col_3									x
11:9	sw_ref_delta_cur_0									x
8:6	sw_ref_delta_cur_1									x
5:3	sw_ref_delta_cur_2									x
2:0	sw_ref_delta_cur_3									x
swreg43										
29:25	sw_binit_rlist_b5	x								
24:20	sw_binit_rlist_f5	x								
19:15	sw_binit_rlist_b4	x								
14:10	sw_binit_rlist_f4	x								

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
9:5	sw_binit_rlist_b3	x								
4:0	sw_binit_rlist_f3	x								
31:22	sw_pred_bc_tap_6_0						x		x	
21:12	sw_pred_bc_tap_6_1						x		x	
11:2	sw_pred_bc_tap_6_2						x		x	
31:24	sw_weight_qp_2									x
23:16	sw_weight_qp_3									x
15:8	sw_weight_qp_4									x
7:0	sw_weight_qp_5									x
swreg44										
29:25	sw_binit_rlist_b8	x								
24:20	sw_binit_rlist_f8	x								
19:15	sw_binit_rlist_b7	x								
14:10	sw_binit_rlist_f7	x								
9:5	sw_binit_rlist_b6	x								
4:0	sw_binit_rlist_f6	x								
31:22	sw_pred_bc_tap_6_3						x		x	
21:12	sw_pred_bc_tap_7_0						x		x	
11:2	sw_pred_bc_tap_7_1						x		x	
26	sw_dec_avsp_ena									x
25	sw_weight_qp_e									x
24:23	sw_weight_qp_model									x
22	sw_avs_aec_e									x
21	sw_no_fwd_ref_e									x
20	sw_pb_field_enhanced_e									x
19:14	sw_qp_delta_cb									x
13:8	sw_qp_delta_cr									x
7:0	sw_weight_qp_0									x
swreg45										
29:25	sw_binit_rlist_b11	x								
24:20	sw_binit_rlist_f11	x								

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
19:15	sw_binit_rlist_b10	x								
14:10	sw_binit_rlist_f10	x								
9:5	sw_binit_rlist_b9	x								
4:0	sw_binit_rlist_f9	x								
31:22	sw_pred_bc_tap_7_2						x		x	
21:12	sw_pred_bc_tap_7_3						x		x	
11:10	sw_pred_tap_2_m1								x	
9:8	sw_pred_tap_2_4								x	
7:6	sw_pred_tap_4_m1								x	
5:4	sw_pred_tap_4_4								x	
3:2	sw_pred_tap_6_m1								x	
1:0	sw_pred_tap_6_4								x	
31:0	sw_dir_mv_base2									x
swreg46										
31:30										
29:25	sw_binit_rlist_b14	x								
24:20	sw_binit_rlist_f14	x								
19:15	sw_binit_rlist_b13	x								
14:10	sw_binit_rlist_f13	x								
9:5	sw_binit_rlist_b12	x								
4:0	sw_binit_rlist_f12	x								
31:27	sw_quant_delta_2								x	
26:22	sw_quant_delta_3								x	
21:11	sw_quant_2								x	
10:0	sw_quant_3								x	
swreg47										
31:30	-									
29:25	sw_pinit_rlist_f3	x								

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
24:20	sw_pinit_rlist_f2	x								
19:15	sw_pinit_rlist_f1	x								
14:10	sw_pinit_rlist_f0	x								
9:5	sw_binit_rlist_b15	x								
4:0	sw_binit_rlist_f15	x								
31:27	sw_quant_delta_4								x	
21:11	sw_quant_4								x	
10:0	sw_quant_5								x	
swreg48										
31:23	sw_startmb_x								x	
22:14	sw_startmb_y								x	
13:12	sw_error_conc_mode								x	
11:0	-									
swreg49										
31:22	sw_pred_bc_tap_0_0	x	x		x		x	x		x
21:12	sw_pred_bc_tap_0_1	x	x		x		x	x		x
11:2	sw_pred_bc_tap_0_2	x	x		x		x	x		x
1:0	-									
swreg50										
31	SW_DEC_MPEG2_PROF					x				
30:29	SW_DEC_VC1_PROF				x					
28	SW_DEC_JPEG_PROF			x						
27:26	SW_DEC_MPEG4_PROF		x							
25:24	SW_DEC_H264_PROF	x								
23	SW_DEC_VP6_PROF						x			
22	SW_DEC_PJPEG_EXIST			x						
21	SW_DEC_OBUF_LEVEL	x	x		x	x	x	x	x	x
20	SW_REF_BUFF_EXIST	x	x		x	x	x	x	x	x

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
19:16	SW_DEC_BUS_STRD	x	x	x	x	x	x	x	x	x
15:14	SW_DEC_SYNTH_LAN	x	x	x	x	x	x	x	x	x
13:12	SW_DEC_BUS_WIDTH	x	x	x	x	x	x	x	x	x
11	SW_DEC_SOREN_PROF		x							
10:0	SW_DEC_MAX_OWIDTH	x	x	x	x	x	x	x	x	x
swreg51										
31	sw_refbu_e	x	x		x	x	x	x	x	x
30:19	sw_refbu_thr	x	x		x	x	x	x	x	x
18:14	sw_refbu_picid	x	x		x	x	x	x	x	x
13	sw_refbu_eval_e	x	x		x	x	x	x	x	x
12	sw_refbu_fparmod_e	x	x		x	x	x	x	x	x
11:9	-									
8:0	sw_refbu_y_offset	x	x		x	x	x	x	x	x
swreg52										
31:16	sw_refbu_hit_sum	x	x		x	x	x	x	x	x
15:0	sw_refbu_intra_sum	x	x		x	x	x	x	x	x
swreg53										
31:22	-									
21:0	sw_refbu_y_mv_sum	x	x		x	x	x	x	x	x
swreg54										
31	SW_DEC_JPEG_EXTENS			x						
30	SW_DEC_REFBU_ILACE	x	x		x	x				
29	SW_DEC_DIVX_PROF		x							
28:0	SW_REF_BUFF2_EXIST	x	x		x	x	x	x	x	x
27:26	SW_DEC_RV_PROF							x		
25	SW_DEC_RTL_ROM		x		x			x		
24	SW_DEC_VP7_PROF								x	

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
23	SW_DEC_VP8_PROF								x	
22	SW_DEC_AVS_PROF									x
21:20	SW_DEC_MVC_PROF	x								
19	SW_DEC_WEBP_E								x	
18:17	SW_DEC_TILED_L	x	x		x	x	x	x	x	x
16	SW_DEC_VP8S_ARCH								x	
15:14	SW_DEC_MAX_OW_EXT	x	x		x	x	x	x	x	x
13:12	SW_DEC_ERRCO_LEVEL								x	
11	SW_VP8_STRIDE_E								x	
10	SW_DPB_FIELD_E	x	x	x	x	x	x	x	x	x
9:7	SW_DEC_CORE_AM	x	x	x	x	x	x	x	x	x
6:0	-									
swreg55										
31	sw_refbu2_buf_e	x	x		x	x	x	x	x	x
30:19	sw_refbu2_thr	x	x		x	x	x	x	x	x
18:14	sw_refbu2_picid	x	x		x	x	x	x	x	x
13:0	sw_apf_threshold	x	x		x	x	x	x	x	x
swreg56										
31:16	sw_refbu_top_sum	x	x		x	x	x	x	x	x
15:0	sw_refbu_bot_sum	x	x		x	x	x	x	x	x
swreg57										
31	fuse_dec_h264	x								
30	fuse_dec_mpeg4		x							
29	fuse_dec_mpeg2					x				
28	fuse_dec_sorenson		x							
27	fuse_dec_jpeg			x						
26	fuse_dec_vp6						x			
25	fuse_dec_vc1				x					
24	fuse_dec_pjpeg			x						

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
23	fuse_dec_divx		x							
22	fuse_dec_rv							x		
21	fuse_dec_vp7								x	
20	fuse_dec_vp8								x	
19	fuse_dec_avs									x
18	fuse_dec_mvc	x								x
17	-									
16	fuse_dec_maxw_4k	x	x	x	x	x	x	x	x	x
15	fuse_dec_maxw_1920	x	x		x	x	x	x	x	x
14	fuse_dec_maxw_1280	x	x		x	x	x	x	x	x
13	fuse_dec_maxw_720	x	x		x	x	x	x	x	x
12	fuse_dec_maxw_352	x	x		x	x	x	x	x	x
11:8	-									
7	fuse_dec_refbuffer	x	x		x	x	x	x	x	x
6:0	-									
swreg58										
31	sw_serv_merge_dis	x	x	x	x	x	x	x	x	x
30	sw_dec_multicore_e	x							x	
29	sw_dec_writestat_e	x							x	
28:27	sw_dec_mc_poll mode	x							x	
26:17	sw_dec_mc_polltime	x							x	
16:0	-									
swreg59										
31:2	sw_dec_ch8pix_base	x								
1:0	-									
Post-processor (swreg60-100)										
swreg102										
31:2	sw_dec_ch_base	x								x
1	-									
0	sw_ch_base_e	x								x

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
swreg103										
31:2	sw_refer0_ch_base	x								x
1:0	-									
swreg104										
31:2	sw_refer1_ch_base	x								x
1:0	-									
swreg105										
31:2	sw_refer2_ch_base	x								x
1:0	-									
swreg106										
31:2	sw_refer3_ch_base	x								x
1:0	-									
swreg107										
31:2	sw_refer4_ch_base	x								x
1:0	-									
swreg108										
31:2	sw_refer5_ch_base	x								x
1:0	-									
swreg109										
31:2	sw_refer6_ch_base	x								x
1:0	-									
swreg110										
31:2	sw_refer7_ch_base	x								x
1:0	-									
swreg111										
31:2	sw_refer8_ch_base	x								x
1:0	-									
swreg112										
31:2	sw_refer9_ch_base	x								x
1:0	-									
swreg113										

Table continues on the next page...

Bit	Name	Dec Modes								
		H.264	H.263	JPEG	VC-1	MPEG2/ MPEG1	VP6	RV	VP7 / VP8	AVS
31:2	sw_refer10_ch_base	x								x
1:0	-									
swreg114										
31:2	sw_refer11_ch_base	x								x
1:0	-									
swreg115										
31:2	sw_refer12_ch_base	x								x
1:0	-									
swreg116										
31:2	sw_refer13_ch_base	x								x
1:0	-									
swreg117										
31:2	sw_refer14_ch_base	x								x
1:0	-									
swreg118										
31:2	sw_refer15_ch_base	x								x
1:0	-									

15.1.5 VPU G1 Memory Map/Register Definition

15.1.5.1 VPU_G1 common registers descriptions

15.1.5.1.1 VPU_G1 common registers memory map

VPU_G1 base address: 3830_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	Interrupt register decoder (SWREG1)	32	RW	0000_0000h
8h	Device configuration register decoder (SWREG2)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
Ch	Decoder control register 0 (decmode,picture type etc) (SWREG3)	32	RW	0000_0000h
30h	Base address for RLC data (RLC) / stream start address/decoded end addr register (VLC) (SWREG12)	32	RW	0000_0000h
34h	Base address for decoded picture (SWREG13)	32	RW	0000_0000h
A0h	Base address for standard dependent tables (SWREG40)	32	RW	0000_0000h
A4h	Base address for direct mode motion vectors (SWREG41)	32	RW	0000_0000h
C0h	Error concealment register (SWREG48)	32	RW	0000_0000h
C4h	Prediction filter tap register for H264 (SWREG49)	32	RW	0000_0000h
C8h	Synthesis configuration register decoder 0 (SWREG50)	32	RO	Table 15-
CCh	Reference picture buffer control register (SWREG51)	32	RW	0000_0000h
D0h	Reference picture buffer information register 1 (SWREG52)	32	RO	0000_0000h
D4h	Reference picture buffer information register 2 (SWREG53)	32	RO	0000_0000h
D8h	Synthesis configuration register decoder 1 (SWREG54)	32	RO	0000_0000h
DCh	Reference picture buffer 2 / Advanced prefetch control register (SWREG55)	32	RW	0000_0000h
E0h	Reference buffer information register 3 (SWREG56)	32	RO	0000_0000h
E4h	Decoder fuse register (SWREG57)	32	RO	Table 15-
E8h	Device configuration register decoder 2 + Multi core control register (SWREG58)	32	RW	0000_0000h
ECh	H264 Chrominance 8 pixel interleaved data base (SWREG59)	32	RW	0000_0000h
F0h	Interrupt register post-processor (SWREG60)	32	RW	0000_0000h
F4h	Device configuration register post-processor (SWREG61)	32	RW	0000_0100h
F8h	Deinterlace control register (SWREG62)	32	RW	0000_0000h
FCh	Base address for reading post-processing input picture luminance (top field/frame) (SWREG63)	32	RW	0000_0000h
100h	Base address for reading post-processing input picture Cb/Ch (top field/frame) (SWREG64)	32	RW	0000_0000h
104h	Base address for reading post-processing input picture Cr (SWRE G65)	32	RW	0000_0000h
108h	Base address for writing post-processed picture luminance/RGB (SWREG66)	32	RW	0000_0000h
10Ch	Base address for writing post-processed picture Ch (SWREG67)	32	RW	0000_0000h
110h	Register for contrast adjusting (SWREG68)	32	RW	0000_0000h
114h	Register for colour conversion and contrast adjusting/YUYV 422 channel orders (SWREG69)	32	RW	0000_0000h
118h	Register for colour conversion 0 (SWREG70)	32	RW	0000_0000h
11Ch	Register for colour conversion 1 + rotation mode (SWREG71)	32	RW	0000_0000h
120h	PP input size and -cropping register (SWREG72)	32	RW	0000_0000h
124h	PP input picture base address for Y bottom field (SWREG73)	32	RW	0000_0000h
128h	PP input picture base for Ch bottom field (SWREG74)	32	RW	0000_0000h
13Ch	Scaling register 0 ratio and padding for R and G (SWREG79)	32	RW	0000_0000h
140h	Scaling ratio register 1 and padding for B (SWREG80)	32	RW	0000_0000h

Table continues on the next page...

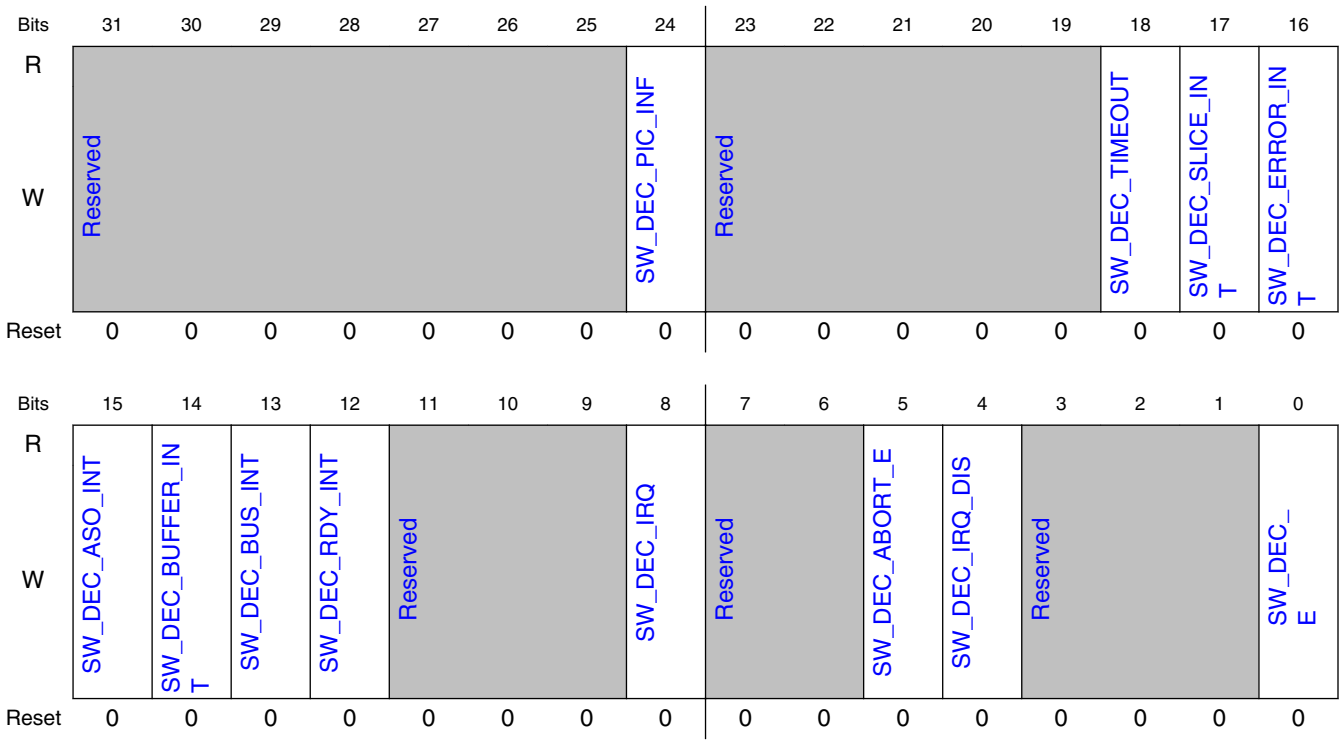
Offset	Register	Width (In bits)	Access	Reset value
144h	Scaling ratio register 2 (SWREG81)	32	RW	0000_0000h
148h	Rmask register (SWREG82)	32	RW	0000_0000h
14Ch	Gmask register (SWREG83)	32	RW	0000_0000h
150h	Bmask register (SWREG84)	32	RW	0000_0000h
154h	Post-processor control register (SWREG85)	32	RW	0000_0000h
158h	Mask 1 start coordinate register (SWREG86)	32	RW	0000_0000h
15Ch	Mask 2 start coordinate register + Mask extensions (SWREG87)	32	RW	0000_0000h
160h	Mask 1 size and PP original width register (SWREG88)	32	RW	0000_0000h
164h	Mask 2 size register + mask extensions (SWREG89)	32	RW	0000_0000h
168h	PiP register 0 (SWREG90)	32	RW	0000_0000h
16Ch	PiP register 1 and dithering control (SWREG91)	32	RW	0000_0000h
170h	Display width and PP input size extension register (SWREG92)	32	RW	0000_0000h
174h	Base address for alpha blend 1 gui component (SWREG93)	32	RW	0000_0000h
178h	Base address for alpha blend 2 gui component (SWREG94)	32	RW	0000_0000h
17Ch	Alpha blend input cropping register (scanline for cropping) (SWREG95)	32	RW	0000_0000h
18Ch	PP fuse register (SWREG99)	32	RO	Table 15-
190h	Synthesis configuration register post-processor (SWREG100)	32	RO	Table 15-
198h	Base address for H264 decoded chroma picture (SWREG102)	32	RW	0000_0000h
19Ch	Base address for reference chroma picture index 0 (SWREG103)	32	RW	0000_0000h
1A0h	Base address for reference chroma picture index 1 (SWREG104)	32	RW	0000_0000h
1A4h	Base address for reference chroma picture index 2 (SWREG105)	32	RW	0000_0000h
1A8h	Base address for reference chroma picture index 3 (SWREG106)	32	RW	0000_0000h
1ACh	Base address for reference chroma picture index 4 (SWREG107)	32	RW	0000_0000h
1B0h	Base address for reference chroma picture index 5 (SWREG108)	32	RW	0000_0000h
1B4h	Base address for reference chroma picture index 6 (SWREG109)	32	RW	0000_0000h
1B8h	Base address for reference chroma picture index 7 (SWREG110)	32	RW	0000_0000h
1BCh	Base address for reference chroma picture index 8 (SWREG111)	32	RW	0000_0000h
1C0h	Base address for reference chroma picture index 9 (SWREG112)	32	RW	0000_0000h
1C4h	Base address for reference chroma picture index 10 (SWREG113)	32	RW	0000_0000h
1C8h	Base address for reference chroma picture index 11 (SWREG114)	32	RW	0000_0000h
1CCh	Base address for reference chroma picture index 12 (SWREG115)	32	RW	0000_0000h
1D0h	Base address for reference chroma picture index 13 (SWREG116)	32	RW	0000_0000h
1D4h	Base address for reference chroma picture index 14 (SWREG117)	32	RW	0000_0000h
1D8h	Base address for reference chroma picture index 15 (SWREG118)	32	RW	0000_0000h

15.1.5.1.2 Interrupt register decoder (SWREG1)

15.1.5.1.2.1 Offset

Register	Offset
SWREG1	4h

15.1.5.1.2.2 Diagram



15.1.5.1.2.3 Fields

Field	Function
31-25 —	Reserved.
24 SW_DEC_PIC_INF	B slice detected. This signal is driven high during picture ready interrupt if B-type slice is found. This bit does not launch interrupt but is used to inform SW about h264 tools.
23-19 —	Reserved.
18 SW_DEC_TIMEOUT	Interrupt status bit decoder timeout. When high the decoder has been idling for too long. HW will self reset. Possible only if timeout interrupt is enabled

Table continues on the next page...

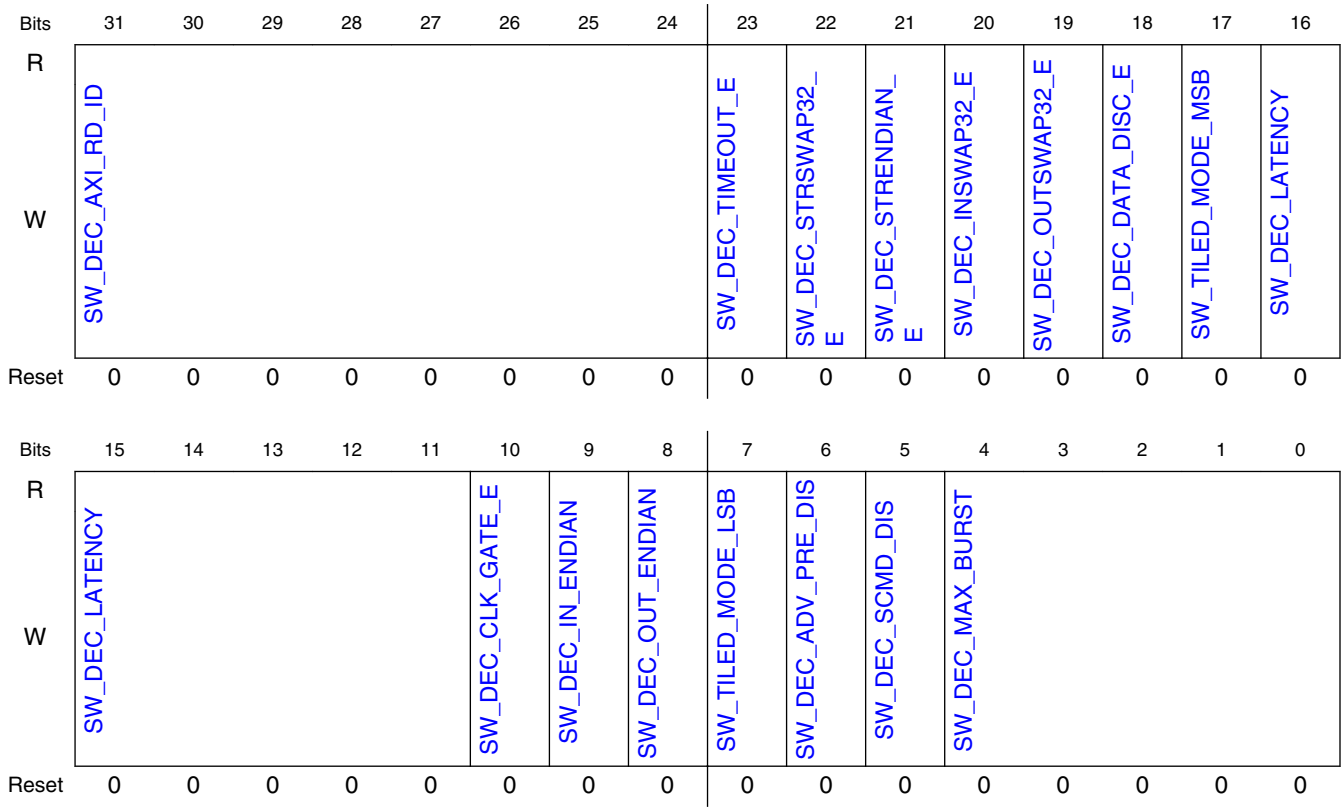
Field	Function
17 SW_DEC_SLICE_INT	Interrupt status bit dec_slice_decoded. When high SW must set new base addresses for sw_dec_out_base and sw_jpg_ch_out_base before resetting this status bit. Used for VP8 web-p modes
16 SW_DEC_ERROR_INT	Interrupt status bit input stream error. When high, an error is found in input data stream decoding. HW will self reset.
15 SW_DEC_ASO_INT	H264: Interrupt status bit ASO (Arbitrary Slice Ordering) detected. When high, ASO detected in input data stream decoding. HW will self reset. VP8: Error detected in Residual data. HW returns MB number in error concealment register for MB it detected it
14 SW_DEC_BUFFER_INT	Interrupt status bit input buffer empty. When high, input stream buffer is empty but picture is not ready. HW will not self reset.
13 SW_DEC_BUS_INT	Interrupt status bit bus. Error response from bus.
12 SW_DEC_RDY_INT	Interrupt status bit decoder. When this bit is high decoder has decoded a picture. HW will self reset.
11-9 —	Reserved.
8 SW_DEC_IRQ	Decoder IRQ. When high, decoder requests an interrupt. SW will reset this after interrupt is handled.
7-6 —	Reserved.
5 SW_DEC_ABORT_E	Abort decoding enable. Setting this bit high will cause HW to abort decoding and safely to reset itself down. After abort is complete the corresponding interrupt status is set and this bit is set low as well as the decoder enable.
4 SW_DEC_IRQ_DIS	Decoder IRQ disable. When high, there are no interrupts concerning decoder from HW. Polling must be used to see the interrupt statuses.
3-1 —	Reserved.
0 SW_DEC_E	Decoder enable. Setting this bit high will start the decoding operation. HW will reset this when picture is processed or ASO or stream error is detected or bus error or timeout interrupt is given.

15.1.5.1.3 Device configuration register decoder (SWREG2)

15.1.5.1.3.1 Offset

Register	Offset
SWREG2	8h

15.1.5.1.3.2 Diagram



15.1.5.1.3.3 Fields

Field	Function
31-24 SW_DEC_AXI_RD_ID	Read ID used for decoder reading services in AXI bus (if connected to AXI).
23 SW_DEC_TIMEOUT_E	Timeout interrupt enable. If enabled HW may return timeout interrupt in case HW gets stuck while decoding picture.
22 SW_DEC_STRSWAP32_E	Decoder input 32bit data swap for stream data (may be used for 64 bit environment): 0b - no swapping of 32 bit words 1b - 32 bit data words are swapped (needed in 64 bit environment to achieve 7-6-5-4-3-2-1-0 byte order(also little endian should be enabled))
21 SW_DEC_STRENDIAN_E	Decoder input endian mode for stream data: 0b - Big endian (0-1-2-3 order) 1b - Little endian (3-2-1-0 order)
20	Decoder input 32bit data swap for other than stream data (may be used for 64 bit environment): 0b - no swapping of 32 bit words

Table continues on the next page...

Field	Function
SW_DEC_INSWAP32_E	1b - 32 bit data words are swapped (needed in 64 bit environment to achieve 7-6-5-4-3-2-1-0 byte order(also little endian should be enabled))
19 SW_DEC_OUTSWAP32_E	Decoder output 32bit data swap (may be used for 64 bit environment): 0b - no swapping of 32 bit words 1b - 32 bit data words are swapped (needed in 64 bit environment to achieve 7-6-5-4-3-2-1-0 byte order(also little endian should be enabled))
18 SW_DEC_DATA_DISC_E	Data discard enable. Precise burst lengths are used with reading services. Extra data is discarded internally. Note. If AHB maxburst 17 is used data discard cannot be enabled (causes conflict)
17 SW_TILED_MODE_MSB	Tiled mode msb. Concatenated to Tiled mode lsb which form 2 bit tiled mode. 0b - Tiled mode not enabled 1b - Tiled mode enabled for 8x4 tile size
16-11 SW_DEC_LATENCY	Decoder master interface additional latency. Can be used to slow down decoder HW between services in steps of 8 clock cycles: 0 = no latency 1 = minimum 8 cycles of IDLE between services 2 = minimum 16 cycles of IDLE between services . . . 63 = minimum latency of 504 cycles of IDLE between services
10 SW_DEC_CLK_GATE_E	Decoder dynamic clock gating enable: 0b - Clock is running for all structures 1b - Clock is gated for decoder structures that are not used. Note: Clock gating value can be changed only when decoder is disabled.
9 SW_DEC_IN_ENDIAN	Decoder input endian mode for other than stream data: 0b - Big endian (0-1-2-3 order) 1b - Little endian (3-2-1-0 order)
8 SW_DEC_OUT_ENDIAN	Decoder output endian mode: 0b - Big endian (0-1-2-3 order) 1b - Little endian (3-2-1-0 order)
7 SW_TILED_MODE_LSB	Tiled mode lsb. Concatenated to Tiled mode msb which form 2 bit tiled mode. Defined in tiled_mode_msb
6 SW_DEC_ADV_PRE_DIS	Advanced PREFETCH mode disable (advanced reference picture reading mode for video)
5 SW_DEC_SCMD_DIS	9170 decoder and later->: AXI Single Command Multiple Data disable. 9170 axi wrapper supports this mode by default (where only the first addresses of the burst are given from address generator). This bit is used to disable the feature (possible SW workaround if something is not working correctly)
4-0 SW_DEC_MAX_BURST	Maximum burst length for decoder bus transactions. Valid values: AHB: 1, 4, 8, 16 and 17. Other values will result in INCR type (undefined length) for all transfers. INCR used for lengths 2 and 3.

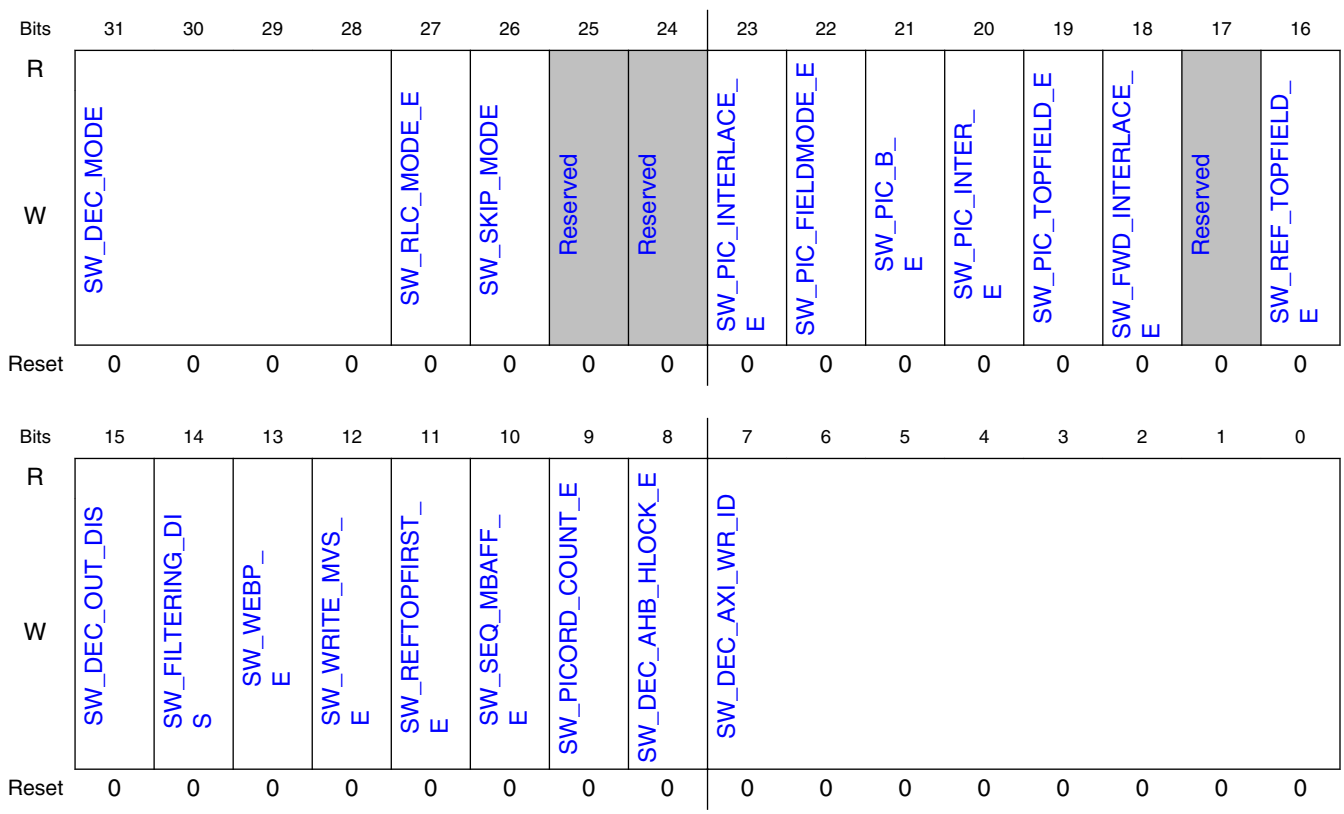
Field	Function
	17 = fixed bursts are used when possible. INCR bursts are used for lengths 2, 3, 5, 6, 7, 9, 10, 11, 12, 13, 14 and 15. OCP: 16-31 AXI: 1-16

15.1.5.1.4 Decoder control register 0 (decmode,picture type etc) (SWREG3)

15.1.5.1.4.1 Offset

Register	Offset
SWREG3	Ch

15.1.5.1.4.2 Diagram



15.1.5.1.4.3 Fields

Field	Function
31-28 SW_DEC_MODE	Decoding mode: 0000b - H.264 0001b - Reserved 0010b - Reserved 0011b - Reserved 0100b - Reserved 0101b - Reserved 0110b - Reserved 0111b - Reserved 1000b - Reserved 1001b - Reserved 1010b - VP8 1011b - Reserved 1100b - Reserved 1101b - Reserved 1110b - Reserved 1111b - Reserved
27 SW_RLC_MODE_E	RLC mode enable: 0b - HW decodes video from bit stream (VLC mode) + side information 1b - HW decodes video from RLC input data + side information (Differential MV's, separate DC coeffs, Intra 4x4 modes, MB control). Valid only for H.264 Baseline.
26 SW_SKIP_MODE	VP8: • 0 : HW decodes mb_coeff_skip -flag • 1 : HW does not decode mb_coeff_skip -flag
25 —	Reserved.
24 —	Reserved.
23 SW_PIC_INTERLACE_E	Coding mode of the current picture: 0b - progressive 1b - interlaced
22 SW_PIC_FIELDMODE_E	Structure of the current picture (residual structure) 0b - Frame structure. For H264, this means MBAFF structured picture for interlaced sequence 1b - Field structure
21 SW_PIC_B_E	B picture enable for current picture: 0b - picture type is I or P depending on sw_pic_inter_e 1b - picture type is B depending on sw_pic_inter_e (not valid for H264 since it is slice based information)
20 SW_PIC_INTER_E	Picture type. Please also see SW_PIC_B_E. 0b - Intra type (I) 1b - Inter type (P)
19 SW_PIC_TOPFIELD_E	If field structure is enabled, this bit informs which one of the fields is being decoded: 0b - bottom field 1b - top field
18 SW_FWD_INTERLACE_E	Coding mode of forward reference picture **NOTE:** For backward reference picture, the coding mode is always the same as for current picture. 0b - progressive

Table continues on the next page...

Field	Function
	1b - interlaced
17 —	Reserved.
16 SW_REF_TOPFIELD_E	Indicates which field should be used as reference if sw_ref_frames = '0': 0b - bottom field 1b - top field
15 SW_DEC_OUT_DIS	Disable decoder output picture writing: 0b - Decoder output picture is written to external memory 1b - Decoder output picture is not written to external memory
14 SW_FILTERING_DIS	De-block filtering disable: 0b - filtering is enabled for current picture 1b - filtering is disabled for current picture
13 SW_WEBP_E	SW_WEBP_E: Webp enable for VP8: • '0' = Normal VP8 • '1' = WEBP picture SW_MVC_E: Multi View Coding enable. Possible for H264 only.
12 SW_WRITE_MVS_E	Direct mode motion vector write enable for current picture / VPX motion vector write enable for error concealment purposes: 0b - Writing disabled for current picture 1b - The direct mode motion vectors are written to external memory. H264 direct mode motion vectors are written to DPB aside with the corresponding reference picture. Other decoding mode dir mode mvs are written to external memory starting from sw_dir_mv_base.
11 SW_REFTOPFIRST_E	Indicates which FWD reference field has been decoded first. 0b - FWD reference bottom field 1b - FWD reference top field
10 SW_SEQ_MBAFF_E	Sequence includes MBAFF coded pictures
9 SW_PICORD_COUNT_E	h264_high config: Picture order count table read enable. If enabled HW will read picture order counts from memory in the beginning of picture
8 SW_DEC_AHB_HLOCK_E	AHB master HLOCK enable. When high the service is locked to decoder as long as it needs the bus (whenever decoder requests the bus it will be granted)
7-0 SW_DEC_AXI_WR_ID	Write ID used for decoder writing services in AXI bus (if connected to AXI)

15.1.5.1.5 Base address for RLC data (RLC) / stream start address/ decoded end addr register (VLC) (SWREG12)

15.1.5.1.5.1 Offset

Register	Offset
SWREG12	30h

15.1.5.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_RLC_VLC_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_RLC_VLC_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.5.3 Fields

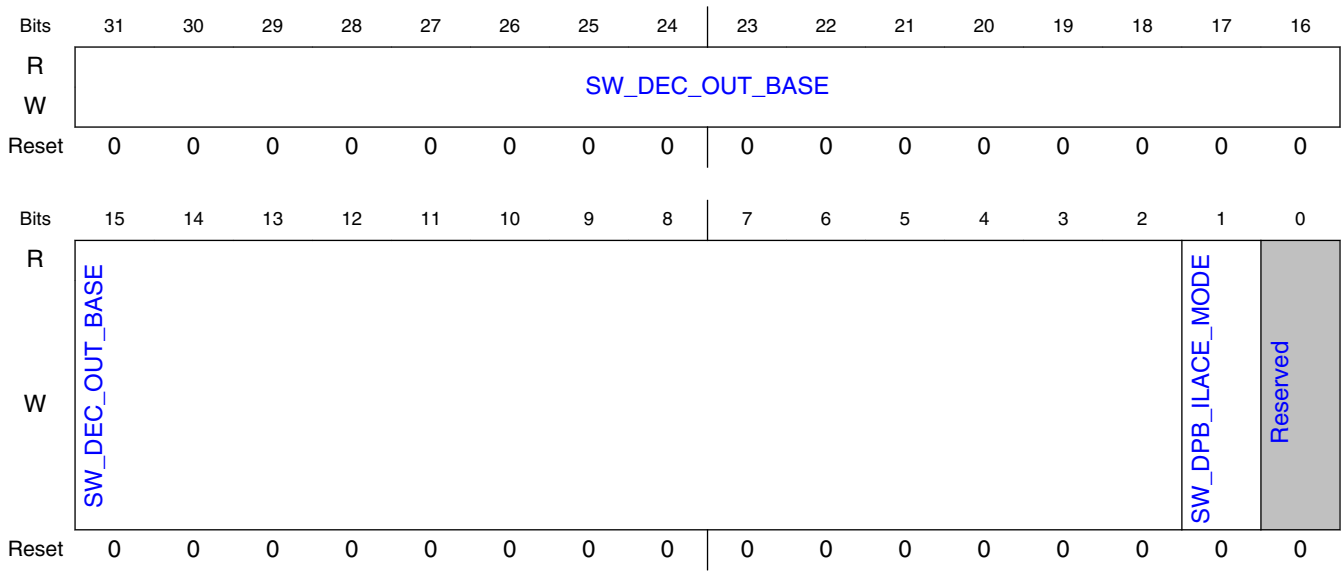
Field	Function
31-0 SW_RLC_VLC_BASE	RLC mode: Base address for RLC data (swreg3.sw_rlc_mode_e = 1). VLC mode: Stream start address / end addr+I288ess with byte precision (swreg4.rlc_mode_en = 0), start bit number in swreg5.stream_start_bit. When sw_dec_buffer_int is high or sw_dec_e is low this register contains HW return value of last_byte_address where stream has been read (and used) in accuracy of byte. For debug purposes the last_byte_address is also written when stream error/ASO is detected even though it may not be accurate. VP8: This base address is used as sw_dct_strm0_base including DCT stream for MB rows 0,2n.

15.1.5.1.6 Base address for decoded picture (SWREG13)

15.1.5.1.6.1 Offset

Register	Offset
SWREG13	34h

15.1.5.1.6.2 Diagram



15.1.5.1.6.3 Fields

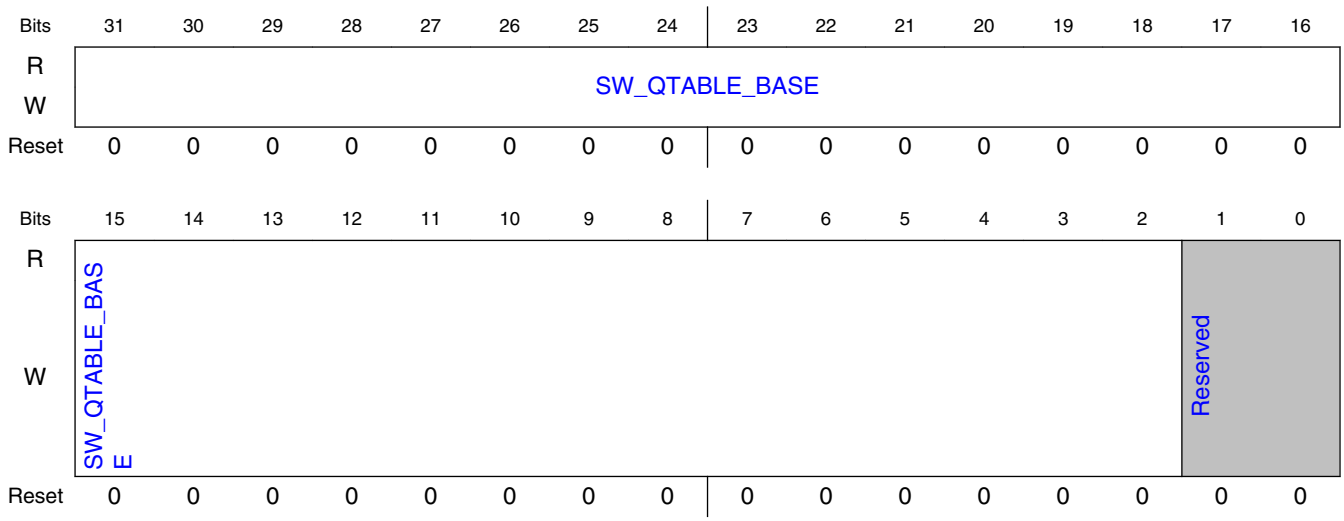
Field	Function
31-2 SW_DEC_OUT_BASE	Video: Base address for decoder output picture. Points directly to start of decoder output picture or field.
1 SW_DPB_ILACE_MODE	DPB ilaced mode: '0' : DPB consist of ilaced/progressive frames '1' : DPB consist of progressive frames / separate fields. This mode requires config support from HW
0 —	Reserved.

15.1.5.1.7 Base address for standard dependent tables (SWREG40)

15.1.5.1.7.1 Offset

Register	Offset
SWREG40	A0h

15.1.5.1.7.2 Diagram



15.1.5.1.7.3 Fields

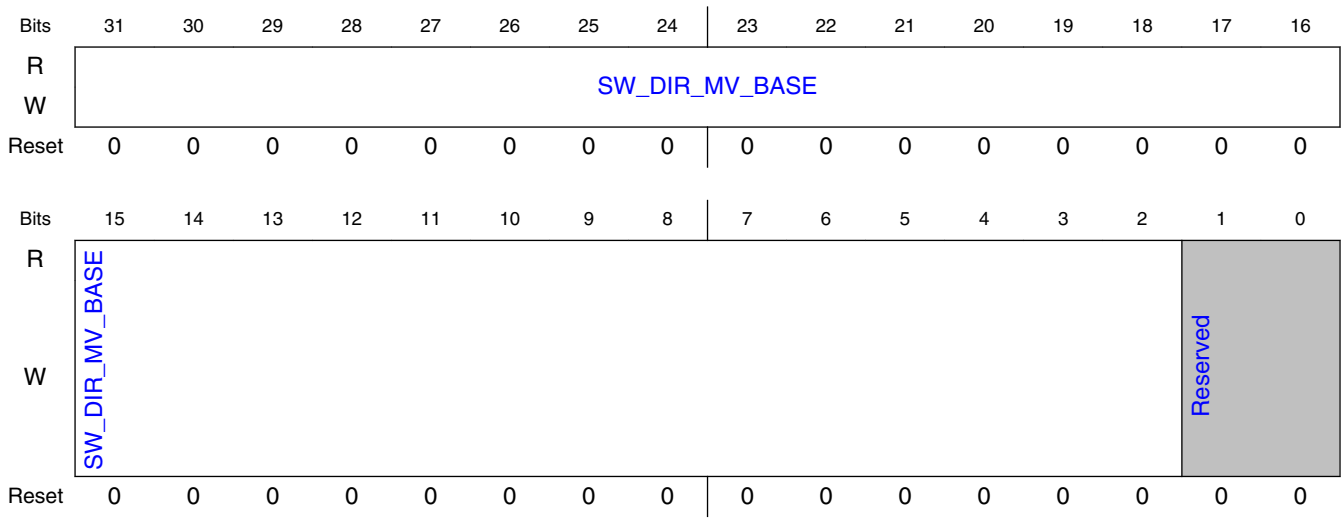
Field	Function
31-2 SW_QTABLE_BASE	Base address for standard dependent tables: H.264: base address for various tables VP8: base address for stream decoding tables
1-0 —	Reserved.

15.1.5.1.8 Base address for direct mode motion vectors (SWREG41)

15.1.5.1.8.1 Offset

Register	Offset
SWREG41	A4h

15.1.5.1.8.2 Diagram



15.1.5.1.8.3 Fields

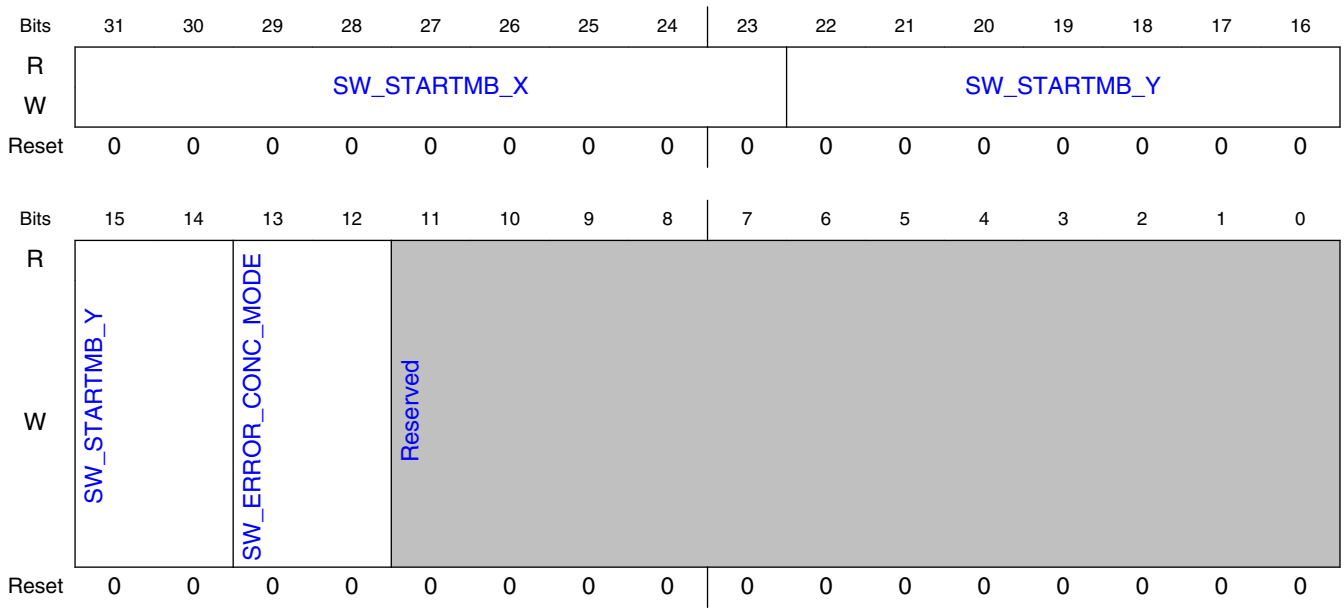
Field	Function
31-2 SW_DIR_MV_BASE	Direct mode motion vector write/read base address. For H264 this is used only for direct mode motion vector write base. VP8: Motion vectors are written for error concealment purposes if sw_write_mvs is high. In error concealment mode motion vectors are read from this base address
1-0 —	Reserved.

15.1.5.1.9 Error concealment register (SWREG48)

15.1.5.1.9.1 Offset

Register	Offset
SWREG48	C0h

15.1.5.1.9.2 Diagram



15.1.5.1.9.3 Fields

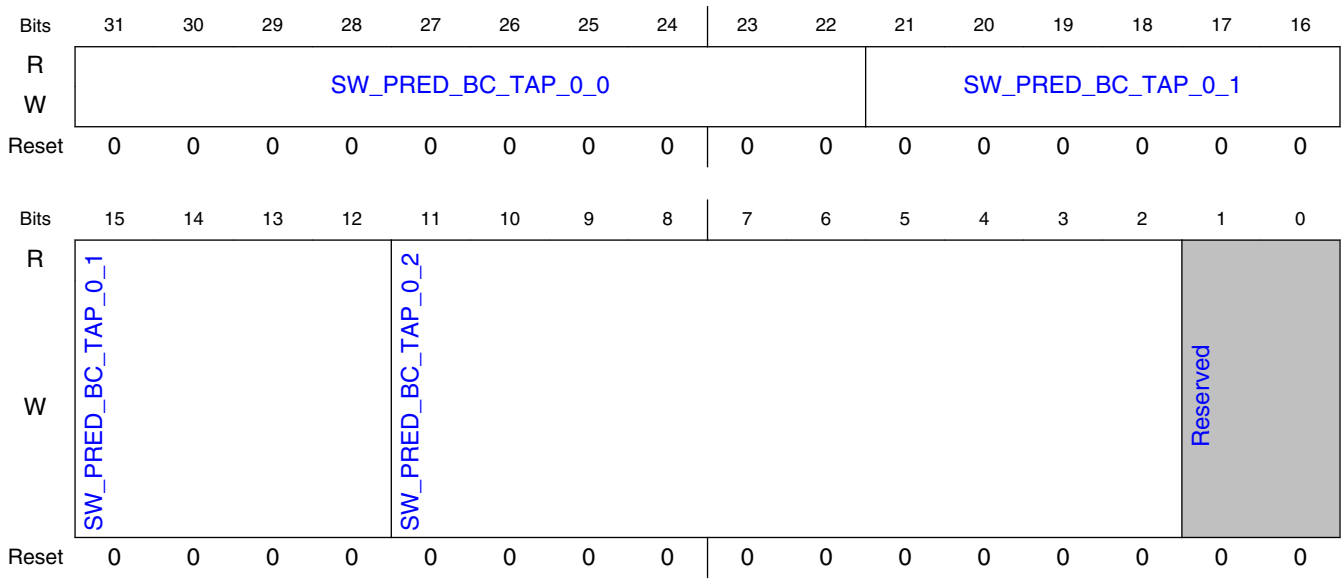
Field	Function
31-23 SW_STARTMB_X	Start MB from SW for X dimension. Used in error concealment case as HW return value if HW founds an error or in HW init mb for error concealment if SW enables error concealment
22-14 SW_STARTMB_Y	Start MB from SW for Y dimension. Used in error concealment case as HW return value if HW founds an error or in HW init mb for error concealment if SW enables error concealment
13-12 SW_ERROR_CONC_MODE	Error concealment mode: 00b - disabled (normal decoding mode) 01b - enabled for direct mode MV usage starting from MB defined by sw_startmb_x, sw_startmb_y
11-0 —	Reserved.

15.1.5.1.10 Prediction filter tap register for H264 (SWREG49)

15.1.5.1.10.1 Offset

Register	Offset
SWREG49	C4h

15.1.5.1.10.2 Diagram



15.1.5.1.10.3 Fields

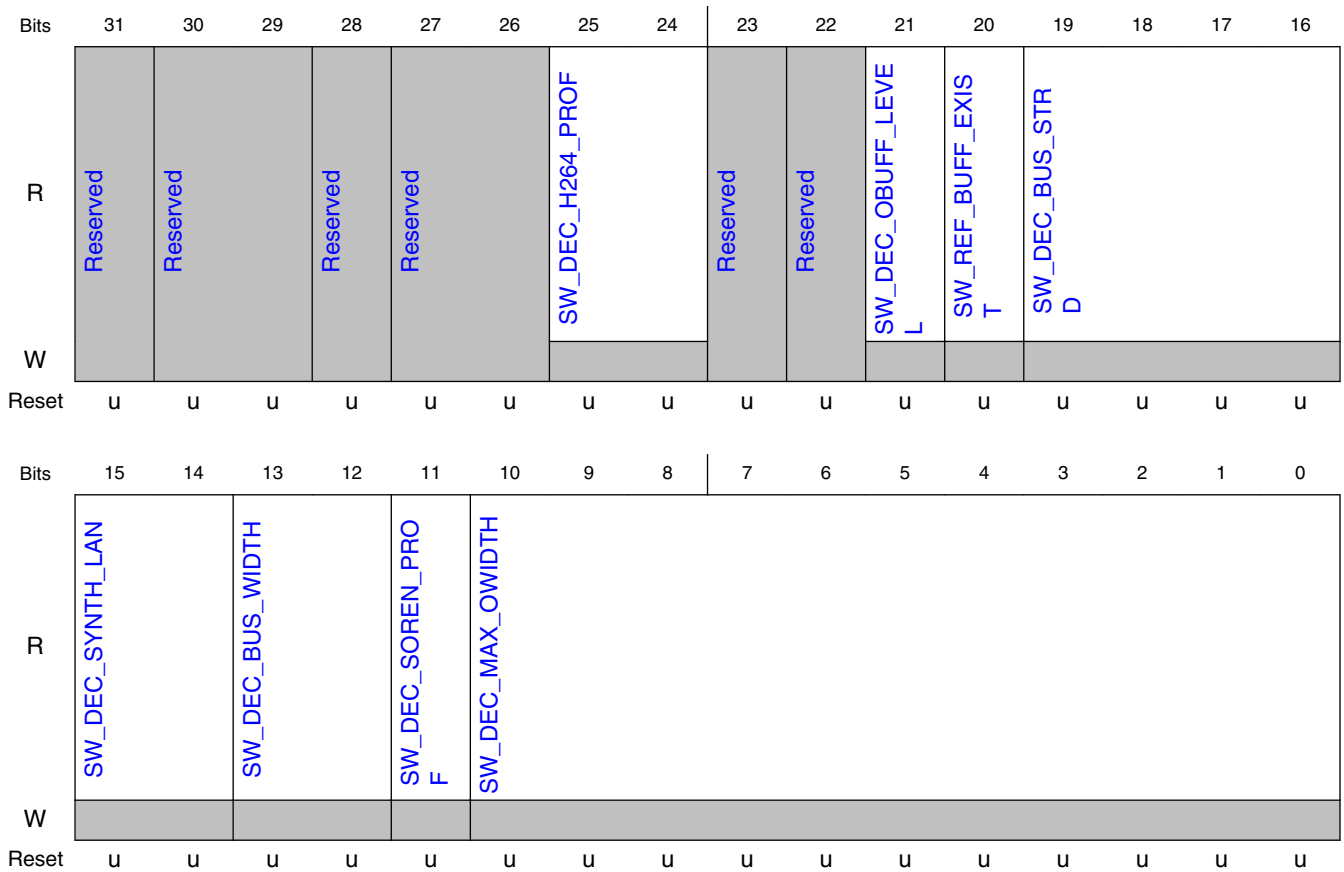
Field	Function
31-22 SW_PRED_BC_TAP_0_0	Prediction filter set 0, tap 0
21-12 SW_PRED_BC_TAP_0_1	Prediction filter set 0, tap 1
11-2 SW_PRED_BC_TAP_0_2	Prediction filter set 0, tap 2
1-0 —	Reserved.

15.1.5.1.11 Synthesis configuration register decoder 0 (SWREG50)

15.1.5.1.11.1 Offset

Register	Offset
SWREG50	C8h

15.1.5.1.11.2 Diagram



15.1.5.1.11.3 Fields

Field	Function
31 —	Reserved.
30-29 —	Reserved.
28 —	Reserved.
27-26 —	Reserved.
25-24 SW_DEC_H264_PROF	Decoding format support, H.264 00b - not supported 01b - supported up to baseline profile 10b - supported up to high profile labeled stream with restricted high profile tools (Tools that are used in Hantro 7280, 8270 encoder) 11b - supported up to high profile

Table continues on the next page...

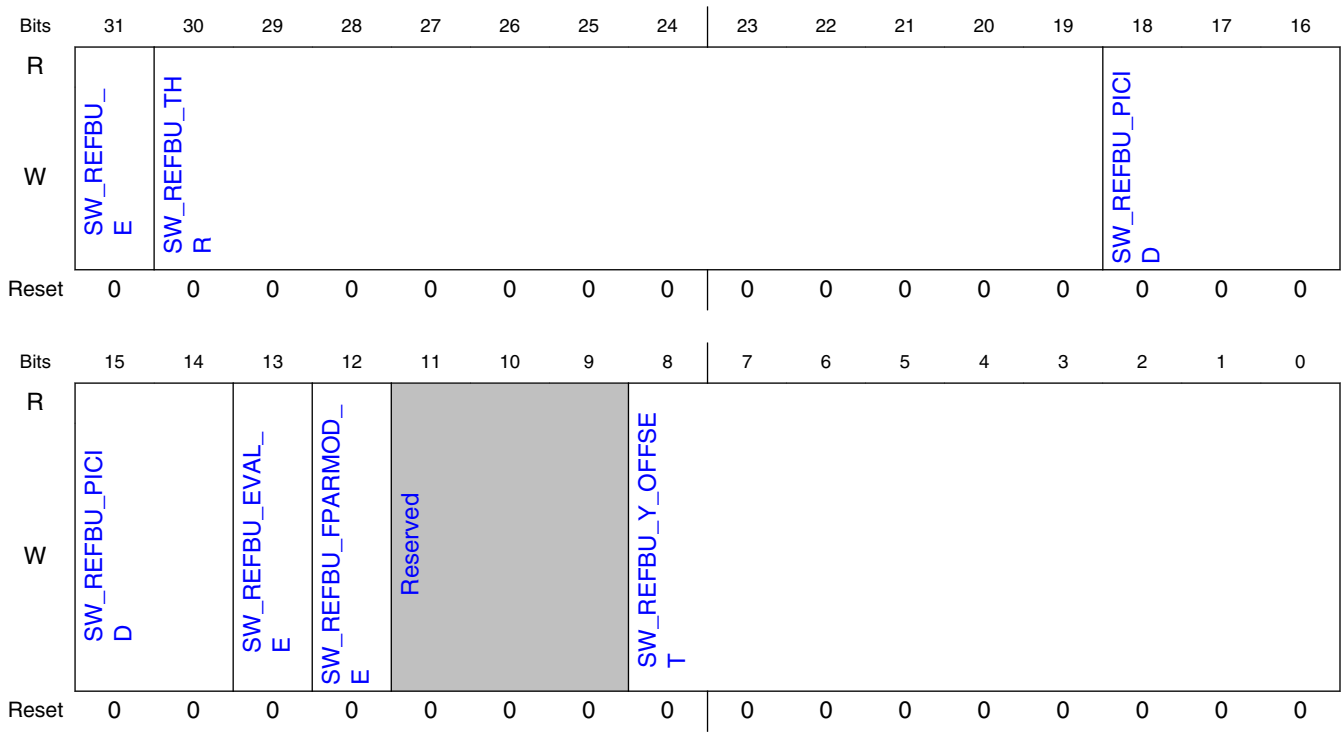
Field	Function
23 —	Reserved.
22 —	Reserved.
21 SW_DEC_OBUFF_LEVEL	Decoder output buffer level: 0b - 1 MB buffering is used 1b - 4 MB buffering is used
20 SW_REF_BUFF_EXIST	Reference picture buffer usage: 0b - not supported 1b - reference buffer is used
19-16 SW_DEC_BUS_STRD	Connected to standard bus: 0000b - error 0001b - AHB master, AHB slave 0010b - OCP master, OCP slave 0011b - AXI master, AXI slave 0100b - AXI master, APB slave 0101b - AXI master, AHB slave
15-14 SW_DEC_SYNTH_LAN	00b - error 01b - vhdl 10b - verilog
13-12 SW_DEC_BUS_WIDTH	00b - error 01b - 32 bit bus 10b - 64 bit bus 11b - 128 bit bus
11 SW_DEC_SOREN_PROF	Decoding format support, Sorenson 0b - not supported 1b - supported
10-0 SW_DEC_MAX_OWIDTH	Max configured decoder video resolution that can be decoded. Informed as width of the picture in pixels.

15.1.5.1.12 Reference picture buffer control register (SWREG51)

15.1.5.1.12.1 Offset

Register	Offset
SWREG51	CCh

15.1.5.1.12.2 Diagram



15.1.5.1.12.3 Fields

Field	Function
31 SW_REFBU_E	Refer picture buffer enable: 0b - refer picture buffer disabled 1b - refer picture buffer enabled. Valid if picture size is QVGA or more.
30-19 SW_REFBU_THR	Reference buffer disable threshold value (cache miss amount). Used to buffer shut down (if more misses than allowed)
18-14 SW_REFBU_PICID	The used reference picture ID for reference buffer usage
13 SW_REFBU_EVAL_E	Enable for HW internal reference ID calculation. If given threshold level is reached by any picture_id after first MB row, that picture_id is used for reference buffer fill for rest of the picture
12 SW_REFBU_FPARMOD_E	Field parity mode enable. Used in refbufferd evaluation mode. 0b - use the result field of the evaluation 1b - use the parity mode field
11-9 —	Reserved.
8-0 SW_REFBU_Y_OFFSET	Y offset for refbufferd. This coordinate is used to compensate the global motion of the video for better buffer hit rate

15.1.5.1.13 Reference picture buffer information register 1 (SWREG52)

15.1.5.1.13.1 Offset

Register	Offset
SWREG52	D0h

15.1.5.1.13.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SW_REFBU_HIT_SUM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SW_REFBU_INTRA_SUM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.13.3 Fields

Field	Function
31-16 SW_REFBU_HIT_SUM	The sum of the refbufferd hits of the picture. Determined for each 8x8 luminance partition of the picture. The proceeding of the HW calculation can be read during HW decoding.
15-0 SW_REFBU_INTRA_SUM	The sum of the luminance 8x8 intra partitions of the picture. The proceeding of the HW calculation can be read during HW decoding.

15.1.5.1.14 Reference picture buffer information register 2 (SWREG53)

15.1.5.1.14.1 Offset

Register	Offset
SWREG53	D4h

15.1.5.1.14.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved										SW_REFBU_Y_MV_SUM					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SW_REFBU_Y_MV_SUM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.14.3 Fields

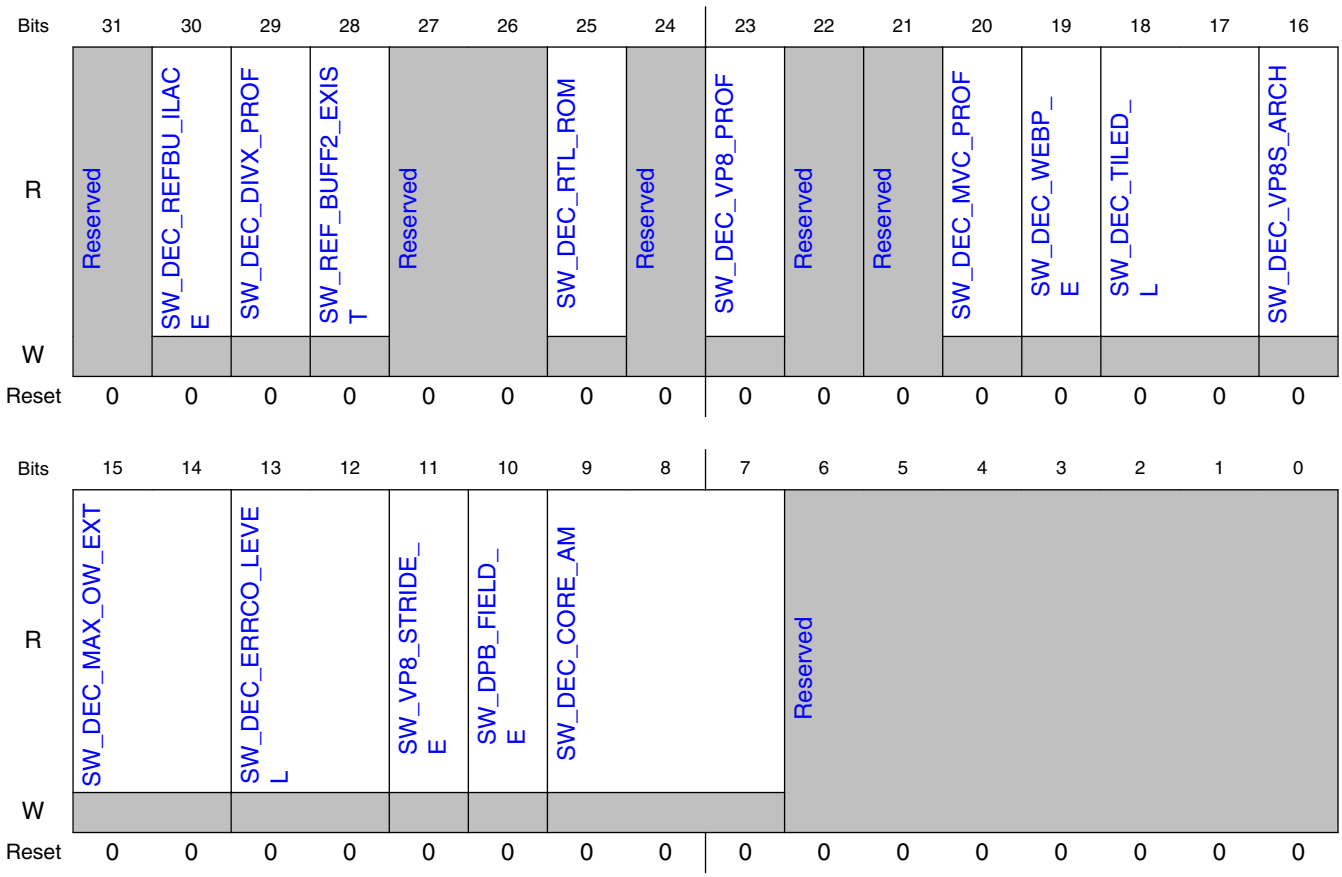
Field	Function
31-22 —	Reserved.
21-0 SW_REFBU_Y_MV_SUM	The sum of the decoded motion vector y-components of the picture. The first luminance motion vector of each MB is used in calculation. Other motion vectors of the MB are discarded. Each motion vector is saturated between -256 - 255 before calculation. The proceeding of the HW calculation can be read during HW decoding.

15.1.5.1.15 Synthesis configuration register decoder 1 (SWREG54)

15.1.5.1.15.1 Offset

Register	Offset
SWREG54	D8h

15.1.5.1.15.2 Diagram



15.1.5.1.15.3 Fields

Field	Function
31 —	Reserved.
30 SW_DEC_REFBU_ILACE	Refbufferd support for interlaced content: 0b - not supported 1b - supported
29 SW_DEC_DIVX_PROF	DIVX Support: 0b - not supported 1b - supported
28 SW_REF_BUFF2_EXIST	Reference picture buffer 2 usage: 0b - not supported 1b - reference buffer 2 is used
27-26 —	Reserved.
25	ROM implementation type (If design includes ROMs) 0b - ROMs are implemented from actual ROM units

Table continues on the next page...

Field	Function
SW_DEC_RTL_ROM	1b - ROMs are implemented from RTL
24 —	Reserved.
23 SW_DEC_VP8_PROF	Decoding format support, VP8 0b - not supported 1b - supported
22 —	Reserved.
21 —	Reserved.
20 SW_DEC_MVC_PROF	Decoding format support, MVC 0b - not supported 1b - supported
19 SW_DEC_WEBP_E	Decoding format support, Web-p 0b - not supported bigger than 1080p resolution 1b - supported upto 16kx16k pixel resolution (defined max)
18-17 SW_DEC_TILED_L	Tiled mode support level 00b - not supported 01b - supported with 8x4 tile size for progressive content 10b - supported with 8x4 tile size for progressive/ilaced content
16 SW_DEC_VP8S_ARCH	VP8 Architecture type (for prediction) 0b - Same prediction architecture as for other decoding formats 1b - Dedicated small architecture for VP8 (refbuffer cannot be used either)
15-14 SW_DEC_MAX_OW_EXT	Max configured decoder video resolution that can be decoded. This is the MSB part of the configuration signal
13-12 SW_DEC_ERRCO_LEVEL	Decoder error concealment support level: 00b - Error concealment not supported (only error detection) 01b - VP8 direct mode motion vector error concealment supported
11 SW_VP8_STRIDE_E	Decoder output stride support for VP8. Separate base addresses for Y/C data and possibility to set scanline bigger than picture width: 0b - not supported, Y and C tables attached. 1b - supported, Y and C tables can be set freely.
10 SW_DPB_FIELD_E	DPB field separate mode support for ilaced content: 0b - Not supported. For ilaced content, DPB is ilaced frame order. 1b - Supported. For ilaced content, DPB can consist of ilaced frames or separate fields (TOP/BOT).
9-7 SW_DEC_CORE_AM	Decoder core amount. If other than 0, the multicore can be used. Each individual cores can be identified from corresponding core ID register: 000b - single core decoder 001b - dual core decoder 010b - 3 core decoder 011b - 4 core decoder 100b - 5 core decoder 101b - 6 core decoder 110b - 7 core decoder 111b - 8 core decoder

Table continues on the next page...

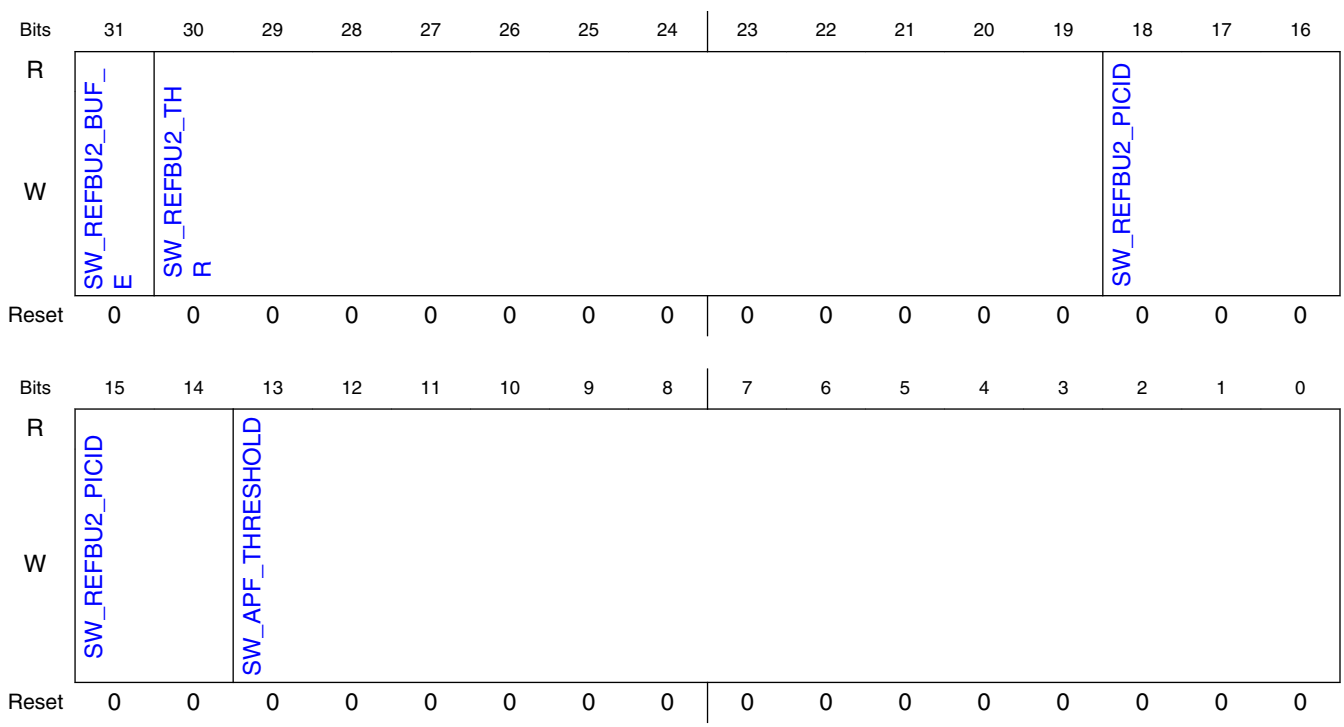
Field	Function
6-0 —	Reserved.

15.1.5.1.16 Reference picture buffer 2 / Advanced prefetch control register (SWREG55)

15.1.5.1.16.1 Offset

Register	Offset
SWREG55	DCh

15.1.5.1.16.2 Diagram



15.1.5.1.16.3 Fields

Field	Function
31 SW_REFBU2_BUF_E	Refer picture buffer 2 enable: 0b - refer picture buffer disabled

Table continues on the next page...

Field	Function
	1b - refer picture buffer enabled. Valid if picture size is QVGA or more (can be turned of by HW if threshold value reached).
30-19 SW_REFBU2_THR	Reference buffer disable threshold value (buffer miss amount). Used to buffer shut down (if more misses than allowed)
18-14 SW_REFBU2_PICID	The used reference picture ID for reference buffer usage
13-0 SW_APF_THRESHOLD	G1 decoder and later :Advanced prefetch threshold value. If current MB exceeds the threshold the advanced mode is not used. Value 0 disables threshold usage and advanced prefetch usage is restricted by internal memory limitation only

15.1.5.1.17 Reference buffer information register 3 (SWREG56)

15.1.5.1.17.1 Offset

Register	Offset
SWREG56	E0h

15.1.5.1.17.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SW_REFBU_TOP_SUM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SW_REFBU_BOT_SUM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.17.3 Fields

Field	Function
31-16 SW_REFBU_TOP_SUM	The sum of the top partitions of the picture
15-0	The sum of the bottom partitions of the picture

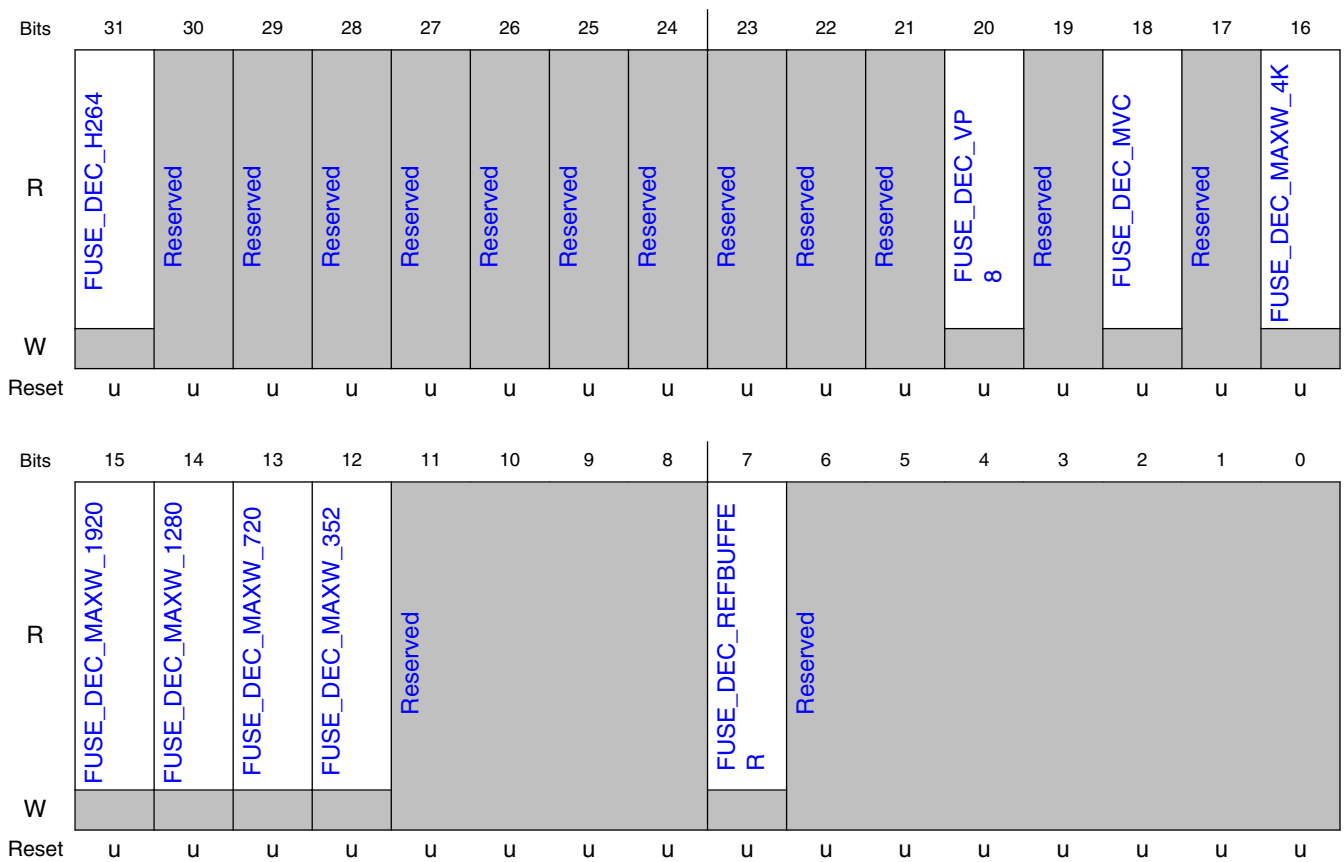
Field	Function
SW_REFBU_BOT_SUM	

15.1.5.1.18 Decoder fuse register (SWREG57)

15.1.5.1.18.1 Offset

Register	Offset
SWREG57	E4h

15.1.5.1.18.2 Diagram



15.1.5.1.18.3 Fields

Field	Function
31 FUSE_DEC_H264	1 = H.264 enabled
30 —	Reserved.
29 —	Reserved.
28 —	Reserved.
27 —	Reserved.
26 —	Reserved.
25 —	Reserved.
24 —	Reserved.
23 —	Reserved.
22 —	Reserved.
21 —	Reserved.
20 FUSE_DEC_VP8	1 = VP8 enabled
19 —	Reserved.
18 FUSE_DEC_MVC	1 = MVC enabled (requires also H264 to be enabled)
17 —	Reserved.
16 FUSE_DEC_MAXW_4K	1 = Max video width up to 4096 pixels enabled. Priority coded with priority 1.
15 FUSE_DEC_MAXW_1920	1 = Max video width up to 1920 pixels enabled. Priority coded with priority 2.

Table continues on the next page...

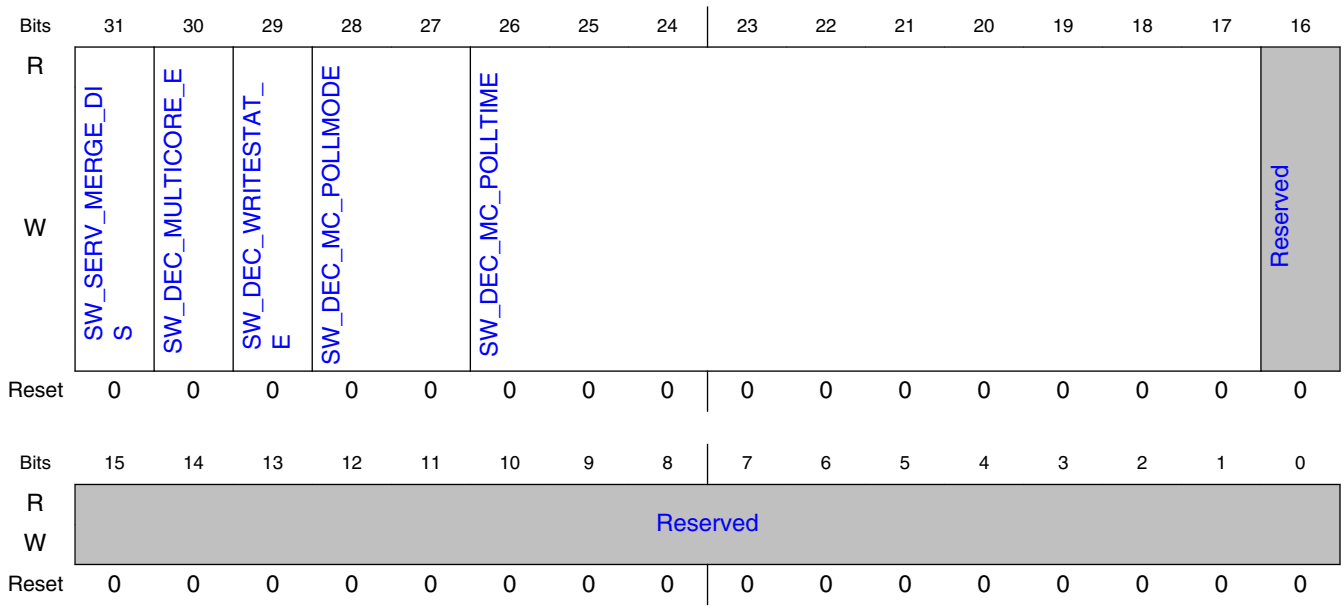
Field	Function
14 FUSE_DEC_MAXW_1280	1 = Max video width up to 1280 pixels enabled. Priority coded with priority 3.
13 FUSE_DEC_MAXW_720	1 = Max video width up to 720 pixels enabled. Priority coded with priority 4.
12 FUSE_DEC_MAXW_352	1 = Max video width up to 352 pixels enabled. Priority coded with priority 5.
11-8 —	Reserved.
7 FUSE_DEC_REFBUFFER	1 = reference buffer used
6-0 —	Reserved.

15.1.5.1.19 Device configuration register decoder 2 + Multi core control register (SWREG58)

15.1.5.1.19.1 Offset

Register	Offset
SWREG58	E8h

15.1.5.1.19.2 Diagram



15.1.5.1.19.3 Fields

Field	Function
31 SW_SERV_MERGE_DIS	Decoder service merge disable: 0b - HW merges simultaneous sub-block requests internally if they are same type (read or write). 1b - decoder serves one sub-block per service and merging is disabled.
30 SW_DEC_MULTICORE_E	Decoder multi core enable: 0b - Multi core disabled or only one core exists in design. 1b - Multi core enable. Each reference picture status must be verified from external memory status field before usage. 128 bits status word exists after each reference picture and include picture proceeding coordinates Y and X.
29 SW_DEC_WRITESTAT_E	Decoder write statusword enable. Must be high if multi core decoding enabled. HW writes output picture data proceeding to external memory after picture data (and after H264 direct mode MVS if they exist)
28-27 SW_DEC_MC_POLLMODE	Decoder multicore status reading mode: 00b - HW internal status polling mechanism is used. Status of reference picture is read only when required coordinate for the reference picture is not big enough. If the status is still not big enough after reading it the HW waits N clock cycles per pixel from the coordinate difference. The N is defined by the sw_dec_mc_polltime (range 0...4). 01b - Dummy status polling mechanism is used for all reference pictures. HW reads status of all reference pictures at frequency defined by sw_dec_mc_polltime.
26-17 SW_DEC_MC_POLLTIME	sw_dec_mc_polltime definition depends on sw_dec_mc_mode. if mode is 0 (HW internal mechanism) the sw_dec_mc_polltime: • 0 = 1/4 cycles per pixel • 1 = wait 1/2 cycles per pixel • 2 = wait one cycle per pixel • 3 = wait 2 cycles per pixel • 4=wait 4 cycles per pixel

Table continues on the next page...

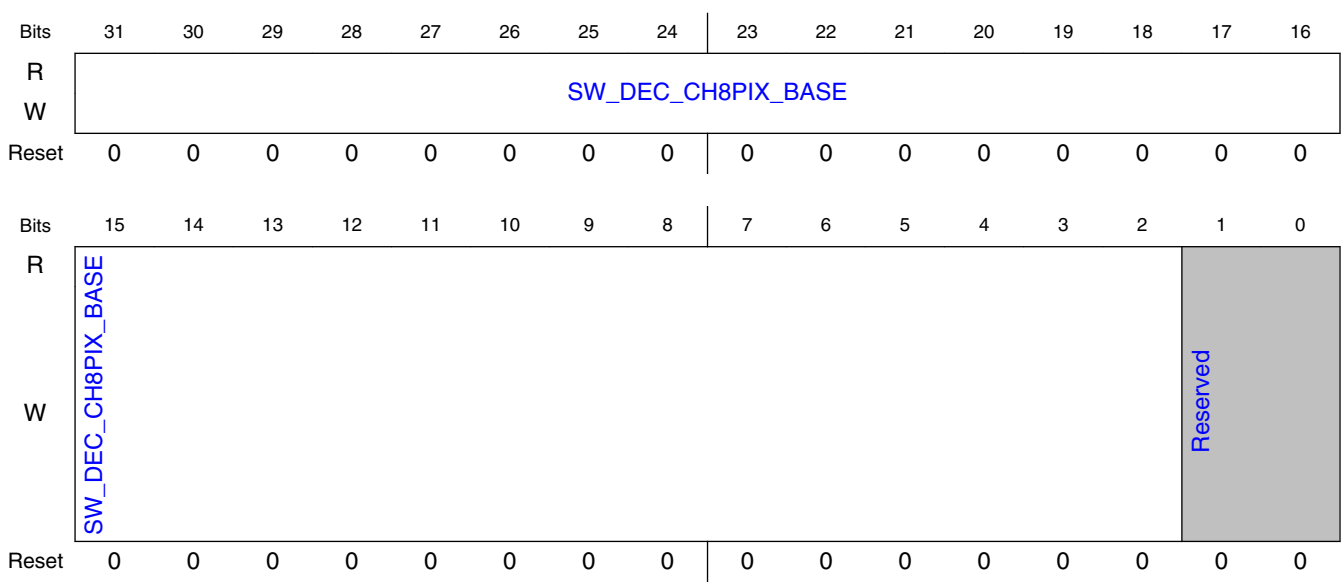
Field	Function
	For dummy polling mechanism the sw_dec_mc_polltime defines the amount cycles between status reads x1024: • 0 = wait 1024 cycles between status reading • 1 = wait 2048 cycles between status reading • 2 = wait 3072 cycles between status reading • N = wait (N+1)x1024 cycles
16-0 —	Reserved.

15.1.5.1.20 H264 Chrominance 8 pixel interleaved data base (SWREG59)

15.1.5.1.20.1 Offset

Register	Offset
SWREG59	ECh

15.1.5.1.20.2 Diagram



15.1.5.1.20.3 Fields

Field	Function
31-2	Base address for additional chrominance data format where chrominance is interleaved in group of 8 pixels. The usage is enabled by sw_ch_8pix_ileav_e

Table continues on the next page...

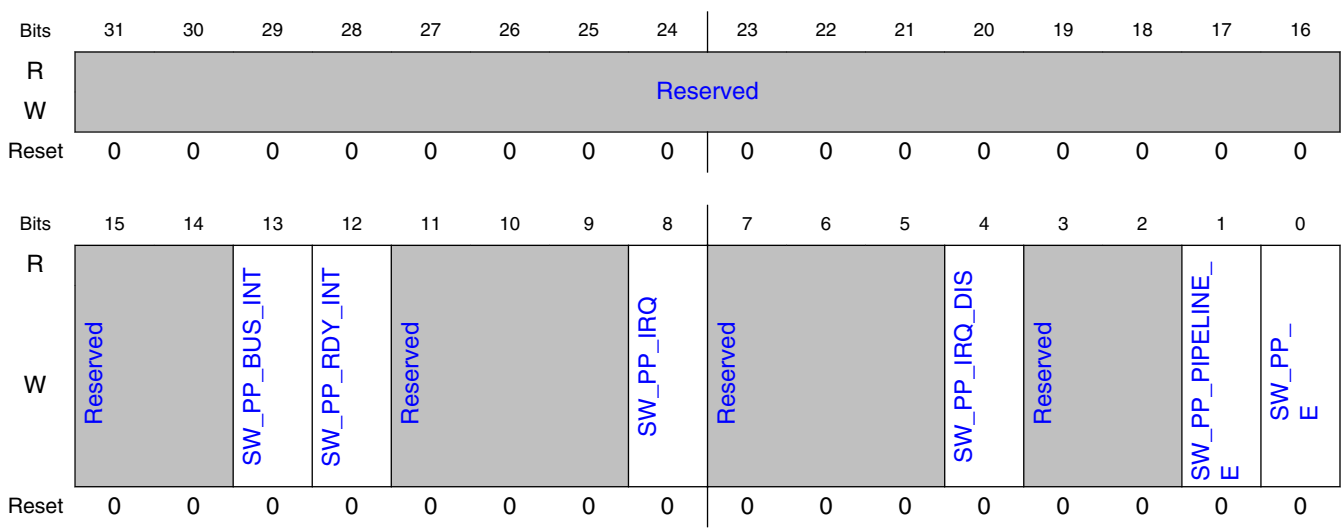
Field	Function
SW_DEC_CH8PIX_BASE	
1-0 —	Reserved.

15.1.5.1.21 Interrupt register post-processor (SWREG60)

15.1.5.1.21.1 Offset

Register	Offset
SWREG60	F0h

15.1.5.1.21.2 Diagram



15.1.5.1.21.3 Fields

Field	Function
31-14 —	Reserved.
13 SW_PP_BUS_INT	Interrupt status bit bus. Error response from bus. In pipeline mode this bit is not used

Table continues on the next page...

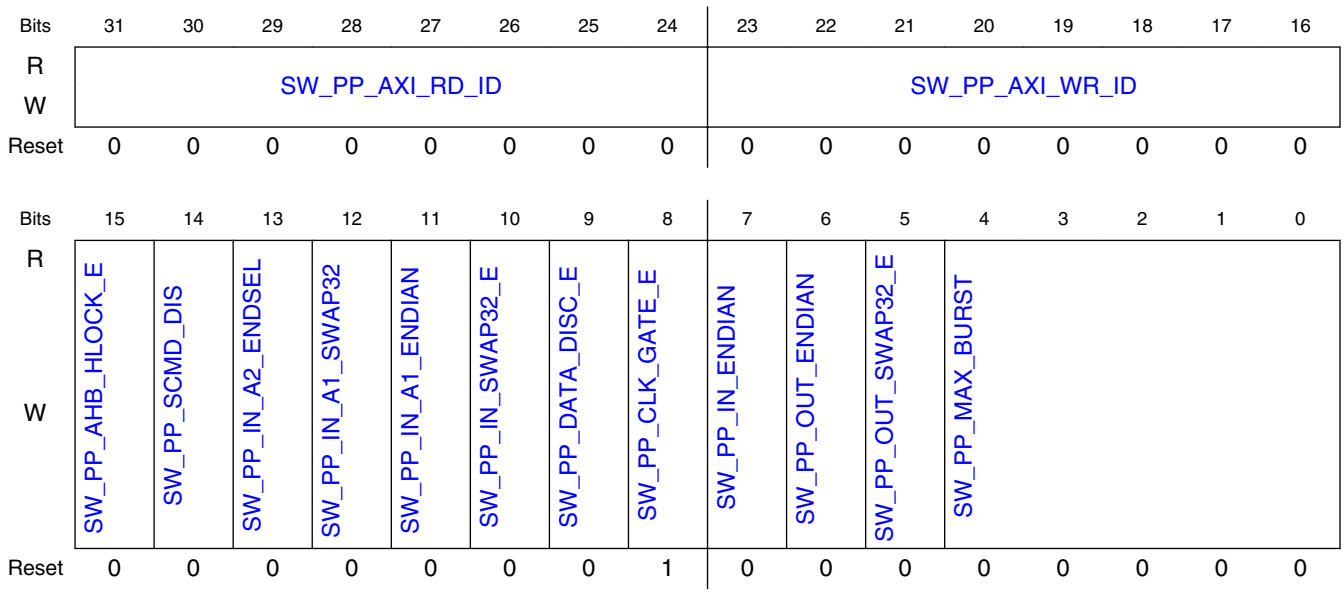
Field	Function
12 SW_PP_RDY_INT	Interrupt status bit pp. When this bit is high post processor has processed a picture in external mode. In pipeline mode this bit is not used.
11-9 —	Reserved.
8 SW_PP_IRQ	Post-processor IRQ. SW will reset this after interrupt is handled. HINTpp is not used for pp if IRQ disable pp is high (sw_pp_irq_n_e = 1). In pipeline mode this bit is not used
7-5 —	Reserved.
4 SW_PP_IRQ_DIS	Post-processor IRQ disable. When high, there are no interrupts from HW concerning post processing. Polling must be used to see the interrupt
3-2 —	Reserved.
1 SW_PP_PIPELINE_E	Decoder – post-processing pipeline enable: 0b - Post-processing is processing different picture than decoder or is disabled 1b - Post-processing is performed in pipeline with decoder
0 SW_PP_E	External mode post-processing enable. This bit will start the post-processing operation. Not to be used if PP is in pipeline with decoder (sw_pp_pipeline_e = 1). HW will reset this when picture is post-processed.

15.1.5.1.22 Device configuration register post-processor (SWREG61)

15.1.5.1.22.1 Offset

Register	Offset
SWREG61	F4h

15.1.5.1.22.2 Diagram



15.1.5.1.22.3 Fields

Field	Function
31-24 SW_PP_AXI_RD_ID	Read ID used for AXI PP read services (if connected to AXI)
23-16 SW_PP_AXI_WR_ID	Write ID used for AXI PP write services (if connected to AXI)
15 SW_PP_AHB_HLOCK_E	AHB master HLOCK enable. When high the service is locked to pp as long as it needs the bus (whenever pp requests the bus it will be granted)
14 SW_PP_SCMD_DIS	9170 decoder: AXI Single Command Multiple Data disable. 9170 axi wrapper supports this mode by default (where only the first addresses of the burst are given from address generator). This bit is used to disable the feature (possible SW workaround if something is not working correctly)
13 SW_PP_IN_A2_ENDSEL	Endian/swap select for Alpha blend input source 2: 0b - Use PP in endian/swap definitions (sw_pp_in_endian, sw_pp_in_swap) 1b - Use Ablend source 1 endian/swap definitions (sw_pp_in_a1_endian, sw_pp_in_a1_swap)
12 SW_PP_IN_A1_SWAP32	Alpha blend source 1 input 32bit data swap (may be used for 64 bit environment): 0b - no swapping of 32 bit words 1b - 32 bit data words are swapped (needed in 64 bit environment to achieve 7-6-5-4-3-2-1-0 byte order(also little endian should be enabled))
11 SW_PP_IN_A1_ENDIAN	Alpha blend source 1 input data byte endian mode. 0b - Big endian (0-1-2-3 order) 1b - Little endian (3-2-1-0 order)
10	PP input 32bit data swap (may be used for 64 bit environment):

Table continues on the next page...

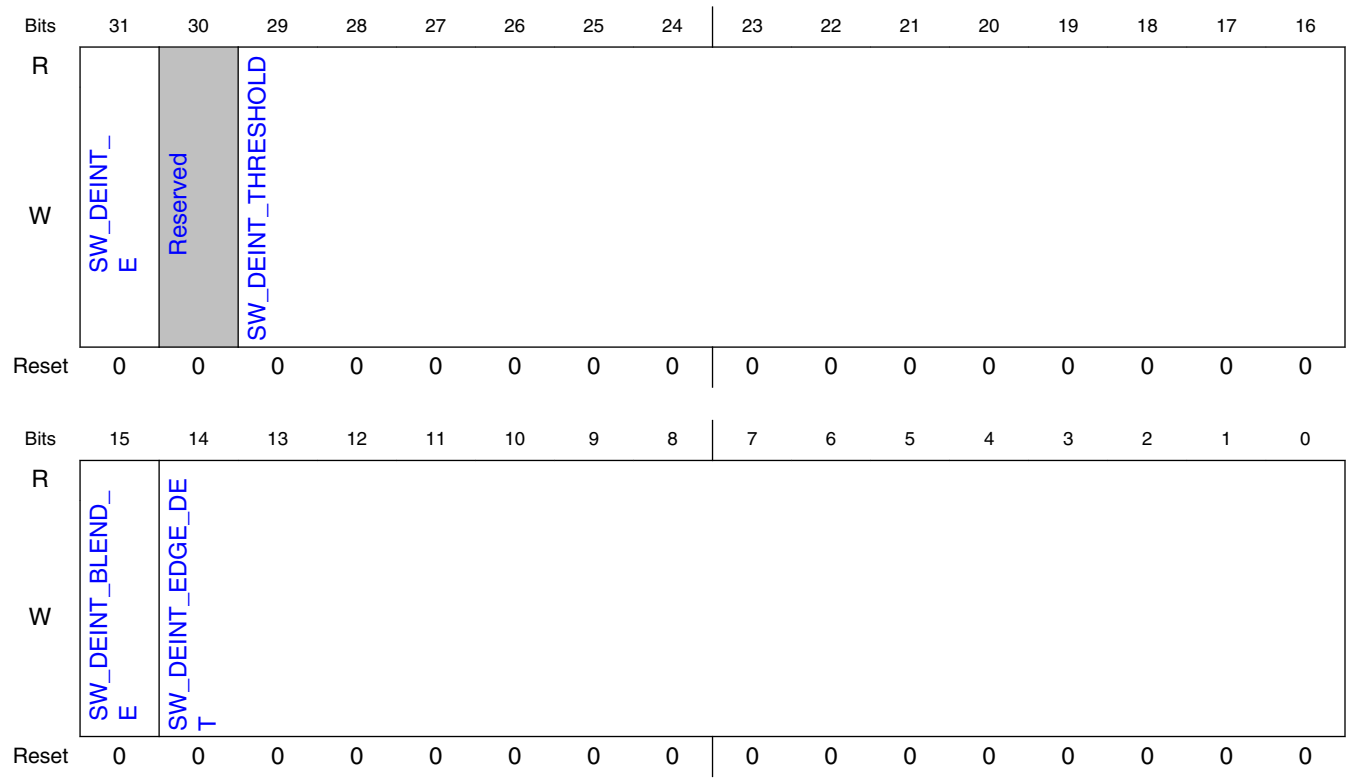
Field	Function
SW_PP_IN_SWAP32_E	0b - no swapping of 32 bit words 1b - 32 bit data words are swapped (needed in 64 bit environment to achieve 7-6-5-4-3-2-1-0 byte order(also little endian should be enabled))
9 SW_PP_DATA_DISC_E	PP data discard enable. Precise burst lengths are used with reading services. Extra data is discarded internally. Note. If AHB maxburst 17 is used data discard cannot be enabled (causes conflict)
8 SW_PP_CLK_GATE_E	PP dynamic clock gating enable. **NOTE:** Clock gating value can be changed only when PP is not enabled. 0b - Clock is running for all PP structures 1b - Clock is gated from PP structures that are not used
7 SW_PP_IN_ENDIAN	PP input picture byte endian mode. Used only if PP is in standalone mode. If PP is running pipelined with the decoder, this bit has no effect. 0b - Big endian (0-1-2-3 order) 1b - Little endian (3-2-1-0 order)
6 SW_PP_OUT_ENDIAN	PP output picture endian mode for YCbCr data or for any data if config value SW_PP_OEN_VERSION = 1. **NOTE:** For SW_PP_OEN_VERSION=0, 16 bit RGB data, this bit works as pixel swapping bit. For 32 bit RGB, this bit has no meaning. 0b - Big endian (0-1-2-3 order) 1b - Little endian (3-2-1-0 order)
5 SW_PP_OUT_SWAP32_E	PP output data word swap (may be used for 64 bit environment): 0b - no swapping of 32 bit words 1b - 32 bit data words are swapped (needed in 64 bit environment to achieve 7-6-5-4-3-2-1-0 byte order (also little endian should be enabled))
4-0 SW_PP_MAX_BURST	Maximum burst length for PP bus transactions. Valid values: AHB: 1, 4, 8 , 16 and 17. Other values will result in INCR type (undefined length) transfers. 17 = fixed bursts are used when possible. INCR bursts for others. OCP: 16-31 AXI: 1-16

15.1.5.1.23 Deinterlace control register (SWREG62)

15.1.5.1.23.1 Offset

Register	Offset
SWREG62	F8h

15.1.5.1.23.2 Diagram



15.1.5.1.23.3 Fields

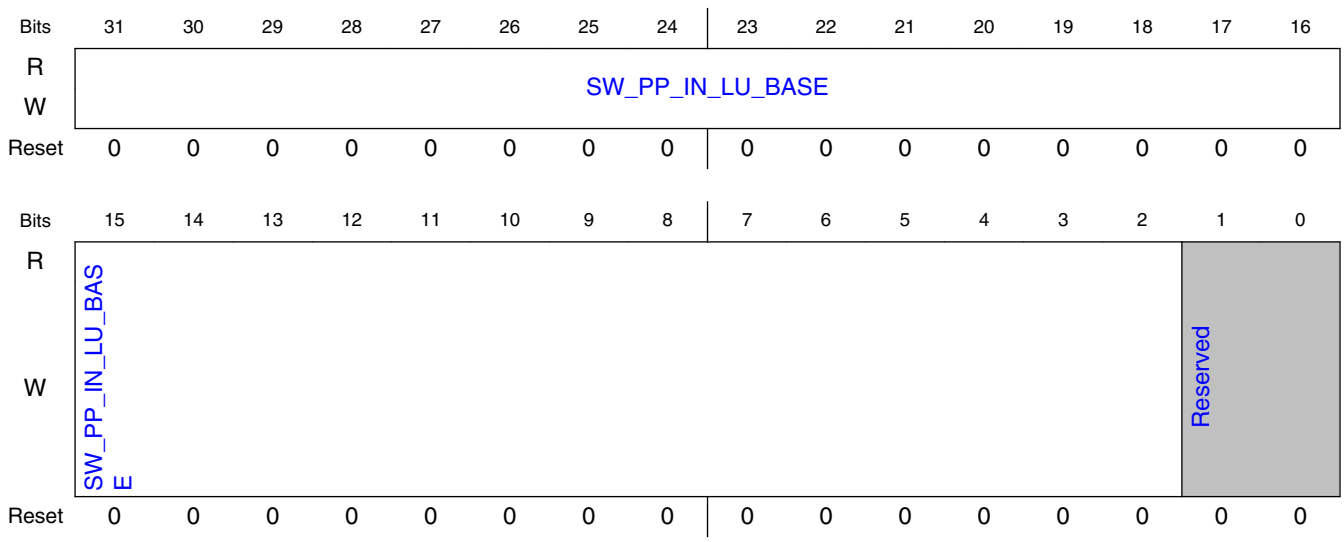
Field	Function
31 SW_DEINT_E	De-interlace enable. Input data is in interlaced format and deinterlacing needs to be performed
30 —	Reserved.
29-16 SW_DEINT_THRESHOLD	Threshold value used in deinterlacing
15 SW_DEINT_BLEND_E	Blend enable for de-interlacing
14-0 SW_DEINT_EDGE_DET	Edge detect value used for deinterlacing

15.1.5.1.24 Base address for reading post-processing input picture luminance (top field/frame) (SWREG63)

15.1.5.1.24.1 Offset

Register	Offset
SWREG63	FCh

15.1.5.1.24.2 Diagram



15.1.5.1.24.3 Fields

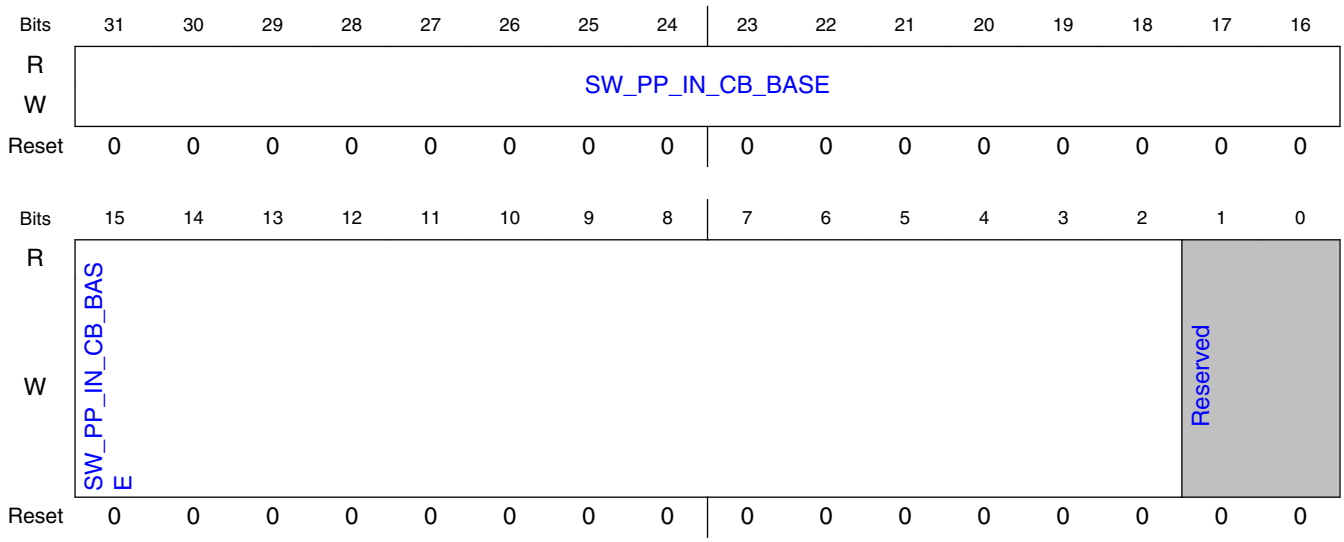
Field	Function
31-2 SW_PP_IN_LU_BASE	Base address for post-processing input luminance picture. If PP input picture is fetched from fields this base address is used to point to top field of the picture. Used in external mode only.
1-0 —	Reserved.

15.1.5.1.25 Base address for reading post-processing input picture Cb/Ch (top field/frame) (SWREG64)

15.1.5.1.25.1 Offset

Register	Offset
SWREG64	100h

15.1.5.1.25.2 Diagram



15.1.5.1.25.3 Fields

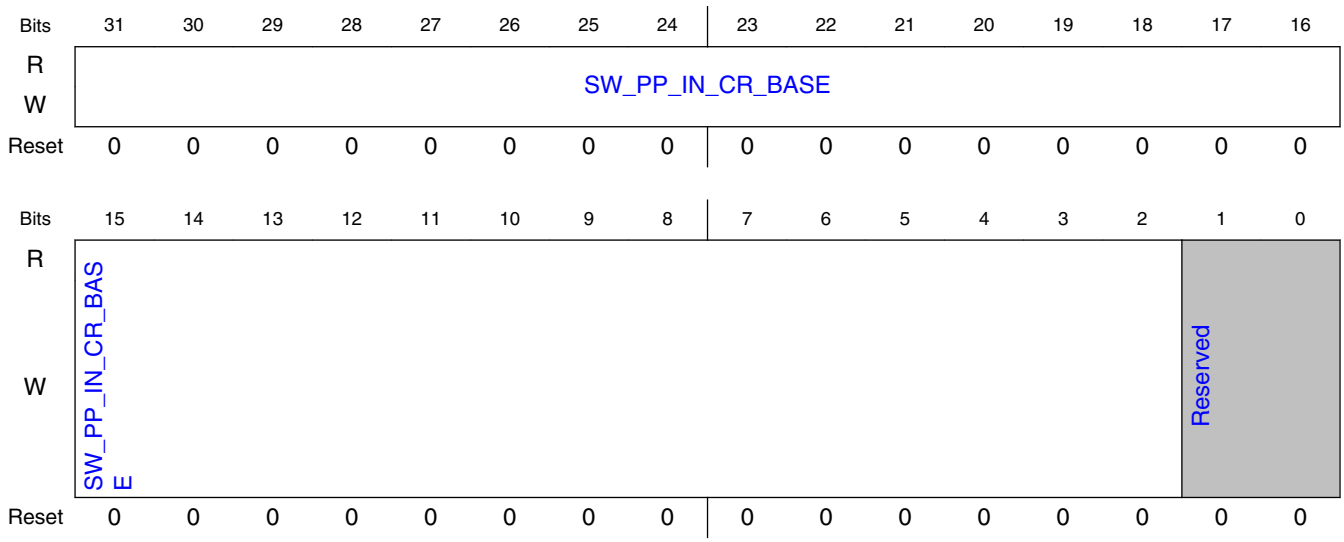
Field	Function
31-2 SW_PP_IN_CB_BASE	Base address for post-processing input Cb picture or for both chrominance pictures (if chrominances interleaved). If PP input picture is fetched from fields this base address is used to point to top field of the picture. Used in external mode only
1-0 —	Reserved.

15.1.5.1.26 Base address for reading post-processing input picture Cr (SWREG65)

15.1.5.1.26.1 Offset

Register	Offset
SWREG65	104h

15.1.5.1.26.2 Diagram



15.1.5.1.26.3 Fields

Field	Function
31-2 SW_PP_IN_CR_BASE	Base address for post-processing input cr picture. Used in external mode only
1-0 —	Reserved.

15.1.5.1.27 Base address for writing post-processed picture luminance/RGB (SWREG66)

15.1.5.1.27.1 Offset

Register	Offset
SWREG66	108h

15.1.5.1.27.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_PP_OUT_LU_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_PP_OUT_LU_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.27.3 Fields

Field	Function
31-0 SW_PP_OUT_LU_BASE	Base address for post-processing output picture (luminance/YUYV/RGB). NOTE: Bits 2:0 are used to adjust the post-processor output to start from zertain byte (1:0 for 32 bit bus). These bits can be other than zero only if Pixel Accurate PP output configuration is enabled

15.1.5.1.28 Base address for writing post-processed picture Ch (SWREG67)

15.1.5.1.28.1 Offset

Register	Offset
SWREG67	10Ch

15.1.5.1.28.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_PP_OUT_CH_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_PP_OUT_CH_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.28.3 Fields

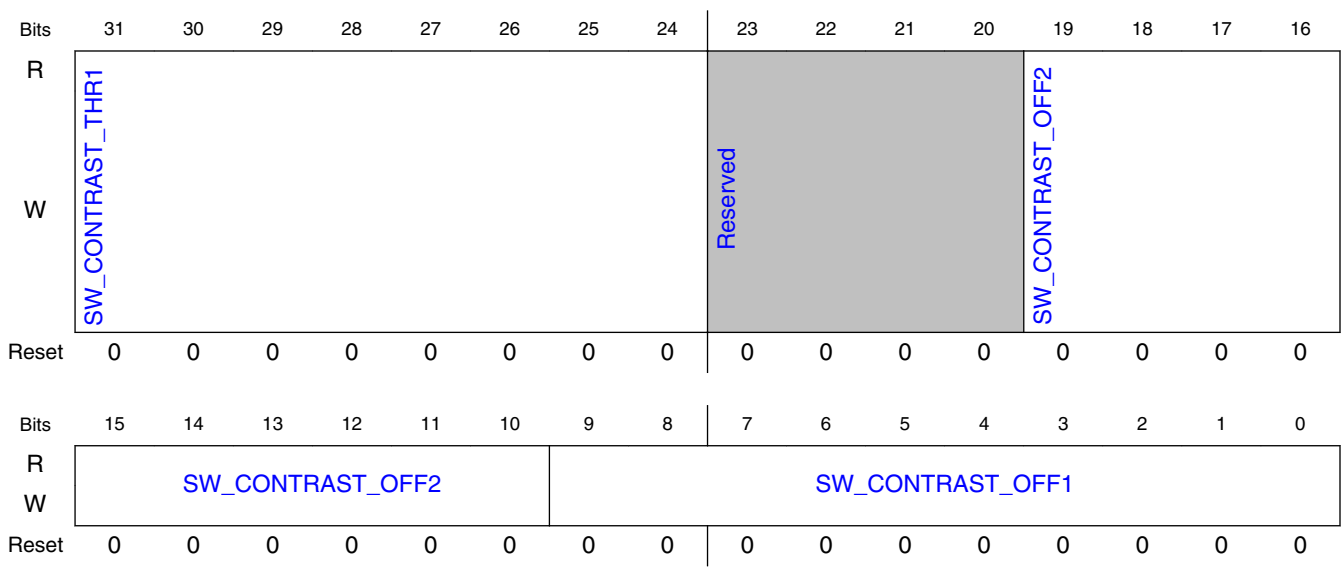
Field	Function
31-0 SW_PP_OUT_CH_BASE	Base address for post-processing output chrominance picture (interleaved chrominance). NOTE: Bits 2:0 are used to adjust the post-processor output to start from zertain byte (1:0 for 32 bit bus). These bits can be other than zero only if Pixel Accurate PP output configuration is enabled

15.1.5.1.29 Register for contrast adjusting (SWREG68)

15.1.5.1.29.1 Offset

Register	Offset
SWREG68	110h

15.1.5.1.29.2 Diagram



15.1.5.1.29.3 Fields

Field	Function
31-24 SW_CONTRAST_THR1	Threshold value 1, used with contrast adjusting

Table continues on the next page...

Field	Function
23-20 —	Reserved.
19-10 SW_CONTRAST_OFF2	Offset value 2, used with contrast adjusting
9-0 SW_CONTRAST_OFF1	Offset value 1, used with contrast adjusting

15.1.5.1.30 Register for colour conversion and contrast adjusting/YUYV 422 channel orders (SWREG69)

15.1.5.1.30.1 Offset

Register	Offset
SWREG69	114h

15.1.5.1.30.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_PP_IN_START_CH	SW_PP_IN_CR_FIRST	SW_PP_OUT_START_CH	SW_PP_OUT_CR_FIRST	SW_COLOR_COEFFA2										SW_COLOR_COEFFA1	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_COLOR_COEFFA1								SW_CONTRAST_THR2							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.30.3 Fields

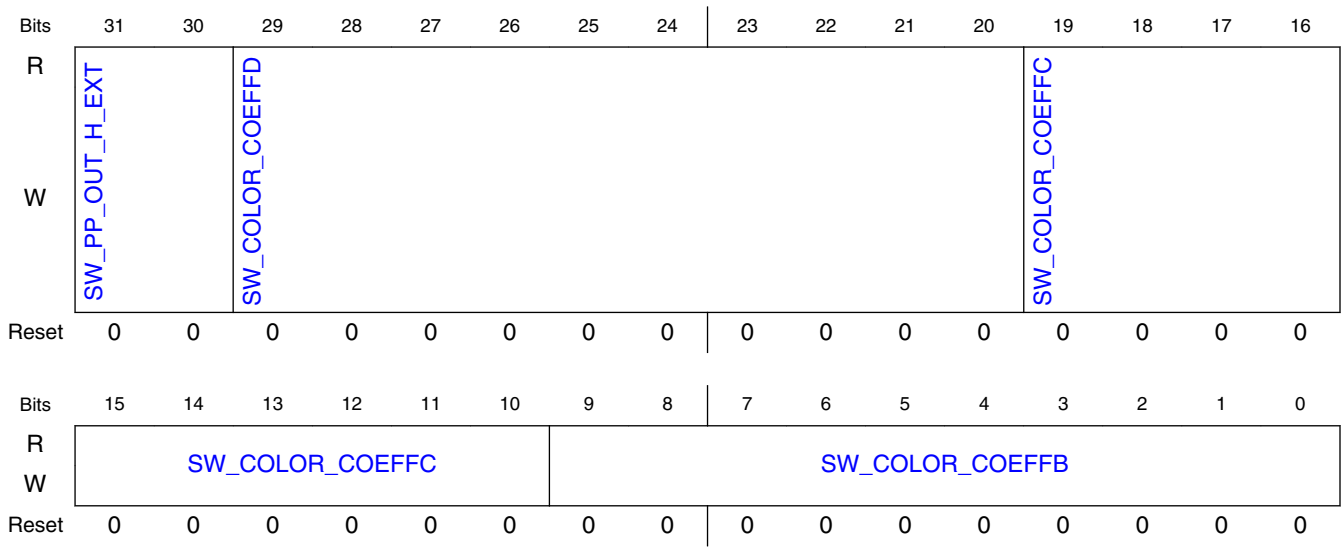
Field	Function
31 SW_PP_IN_START_CH	For YUYV 422 input format. Enable for start_with_chrominance. 0b - the order is Y0CbY0Cr or Y0CrY0Cb 1b - the order is CbY0CrY0 or CrY0CbY0
30 SW_PP_IN_CR_FIRST	For YUYV 422 input format. Enable for Cr first (before Cb). 0b - the order is Y0CbY0Cr or CbY0CrY0 (if 420 semiplanar chrominance: CbCrCbCr) 1b - the order is Y0CrY0Cb or CrY0CbY0 (if 420 semiplanar chrominance: CrCbCrCb)
29 SW_PP_OUT_START_CH	For YUYV 422 output format. Enable for start_with_chrominance. 0b - the order is Y0CbY0Cr or Y0CrY0Cb 1b - the order is CbY0CrY0 or CrY0CbY0
28 SW_PP_OUT_CR_FIRST	For YUYV 422 output format. Enable for Cr first (before Cb). 0b - the order is Y0CbY0Cr or CbY0CrY0 1b - the order is Y0CrY0Cb or CrY0CbY0
27-18 SW_COLOR_COEFFA2	Coefficient a2, used with Y pixel to calculate all color components
17-8 SW_COLOR_COEFFA1	Coefficient a1, used with Y pixel to calculate all color components
7-0 SW_CONTRAST_THR2	Threshold value 2, used with contrast adjusting

15.1.5.1.31 Register for colour conversion 0 (SWREG70)

15.1.5.1.31.1 Offset

Register	Offset
SWREG70	118h

15.1.5.1.31.2 Diagram



15.1.5.1.31.3 Fields

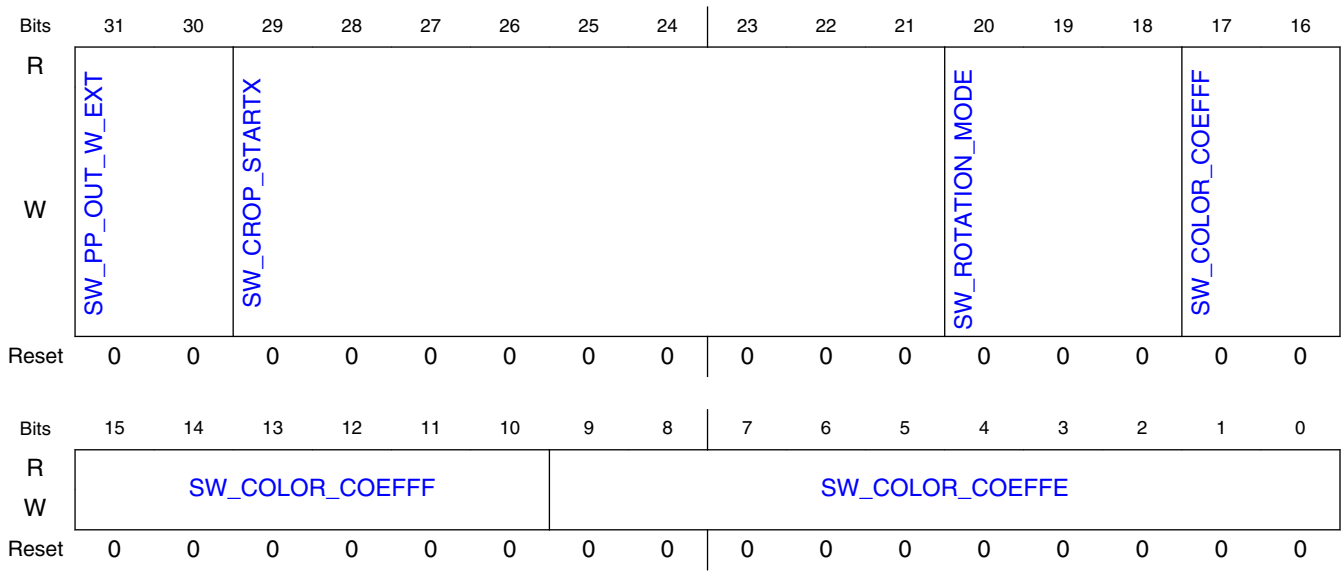
Field	Function
31-30 SW_PP_OUT_H_EXT	Extended output height for 4k resolution
29-20 SW_COLOR_COEFFD	Coefficient d, used with Cb to calculate green component value
19-10 SW_COLOR_COEFFC	Coefficient c, used with Cr to calculate green component value
9-0 SW_COLOR_COEFFB	Coefficient b, used with Cr to calculate red component value

15.1.5.1.32 Register for colour conversion 1 + rotation mode (SWREG71)

15.1.5.1.32.1 Offset

Register	Offset
SWREG71	11Ch

15.1.5.1.32.2 Diagram



15.1.5.1.32.3 Fields

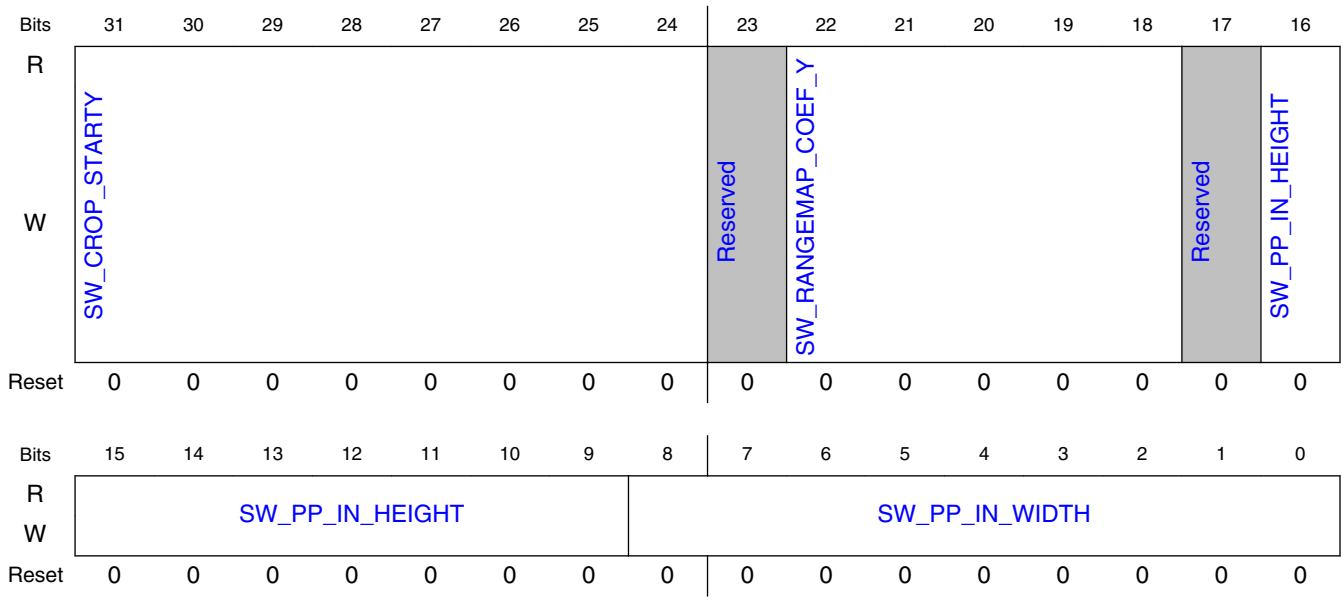
Field	Function
31-30 SW_PP_OUT_W_EXT	Extended output width for 4k resolution
29-21 SW_CROP_STARTX	Start coordinate x for the cropped area in macroblocks.
20-18 SW_ROTATION_MODE	Rotation mode: 000b - rotation disabled 001b - rotate + 90 010b - rotate – 90 011b - horizontal flip (mirror) 100b - vertical flip 101b - rotate 180
17-10 SW_COLOR_COEFFF	Coefficient f, used with Y to adjust brightness
9-0 SW_COLOR_COEFFE	Coefficient e, used with Cb to calculate blue component value

15.1.5.1.33 PP input size and -cropping register (SWREG72)

15.1.5.1.33.1 Offset

Register	Offset
SWREG72	120h

15.1.5.1.33.2 Diagram



15.1.5.1.33.3 Fields

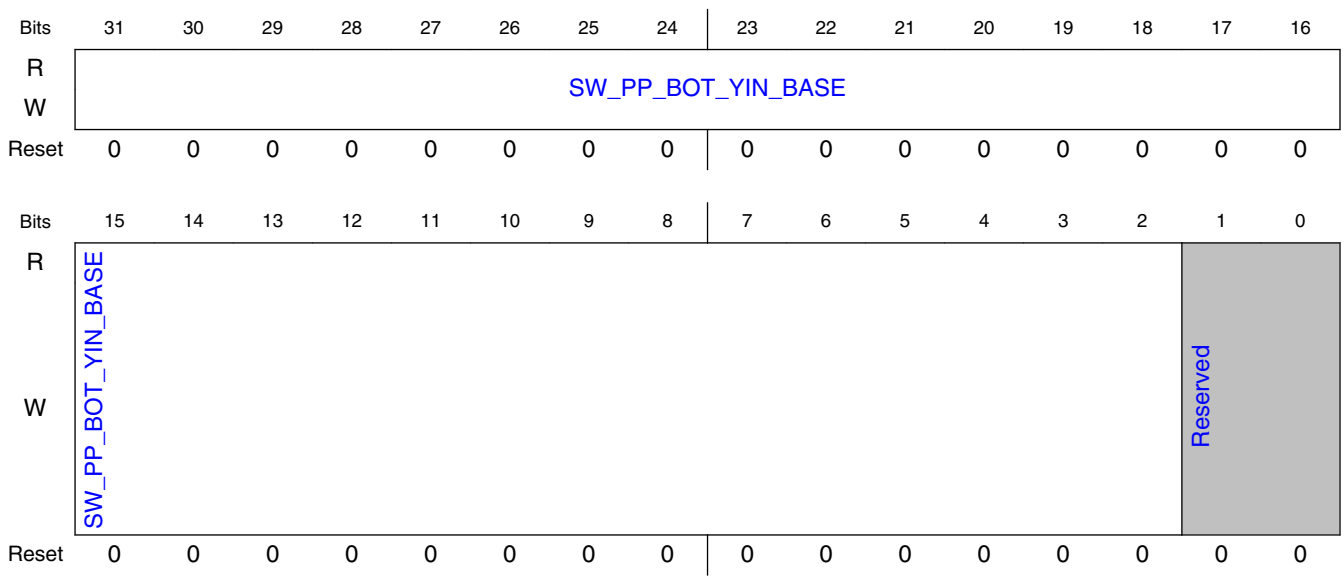
Field	Function
31-24 SW_CROP_STARTY	Start coordinate y for the cropped area in macroblocks.
23 —	Reserved.
22-18 SW_RANGEMAP_COEF_Y	Range map value for Y component
17 —	Reserved.
16-9 SW_PP_IN_HEIGHT	PP input picture height in MBs. Can be cropped from a bigger input picture in external mode
8-0 SW_PP_IN_WIDTH	PP input picture width in MBs. Can be cropped from a bigger input picture in external mode

15.1.5.1.34 PP input picture base address for Y bottom field (SWREG73)

15.1.5.1.34.1 Offset

Register	Offset
SWREG73	124h

15.1.5.1.34.2 Diagram



15.1.5.1.34.3 Fields

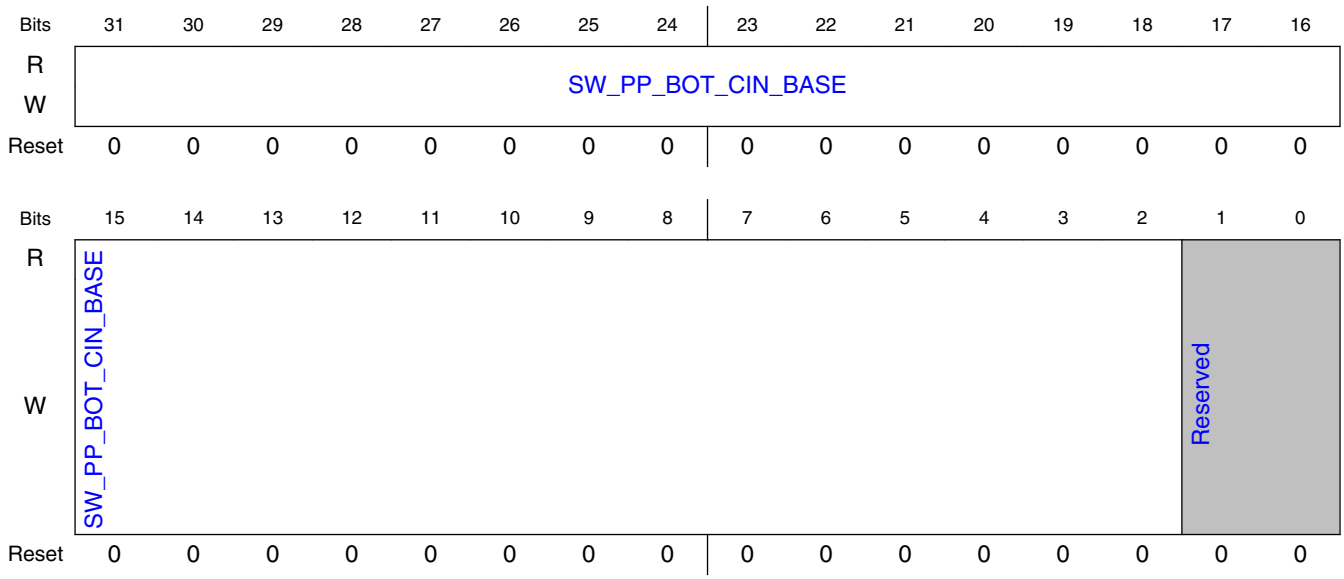
Field	Function
31-2 SW_PP_BOT_YIN_BASE	PP input Y base for bottom field
1-0 —	Reserved.

15.1.5.1.35 PP input picture base for Ch bottom field (SWREG74)

15.1.5.1.35.1 Offset

Register	Offset
SWREG74	128h

15.1.5.1.35.2 Diagram



15.1.5.1.35.3 Fields

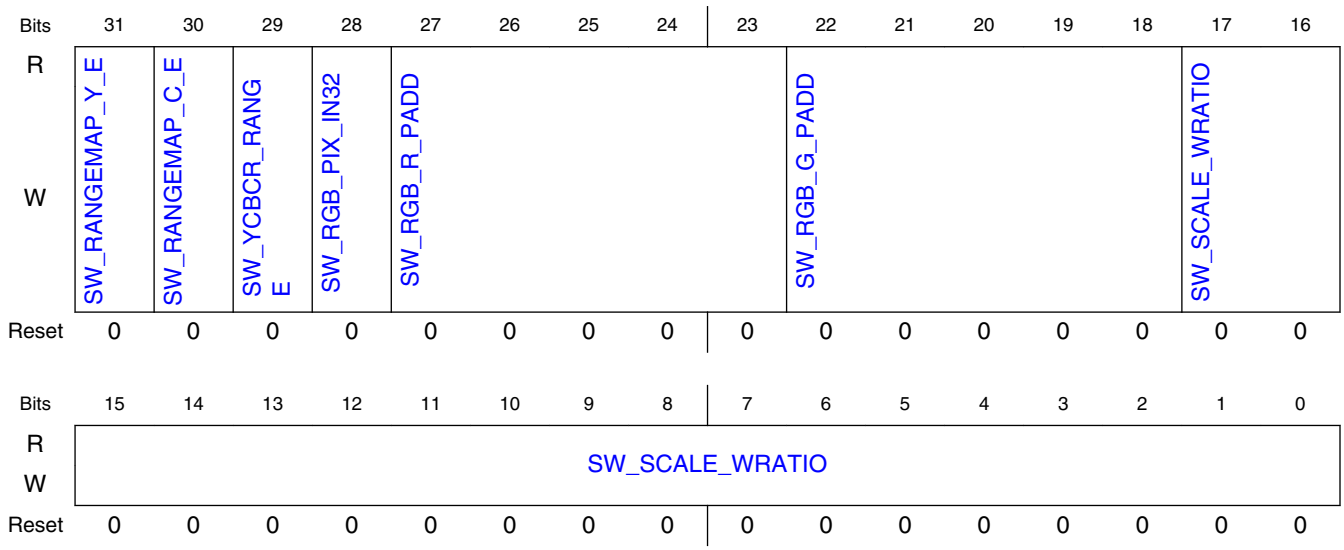
Field	Function
31-2 SW_PP_BOT_CIN_BASE	PP input C base for bottom field (mixed chrominance)
1-0 —	Reserved.

15.1.5.1.36 Scaling register 0 ratio and padding for R and G (SWREG79)

15.1.5.1.36.1 Offset

Register	Offset
SWREG79	13Ch

15.1.5.1.36.2 Diagram



15.1.5.1.36.3 Fields

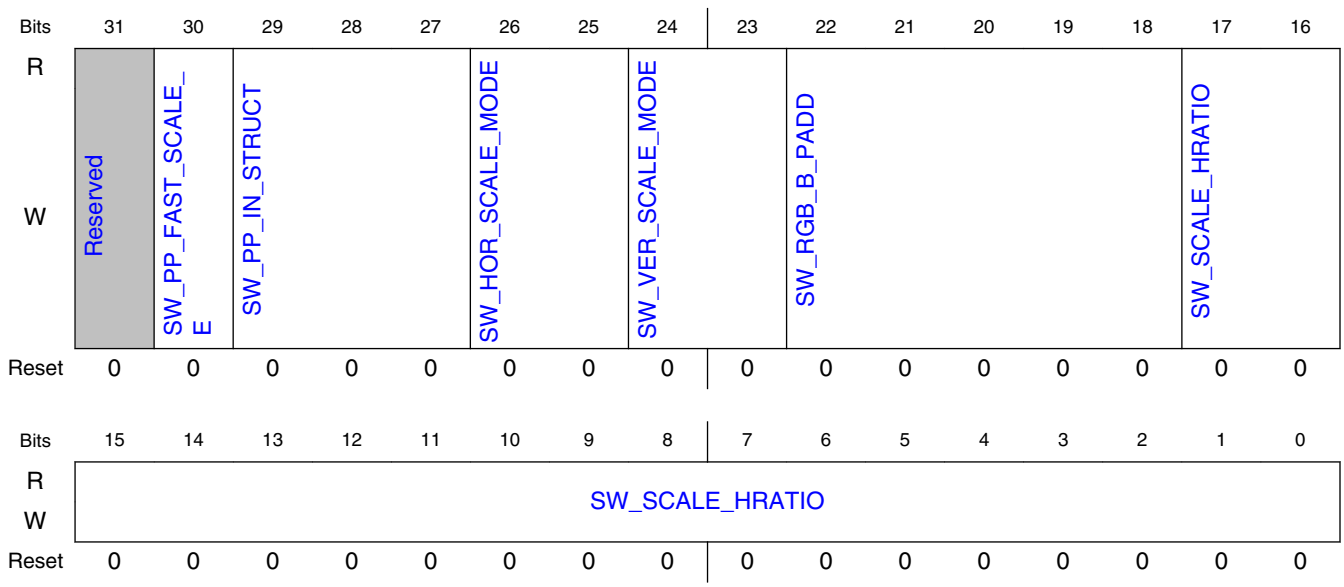
Field	Function
31 SW_RANGEMAP_Y_E	Range map enable for Y component
30 SW_RANGEMAP_C_E	Range map enable for chrominance component
29 SW_YCBCR_RANGE	Defines the YCbCr range in RGB conversion: 0b - 16…235 for Y, 16…240 for Chrominance. 1b - 0…255 for all components
28 SW_RGB_PIX_IN32	RGB pixel amount/ 32 bit word 0b - 1 RGB pixel/32 bit 1b - 2 RGB pixels/32 bit
27-23 SW_RGB_R_PADD	Amount of ones that will be padded in front of the R-component
22-18 SW_RGB_G_PADD	Amount of ones that will be padded in front of the G-component
17-0 SW_SCALE_WRATIO	Scaling ratio for width (outputw-1/inputw-1)

15.1.5.1.37 Scaling ratio register 1 and padding for B (SWREG80)

15.1.5.1.37.1 Offset

Register	Offset
SWREG80	140h

15.1.5.1.37.2 Diagram



15.1.5.1.37.3 Fields

Field	Function
31 —	Reserved.
30 SW_PP_FAST_SCALE_E	0b - fast downscaling is not enabled 1b - fast downscaling is enabled. The quality of the picture is decreased but performance is improved.
29-27 SW_PP_IN_STRUCT	PP input data picture structure: 000b - Top field / progressive frame structure: Read input data from top field base address /frame base address and read every line. 001b - Bottom field structure: Read input data from bottom field base address and read every line. 010b - Interlaced field structure: Read input data from both top and bottom field base address and take every line from each field. 011b - Interlaced frame structure: Read input data from both top and bottom field base address and take every second line from each field. 100b - Ripped top field structure: Read input data from top field base address and read every second line.

Table continues on the next page...

Field	Function
	101b - Ripped bottom field structure: Read input data from bottom field base address and read every second line.
26-25 SW_HOR_SCALE_MODE	Horizontal scaling mode: 00b - Off 01b - Upscale 10b - Downscale
24-23 SW_VER_SCALE_MODE	Vertical scaling mode: 00b - Off 01b - Upscale 10b - Downscale
22-18 SW_RGB_B_PADD	Amount of ones that will be padded in front of the B-component
17-0 SW_SCALE_HRATIO	Scaling ratio for height (outputh-1/inputh-1)

15.1.5.1.38 Scaling ratio register 2 (SWREG81)

15.1.5.1.38.1 Offset

Register	Offset
SWREG81	144h

15.1.5.1.38.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_WSCALE_INVRA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_HSCALE_INVRA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.38.3 Fields

Field	Function
31-16	Inverse scaling ratio for width, or ch (inputw-1 / outputw-1)

Table continues on the next page...

Field	Function
SW_WSCALE_INVRA	
15-0 SW_HSCALE_INVRA	Inverse scaling ratio for height or cv (inputh-1 / outputh-1)

15.1.5.1.39 Rmask register (SWREG82)

15.1.5.1.39.1 Offset

Register	Offset
SWREG82	148h

15.1.5.1.39.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_R_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_R_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.39.3 Fields

Field	Function
31-0 SW_R_MASK	Bit mask for R component (and alpha channel)

15.1.5.1.40 Gmask register (SWREG83)

15.1.5.1.40.1 Offset

Register	Offset
SWREG83	14Ch

15.1.5.1.40.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_G_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_G_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.40.3 Fields

Field	Function
31-0 SW_G_MASK	Bit mask for G component (and alpha channel)

15.1.5.1.41 Bmask register (SWREG84)

15.1.5.1.41.1 Offset

Register	Offset
SWREG84	150h

15.1.5.1.41.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_B_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_B_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.41.3 Fields

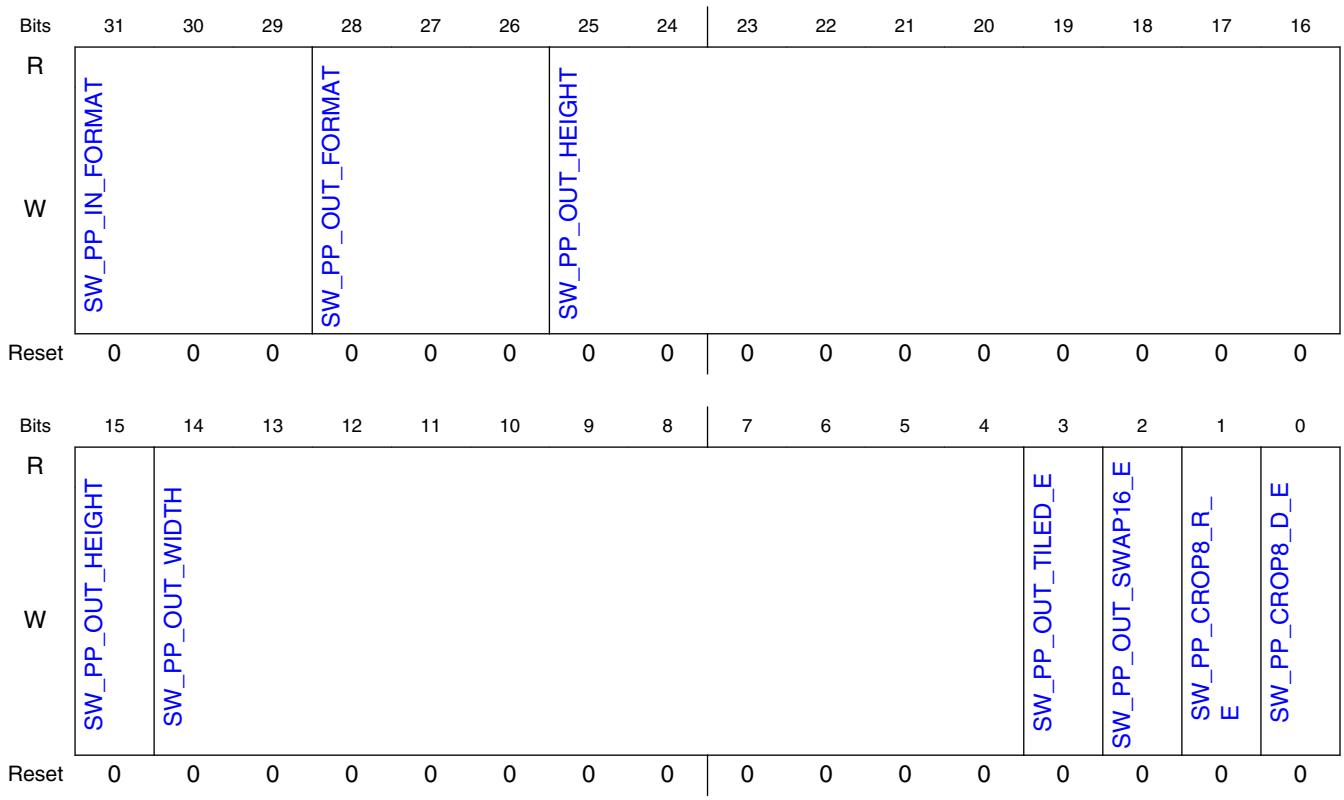
Field	Function
31-0 SW_B_MASK	Bit mask for B component (and alpha channel)

15.1.5.1.42 Post-processor control register (SWREG85)

15.1.5.1.42.1 Offset

Register	Offset
SWREG85	154h

15.1.5.1.42.2 Diagram



15.1.5.1.42.3 Fields

Field	Function
31-29 SW_PP_IN_FORMAT	PP input picture data format 000b - YUYV 4:2:2 interleaved (supported only in external mode) 001b - YCbCr 4:2:0 Semi-planar in linear raster-scan format 010b - YCbCr 4:2:0 planar (supported only in external mode) 011b - YCbCr 4:0:0 (supported only in pipelined mode) 100b - YCbCr 4:2:2 Semi-planar (supported only in pipelined mode) 101b - YCbCr 4:2:0 Semi-planar in tiled format (supported only in external mode (8170 decoder only) 110b - Reserved 111b - Escape pp input data format. Defined in swreg86.
28-26 SW_PP_OUT_FORMAT	PP output picture data format: 000b - RGB 001b - YCbCr 4:2:0 planar (Not supported) 010b - YCbCr 4:2:2 planar (Not supported) 011b - YUYV 4:2:2 interleaved 100b - YCbCr 4:4:4 planar (Not supported) 101b - YCh 4:2:0 chrominance interleaved 110b - YCh 4:2:2 (Not supported) 111b - YCh 4:4:4 (Not supported)

Table continues on the next page...

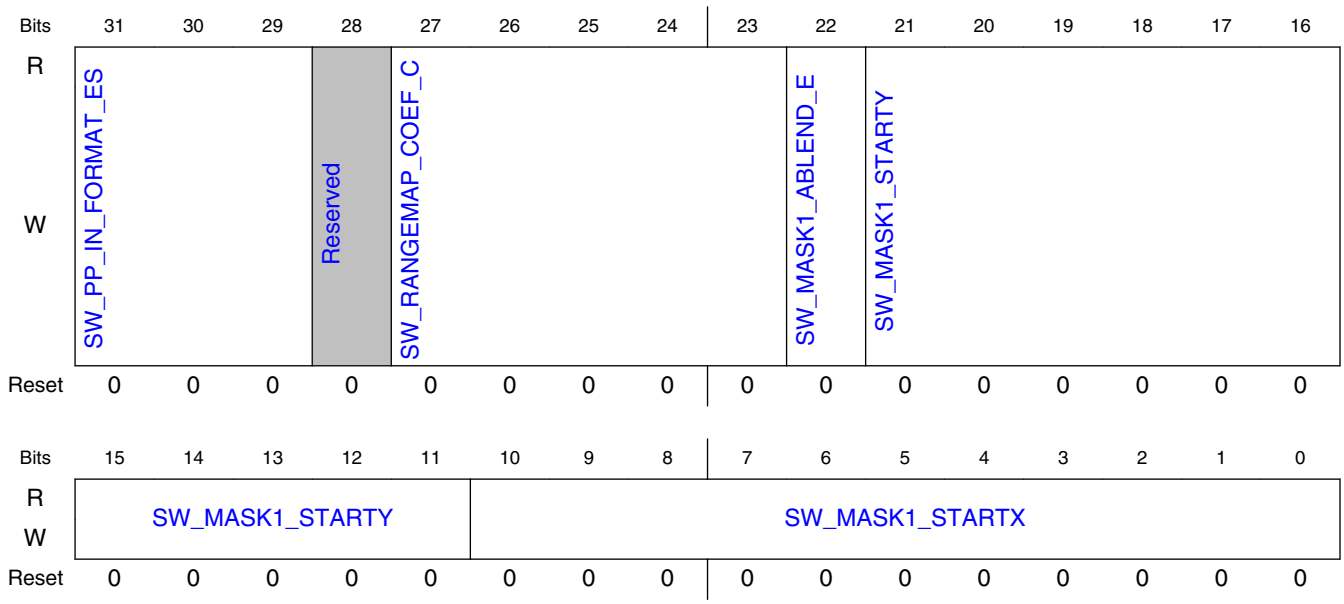
Field	Function
25-15 SW_PP_OUT_HEIGHT	Scaled picture height in pixels (Must be dividable by 2 or by any if Pixel Accurate PP output configuration is enabled) Max scaled picture height is 1920 pixels or maximum three times the input source height minus 8 pixels
14-4 SW_PP_OUT_WIDTH	Scaled picture width in pixels. Must be dividable by 8 or by any if Pixel Accurate PP output configuration is enabled. Max scaled picture width is 1920 pixels or maximum three times the input source width minus 8 pixels
3 SW_PP_OUT_TILED_E	Tiled mode enable for PP output. Can be used only for YCbYCr 422 output format. Can be used only if corresponding configuration supports this feature. Tile size is 4x4 pixels.
2 SW_PP_OUT_SWAP16_E	PP output swap 16, swaps 16 bit half inside of 32 bit word. Can be used for 16 bit RGB to change pixel orders but is valid also for any output format. NOTE: requires that configuration of SW_PPD_OEN_VERSION=1
1 SW_PP_CROP8_R_E	PP input picture width is not 16 pixels multiple. Only 8 pixels of the most right MB of the unrotated input picture is used for PP input.
0 SW_PP_CROP8_D_E	PP input picture height is not 16 pixels multiple. Only 8 pixel rows of the most down MB of the unrotated input picture is used for PP input.

15.1.5.1.43 Mask 1 start coordinate register (SWREG86)

15.1.5.1.43.1 Offset

Register	Offset
SWREG86	158h

15.1.5.1.43.2 Diagram



15.1.5.1.43.3 Fields

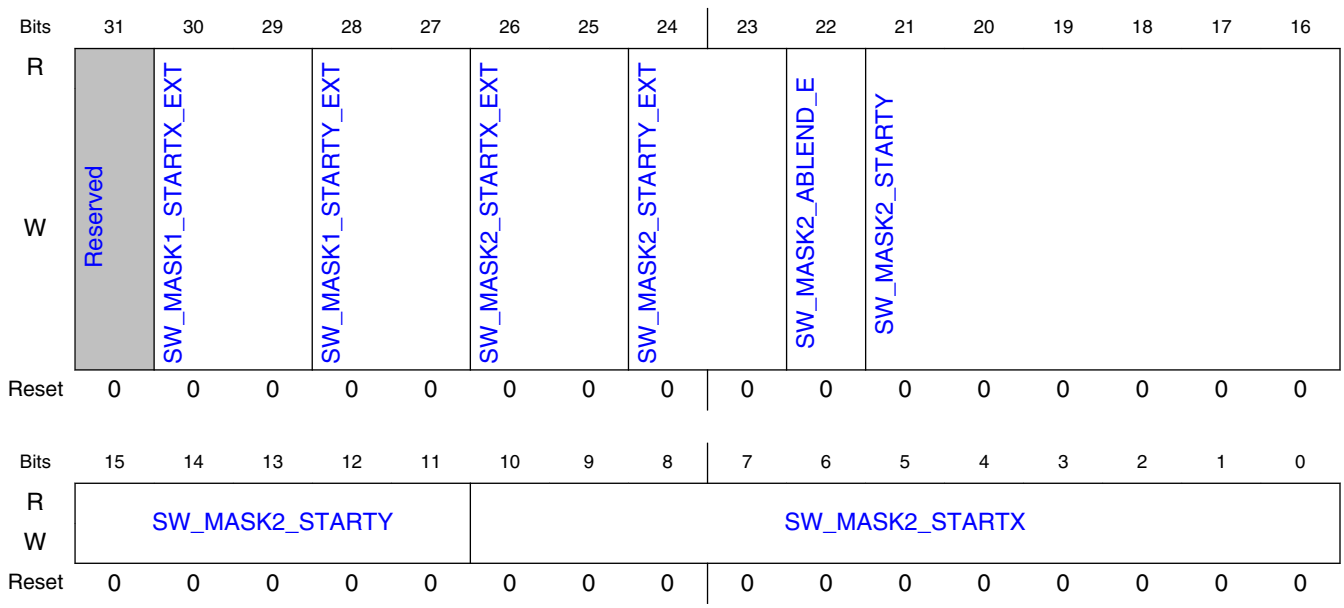
Field	Function
31-29 SW_PP_IN_FORMAT_ES	Escape PP in format. Used if sw_pp_in_format is defined to 7. 000b - YCbCr 4:4:4 001b - YCbCr 4:1:1
28 —	Reserved.
27-23 SW_RANGEMAP_COEF_C	Range map value for chrominance component
22 SW_MASK1_ABLEND_E	Mask 1 alpha blending enable. Instead of masking the output picture the alpha blending is performed. Alpha blending source can be found from alpha blend 1 base address. Alpha blending can be enabled only for RGB/ YUYV 422 data.
21-11 SW_MASK1_STARTY	Vertical start pixel for mask area 1. Defines the y coordinate. Coordinate 0,0 means the up-left corner in PP output luminance picture. See Table 47 for restrictions
10-0 SW_MASK1_STARTX	Horizontal start pixel for mask area 1. Defines the x coordinate. Coordinate 0,0 means the up-left corner in PP output luminance picture. See Table 47 for restrictions

15.1.5.1.44 Mask 2 start coordinate register + Mask extensions (SWREG87)

15.1.5.1.44.1 Offset

Register	Offset
SWREG87	15Ch

15.1.5.1.44.2 Diagram



15.1.5.1.44.3 Fields

Field	Function
31 —	Reserved.
30-29 SW_MASK1_STARTX_EXT	Extended coordinate upto 4k resolution
28-27 SW_MASK1_STARTY_EXT	Extended coordinate upto 4k resolution
26-25 SW_MASK2_STARTX_EXT	Extended coordinate upto 4k resolution

Table continues on the next page...

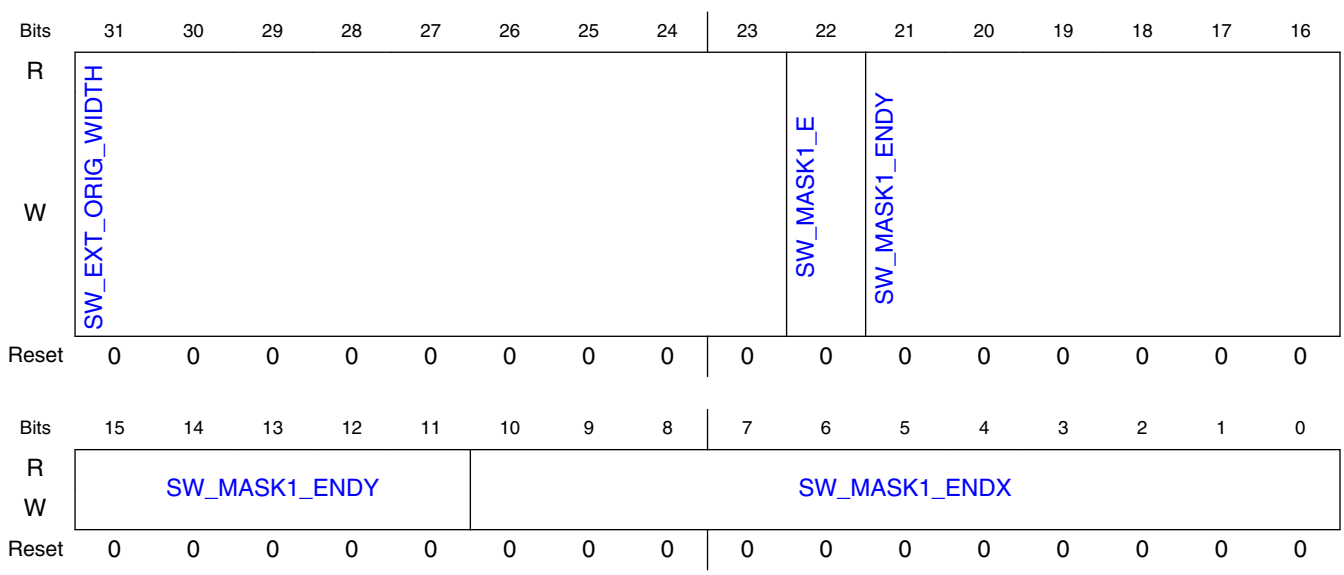
Field	Function
24-23 SW_MASK2_STARTY_EXT	Extended coordinate upto 4k resolution
22 SW_MASK2_ABLEND_E	Mask 2 alpha blending enable. Instead of masking the output picture the alpha blending is performed. Alpha blending source can be found from alpha blend 2 base address. Alpha blending can be enabled only for RGB/YUYV 422 data.
21-11 SW_MASK2_STARTY	Vertical start pixel for mask area 2. Defines the y coordinate. Coordinate 0,0 means the up-left corner in PP output Y picture. See Table 47 for restrictions
10-0 SW_MASK2_STARTX	Horizontal start pixel for mask area 2. Defines the x coordinate. Coordinate 0,0 means the up-left corner in PP output Y picture. See Table 47 for restrictions

15.1.5.1.45 Mask 1 size and PP original width register (SWREG88)

15.1.5.1.45.1 Offset

Register	Offset
SWREG88	160h

15.1.5.1.45.2 Diagram



15.1.5.1.45.3 Fields

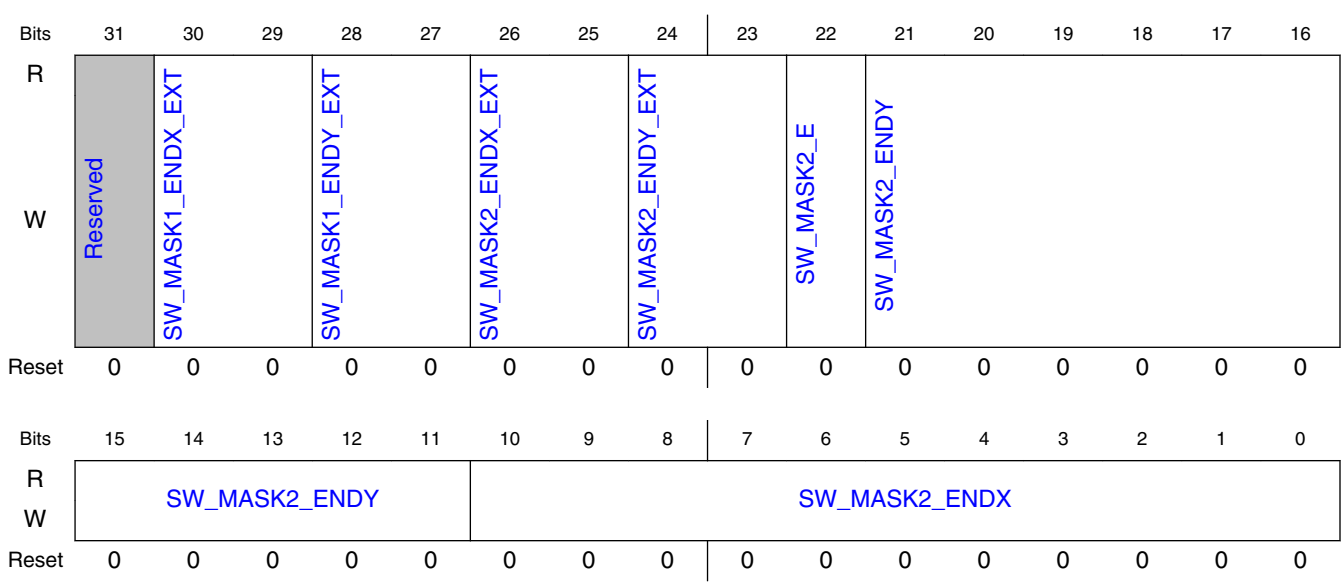
Field	Function
31-23 SW_EXT_ORIG_WIDTH	PP input picture original width in macro blocks.
22 SW_MASK1_E	Mask 1 enable. If mask 1 is used this bit is high
21-11 SW_MASK1_ENDY	Mask 1 end coordinate y in pixels (inside of PPD output picture). Range must be between [Mask1StartCoordinateY, ScaledHeight].
10-0 SW_MASK1_ENDX	Mask 1 end coordinate x in pixels (inside of PPD output picture). Range must be between [Mask1StartCoordinateX, ScaledWidth]

15.1.5.1.46 Mask 2 size register + mask extensions (SWREG89)

15.1.5.1.46.1 Offset

Register	Offset
SWREG89	164h

15.1.5.1.46.2 Diagram



15.1.5.1.46.3 Fields

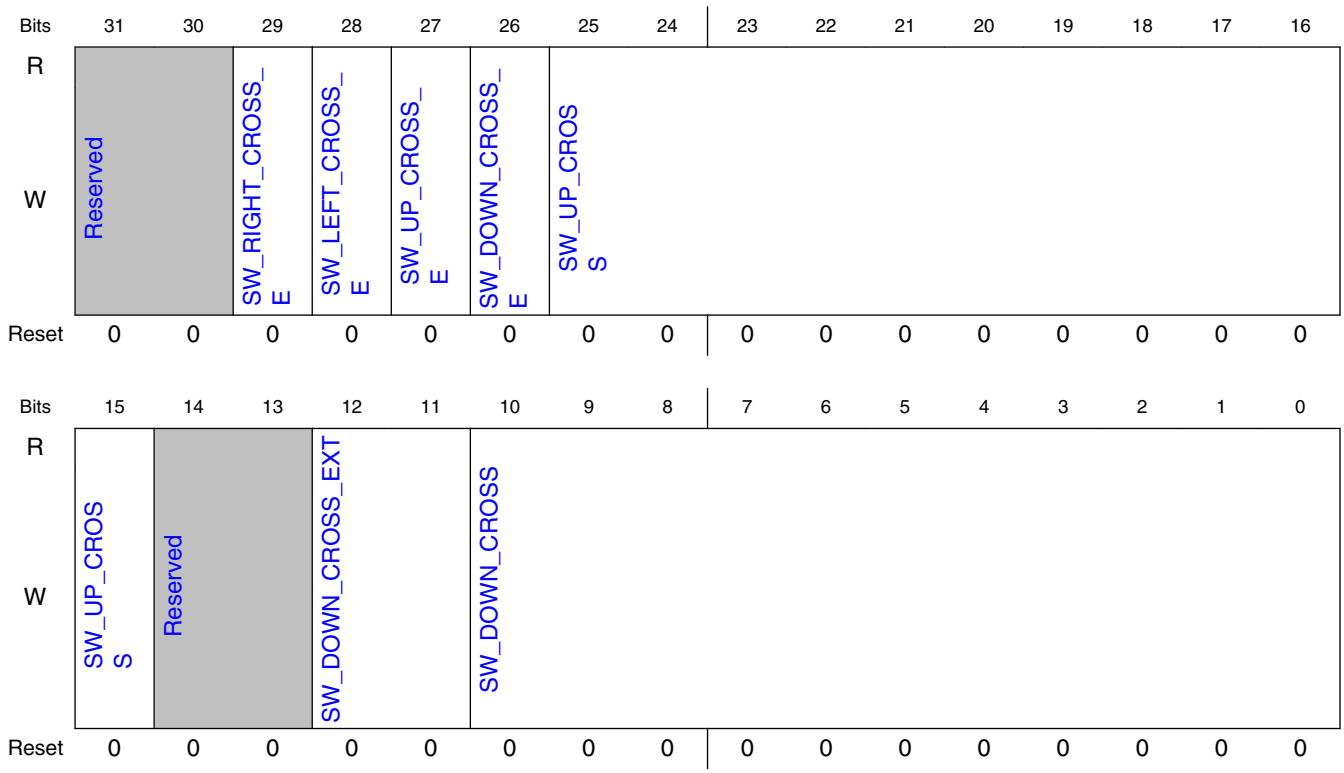
Field	Function
31 —	Reserved.
30-29 SW_MASK1_END X_EXT	Extended coordinate upto 4k resolution
28-27 SW_MASK1_END Y_EXT	Extended coordinate upto 4k resolution
26-25 SW_MASK2_END X_EXT	Extended coordinate upto 4k resolution
24-23 SW_MASK2_END Y_EXT	Extended coordinate upto 4k resolution
22 SW_MASK2_E	Mask 2 enable. If mask 1 is used this bit is high
21-11 SW_MASK2_END Y	Mask 2 end coordinate y in pixels (inside of PP output picture). Range must be between [Mask2StartCoordinateY, ScaledHeight].
10-0 SW_MASK2_END X	Mask 2 end coordinate x in pixels (inside of PP output picture). Range must be between [Mask2StartCoordinateX, ScaledWidth].

15.1.5.1.47 PiP register 0 (SWREG90)

15.1.5.1.47.1 Offset

Register	Offset
SWREG90	168h

15.1.5.1.47.2 Diagram



15.1.5.1.47.3 Fields

Field	Function
31-30 —	Reserved.
29 SW_RIGHT_CROSS_E	Right side overcross enable. 0b - No right side overcross 1b - Right side overcross
28 SW_LEFT_CROSS_E	Left side overcross enable. 0b - No left side overcross 1b - Left side overcross
27 SW_UP_CROSS_E	Upward overcross enable. 0b - No upward overcross 1b - Upward overcross
26 SW_DOWN_CROSS_E	Downward overcross enable. 0b - No downward overcross 1b - Downward overcross
25-15 SW_UP_CROSS	Amount of upward overcross (vertical pixels outside of display from the upper side). Range must be between [0, ScaledHeight].

Table continues on the next page...

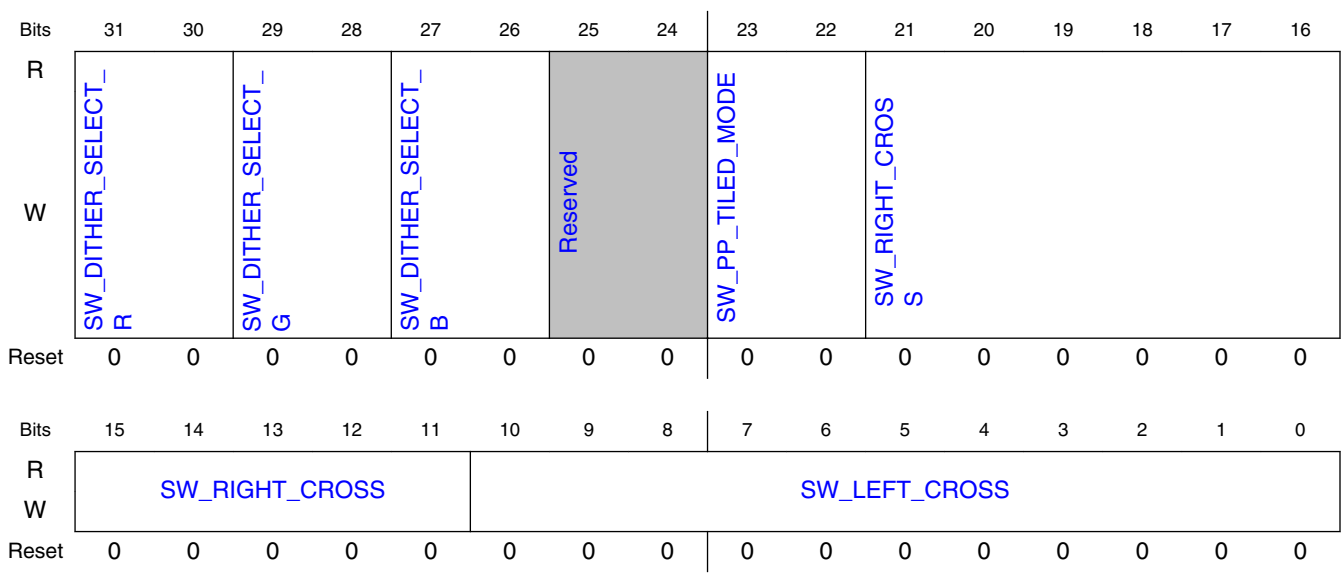
Field	Function
14-13 —	Reserved.
12-11 SW_DOWN_CROSS_EXT	Extended coordinate for 4k resolution
10-0 SW_DOWN_CROSS	Amount of downward overcross (vertical pixels outside of display from the down side). Range must be between [0, ScaledHeight].

15.1.5.1.48 PiP register 1 and dithering control (SWREG91)

15.1.5.1.48.1 Offset

Register	Offset
SWREG91	16Ch

15.1.5.1.48.2 Diagram



15.1.5.1.48.3 Fields

Field	Function
31-30	Dithering control for R channel: 00b - dithering disabled

Table continues on the next page...

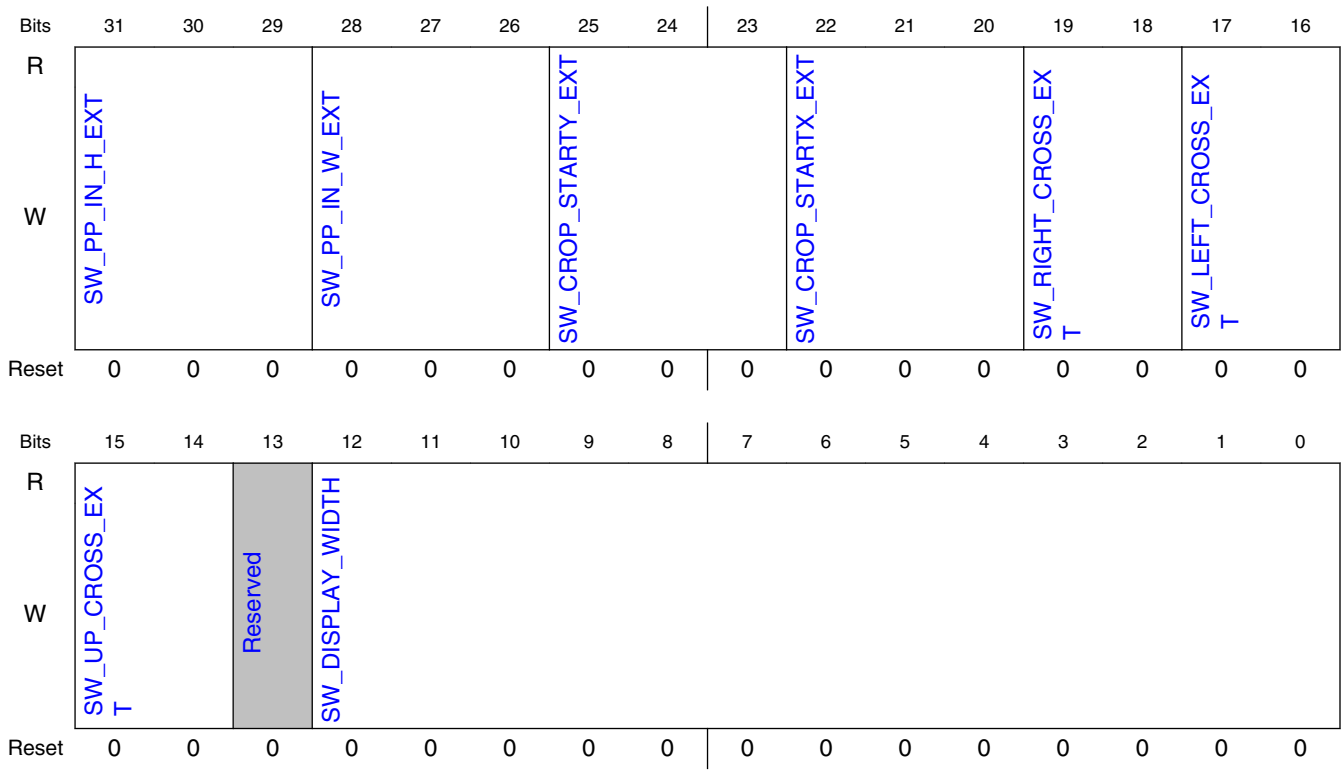
Field	Function
SW_DITHER_SELECT_R	01b - use four-bit dither matrix 10b - use five-bit dither matrix 11b - use six-bit dither matrix
29-28 SW_DITHER_SELECT_G	Dithering control for G channel: 00b - dithering disabled 01b - use four-bit dither matrix 10b - use five-bit dither matrix 11b - use six-bit dither matrix
27-26 SW_DITHER_SELECT_B	Dithering control for B channel: 00b - dithering disabled 01b - use four-bit dither matrix 10b - use five-bit dither matrix 11b - use six-bit dither matrix
25-24 —	Reserved.
23-22 SW_PP_TILED_MODE	Input data is in tiled mode (at the moment valid only for YCbCr 420 data, pipeline or external mode): 00b - Tiled mode not used 01b - Tiled mode enabled for 8x4 sized tiles
21-11 SW_RIGHT_CROSS	Amount of right side overcross (Horizontal pixels outside of display from the right side). Range must be between [0, ScaledWidth].
10-0 SW_LEFT_CROSS	Amount of left side overcross (Horizontal pixels outside of display from the left side). Range must be between [0, ScaledWidth].

15.1.5.1.49 Display width and PP input size extension register (SWREG92)

15.1.5.1.49.1 Offset

Register	Offset
SWREG92	170h

15.1.5.1.49.2 Diagram



15.1.5.1.49.3 Fields

Field	Function
31-29 SW_PP_IN_H_EXT	Extended PP input height. Used with WEBP
28-26 SW_PP_IN_W_EXT	Extended PP input width. Used with WEBP
25-23 SW_CROP_STARTY_EXT	Extended PP input crop start coordinate x. Used with WEBP
22-20 SW_CROP_STARTX_EXT	Extended PP input crop start coordinate y. Used with WEBP
19-18 SW_RIGHT_CROSS_EXT	Extended coordinate for 4k resolution
17-16 SW_LEFT_CROSS_EXT	Extended coordinate for 4k resolution

Table continues on the next page...

Field	Function
15-14 SW_UP_CROSS_EXT	Extended coordinate for 4k resolution
13 —	Reserved.
12-0 SW_DISPLAY_WIDTH	Width of the display in pixels. Max 4k (depends on HW config support)

15.1.5.1.50 Base address for alpha blend 1 gui component (SWREG93)

15.1.5.1.50.1 Offset

Register	Offset
SWREG93	174h

15.1.5.1.50.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_ABLEND1_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_ABLEND1_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.50.3 Fields

Field	Function
31-0 SW_ABLEND1_BASE	Base address for alpha blending input 1 (if mask1 is used in alpha blending mode). Format of data is 24 bit RGB/ YCbCr and endian/swap -mode is as in PP input. Amount of data is informed with mask 1 size or with ablend1_scanline if ablend cropping is supported in configuration.

15.1.5.1.51 Base address for alpha blend 2 gui component (SWREG94)

15.1.5.1.51.1 Offset

Register	Offset
SWREG94	178h

15.1.5.1.51.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_ABLEND2_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_ABLEND2_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.1.51.3 Fields

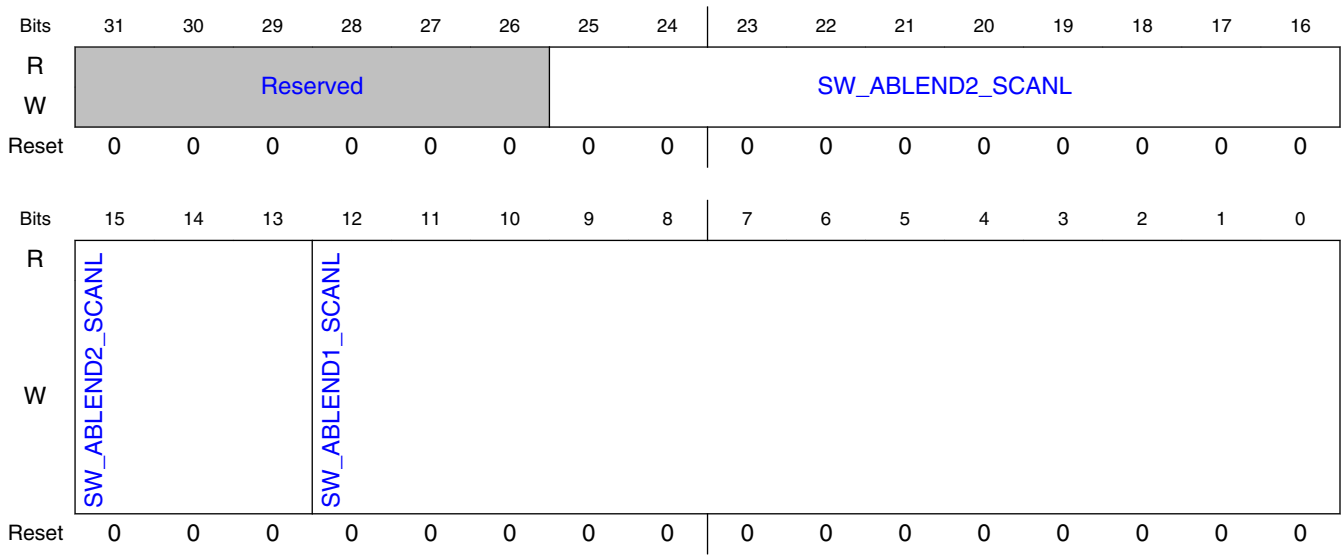
Field	Function
31-0 SW_ABLEND2_BASE	Base address for alpha blending input 2 (if mask2 is used in alpha blending mode). Format of data is 24 bit RGB/ YCbCr and endian/swap -mode is as in PP input. Amount of data is informed with mask 2 size or with ablend2_scanline if ablend cropping is supported in configuration.

15.1.5.1.52 Alpha blend input cropping register (scanline for cropping) (SWREG95)

15.1.5.1.52.1 Offset

Register	Offset
SWREG95	17Ch

15.1.5.1.52.2 Diagram



15.1.5.1.52.3 Fields

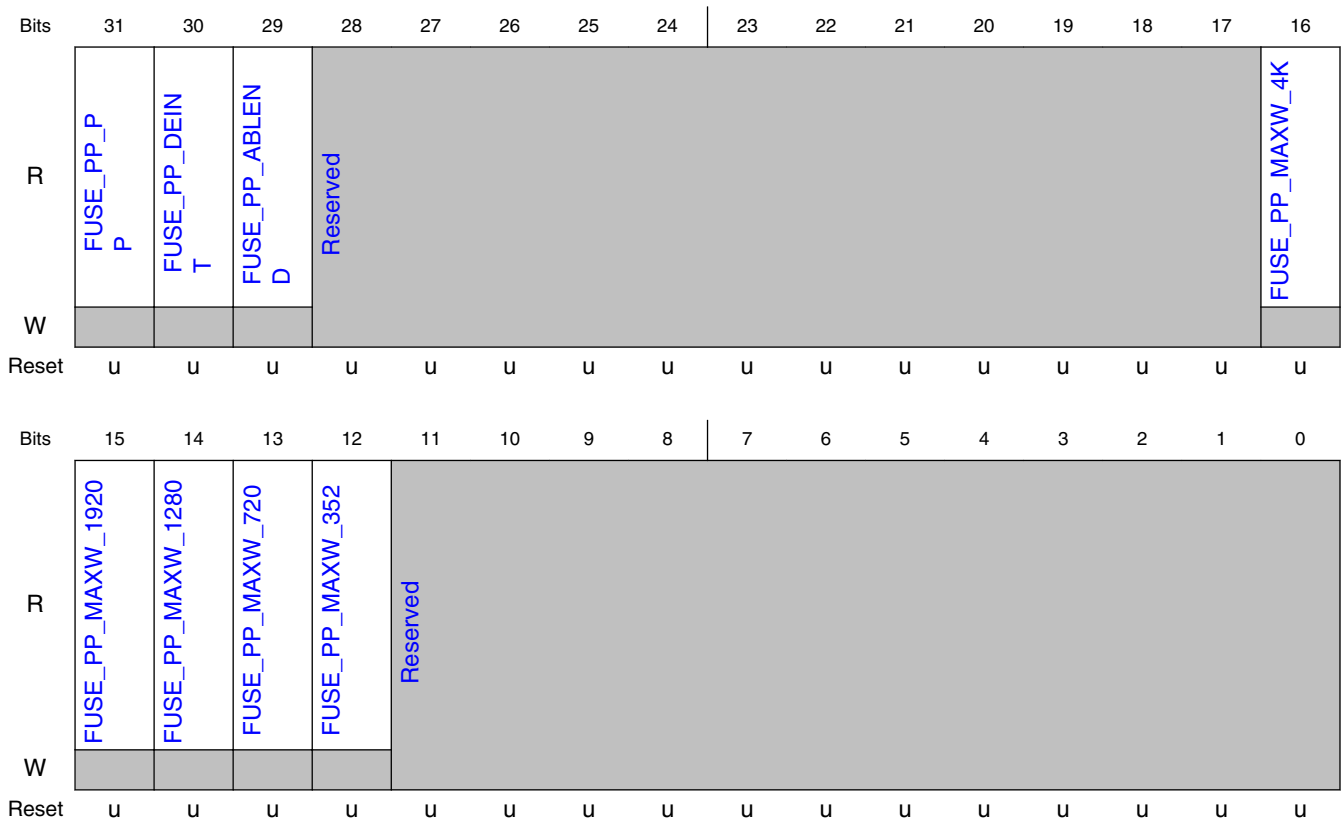
Field	Function
31-26 —	Reserved.
25-13 SW_ABLEND2_SCANL	Scanline width in pixels for Ablend 2. Usage enabled if corresponding configuration bit is enabled
12-0 SW_ABLEND1_SCANL	Scanline width in pixels for Ablend 1. Usage enabled if corresponding configuration bit is enabled

15.1.5.1.53 PP fuse register (SWREG99)

15.1.5.1.53.1 Offset

Register	Offset
SWREG99	18Ch

15.1.5.1.53.2 Diagram



15.1.5.1.53.3 Fields

Field	Function
31 FUSE_PP_PP	1 = PP enabled
30 FUSE_PP_DEINT	1 = Deinterlacing enabled
29 FUSE_PP_ABLEND	1 = Alpha Blending enabled
28-17 —	Reserved.
16 FUSE_PP_MAXW_4K	1 = Max PP output width up to 4096 pixels enabled. Priority coded with priority 1
15 FUSE_PP_MAXW_1920	1 = Max PP output width up to 1920 pixels enabled. Priority coded with priority 2

Table continues on the next page...

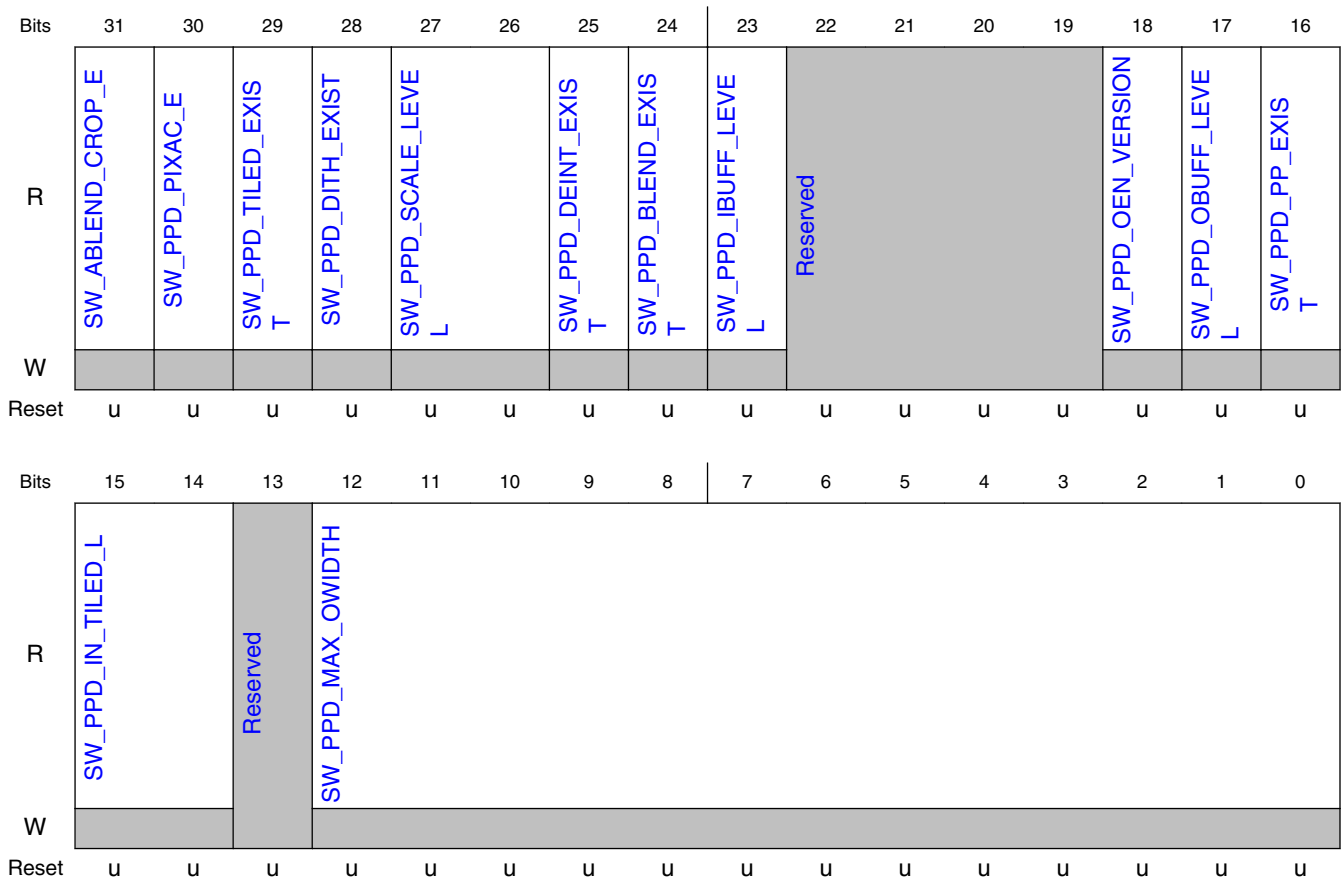
Field	Function
14 FUSE_PP_MAX W_1280	1 = Max PP output width up to 1280 pixels enabled. Priority coded with priority 3
13 FUSE_PP_MAX W_720	1 = Max PP output width up to 720 pixels enabled. Priority coded with priority 4
12 FUSE_PP_MAX W_352	1 = Max PP output width up to 352 pixels enabled. Priority coded with priority 5
11-0 —	Reserved.

15.1.5.1.54 Synthesis configuration register post-processor (SWREG100)

15.1.5.1.54.1 Offset

Register	Offset
SWREG100	190h

15.1.5.1.54.2 Diagram



15.1.5.1.54.3 Fields

Field	Function
31 SW_ABLEND_CROP_E	Alpha blending support for input cropping: 0b - Not supported. External memory must include the exact image of the area being alpha blended. 1b - Supported. External memory can include a picture from blended area can be cropped. Requires usage of swreg95.
30 SW_PPD_PIXAC_E	Pixel Accurate PP output mode exists: 0b - PIP, Scaling and masks can be adjusted by steps of 8 pixels (width) or 2 pixels (height) 1b - PIP, Scaling and masks can be adjusted by steps of 1 pixel for RGB and 2 pixels for subsampled chroma formats (by using bus specific write strobe functionality)
29 SW_PPD_TILED_EXIST	PP output YCbYCr 422 tiled support (4x4 pixel tiles) 0b - Not supported 1b - Supported
28 SW_PPD_DITH_EXIST	Dithering exists: 0b - No 1b - Yes
27-26	Scaling support:

Table continues on the next page...

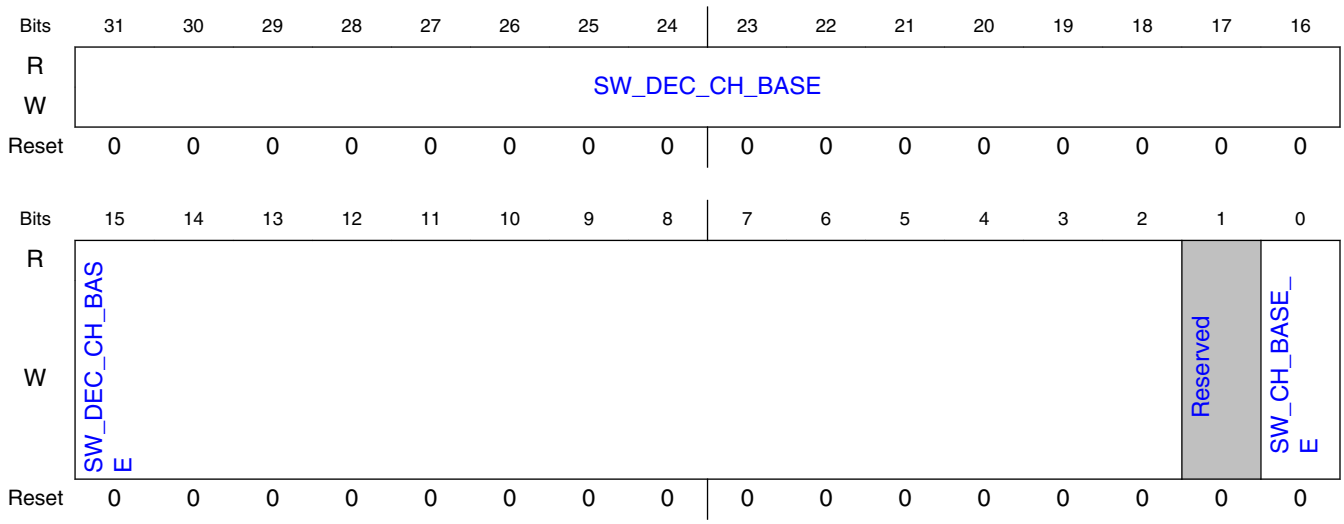
Field	Function
SW_PPD_SCALE_LEVEL	00b - No scaling 01b - Scaling with lo performance architecture 10b - Scaling with high performance architecture 11b - Scaling with high performance architecture + fast downscaling enabled
25 SW_PPD_DEINT_EXIST	De-interlacing exits: 0b - No 1b - Yes
24 SW_PPD_BLEND_EXIST	Alpha blending exists: 0b - No 1b - Yes
23 SW_PPD_IBUFF_LEVEL	PP input buffering level: 0b - 1 MB input buffering is used 1b - 4 MB input buffering is used
22-19 —	Reserved.
18 SW_PPD_OEN_VERSION	PP output endian version: 0b - Endian mode supported for other than RGB 1b - Endian mode supported for any output format
17 SW_PPD_OBUFF_LEVEL	PP output buffering level: 0b - 1 unit output buffering is used 1b - 4 unit output buffering is used
16 SW_PPD_PP_EXIST	PPD exists: 0b - No 1b - Yes
15-14 SW_PPD_IN_TILED_L	PPD input tiled mode support level 00b - not supported 01b - 8x4 tile size supported
13 —	Reserved.
12-0 SW_PPD_MAX_OWIDTH	Max supported PP output width in pixels

15.1.5.1.55 Base address for H264 decoded chroma picture (SWREG102)

15.1.5.1.55.1 Offset

Register	Offset
SWREG102	198h

15.1.5.1.55.2 Diagram



15.1.5.1.55.3 Fields

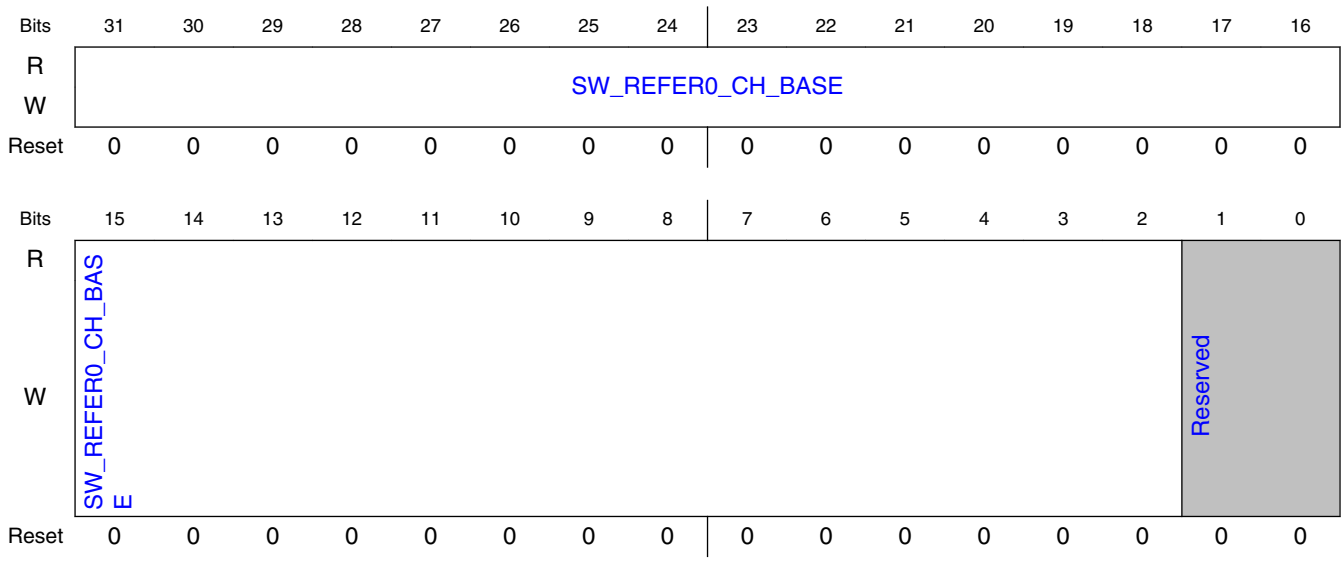
Field	Function
31-2 SW_DEC_CH_BASE	Valid only if chroma address separate mode is enabled. H264: Base address for decoder output chroma picture. Points directly to start of decoder output chroma picture or field.
1 —	Reserved.
0 SW_CH_BASE_E	chroma address separate mode enable: 0b - HW outputs decoded chroma picture to the end of decoded luma picture. HW calculates the chroma picture address according to sw_dec_base and luma data length. 1b - HW outputs decoded chroma picture to independent memory address

15.1.5.1.56 Base address for reference chroma picture index 0 (SWREG103)

15.1.5.1.56.1 Offset

Register	Offset
SWREG103	19Ch

15.1.5.1.56.2 Diagram



15.1.5.1.56.3 Fields

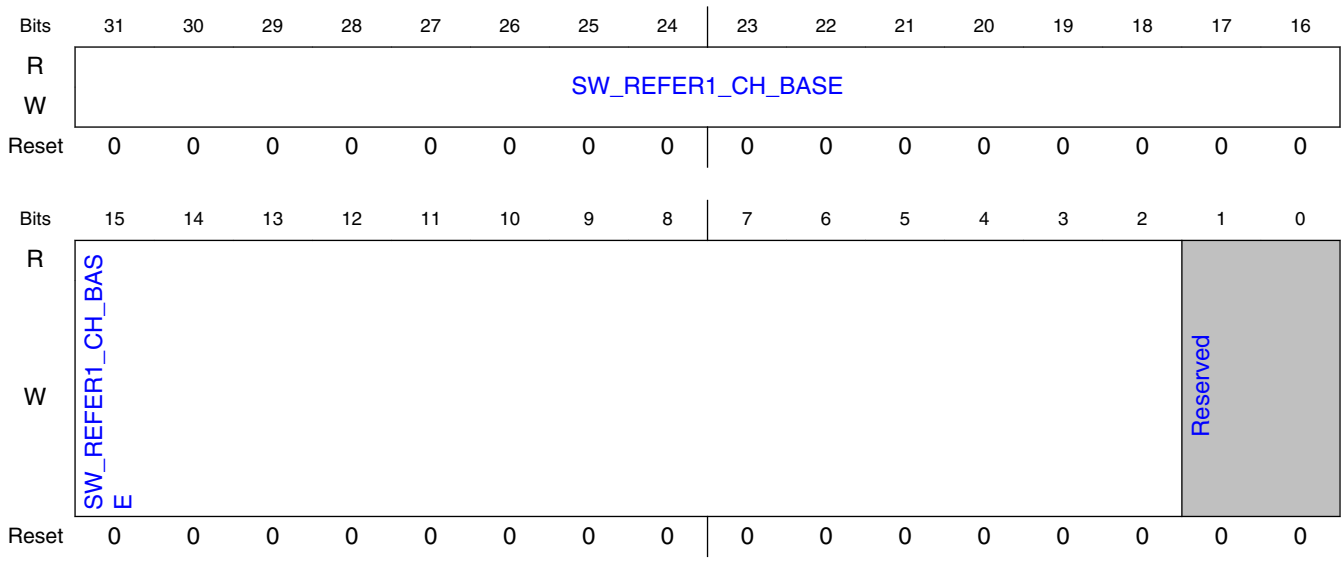
Field	Function
31-2 SW_REFER0_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 0.
1-0 —	Reserved.

15.1.5.1.57 Base address for reference chroma picture index 1 (SWREG104)

15.1.5.1.57.1 Offset

Register	Offset
SWREG104	1A0h

15.1.5.1.57.2 Diagram



15.1.5.1.57.3 Fields

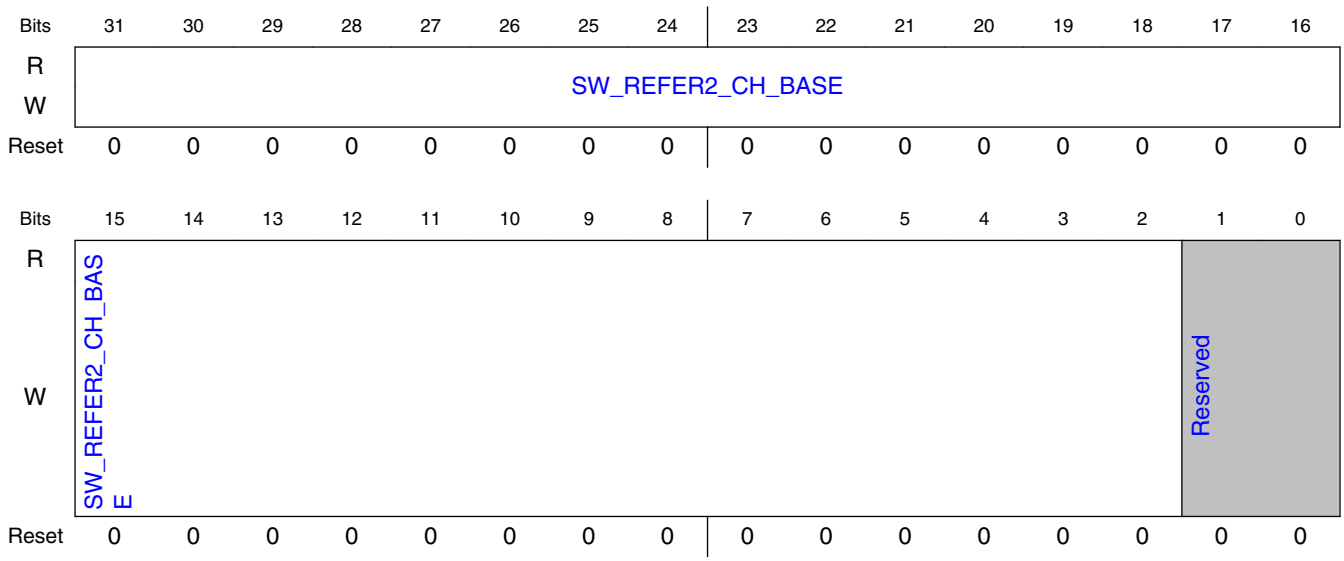
Field	Function
31-2 SW_REFER1_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 1.
1-0 —	Reserved.

15.1.5.1.58 Base address for reference chroma picture index 2 (SWREG105)

15.1.5.1.58.1 Offset

Register	Offset
SWREG105	1A4h

15.1.5.1.58.2 Diagram



15.1.5.1.58.3 Fields

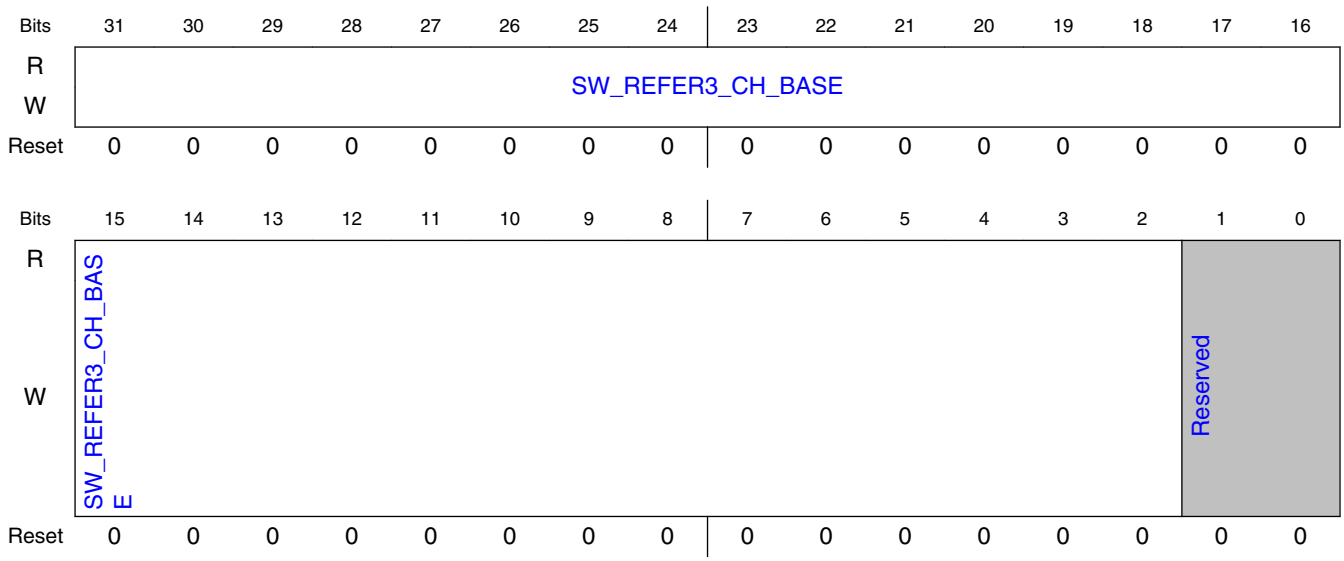
Field	Function
31-2 SW_REFER2_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 2.
1-0 —	Reserved.

15.1.5.1.59 Base address for reference chroma picture index 3 (SWREG106)

15.1.5.1.59.1 Offset

Register	Offset
SWREG106	1A8h

15.1.5.1.59.2 Diagram



15.1.5.1.59.3 Fields

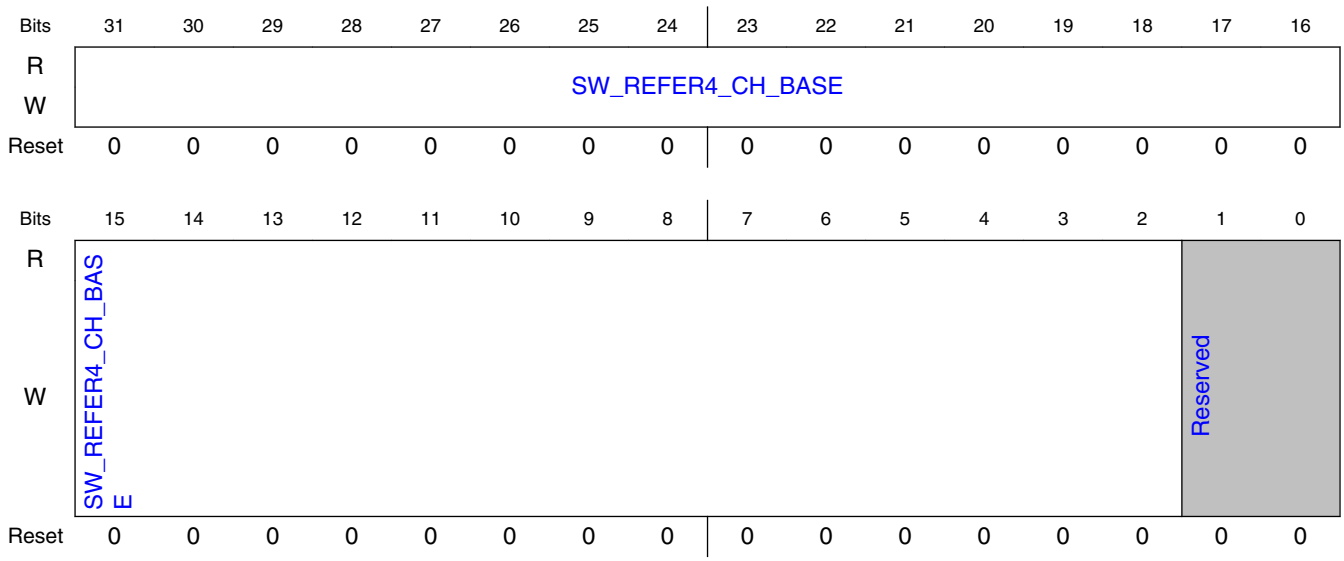
Field	Function
31-2 SW_REFER3_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 3.
1-0 —	Reserved.

15.1.5.1.60 Base address for reference chroma picture index 4 (SWREG107)

15.1.5.1.60.1 Offset

Register	Offset
SWREG107	1ACh

15.1.5.1.60.2 Diagram



15.1.5.1.60.3 Fields

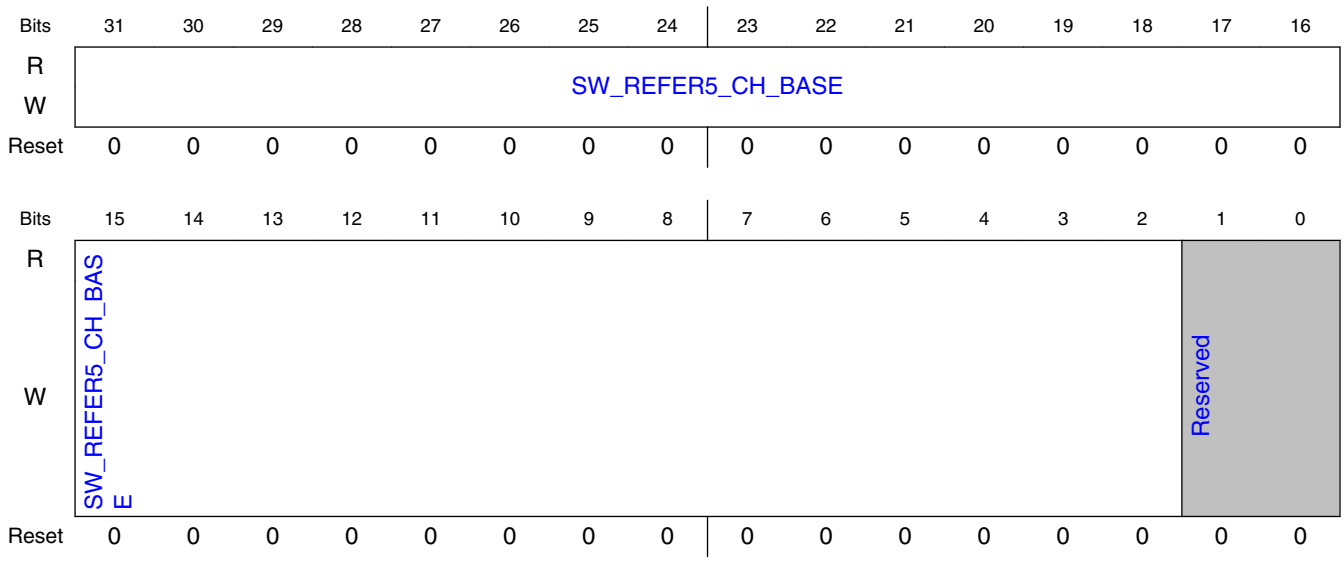
Field	Function
31-2 SW_REFER4_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 4.
1-0 —	Reserved.

15.1.5.1.61 Base address for reference chroma picture index 5 (SWREG108)

15.1.5.1.61.1 Offset

Register	Offset
SWREG108	1B0h

15.1.5.1.61.2 Diagram



15.1.5.1.61.3 Fields

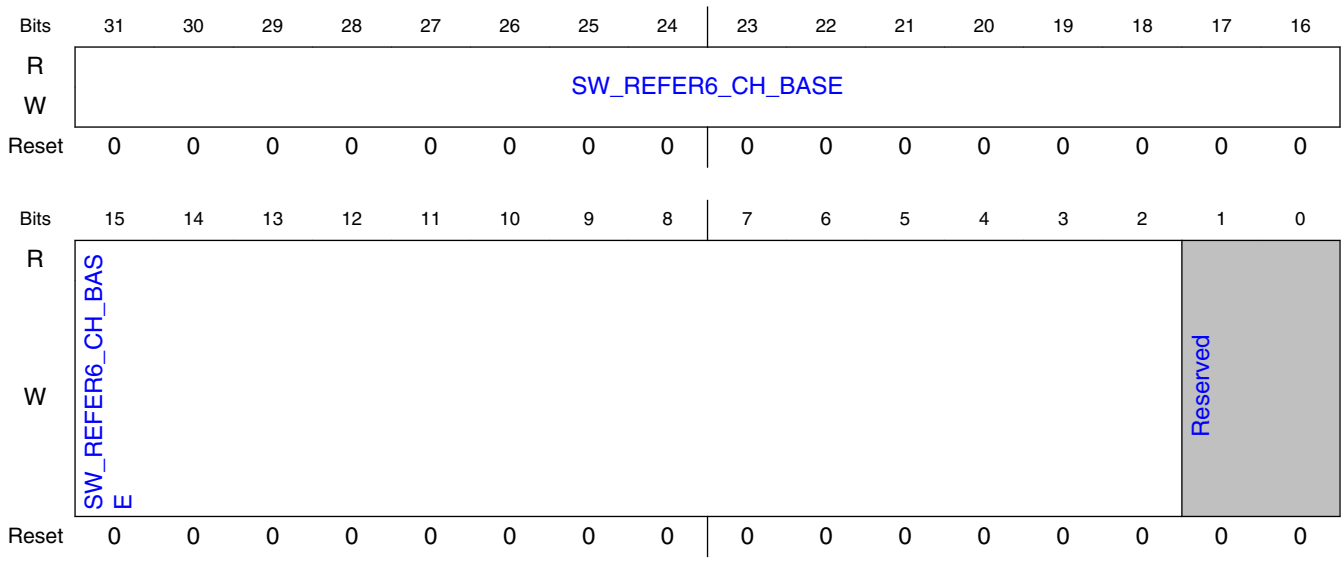
Field	Function
31-2 SW_REFER5_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 5.
1-0 —	Reserved.

15.1.5.1.62 Base address for reference chroma picture index 6 (SWREG109)

15.1.5.1.62.1 Offset

Register	Offset
SWREG109	1B4h

15.1.5.1.62.2 Diagram



15.1.5.1.62.3 Fields

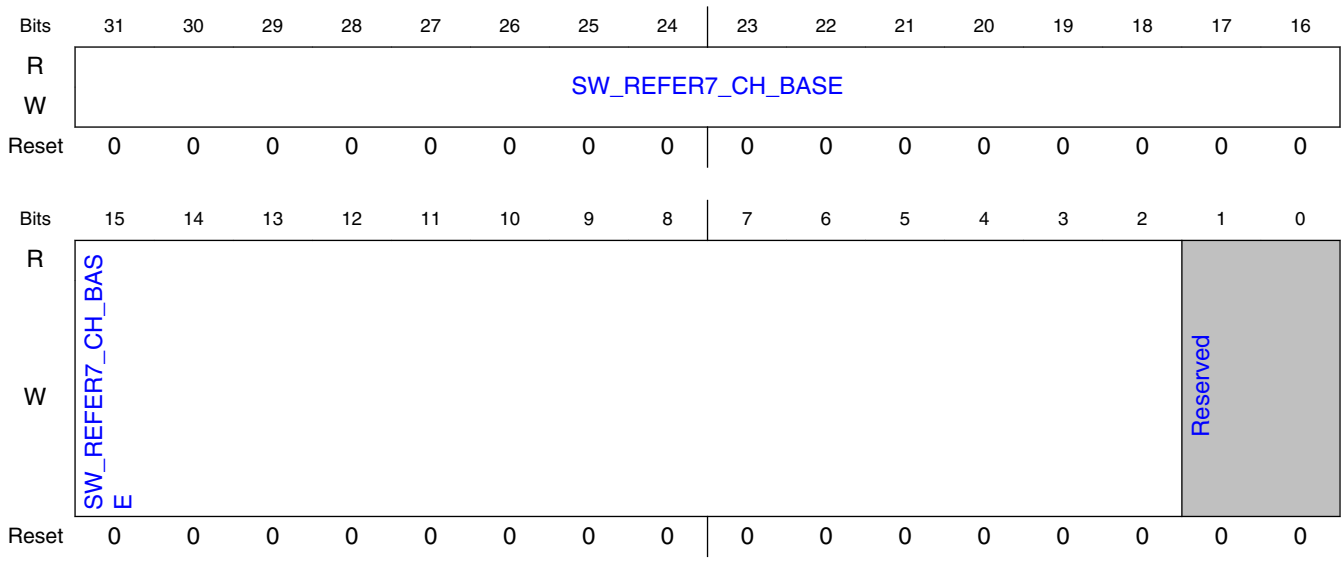
Field	Function
31-2 SW_REFER6_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 6.
1-0 —	Reserved.

15.1.5.1.63 Base address for reference chroma picture index 7 (SWREG110)

15.1.5.1.63.1 Offset

Register	Offset
SWREG110	1B8h

15.1.5.1.63.2 Diagram



15.1.5.1.63.3 Fields

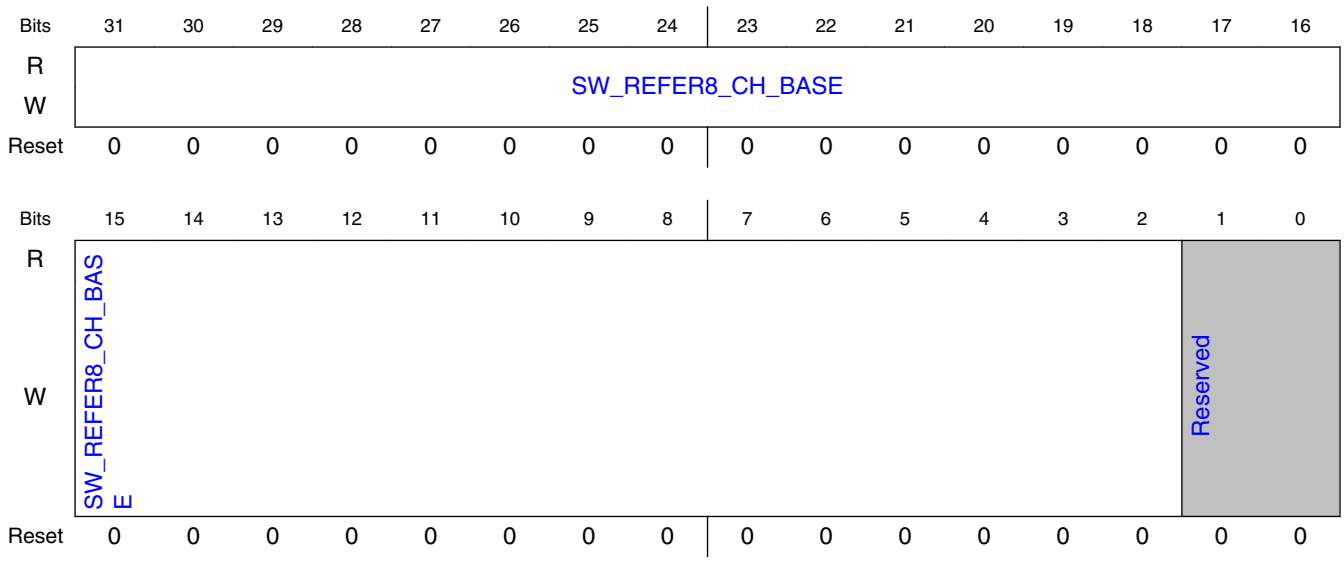
Field	Function
31-2 SW_REFER7_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 7.
1-0 —	Reserved.

15.1.5.1.64 Base address for reference chroma picture index 8 (SWREG111)

15.1.5.1.64.1 Offset

Register	Offset
SWREG111	1BCh

15.1.5.1.64.2 Diagram



15.1.5.1.64.3 Fields

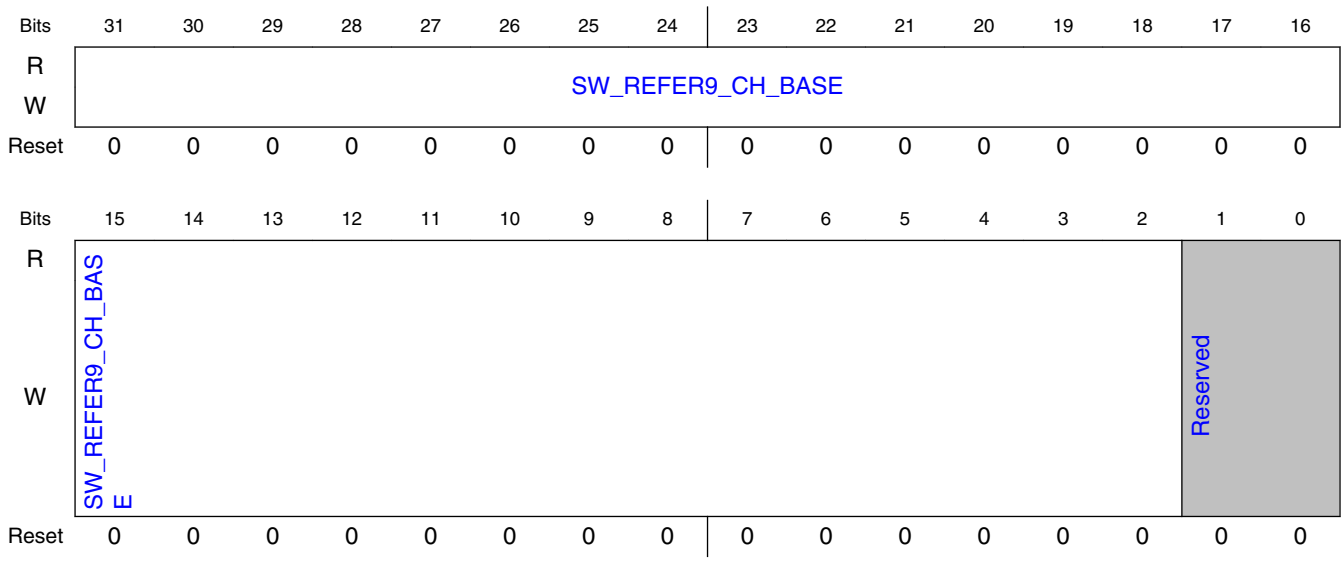
Field	Function
31-2 SW_REFER8_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 8.
1-0 —	Reserved.

15.1.5.1.65 Base address for reference chroma picture index 9 (SWREG112)

15.1.5.1.65.1 Offset

Register	Offset
SWREG112	1C0h

15.1.5.1.65.2 Diagram



15.1.5.1.65.3 Fields

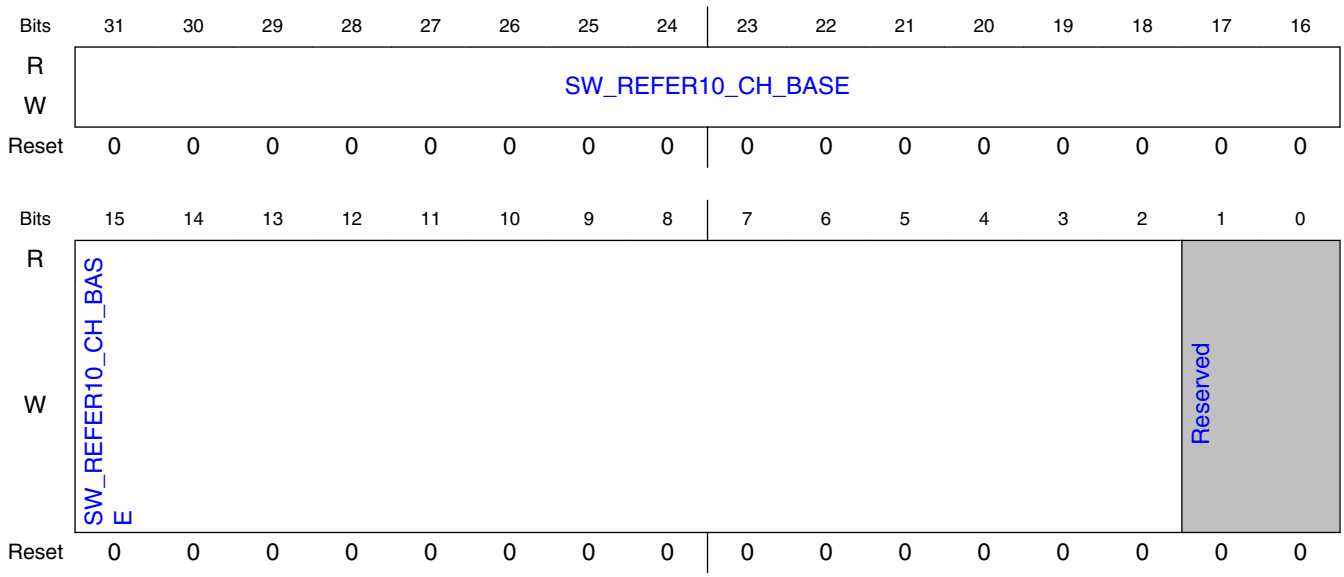
Field	Function
31-2 SW_REFER9_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 9.
1-0 —	Reserved.

15.1.5.1.66 Base address for reference chroma picture index 10 (SWREG113)

15.1.5.1.66.1 Offset

Register	Offset
SWREG113	1C4h

15.1.5.1.66.2 Diagram



15.1.5.1.66.3 Fields

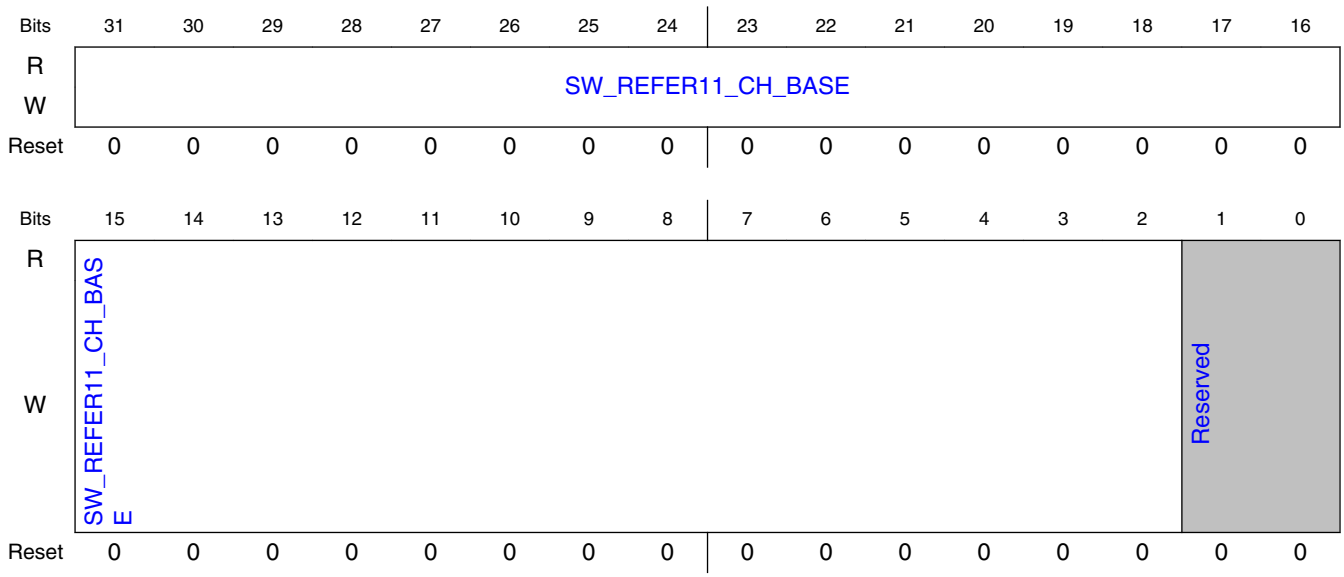
Field	Function
31-2 SW_REFER10_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 10.
1-0 —	Reserved.

15.1.5.1.67 Base address for reference chroma picture index 11 (SWREG114)

15.1.5.1.67.1 Offset

Register	Offset
SWREG114	1C8h

15.1.5.1.67.2 Diagram



15.1.5.1.67.3 Fields

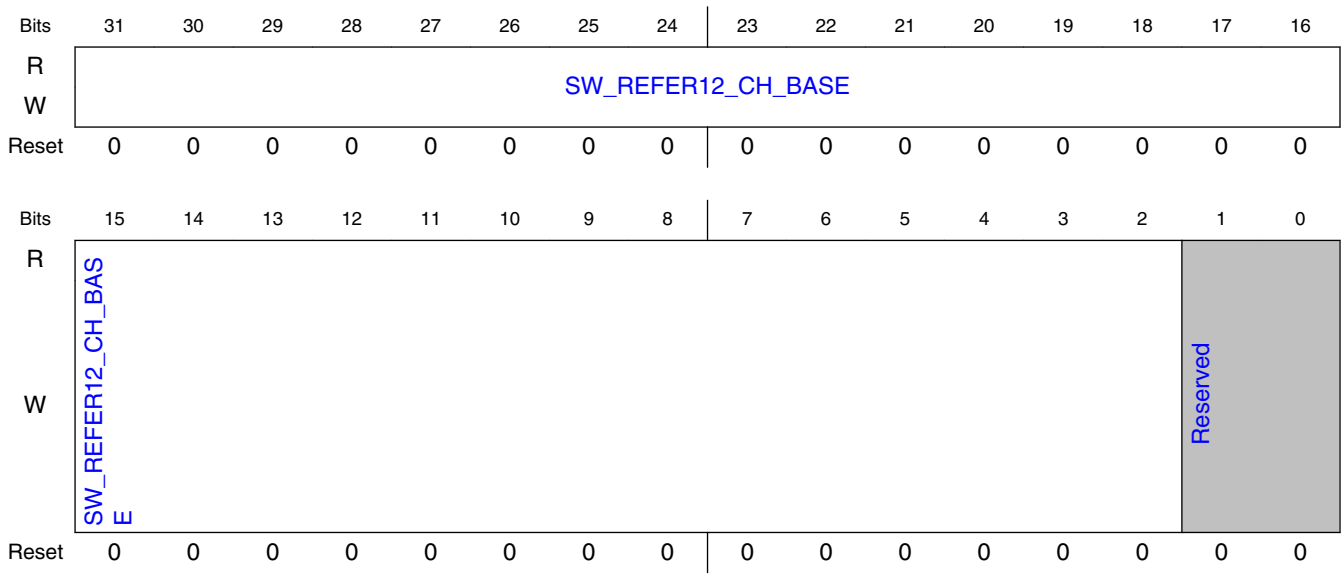
Field	Function
31-2 SW_REFER11_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 11.
1-0 —	Reserved.

15.1.5.1.68 Base address for reference chroma picture index 12 (SWREG115)

15.1.5.1.68.1 Offset

Register	Offset
SWREG115	1CCh

15.1.5.1.68.2 Diagram



15.1.5.1.68.3 Fields

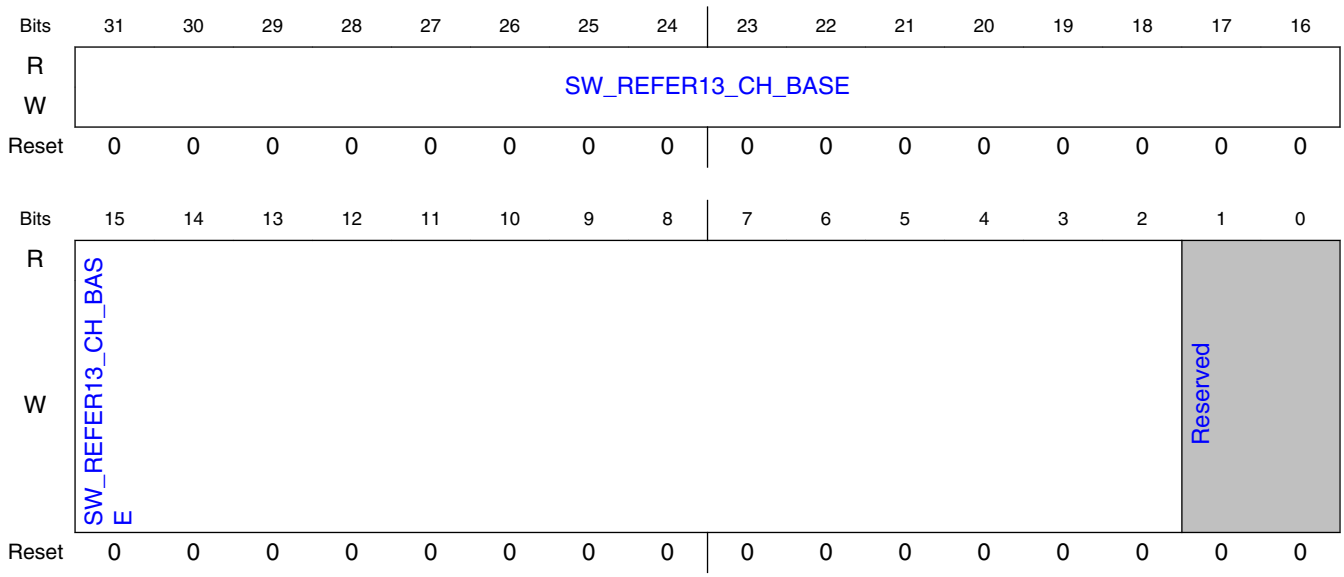
Field	Function
31-2 SW_REFER12_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 12.
1-0 —	Reserved.

15.1.5.1.69 Base address for reference chroma picture index 13 (SWREG116)

15.1.5.1.69.1 Offset

Register	Offset
SWREG116	1D0h

15.1.5.1.69.2 Diagram



15.1.5.1.69.3 Fields

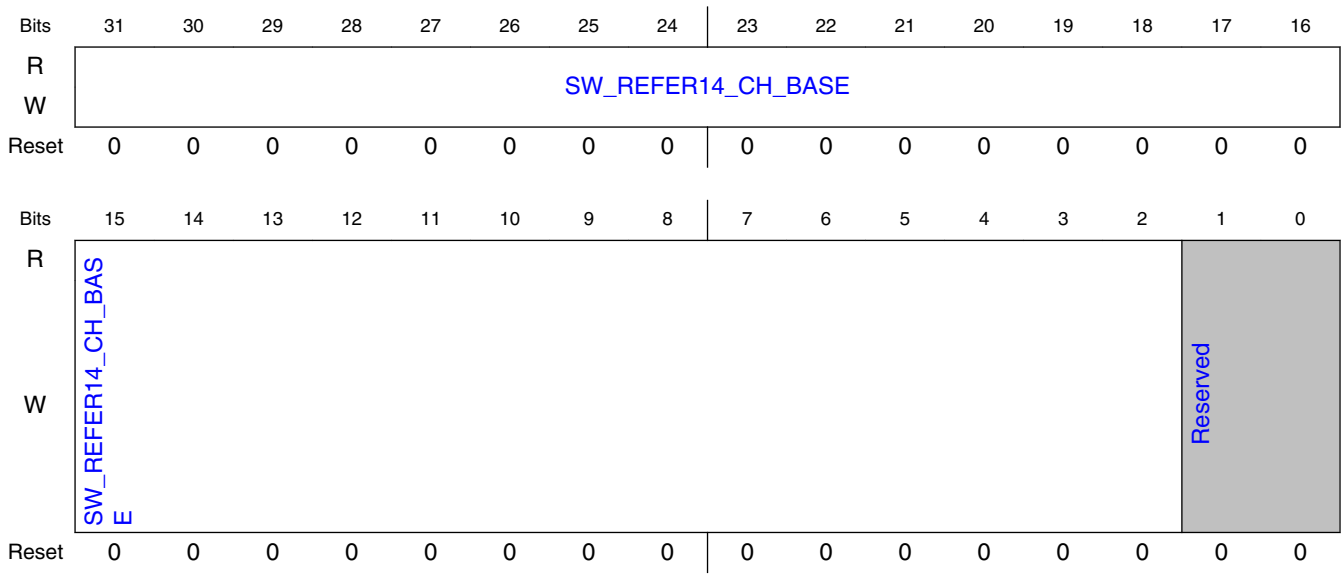
Field	Function
31-2 SW_REFER13_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 13.
1-0 —	Reserved.

15.1.5.1.70 Base address for reference chroma picture index 14 (SWREG117)

15.1.5.1.70.1 Offset

Register	Offset
SWREG117	1D4h

15.1.5.1.70.2 Diagram



15.1.5.1.70.3 Fields

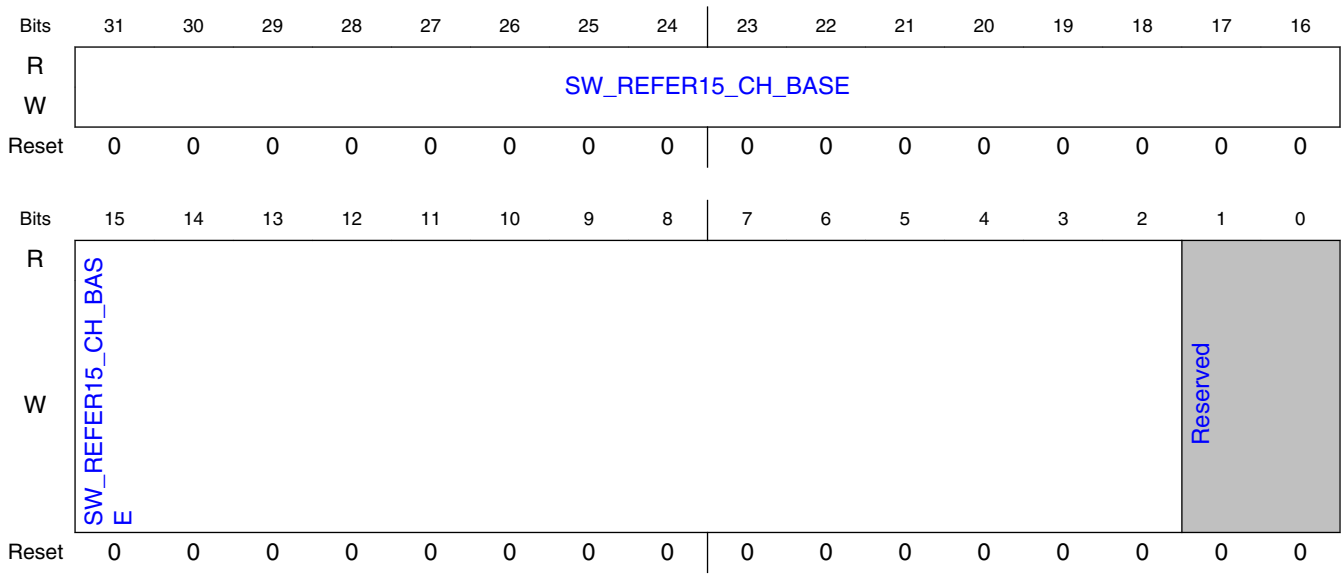
Field	Function
31-2 SW_REFER14_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 14.
1-0 —	Reserved.

15.1.5.1.71 Base address for reference chroma picture index 15 (SWREG118)

15.1.5.1.71.1 Offset

Register	Offset
SWREG118	1D8h

15.1.5.1.71.2 Diagram



15.1.5.1.71.3 Fields

Field	Function
31-2 SW_REFER15_CH_BASE	Valid only if chroma address separate mode is enabled. Base address for reference chroma picture index 15.
1-0 —	Reserved.

15.1.5.2 VPU_G1 H264 decoder register descriptions

15.1.5.2.1 VPU_G1 H264 decoder memory map

VPU_G1_H264 base address: 3830_0000h

Offset	Register	Width (In bits)	Access	Reset value
10h	Decoder control register 1 (picture parameters) (SWREG4)	32	RW	0000_0000h
14h	Decoder control register 2 (stream decoding table selects) (SWREG5)	32	RW	0000_0000h
18h	Decoder control register 3 (stream buffer information) (SWREG6)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
1Ch	Decoder control register 4 (H264, VC-1, VP6 and progressive JPEG control) (SWREG7)	32	RW	0000_0000h
20h	Decoder control register 5 (H264, VC-1, VP6, Progressive JPEG and RV control) (SWREG8)	32	RW	0000_0000h
24h	Decoder control register 6 / base address for MB-control (RLC) / VC-1 intensity control 0/ VP6,VP7,VP8 ctrl-stream length/ RV pic slice amount (SWREG9)	32	RW	0000_0000h
28h	Base address for differential motion vector base address (RLC-mode) /H264 P initial fwd ref pic list register (4-9)/ VC-1 intensity control 1/ VP7 and VP8 segmentation base register (SWREG10)	32	RW	0000_0000h
2Ch	Decoder control register 7 (VLC) / base address for H.264 intra prediction 4x4 / base address for MPEG-4 DC component (RLC) / H264 P initial fwd ref pic list register (10-15) / VC-1 intensity control 2 (SWREG11)	32	RW	0000_0000h
38h	Base address for reference picture index 0 / base address for JPEG decoder output chrominance picture (SWREG14)	32	RW	0000_0000h
3Ch	Base address for reference picture index 1 / JPEG control (SWRE G15)	32	RW	0000_0000h
40h	Base address for reference picture index 2 / List of VLC code lengths in first JPEG AC table (SWREG16)	32	RW	0000_0000h
44h	Base address for reference picture index 3 / List of VLC code lengths in first JPEG AC table (SWREG17)	32	RW	0000_0000h
48h	Base address for reference picture index 4 / VC1 control / MPEG4 MVD control/ List of VLC code lengths in first JPEG AC table / VC-1 intensity control 4 / VP6/VP7, VP8 Golden refer picture base (SWRE G18)	32	RW	0000_0000h
4Ch	Base address for reference picture index 5 / MPEG4 TRB/TRD delta 0 / VC-1 intensity control 3 List of VLC code lengths in first/second JPEG AC table / VP6/VP7 scan maps (SWREG19)	32	RW	0000_0000h
50h	Base address for reference picture index 6 / / MPEG4 TRB/TRD delta -1 / List of VLC code lengths in second JPEG AC table / VP6/VP7 scan maps (SWREG20)	32	RW	0000_0000h
54h	Base address for reference picture index 7 / MPEG4 TRB/TRD delta 1 / List of VLC code lengths in second JPEG AC table / VP6/VP7 scan maps (SWREG21)	32	RW	0000_0000h
58h	Base address for reference picture index 8 / List of VLC code lengths in second JPEG AC table / VP6 scan maps / VP7,VP8 DCT stream 1 base (SWREG22)	32	RW	0000_0000h
5Ch	Base address for reference picture index 9 / List of VLC code lengths in first JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 2 base (SWREG23)	32	RW	0000_0000h
60h	Base address for reference picture index 10 / List of VLC code lengths in first JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 3 base (SWREG24)	32	RW	0000_0000h
64h	Base address for reference picture index 11 / List of VLC code lengths in second JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 4 base (SWREG25)	32	RW	0000_0000h

Table continues on the next page...

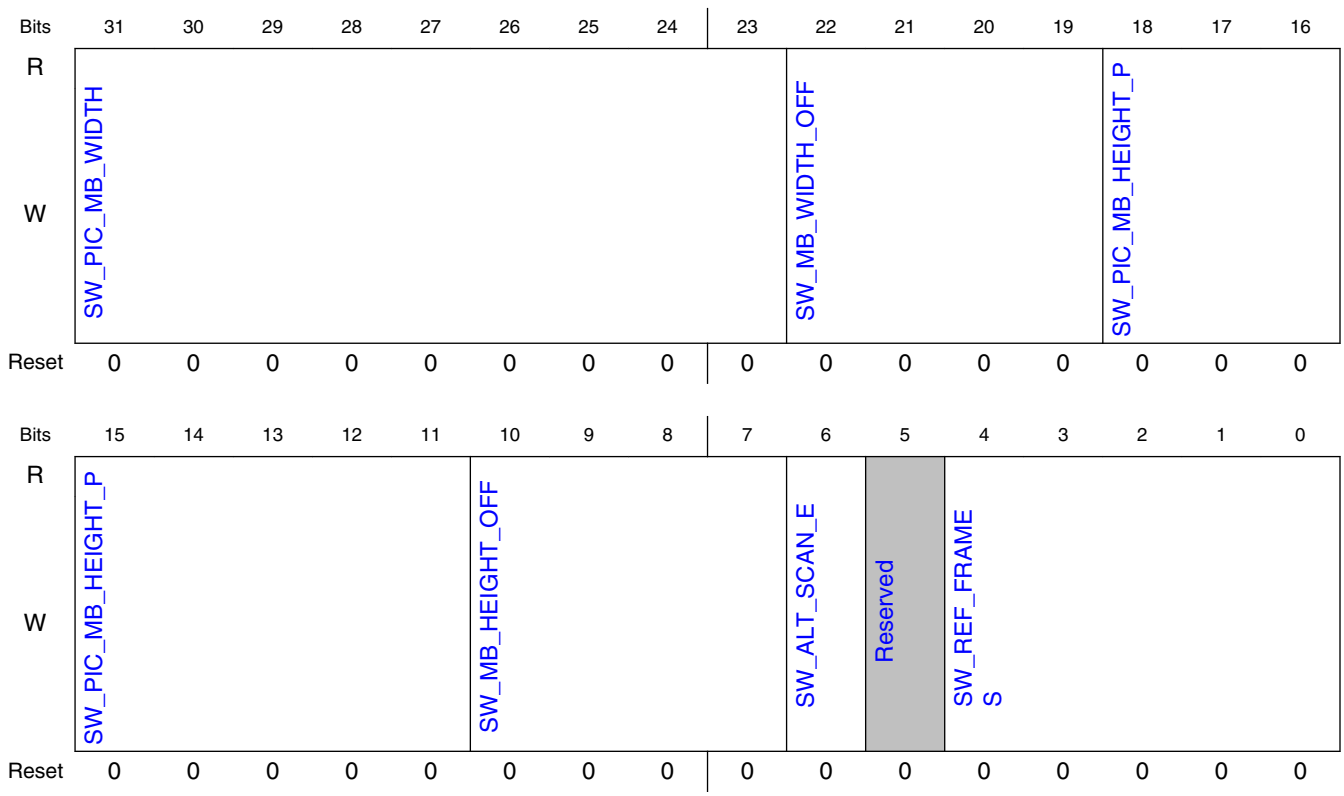
Offset	Register	Width (In bits)	Access	Reset value
68h	Base address for reference picture index 12 / List of VLC code lengths in second JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 5 base (SWREG26)	32	RW	0000_0000h
6Ch	Base address for reference picture index 13 / VC-1 bitpl mbctrl or VP6,VP7,VP8 ctrl stream base /Progressive JPEG DC table (SWREG27)	32	RW	0000_0000h
70h	Base address for reference picture index 14 / VP6 scan maps / Progressive JPEG DC table / VP7,VP8 DCT stream 6 base (SWREG28)	32	RW	0000_0000h
74h	Base address for reference picture index 15 / VP6 scan maps / VP7,VP8 DCT stream 7 base (SWREG29)	32	RW	0000_0000h
78h	Reference picture numbers for index 0 and 1 (H264 VLC) / VP6 scan maps / VP7,VP8 loop filter mb level adjusts (SWREG30)	32	RW	0000_0000h
7Ch	Reference picture numbers for index 2 and 3 (H264 VLC) / VP6 scan maps / VP7,VP8 loop filter ref pic level adjusts (SWREG31)	32	RW	0000_0000h
80h	Reference picture numbers for index 4 and 5 (H264 VLC) / VP6 scan maps / VP7,VP8 loop filter levels (SWREG32)	32	RW	0000_0000h
84h	Reference picture numbers for index 6 and 7 (H264 VLC) / VP6 scan maps / VP7,VP8 quantization values (SWREG33)	32	RW	0000_0000h
88h	Reference picture numbers for index 8 and 9 (H264 VLC) / MPEG4, VC1, VPx prediction filter taps (SWREG34)	32	RW	0000_0000h
8Ch	Reference picture numbers for index 10 and 11 (H264 VLC) / VC1, VPx prediction filter taps (SWREG35)	32	RW	0000_0000h
90h	Reference picture numbers for index 12 and 13 (H264 VLC) / VC1, VPx prediction filter taps (SWREG36)	32	RW	0000_0000h
94h	Reference picture numbers for index 14 and 15 (H264 VLC) / VPx prediction filter taps (SWREG37)	32	RW	0000_0000h
98h	Reference picture long term flags (H264 VLC) / VPx prediction filter taps (SWREG38)	32	RW	0000_0000h
9Ch	Reference picture valid flags (H264 VLC) / VPx prediction filter taps (SWREG39)	32	RW	0000_0000h
A8h	bi_dir initial ref pic list register (0-2) / VP6 prediction filter taps / Progressive JPEG Cb ACDC coefficient base (SWREG42_H264)	32	RW	0000_0000h
ACh	bi-dir initial ref pic list register (3-5) / VP6 prediction filter taps / Progressive JPEG Cr ACDC coefficient base (SWREG43_H264)	32	RW	0000_0000h
B0h	bi-dir initial ref pic list register (6-8) / VP6 prediction filter taps (SWREG44_H264)	32	RW	0000_0000h
B4h	bi-dir initial ref pic list register (9-11) / VP6 prediction filter taps (SWREG45)	32	RW	0000_0000h
B8h	bi-dir initial ref pic list register (12-14) / VP7,VP8 quantization values (SWREG46)	32	RW	0000_0000h
BCh	bi-dir and P fwd initial ref pic list register (15 and P 0-3) / VP7,VP8 quantization values (SWREG47)	32	RW	0000_0000h

15.1.5.2.2 Decoder control register 1 (picture parameters) (SWREG4)

15.1.5.2.2.1 Offset

Register	Offset
SWREG4	10h

15.1.5.2.2.2 Diagram



15.1.5.2.2.3 Fields

Field	Function
31-23 SW_PIC_MB_WIDTH	Picture width in macroblocks = ((width in pixels + 15) /16)
22-19 SW_MB_WIDTH_OFF	The amount of meaningfull horizontal pixels in last MB (width offset) 0 if exactly 16 pixels multiple picture and all the horizontal pixels in last MB are meaningfull
18-11	Picture height in macroblocks =((height in pixels+15)/16). Picture height is informed as size of the (progressive) frame also for single field (of interlaced content) is being decoded

Table continues on the next page...

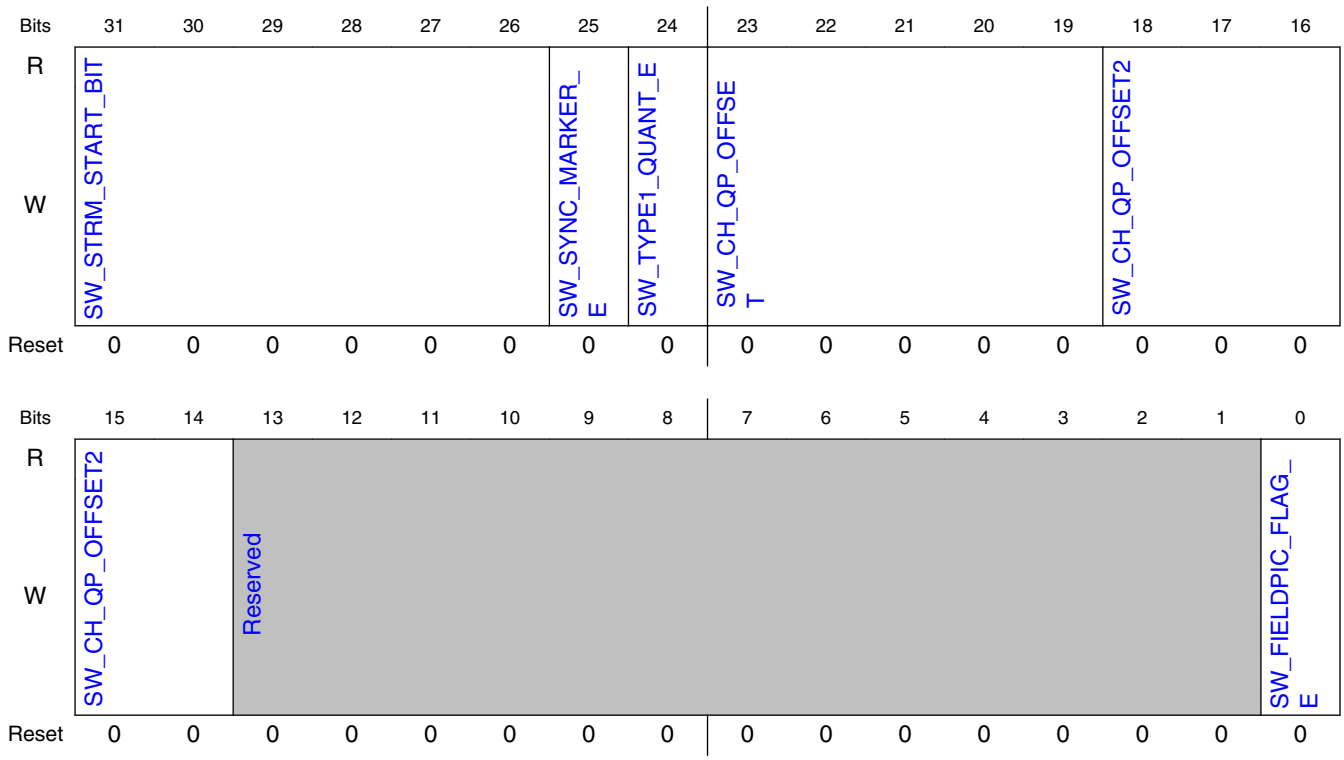
Field	Function
SW_PIC_MB_HEIGHT_P	
10-7 SW_MB_HEIGHT_OFF	The amount of meaningful vertical pixels in last MB (height offset 0 if exactly 16 pixels multiple picture and all the vertical pixels in last MB are meaningfull
6 SW_ALT_SCAN_E	indicates alternative vertical scan method used for interlaced frames
5 —	Reserved.
4-0 SW_REF_FRAMES	H.264: num_ref_frames, maximum number of short and long term reference frames in decoded picture buffer VC-1: num_ref semantics

15.1.5.2.3 Decoder control register 2 (stream decoding table selects) (SWREG5)

15.1.5.2.3.1 Offset

Register	Offset
SWREG5	14h

15.1.5.2.3.2 Diagram



15.1.5.2.3.3 Fields

Field	Function
31-26 SW_STRM_START_BIT	Exact bit of stream start word where decoding can be started (assosiates with sw_rlc_vlc_base)
25 SW_SYNC_MARKER_E	Sync markers enable For progressive JPEG, this indicates that there are restart markers in the stream after restart interval steps. 0b - synch markers are not used 1b - synch markers are used.
24 SW_TYPE1_QUANT_E	MPEG4: Type 1 quantization enable • '0' = type 2 inverse Q method • '1' = type 1 inverse Q method (Q-tables used) H264 (h264_high config): scaling matrix enable • '0' = normal transform • '1' = use scaling matrix for transform (read from external memory)
23-19 SW_CH_QP_OFFSET	Chroma Qp filter offset. (For H.264 this offset concerns Cb only)
18-14	Chroma Qp filter offset for cr type

Table continues on the next page...

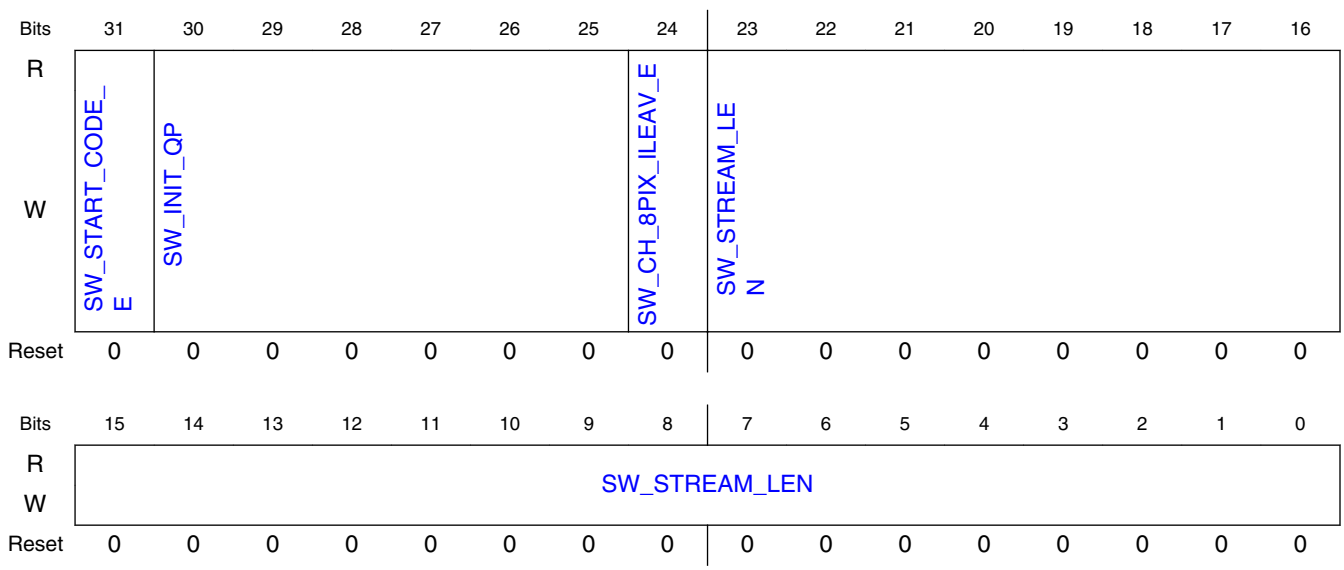
Field	Function
SW_CH_QP_OFFSET2	
13-1 —	Reserved.
0 SW_FIELDPIC_FLAG_E	Flag for streamd that field_pic_flag exists in stream

15.1.5.2.4 Decoder control register 3 (stream buffer information) (SWREG6)

15.1.5.2.4.1 Offset

Register	Offset
SWREG6	18h

15.1.5.2.4.2 Diagram



15.1.5.2.4.3 Fields

Field	Function
31	Bit for indicating stream start code existence: 0b - stream doesn't contain start codes

Table continues on the next page...

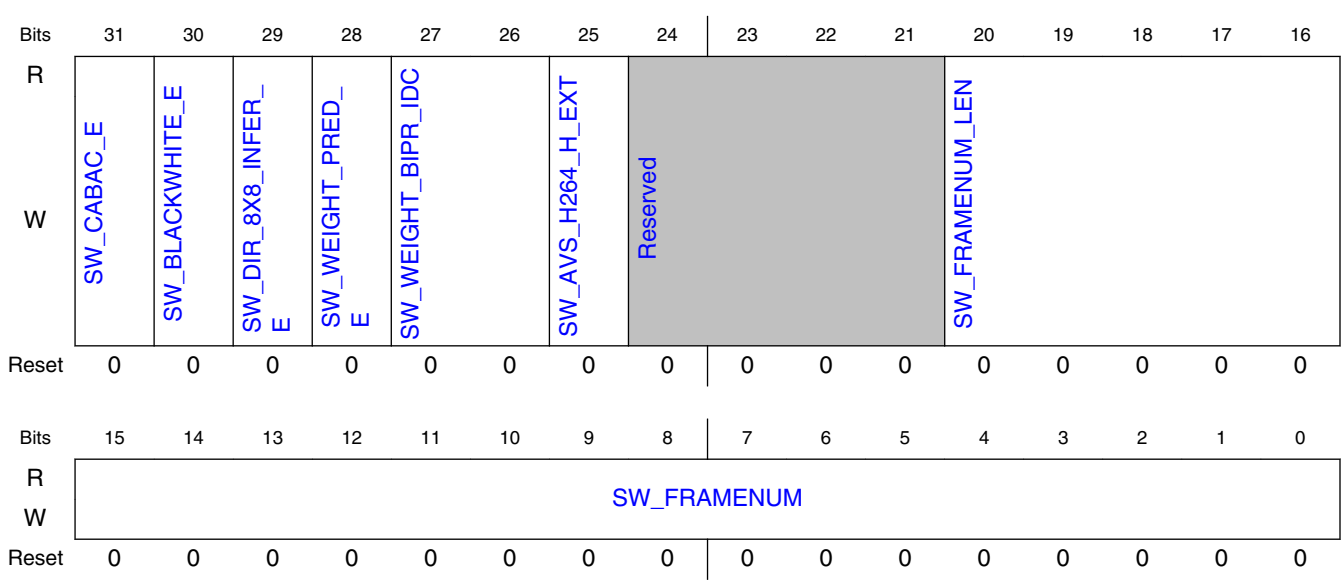
Field	Function
SW_START_CODE_E	1b - stream contains start codes
30-25 SW_INIT_QP	Initial value for quantization parameter (picture quantizer).
24 SW_CH_8PIX_ILEAV_E	Enable for additional chrominance data format writing where decoder writes chrominance in group of 8 pixels of Cb and then corresponding 8 pixels of Cr. Data is written to sw_dec_ch8pix_base. Cannot be used if tiled mode is enabled
23-0 SW_STREAM_LEN	Amount of stream data bytes in input buffer. If the given buffer size is not enough for finishing the picture the corresponding interrupt is given and new stream buffer base address and stream buffer size information should be given (assosiates with sw_rlc_vlc_base). For VC-1/VP6 the buffer must include data for one picture/slice of the picture For H264/MPEG4/H263/MPEG2/MPEG1 the buffer must include at least data for one slice/VP of the picture For JPEG the buffer size must be a multiple of 256 bytes or the amount of data for one picture.

15.1.5.2.5 Decoder control register 4 (H264, VC-1, VP6 and progressive JPEG control) (SWREG7)

15.1.5.2.5.1 Offset

Register	Offset
SWREG7	1Ch

15.1.5.2.5.2 Diagram



15.1.5.2.5.3 Fields

Field	Function
31 SW_CABAC_E	CABAC enable
30 SW_BLACKWHITE_E	0b - 4:2:0 sampling format 1b - 4:0:0 sampling format (H264 monochroma)
29 SW_DIR_8X8_INFER_E	Specifies the method to use to derive luma motion vectors in B_skip, B_Direct_16x16 and B_direct_8x8_inference_flag (see direct_8x8_inference flag)
28 SW_WEIGHT_PRED_E	Weighted prediction enable for P slices
27-26 SW_WEIGHT_BIPR_IDC	weighted prediction specification for B slices: 00b - default weighted prediction is applied to B slices 01b - explicit weighted prediction shall be applied to B slices 10b - implicit weighted prediction shall be applied to B slices
25 SW_AVS_H264_H_EXT	Resolution extension to support 4k resolution for AVS/H264. Used as MSB of sw_pic_mb_height
24-21 —	Reserved.
20-16 SW_FRAMENUM_LEN	H.264: Bit length of frame_num in data stream RV: frame size length. Informs how many bits in stream are used for frame size (HW discards these bits)
15-0 SW_FRAMENUM	current frame_num, used to identify short-term reference frames. Used in reference picture reordering

15.1.5.2.6 Decoder control register 5 (H264, VC-1, VP6, Progressive JPEG and RV control) (SWREG8)

15.1.5.2.6.1 Offset

Register	Offset
SWREG8	20h

15.1.5.2.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SW_CONST_INTRA_E	SW_FILT_CTRL_PRES	SW_RDPIC_CNT_PRES	SW_8X8TRANS_FLAG_E	SW_REFPIC_MK_LEN											SW_IDR_PIC_E
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SW_IDR_PIC_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.6.3 Fields

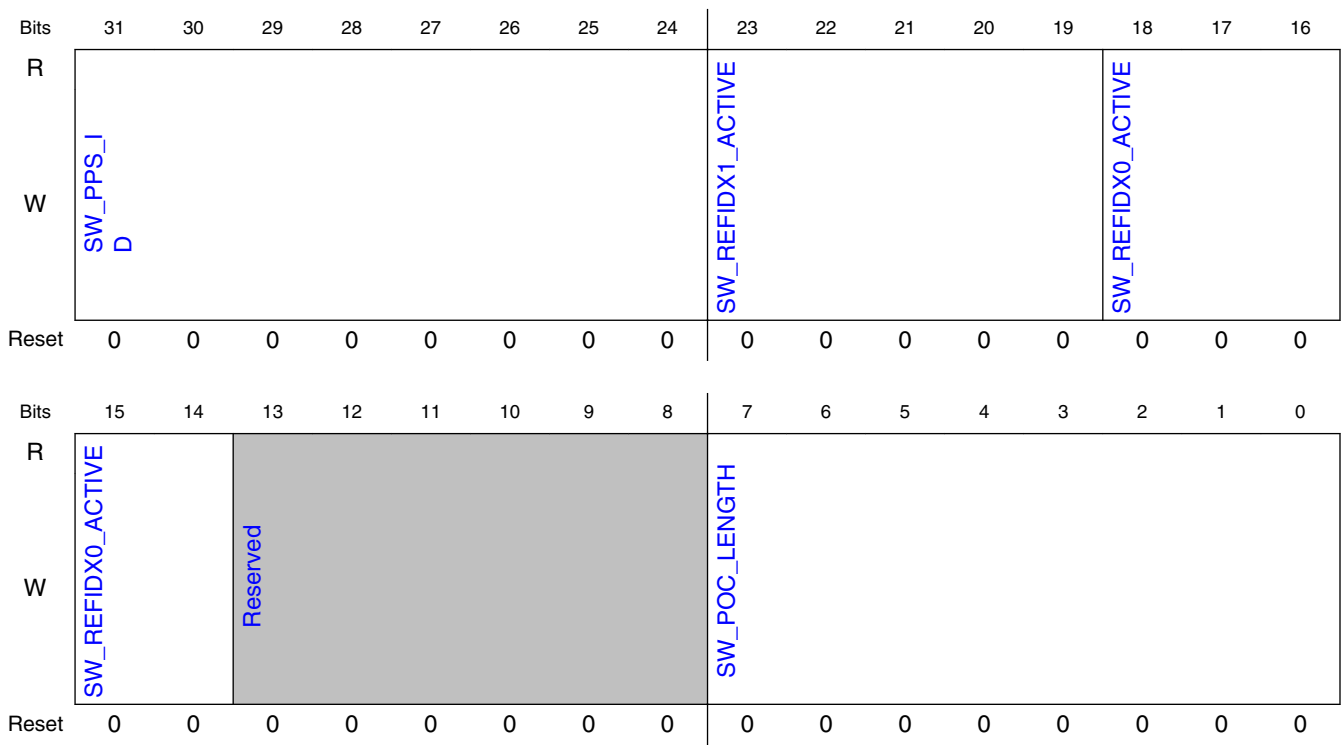
Field	Function
31 SW_CONST_INTRA_E	constrained_intra_pred_flag equal to 1 specifies that intra prediction uses only neighbouring intra macroblocks in prediction. When equal to 0 also neighbouring inter macroblocks are used in intra prediction process.
30 SW_FILT_CTRL_PRES	deblocking_filter_control_present_flag indicates whether extra variables controlling characteristics of the deblocking filter are present in the slice header.
29 SW_RDPIC_CNT_PRES	redundant_pic_cnt_present_flag specifies whether redundant_pic_cnt syntax elements are present in the slice header.
28 SW_8X8TRANS_FLAG_E	8x8 transform flag enable for stream decoding
27-17 SW_REFPIC_MK_LEN	Length of decoded reference picture marking bits
16 SW_IDR_PIC_E	IDR (instantaneous decoding refresh) picture flag.
15-0 SW_IDR_PIC_ID	idr_pic_id, identifies IDR (instantaneous decoding refresh) picture

15.1.5.2.7 Decoder control register 6 / base address for MB-control (RLC) / VC-1 intensity control 0/ VP6,VP7,VP8 ctrl-stream length/ RV pic slice amount (SWREG9)

15.1.5.2.7.1 Offset

Register	Offset
SWREG9	24h

15.1.5.2.7.2 Diagram



15.1.5.2.7.3 Fields

Field	Function
31-24 SW_PPS_ID	pic_parameter_set_id, identifies the picture parameter set that is referred to in the slice header.
23-19 SW_REFIDX1_ACTIVE	Specifies the maximum reference index that can be used while decoding inter predicted macro blocks.

Table continues on the next page...

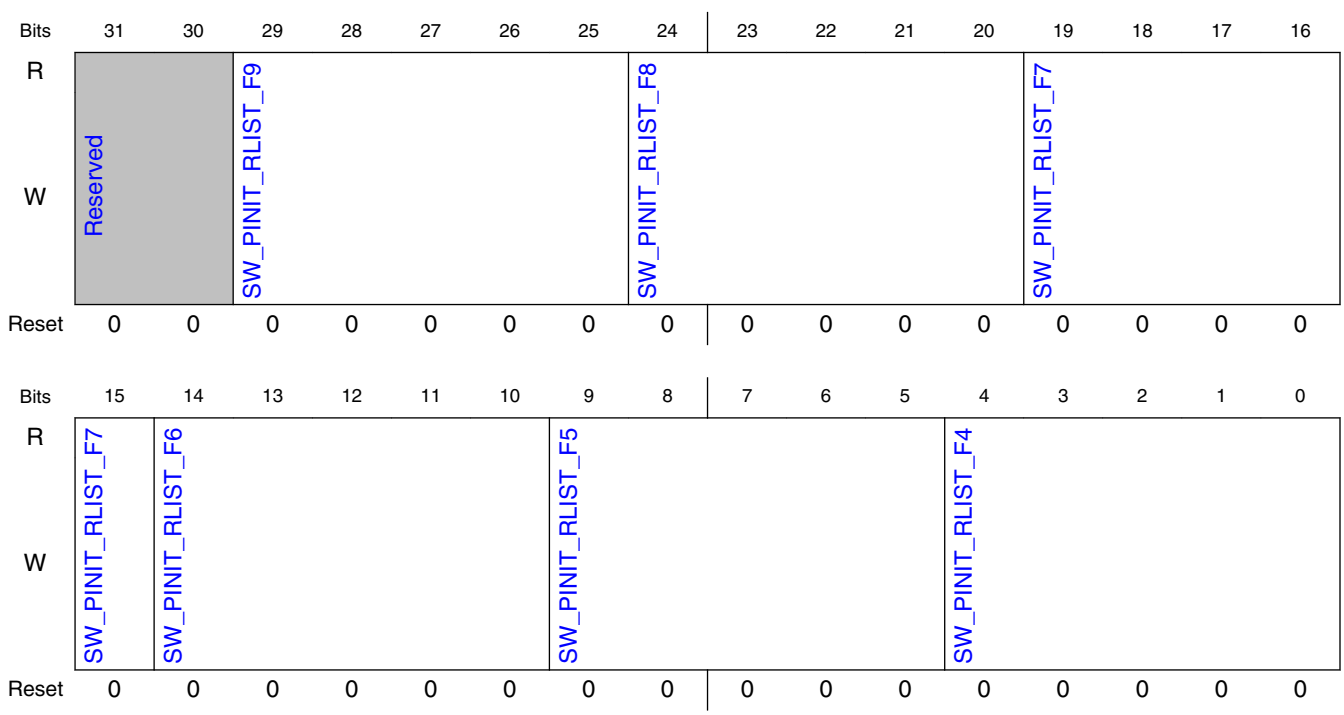
Field	Function
18-14 SW_REFIDX0_ACTIVE	Specifies the maximum reference index that can be used while decoding inter predicted macro blocks. This is same as in previous decoders (width increased with q bit)
13-8 —	Reserved.
7-0 SW_POC_LENGTH	Length of picture order count field in stream

15.1.5.2.8 Base address for differential motion vector base address (RLC-mode) /H264 P initial fwd ref pic list register (4-9)/ VC-1 intensity control 1/ VP7 and VP8 segmentation base register (SWREG10)

15.1.5.2.8.1 Offset

Register	Offset
SWREG10	28h

15.1.5.2.8.2 Diagram



15.1.5.2.8.3 Fields

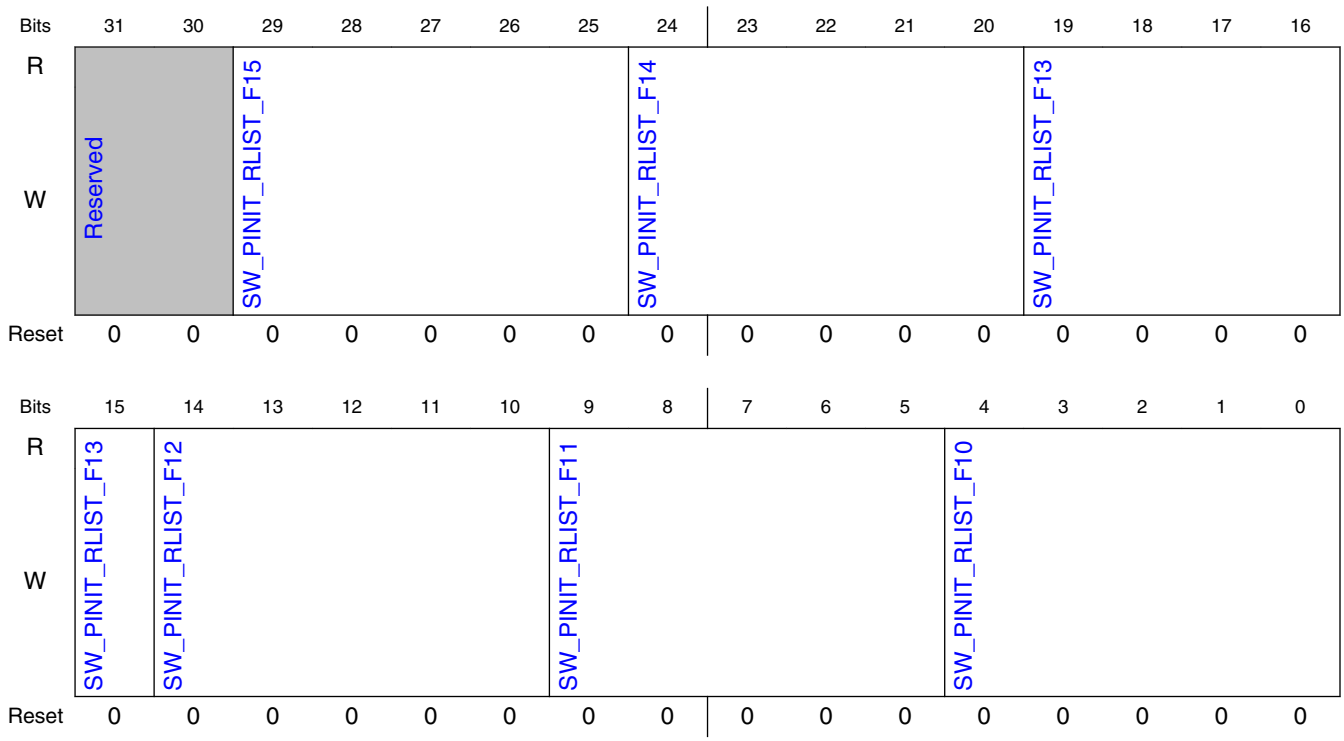
Field	Function
31-30 —	Reserved.
29-25 SW_PINIT_RLIST_F9	Initial reference picture list for P forward picid 9
24-20 SW_PINIT_RLIST_F8	Initial reference picture list for P forward picid 8
19-15 SW_PINIT_RLIST_F7	Initial reference picture list for P forward picid 7
14-10 SW_PINIT_RLIST_F6	Initial reference picture list for P forward picid 6
9-5 SW_PINIT_RLIST_F5	Initial reference picture list for P forward picid 5
4-0 SW_PINIT_RLIST_F4	Initial reference picture list for P forward picid 4

15.1.5.2.9 Decoder control register 7 (VLC) / base address for H.264 intra prediction 4x4 / base address for MPEG-4 DC component (RLC) / H264 P initial fwd ref pic list register (10-15) / VC-1 intensity control 2 (SWREG11)

15.1.5.2.9.1 Offset

Register	Offset
SWREG11	2Ch

15.1.5.2.9.2 Diagram



15.1.5.2.9.3 Fields

Field	Function
31-30 —	Reserved.
29-25 SW_PINIT_RLIST_F15	Initial reference picture list for P forward picid 15
24-20 SW_PINIT_RLIST_F14	Initial reference picture list for P forward picid 14
19-15 SW_PINIT_RLIST_F13	Initial reference picture list for P forward picid 13
14-10 SW_PINIT_RLIST_F12	Initial reference picture list for P forward picid 12
9-5 SW_PINIT_RLIST_F11	Initial reference picture list for P forward picid 11
4-0	Initial reference picture list for P forward picid 10

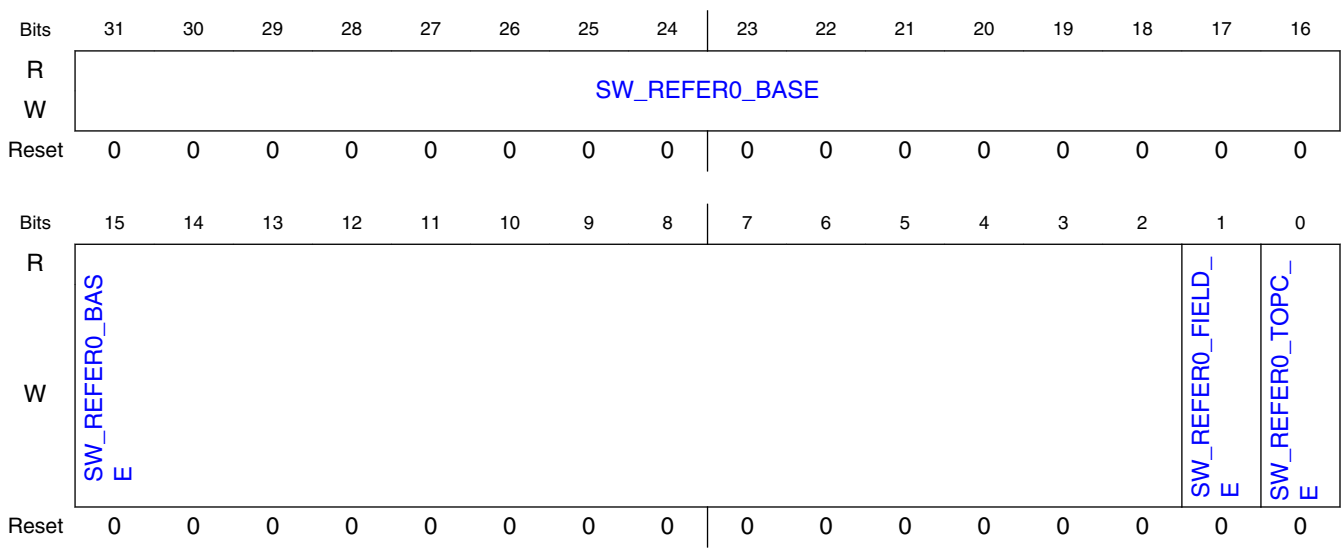
Field	Function
SW_PINIT_RLIST_F10	

15.1.5.2.10 Base address for reference picture index 0 / base address for JPEG decoder output chrominance picture (SWREG14)

15.1.5.2.10.1 Offset

Register	Offset
SWREG14	38h

15.1.5.2.10.2 Diagram



15.1.5.2.10.3 Fields

Field	Function
31-2 SW_REFER0_BASE	Base address for reference picture index 0. See picture index definition from toplevel_sp
1 SW_REFER0_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0	Which field of reference picture is closer to current picture:

Field	Function
SW_REFER0_TOPC_E	0b - Bottom field is closer to current picture 1b - Top field is closer to current picture

15.1.5.2.11 Base address for reference picture index 1 / JPEG control (SWREG15)

15.1.5.2.11.1 Offset

Register	Offset
SWREG15	3Ch

15.1.5.2.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER1_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER1_BASE														SW_REFER1_FIELD_E	SW_REFER1_TOPC_E
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.11.3 Fields

Field	Function
31-2 SW_REFER1_BASE	Base address for reference picture index 1. See picture index definition from toplevel_sp. For VP8 this base address is used as Chrominance base address for reference picture 0 (if vp8 stride configuration is enabled)
1 SW_REFER1_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0	Which field of reference picture is closer to current picture:

Field	Function
SW_REFER1_TOPC_E	0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.12 Base address for reference picture index 2 / List of VLC code lengths in first JPEG AC table (SWREG16)

15.1.5.2.12.1 Offset

Register	Offset
SWREG16	40h

15.1.5.2.12.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER2_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER2_BASE														SW_REFER2_FIELD_E	SW_REFER2_TOPC_E
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.12.3 Fields

Field	Function
31-2 SW_REFER2_BASE	Base address for reference picture index 2. See picture index definition from toplevel_sp. For VP8 video this base address is used as Golden reference chrominance base address (if vp8 stride configuration is enabled)
1 SW_REFER2_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0	Which field of reference picture is closer to current picture:

Field	Function
SW_REFER2_TOPC_E	0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.13 Base address for reference picture index 3 / List of VLC code lengths in first JPEG AC table (SWREG17)

15.1.5.2.13.1 Offset

Register	Offset
SWREG17	44h

15.1.5.2.13.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER3_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER3_BASE														SW_REFER3_FIELD_E	SW_REFER3_TOPC_E
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.13.3 Fields

Field	Function
31-2 SW_REFER3_BASE	Base address for reference picture index 3. See picture index definition from toplevel_sp. For VP8 video this base address is used as Alternate reference chrominance base address (if vp8 stride configuration is enabled)
1 SW_REFER3_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0	Which field of reference picture is closer to current picture:

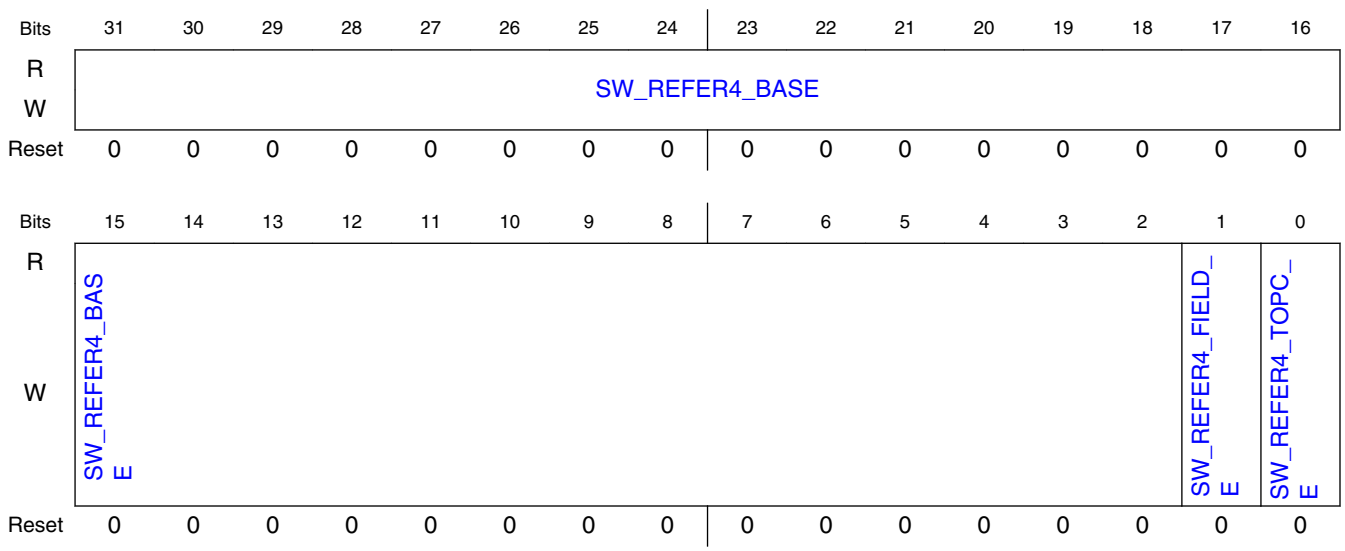
Field	Function
SW_REFER3_TOPC_E	0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.14 Base address for reference picture index 4 / VC1 control / MPEG4 MVD control/ List of VLC code lengths in first JPEG AC table / VC-1 intensity control 4 / VP6/VP7, VP8 Golden refer picture base (SWREG18)

15.1.5.2.14.1 Offset

Register	Offset
SWREG18	48h

15.1.5.2.14.2 Diagram



15.1.5.2.14.3 Fields

Field	Function
31-2 SW_REFER4_BASE	H264: Base address for reference picture index 4 VP6/VP7/VP8: Base address for Golden reference picture (corresponds picid 4)
1	Refer picture consist of single fields or frame: 0b - reference picture consists of frame

Table continues on the next page...

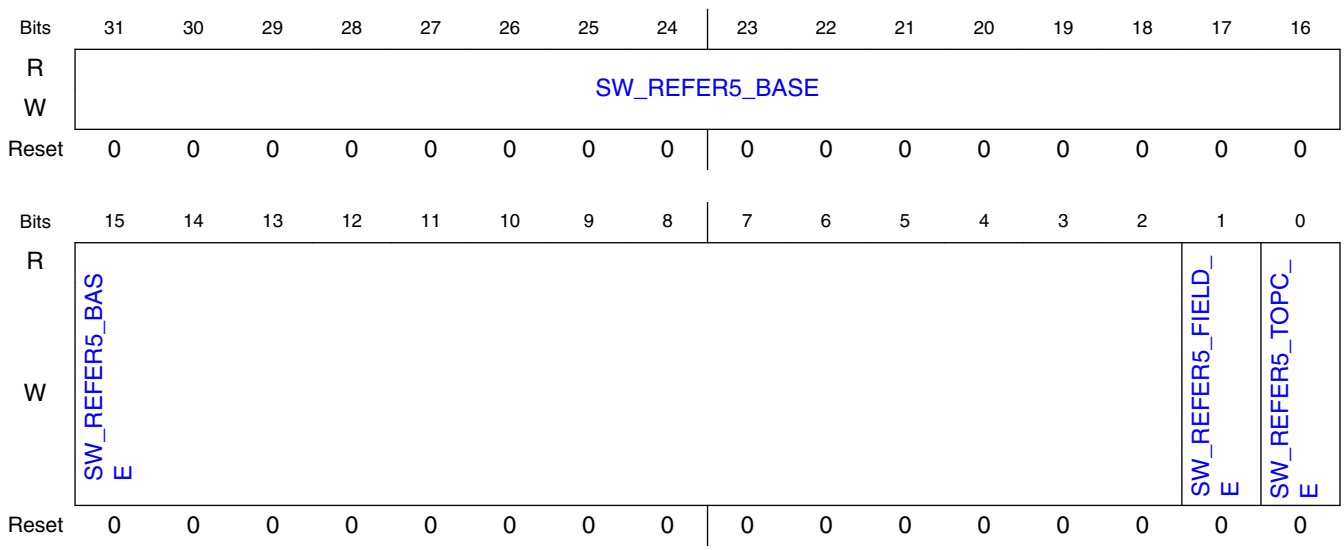
Field	Function
SW_REFER4_FIELD_E	1b - reference picture consists of fields
0 SW_REFER4_TOPC_E	Which field of reference picture is closer to current picture: 0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.15 Base address for reference picture index 5 / MPEG4 TRB/TRD delta 0 / VC-1 intensity control 3 List of VLC code lengths in first/second JPEG AC table / VP6/VP7 scan maps (SWREG19)

15.1.5.2.15.1 Offset

Register	Offset
SWREG19	4Ch

15.1.5.2.15.2 Diagram



15.1.5.2.15.3 Fields

Field	Function
31-2 SW_REFER5_BASE	H.264: Base address for reference picture index 5 VP8: Base address for alternate reference picture (corresponds picid 5)

Table continues on the next page...

Field	Function
1 SW_REFER5_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0 SW_REFER5_TOPC_E	Which field of reference picture is closer to current picture: 0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.16 Base address for reference picture index 6 / / MPEG4 TRB/TRD delta -1 / List of VLC code lengths in second JPEG AC table / VP6/VP7 scan maps (SWREG20)

15.1.5.2.16.1 Offset

Register	Offset
SWREG20	50h

15.1.5.2.16.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER6_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER6_BASE														SW_REFER6_FIELD_E	SW_REFER6_TOPC_E
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.16.3 Fields

Field	Function
31-2	Base address for reference picture index 6. For VP8 video this base address is used as decoder output chrominance base address (if vp8 stride configuration is enabled)

Table continues on the next page...

Field	Function
SW_REFER6_BASE	
1 SW_REFER6_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0 SW_REFER6_TOPC_E	Which field of reference picture is closer to current picture: 0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.17 Base address for reference picture index 7 / MPEG4 TRB/TRD delta 1 / List of VLC code lengths in second JPEG AC table / VP6/VP7 scan maps (SWREG21)

15.1.5.2.17.1 Offset

Register	Offset
SWREG21	54h

15.1.5.2.17.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER7_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER7_BASE														SW_REFER7_FIELD_E	SW_REFER7_TOPC_E
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.17.3 Fields

Field	Function
31-2	Base address for reference picture index 7

Table continues on the next page...

Field	Function
SW_REFER7_BASE	
1 SW_REFER7_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0 SW_REFER7_TOPC_E	Which field of reference picture is closer to current picture: 0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.18 Base address for reference picture index 8 / List of VLC code lengths in second JPEG AC table / VP6 scan maps / VP7,VP8 DCT stream 1 base (SWREG22)

15.1.5.2.18.1 Offset

Register	Offset
SWREG22	58h

15.1.5.2.18.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER8_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER8_BASE														SW_REFER8_FIELD_E	SW_REFER8_TOPC_E
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.18.3 Fields

Field	Function
31-2	Base address for reference picture index 8

Table continues on the next page...

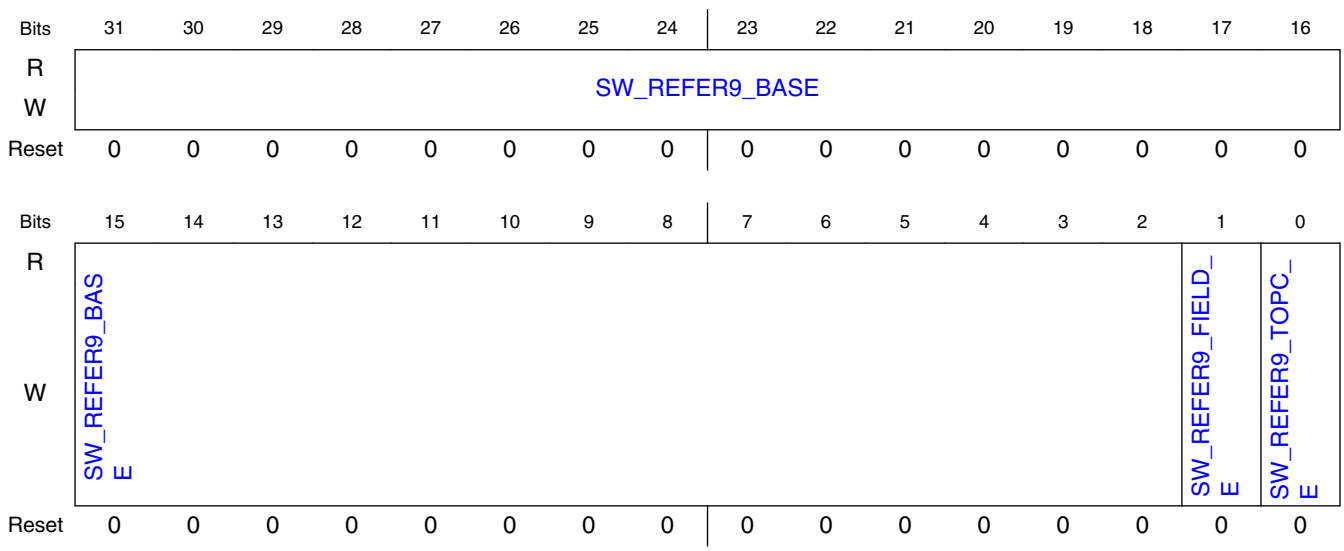
Field	Function
SW_REFER8_BASE	
1 SW_REFER8_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0 SW_REFER8_TOPC_E	Which field of reference picture is closer to current picture: 0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.19 Base address for reference picture index 9 / List of VLC code lengths in first JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 2 base (SWREG23)

15.1.5.2.19.1 Offset

Register	Offset
SWREG23	5Ch

15.1.5.2.19.2 Diagram



15.1.5.2.19.3 Fields

Field	Function
31-2	Base address for reference picture index 9

Table continues on the next page...

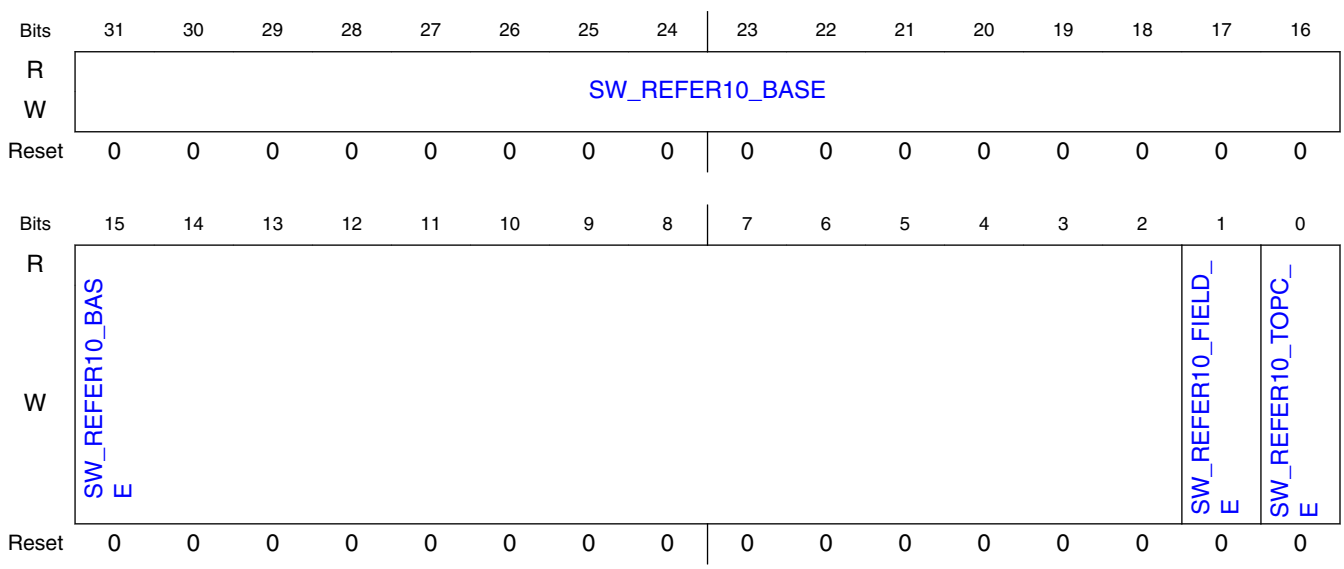
Field	Function
SW_REFER9_BASE	
1 SW_REFER9_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0 SW_REFER9_TOPC_E	Which field of reference picture is closer to current picture: 0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.20 Base address for reference picture index 10 / List of VLC code lengths in first JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 3 base (SWREG24)

15.1.5.2.20.1 Offset

Register	Offset
SWREG24	60h

15.1.5.2.20.2 Diagram



15.1.5.2.20.3 Fields

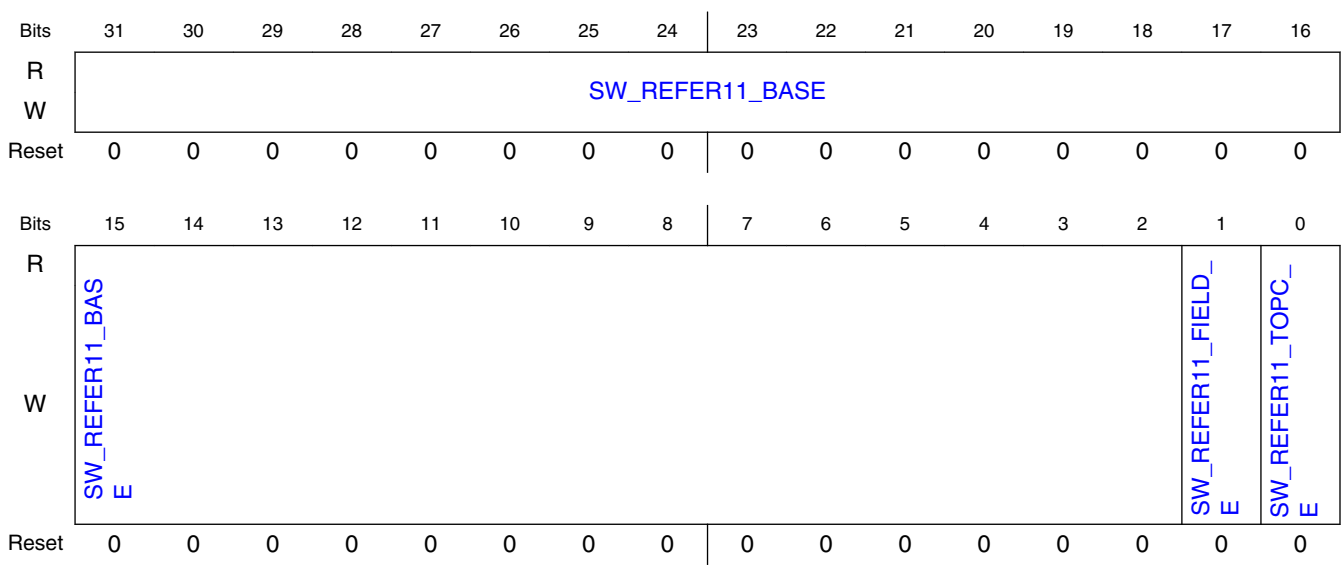
Field	Function
31-2 SW_REFER10_BASE	Base address for reference picture index 10
1 SW_REFER10_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0 SW_REFER10_TOPC_E	Which field of reference picture is closer to current picture: 0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.21 Base address for reference picture index 11 / List of VLC code lengths in second JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 4 base (SWREG25)

15.1.5.2.21.1 Offset

Register	Offset
SWREG25	64h

15.1.5.2.21.2 Diagram



15.1.5.2.21.3 Fields

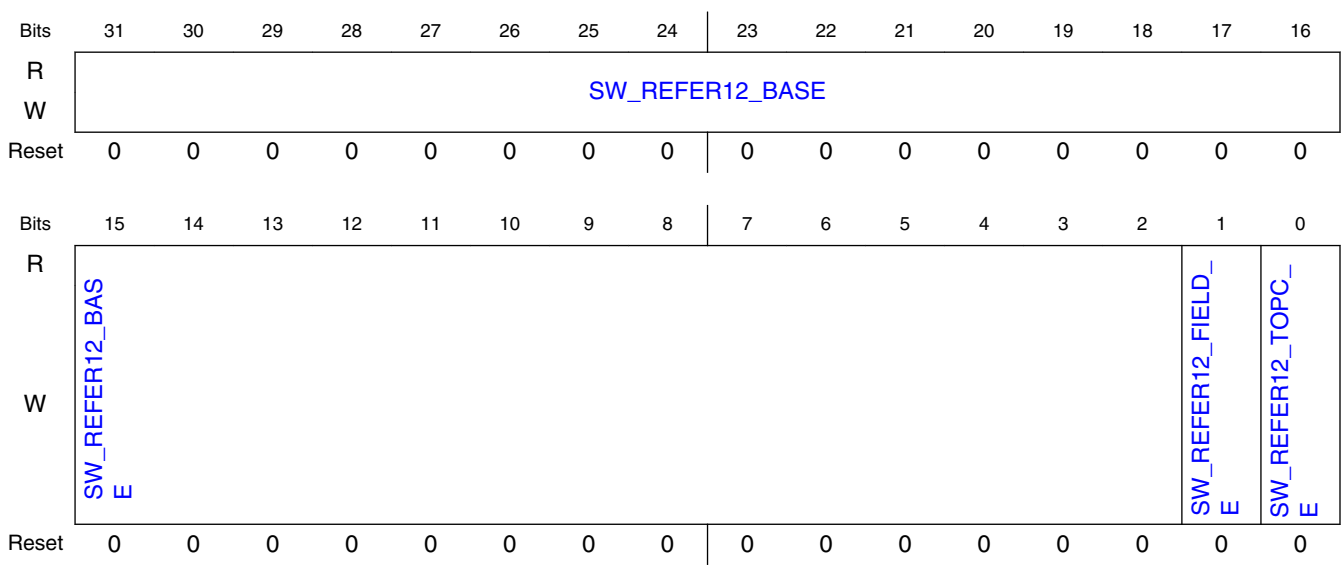
Field	Function
31-2 SW_REFER11_BASE	Base address for reference picture index 11
1 SW_REFER11_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0 SW_REFER11_TOPC_E	Which field of reference picture is closer to current picture: 0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.22 Base address for reference picture index 12 / List of VLC code lengths in second JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 5 base (SWREG26)

15.1.5.2.22.1 Offset

Register	Offset
SWREG26	68h

15.1.5.2.22.2 Diagram



15.1.5.2.22.3 Fields

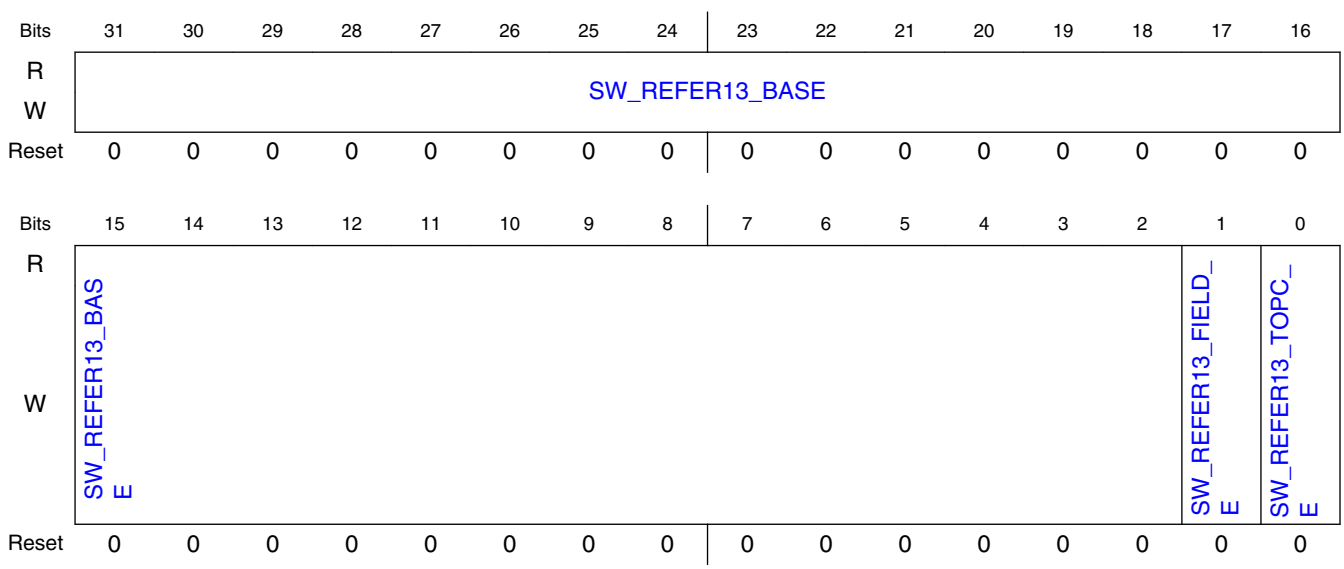
Field	Function
31-2 SW_REFER12_BASE	Base address for reference picture index 12
1 SW_REFER12_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0 SW_REFER12_TOPC_E	Which field of reference picture is closer to current picture: 0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.23 Base address for reference picture index 13 / VC-1 bitpl mbctrl or VP6,VP7,VP8 ctrl stream base /Progressive JPEG DC table (SWREG27)

15.1.5.2.23.1 Offset

Register	Offset
SWREG27	6Ch

15.1.5.2.23.2 Diagram



15.1.5.2.23.3 Fields

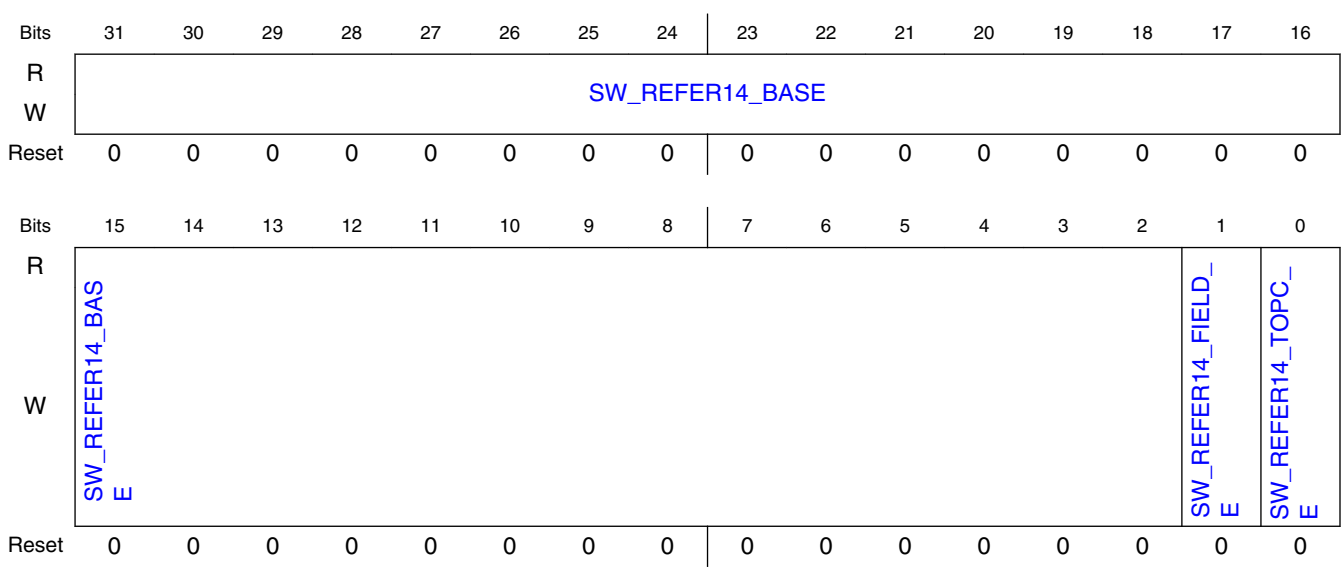
Field	Function
31-2 SW_REFER13_BASE	Base address for reference picture index 13
1 SW_REFER13_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0 SW_REFER13_TOPC_E	Which field of reference picture is closer to current picture: 0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.24 Base address for reference picture index 14 / VP6 scan maps / Progressive JPEG DC table / VP7,VP8 DCT stream 6 base (SWREG28)

15.1.5.2.24.1 Offset

Register	Offset
SWREG28	70h

15.1.5.2.24.2 Diagram



15.1.5.2.24.3 Fields

Field	Function
31-2 SW_REFER14_BASE	Base address for reference picture index 14
1 SW_REFER14_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0 SW_REFER14_TOPC_E	Which field of reference picture is closer to current picture: 0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.25 Base address for reference picture index 15 / VP6 scan maps / VP7,VP8 DCT stream 7 base (SWREG29)

15.1.5.2.25.1 Offset

Register	Offset
SWREG29	74h

15.1.5.2.25.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER15_BASE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER15_BASE														SW_REFER15_FIELD_E	SW_REFER15_TOPC_E
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.25.3 Fields

Field	Function
31-2 SW_REFER15_BASE	Base address for reference picture index 15. For Multi View Coding this base address refers to inter view base address
1 SW_REFER15_FIELD_E	Refer picture consist of single fields or frame: 0b - reference picture consists of frame 1b - reference picture consists of fields
0 SW_REFER15_TOPC_E	Which field of reference picture is closer to current picture: 0b - bottom field is closer to current picture 1b - top field is closer to current picture

15.1.5.2.26 Reference picture numbers for index 0 and 1 (H264 VLC) / VP6 scan maps / VP7,VP8 loop filter mb level adjusts (SWREG30)

15.1.5.2.26.1 Offset

Register	Offset
SWREG30	78h

15.1.5.2.26.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER1_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER0_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.26.3 Fields

Field	Function
31-16 SW_REFER1_NBR	Number for reference picture index 1

Table continues on the next page...

Field	Function
15-0 SW_REFER0_NBR	Number for reference picture index 0

15.1.5.2.27 Reference picture numbers for index 2 and 3 (H264 VLC) / VP6 scan maps / VP7,VP8 loop filter ref pic level adjusts (SWREG31)

15.1.5.2.27.1 Offset

Register	Offset
SWREG31	7Ch

15.1.5.2.27.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER3_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER2_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.27.3 Fields

Field	Function
31-16 SW_REFER3_NBR	Number for reference picture index 3
15-0 SW_REFER2_NBR	Number for reference picture index 2

15.1.5.2.28 Reference picture numbers for index 4 and 5 (H264 VLC) / VP6 scan maps / VP7,VP8 loop filter levels (SWREG32)

15.1.5.2.28.1 Offset

Register	Offset
SWREG32	80h

15.1.5.2.28.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER5_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER4_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.28.3 Fields

Field	Function
31-16 SW_REFER5_NBR	Number for reference picture index 5
15-0 SW_REFER4_NBR	Number for reference picture index 4

15.1.5.2.29 Reference picture numbers for index 6 and 7 (H264 VLC) / VP6 scan maps / VP7,VP8 quantization values (SWREG33)

15.1.5.2.29.1 Offset

Register	Offset
SWREG33	84h

15.1.5.2.29.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER7_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER6_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.29.3 Fields

Field	Function
31-16 SW_REFER7_NBR	Number for reference picture index 7
15-0 SW_REFER6_NBR	Number for reference picture index 6

15.1.5.2.30 Reference picture numbers for index 8 and 9 (H264 VLC) / MPEG4, VC1, VPx prediction filter taps (SWREG34)

15.1.5.2.30.1 Offset

Register	Offset
SWREG34	88h

15.1.5.2.30.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER9_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER8_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.30.3 Fields

Field	Function
31-16 SW_REFER9_NBR	Number for reference picture index 9
15-0 SW_REFER8_NBR	Number for reference picture index 8

15.1.5.2.31 Reference picture numbers for index 10 and 11 (H264 VLC) / VC1, VPx prediction filter taps (SWREG35)

15.1.5.2.31.1 Offset

Register	Offset
SWREG35	8Ch

15.1.5.2.31.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER11_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER10_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.31.3 Fields

Field	Function
31-16 SW_REFER11_NBR	Number for reference picture index 11
15-0 SW_REFER10_NBR	Number for reference picture index 10

15.1.5.2.32 Reference picture numbers for index 12 and 13 (H264 VLC) / VC1, VPx prediction filter taps (SWREG36)

15.1.5.2.32.1 Offset

Register	Offset
SWREG36	90h

15.1.5.2.32.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER13_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER12_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.32.3 Fields

Field	Function
31-16 SW_REFER13_NBR	Number for reference picture index 13
15-0 SW_REFER12_NBR	Number for reference picture index 12

15.1.5.2.33 Reference picture numbers for index 14 and 15 (H264 VLC) / VPx prediction filter taps (SWREG37)

15.1.5.2.33.1 Offset

Register	Offset
SWREG37	94h

15.1.5.2.33.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER15_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER14_NBR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.33.3 Fields

Field	Function
31-16 SW_REFER15_NBR	Number for reference picture index 15
15-0 SW_REFER14_NBR	Number for reference picture index 14

15.1.5.2.34 Reference picture long term flags (H264 VLC) / VPx prediction filter taps (SWREG38)

15.1.5.2.34.1 Offset

Register	Offset
SWREG38	98h

15.1.5.2.34.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER_LTERM_E															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER_LTERM_E															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.34.3 Fields

Field	Function
31-0 SW_REFER_LTERM_E	Long term flag for reference picture index [31:0]. Definition: If frame is being decoded the bits 31:15 are used, Bit 31 for picture index 0, Bit 30 for picture index 1 etc… IF field is being decoded the bits 31:0 are used, Bit 31 for reference picture 0 top field, bit 30 for reference picture 0 bottom field etc...

15.1.5.2.35 Reference picture valid flags (H264 VLC) / VPx prediction filter taps (SWREG39)

15.1.5.2.35.1 Offset

Register	Offset
SWREG39	9Ch

15.1.5.2.35.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER_VALID_E															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER_VALID_E															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.2.35.3 Fields

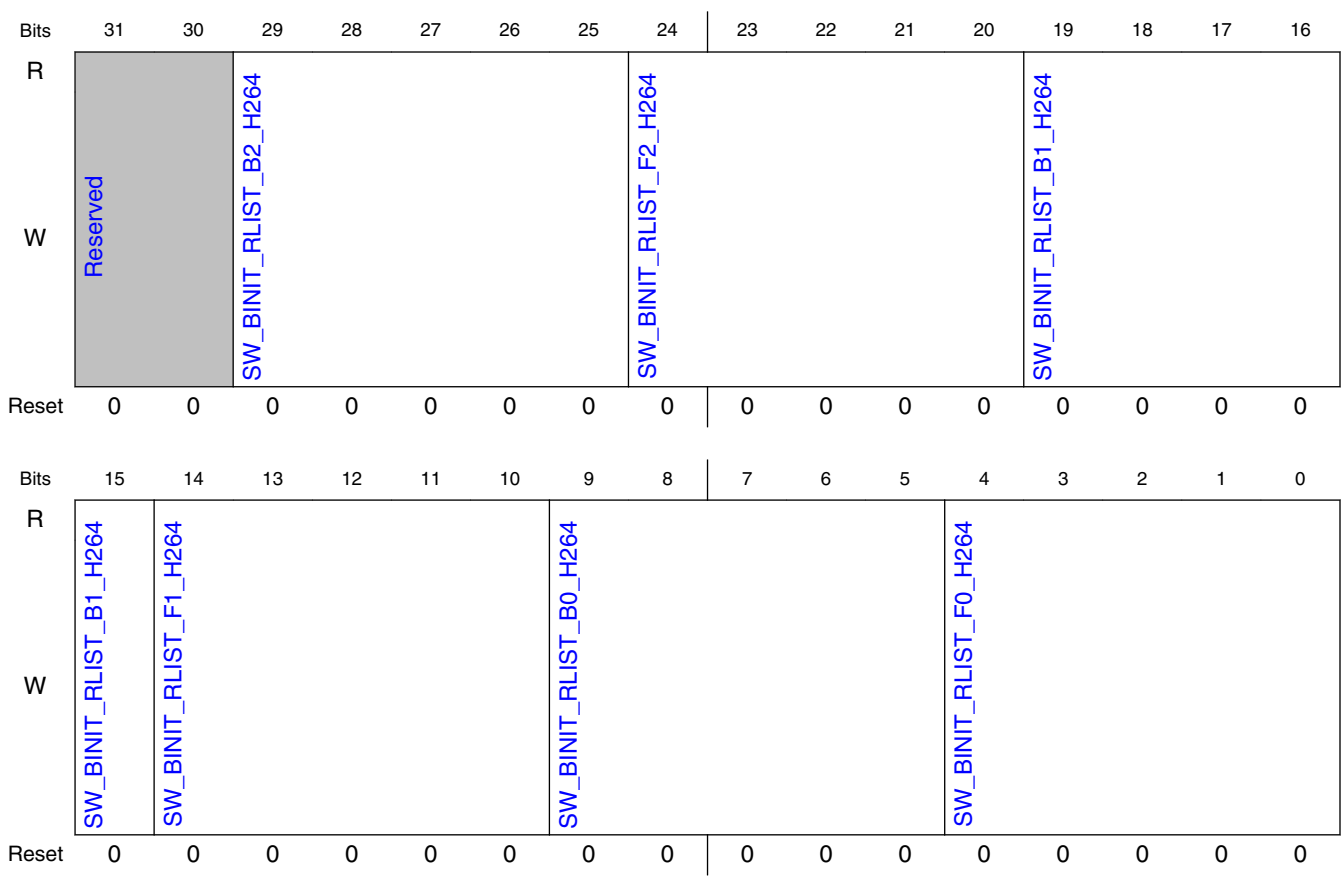
Field	Function
31-0 SW_REFER_VALID_E	Valid flag for reference picture index [31:0].Definition: If frame is being decoded the bits 31:15 are used, Bit 31 for picture index 0, Bit 30 for picture index 1 etc… IF field is being decoded the bits 31:0 are used, Bit 31 for reference picture 0 top field, bit 30 for reference picture 0 bottom field etc...

15.1.5.2.36 bi_dir initial ref pic list register (0-2) / VP6 prediction filter taps / Progressive JPEG Cb ACDC coefficient base (SWREG42_H264)

15.1.5.2.36.1 Offset

Register	Offset
SWREG42_H264	A8h

15.1.5.2.36.2 Diagram



15.1.5.2.36.3 Fields

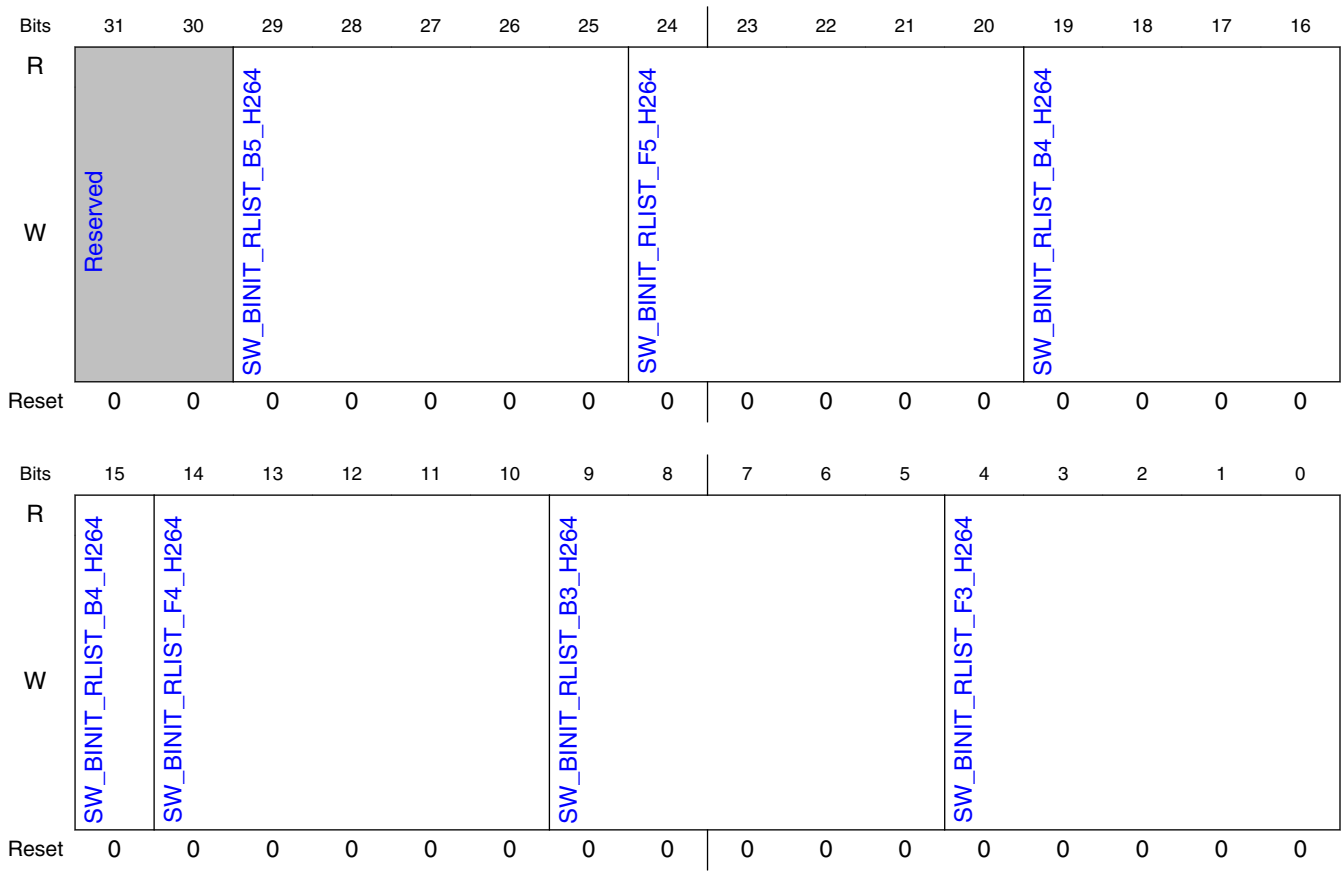
Field	Function
31-30 —	Reserved.
29-25 SW_BINIT_RLIST_B2_H264	Initial reference picture list for bi-direct backward picid 2
24-20 SW_BINIT_RLIST_F2_H264	Initial reference picture list for bi-direct forward picid 2
19-15 SW_BINIT_RLIST_B1_H264	Initial reference picture list for bi-direct backward picid 1
14-10 SW_BINIT_RLIST_F1_H264	Initial reference picture list for bi-direct forward picid 1
9-5 SW_BINIT_RLIST_B0_H264	Initial reference picture list for bi-direct backward picid 0
4-0 SW_BINIT_RLIST_F0_H264	Initial reference picture list for bi-direct forward picid 0

15.1.5.2.37 bi-dir initial ref pic list register (3-5) / VP6 prediction filter taps / Progressive JPEG Cr ACDC coefficient base (SWREG43_H264)

15.1.5.2.37.1 Offset

Register	Offset
SWREG43_H264	ACh

15.1.5.2.37.2 Diagram



15.1.5.2.37.3 Fields

Field	Function
31-30 —	Reserved.
29-25 SW_BINIT_RLIST_B5_H264	Initial reference picture list for bi-direct backward picid 5
24-20 SW_BINIT_RLIST_F5_H264	Initial reference picture list for bi-direct forward picid 5
19-15 SW_BINIT_RLIST_B4_H264	Initial reference picture list for bi-direct backward picid 4
14-10 SW_BINIT_RLIST_F4_H264	Initial reference picture list for bi-direct forward picid 4
9-5	Initial reference picture list for bi-direct backward picid 3

Table continues on the next page...

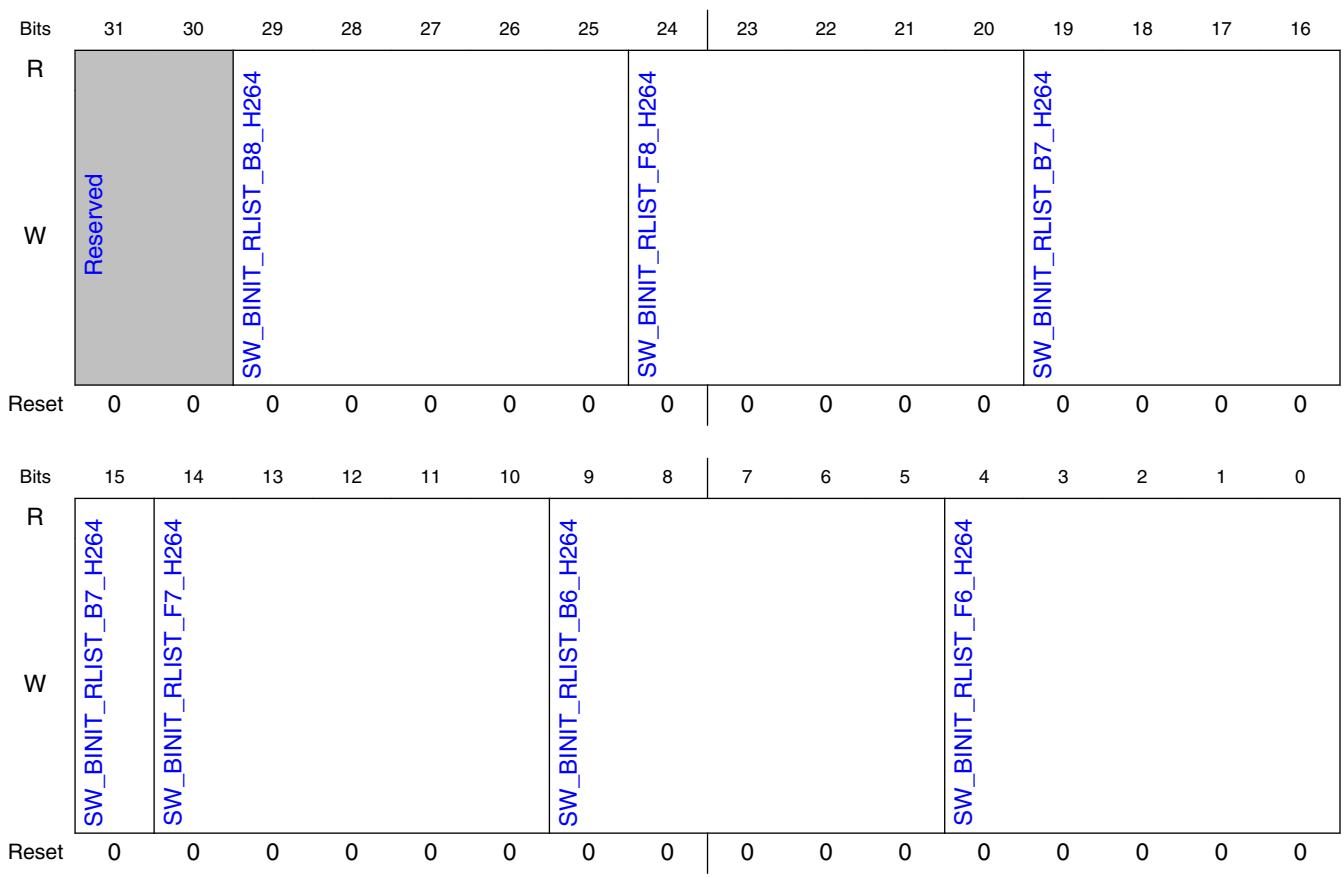
Field	Function
SW_BINIT_RLIST_B3_H264	
4-0 SW_BINIT_RLIST_F3_H264	Initial reference picture list for bi-direct forward picid 3

15.1.5.2.38 bi-dir initial ref pic list register (6-8) / VP6 prediction filter taps (SWREG44_H264)

15.1.5.2.38.1 Offset

Register	Offset
SWREG44_H264	B0h

15.1.5.2.38.2 Diagram



15.1.5.2.38.3 Fields

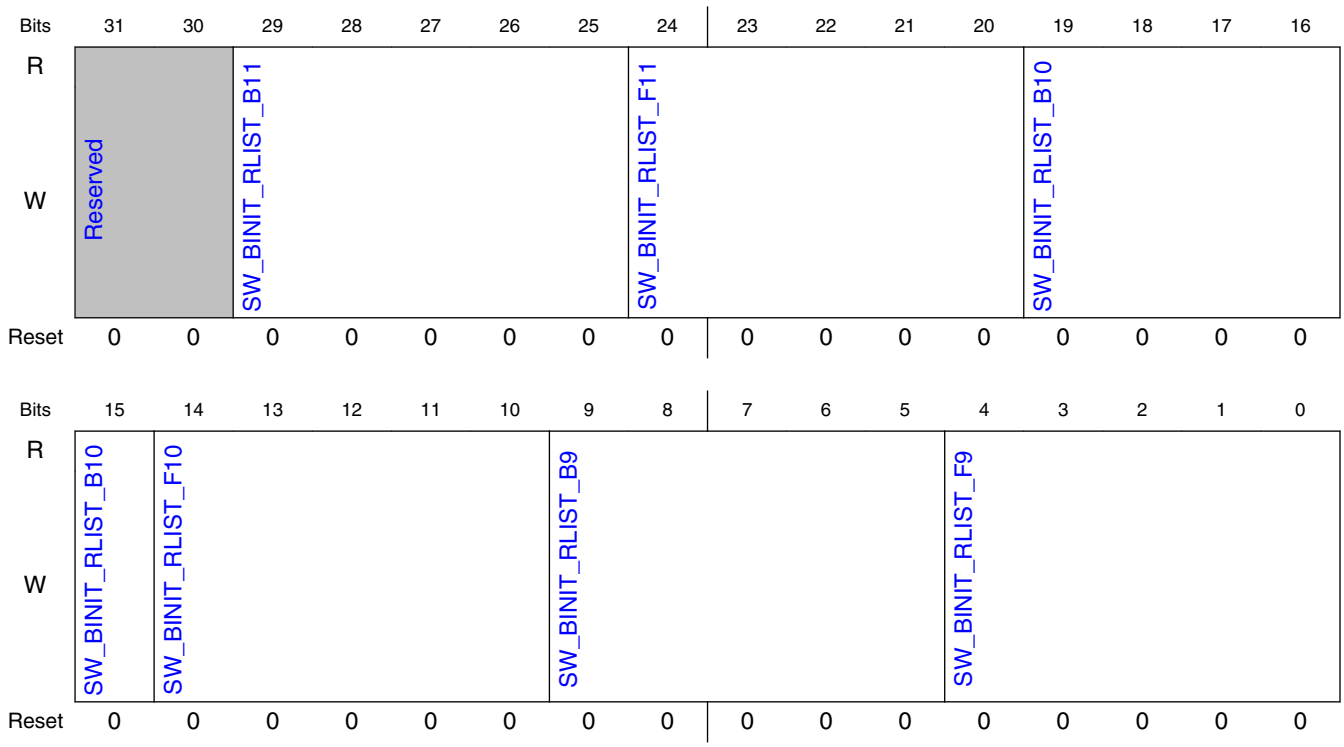
Field	Function
31-30 —	Reserved.
29-25 SW_BINIT_RLIST_B8_H264	Initial reference picture list for bi-direct backward picid 8
24-20 SW_BINIT_RLIST_F8_H264	Initial reference picture list for bi-direct forward picid 8
19-15 SW_BINIT_RLIST_B7_H264	Initial reference picture list for bi-direct backward picid 7
14-10 SW_BINIT_RLIST_F7_H264	Initial reference picture list for bi-direct forward picid 7
9-5 SW_BINIT_RLIST_B6_H264	Initial reference picture list for bi-direct backward picid 6
4-0 SW_BINIT_RLIST_F6_H264	Initial reference picture list for bi-direct forward picid 6

15.1.5.2.39 bi-dir initial ref pic list register (9-11) / VP6 prediction filter taps (SWREG45)

15.1.5.2.39.1 Offset

Register	Offset
SWREG45	B4h

15.1.5.2.39.2 Diagram



15.1.5.2.39.3 Fields

Field	Function
31-30 —	Reserved.
29-25 SW_BINIT_RLIST_B11	Initial reference picture list for bi-direct backward picid 11
24-20 SW_BINIT_RLIST_F11	Initial reference picture list for bi-direct forward picid 11
19-15 SW_BINIT_RLIST_B10	Initial reference picture list for bi-direct backward picid 10
14-10 SW_BINIT_RLIST_F10	Initial reference picture list for bi-direct forward picid 10
9-5 SW_BINIT_RLIST_B9	Initial reference picture list for bi-direct backward picid 9
4-0	Initial reference picture list for bi-direct forward picid 9

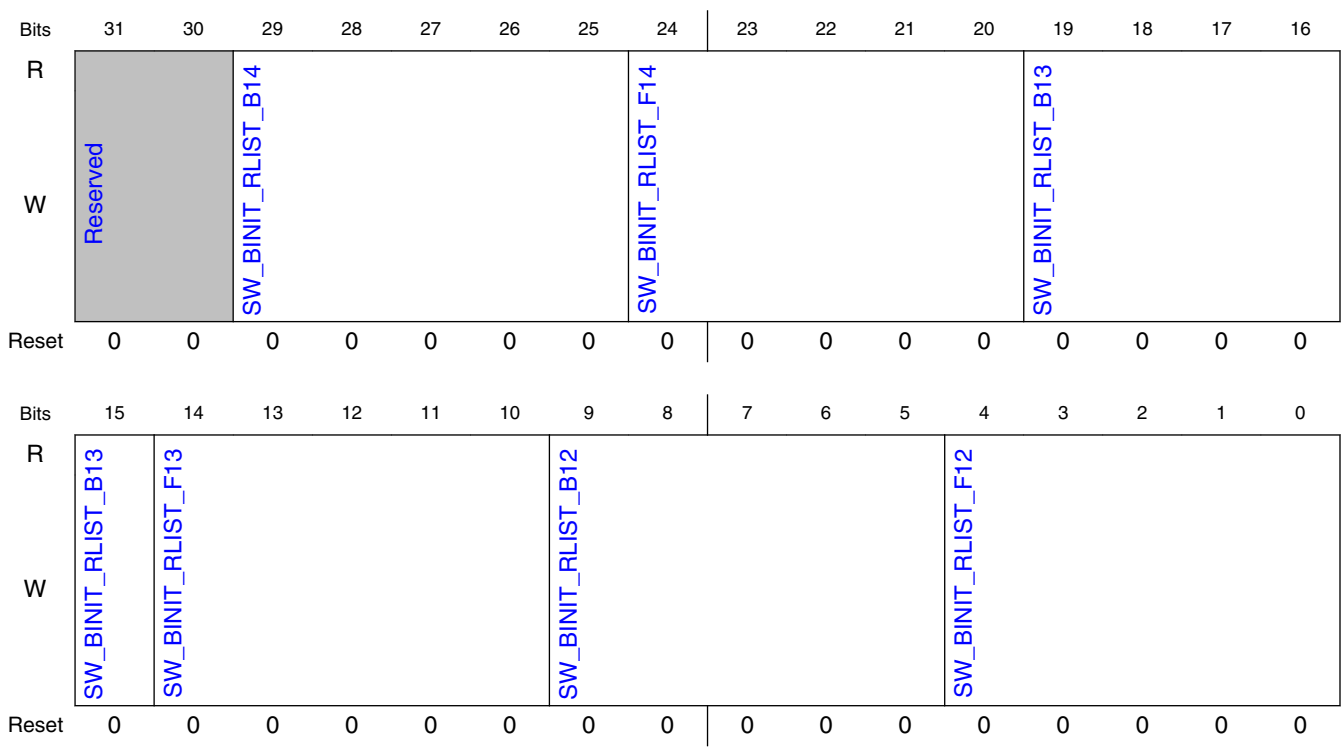
Field	Function
SW_BINIT_RLIST_F9	

15.1.5.2.40 bi-dir initial ref pic list register (12-14) / VP7,VP8 quantization values (SWREG46)

15.1.5.2.40.1 Offset

Register	Offset
SWREG46	B8h

15.1.5.2.40.2 Diagram



15.1.5.2.40.3 Fields

Field	Function
31-30 —	Reserved.

Table continues on the next page...

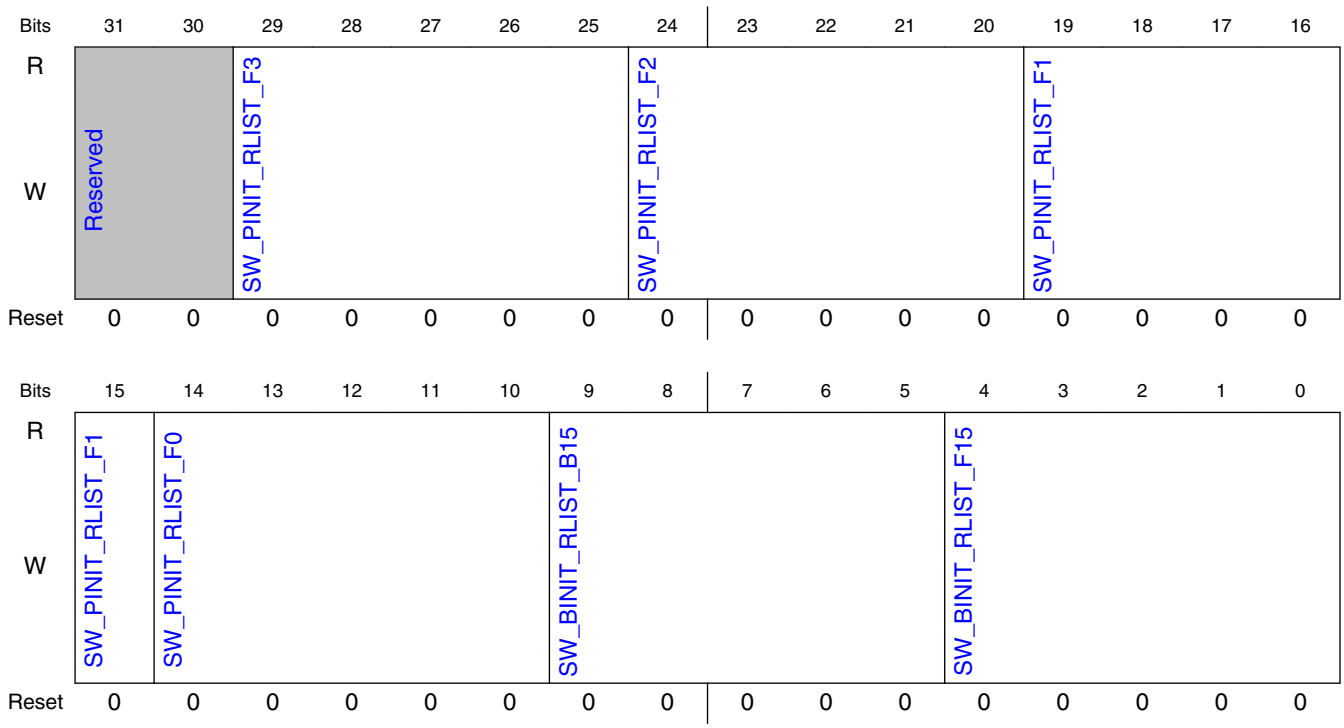
Field	Function
29-25 SW_BINIT_RLIST_B14	Initial reference picture list for bi-direct backward picid 14
24-20 SW_BINIT_RLIST_F14	Initial reference picture list for bi-direct forward picid 14
19-15 SW_BINIT_RLIST_B13	Initial reference picture list for bi-direct backward picid 13
14-10 SW_BINIT_RLIST_F13	Initial reference picture list for bi-direct forward picid 13
9-5 SW_BINIT_RLIST_B12	Initial reference picture list for bi-direct backward picid 12
4-0 SW_BINIT_RLIST_F12	Initial reference picture list for bi-direct forward picid 12

15.1.5.2.41 bi-dir and P fwd initial ref pic list register (15 and P 0-3) / VP7,VP8 quantization values (SWREG47)

15.1.5.2.41.1 Offset

Register	Offset
SWREG47	BCh

15.1.5.2.41.2 Diagram



15.1.5.2.41.3 Fields

Field	Function
31-30 —	Reserved.
29-25 SW_PINIT_RLIST_F3	Initial reference picture list for P forward picid 3
24-20 SW_PINIT_RLIST_F2	Initial reference picture list for P forward picid 2
19-15 SW_PINIT_RLIST_F1	Initial reference picture list for P forward picid 1
14-10 SW_PINIT_RLIST_F0	Initial reference picture list for P forward picid 0
9-5 SW_BINIT_RLIST_B15	Initial reference picture list for bi-direct backward picid 15
4-0	Initial reference picture list for bi-direct forward picid 15

Field	Function
SW_BINIT_RLIST_F15	

15.1.5.3 VPU_G1 VP8 decoder register descriptions

15.1.5.3.1 VPU_G1 VP8 decoder memory map

VPU_G1_VP8 base address: 3830_0000h

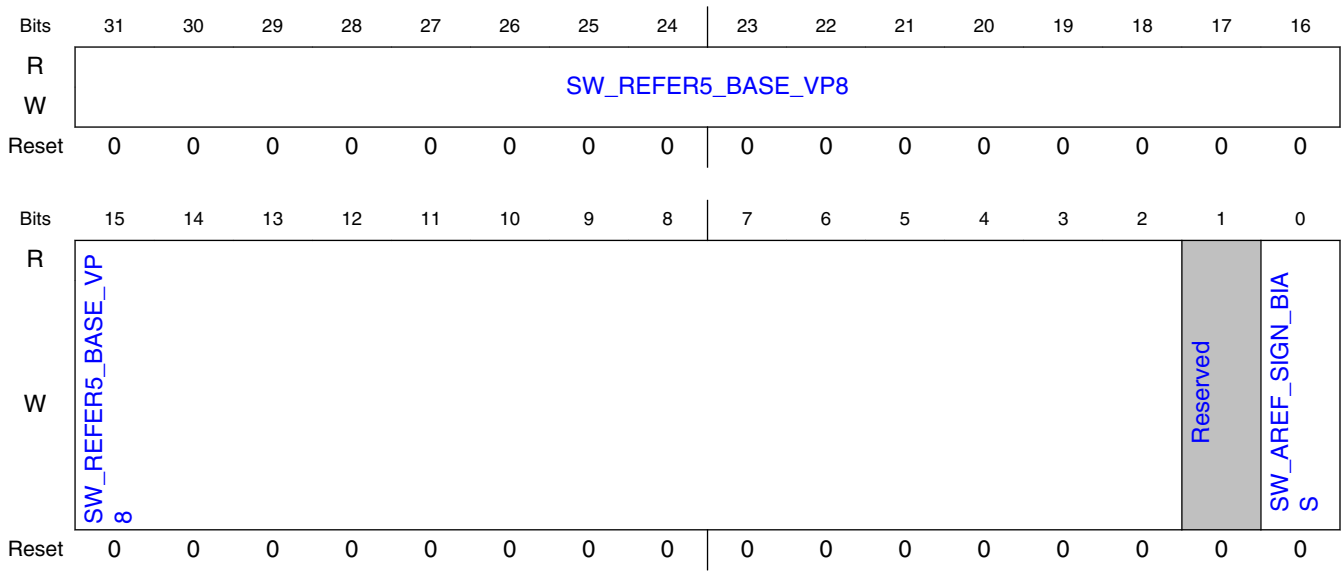
Offset	Register	Width (In bits)	Access	Reset value
4Ch	Base address for reference picture index 5 / MPEG4 TRB/TRD delta 0 / VC-1 intensity control 3 List of VLC code lengths in first/second JPEG AC table / VP6/VP7 scan maps (SWREG19_VP8)	32	RW	0000_0000h
50h	Base address for reference picture index 6 / / MPEG4 TRB/TRD delta -1 / List of VLC code lengths in second JPEG AC table / VP6/VP7 scan maps (SWREG20_VP8)	32	RW	0000_0000h
54h	Base address for reference picture index 7 / MPEG4 TRB/TRD delta 1 / List of VLC code lengths in second JPEG AC table / VP6/VP7 scan maps (SWREG21_VP8)	32	RW	0000_0000h

15.1.5.3.2 Base address for reference picture index 5 / MPEG4 TRB/TRD delta 0 / VC-1 intensity control 3 List of VLC code lengths in first/second JPEG AC table / VP6/VP7 scan maps (SWREG19_VP8)

15.1.5.3.2.1 Offset

Register	Offset
SWREG19_VP8	4Ch

15.1.5.3.2.2 Diagram



15.1.5.3.2.3 Fields

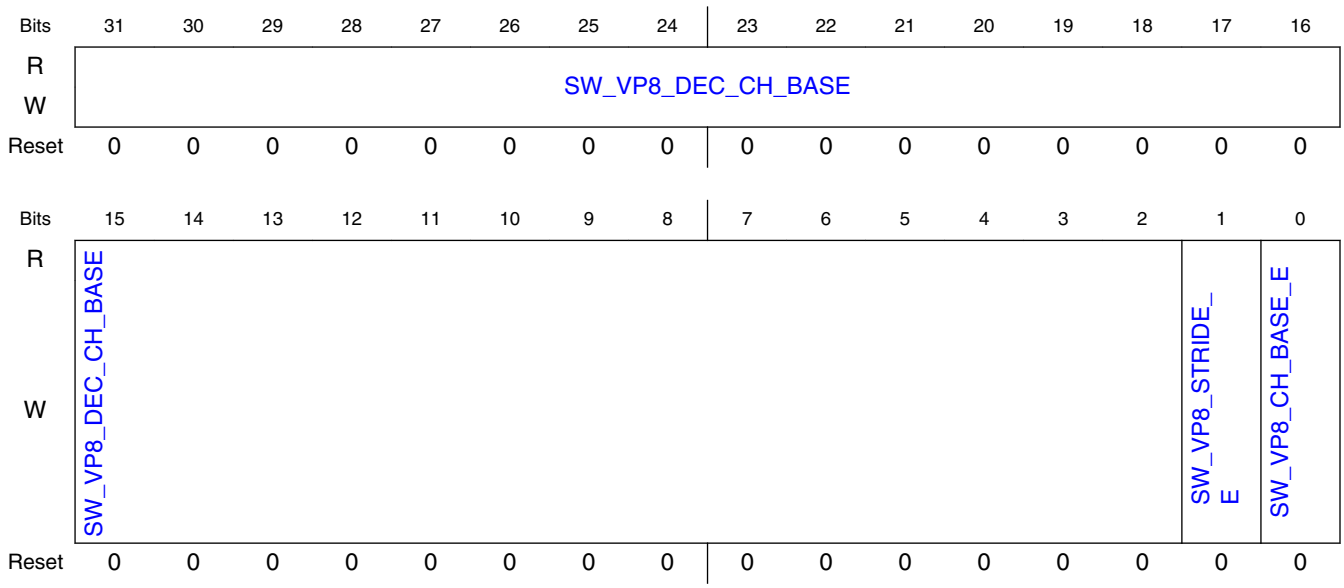
Field	Function
31-2 SW_REFER5_BASE_VP8	H.264: Base address for reference picture index 5 VP8: Base address for alternate reference picture (corresponds picid 5)
1 —	Reserved.
0 SW_AREF_SIGN_BIAS	VP8 only: Reference picture sign bias for Alternate reference frame

15.1.5.3.3 Base address for reference picture index 6 / / MPEG4 TRB/TRD delta -1 / List of VLC code lengths in second JPEG AC table / VP6/VP7 scan maps (SWREG20_VP8)

15.1.5.3.3.1 Offset

Register	Offset
SWREG20_VP8	50h

15.1.5.3.3.2 Diagram



15.1.5.3.3.3 Fields

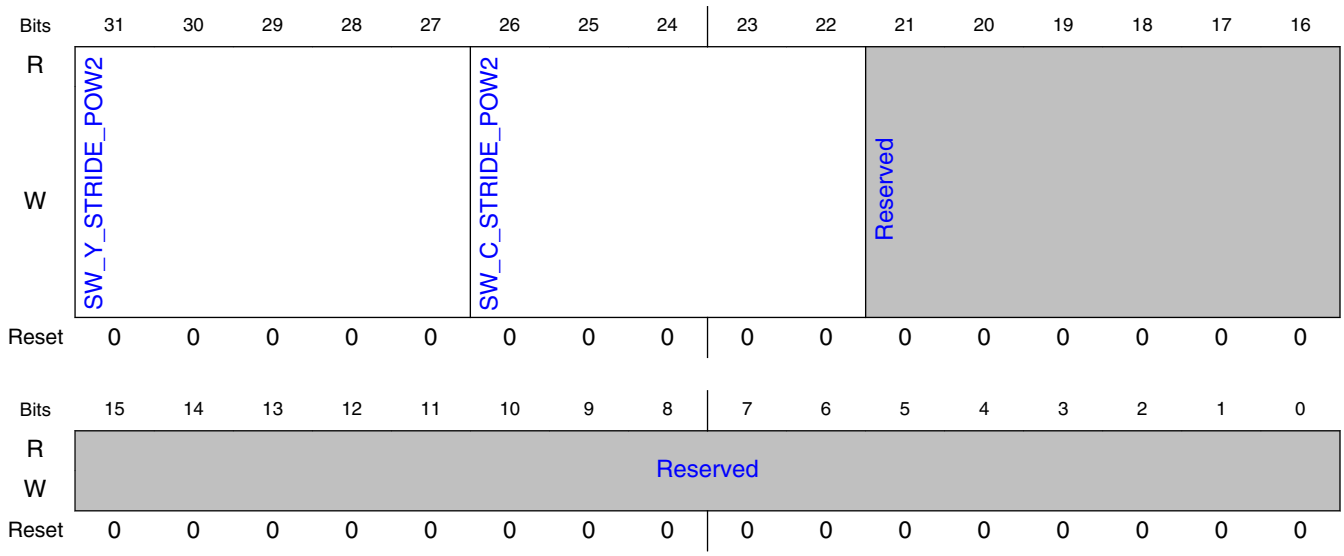
Field	Function
31-2 SW_VP8_DEC_CH_BASE	VP8 video base address for decoder output chrominance data (if vp8 stride configuration is enabled)
1 SW_VP8_STRIDE_E	VP8 stride enable. Can be set high only if HW configuration supports strides. Y and C strides are used instead of picture width. Separate chrominance base addresses are used instead of internal chrominance offsets. 0b - Not enabled 1b - Enabled
0 SW_VP8_CH_BASE_E	VP8 separate chrominance enable: 0b - Write/Read chrominance data from internal offset after the luminance data 1b - Write/Read chrominance data from separate base addresses given by SW

15.1.5.3.4 Base address for reference picture index 7 / MPEG4 TRB/TRD delta 1 / List of VLC code lengths in second JPEG AC table / VP6/VP7 scan maps (SWREG21_VP8)

15.1.5.3.4.1 Offset

Register	Offset
SWREG21_VP8	54h

15.1.5.3.4.2 Diagram



15.1.5.3.4.3 Fields

Field	Function
31-27 SW_Y_STRIDE_POW2	VP8 Y stride length informed by 2^n (n=sw_y_stride_pow2). Valid range 10-17 for 32 bit bus and 10-18 for 64 bit bus
26-22 SW_C_STRIDE_POW2	VP8 C stride length informed by 2^n (n=sw_c_stride_pow2). Valid range 10-17 for 32 bit bus and 10-18 for 64 bit bus
21-0 —	Reserved.

15.1.5.4 VPU_G1 VP7/VP8 decoder register descriptions

15.1.5.4.1 VPU_G1 VP7/VP8 decoder memory map

VPU_G1_VP7_VP8 base address: 3830_0000h

Offset	Register	Width (In bits)	Access	Reset value
10h	Decoder control register 1 (picture parameters) (SWREG4_JPEG_VP7_VP8)	32	RW	0000_0000h
14h	Decoder control register 2 (stream decoding table selects) (SWREG5_VP7_VP8)	32	RW	0000_0000h
18h	Decoder control register 3 (stream buffer information) (SWREG6_VP7_VP8)	32	RW	0000_0000h
1Ch	Decoder control register 4 (H264, VC-1, VP6 and progressive JPEG control) (SWREG7_VP7_VP8)	32	RW	0000_0000h
28h	Base address for differential motion vector base address (RLC-mode) /H264 P initial fwd ref pic list register (4-9)/ VC-1 intensity control 1/ VP7 and VP8 segmentation base register (SWREG10_VP7_VP8)	32	RW	0000_0000h
2Ch	Decoder control register 7 (VLC) / base address for H.264 intra prediction 4x4 / base address for MPEG-4 DC component (RLC) / H264 P initial fwd ref pic list register (10-15) / VC-1 intensity control 2 (SWREG11_VP7_VP8)	32	RW	0000_0000h
38h	Base address for reference picture index 0 / base address for JPEG decoder output chrominance picture (SWREG14_VP7_VP8)	32	RW	0000_0000h
3Ch	Base address for reference picture index 1 / JPEG control (SWREG15_VP7_VP8)	32	RW	0000_0000h
48h	Base address for reference picture index 4 / VC1 control / MPEG4 MVD control/ List of VLC code lengths in first JPEG AC table / VC-1 intensity control 4 / VP6/VP7, VP8 Golden refer picture base (SWREG18_VP7_VP8)	32	RW	0000_0000h
58h	Base address for reference picture index 8 / List of VLC code lengths in second JPEG AC table / VP6 scan maps / VP7,VP8 DCT stream 1 base (SWREG22_VP7_VP8)	32	RW	0000_0000h
5Ch	Base address for reference picture index 9 / List of VLC code lengths in first JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 2 base (SWREG23_VP7_VP8)	32	RW	0000_0000h
60h	Base address for reference picture index 10 / List of VLC code lengths in first JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 3 base (SWREG24_VP7_VP8)	32	RW	0000_0000h
64h	Base address for reference picture index 11 / List of VLC code lengths in second JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 4 base (SWREG25_VP7_VP8)	32	RW	0000_0000h
68h	Base address for reference picture index 12 / List of VLC code lengths in second JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 5 base (SWREG26_VP7_VP8)	32	RW	0000_0000h
6Ch	Base address for reference picture index 13 / VC-1 bitpl mbctrl or VP6,VP7,VP8 ctrl stream base /Progressive JPEG DC table (SWREG27_VC1)	32	RW	0000_0000h
70h	Base address for reference picture index 14 / VP6 scan maps / Progressive JPEG DC table / VP7,VP8 DCT stream 6 base (SWREG28_VP7_VP8)	32	RW	0000_0000h
74h	Base address for reference picture index 15 / VP6 scan maps / VP7,VP8 DCT stream 7 base (SWREG29_VP7_VP8)	32	RW	0000_0000h
78h	Reference picture numbers for index 0 and 1 (H264 VLC) / VP6 scan maps / VP7,VP8 loop filter mb level adjusts (SWREG30_VP7_VP8)	32	RW	0000_0000h

Table continues on the next page...

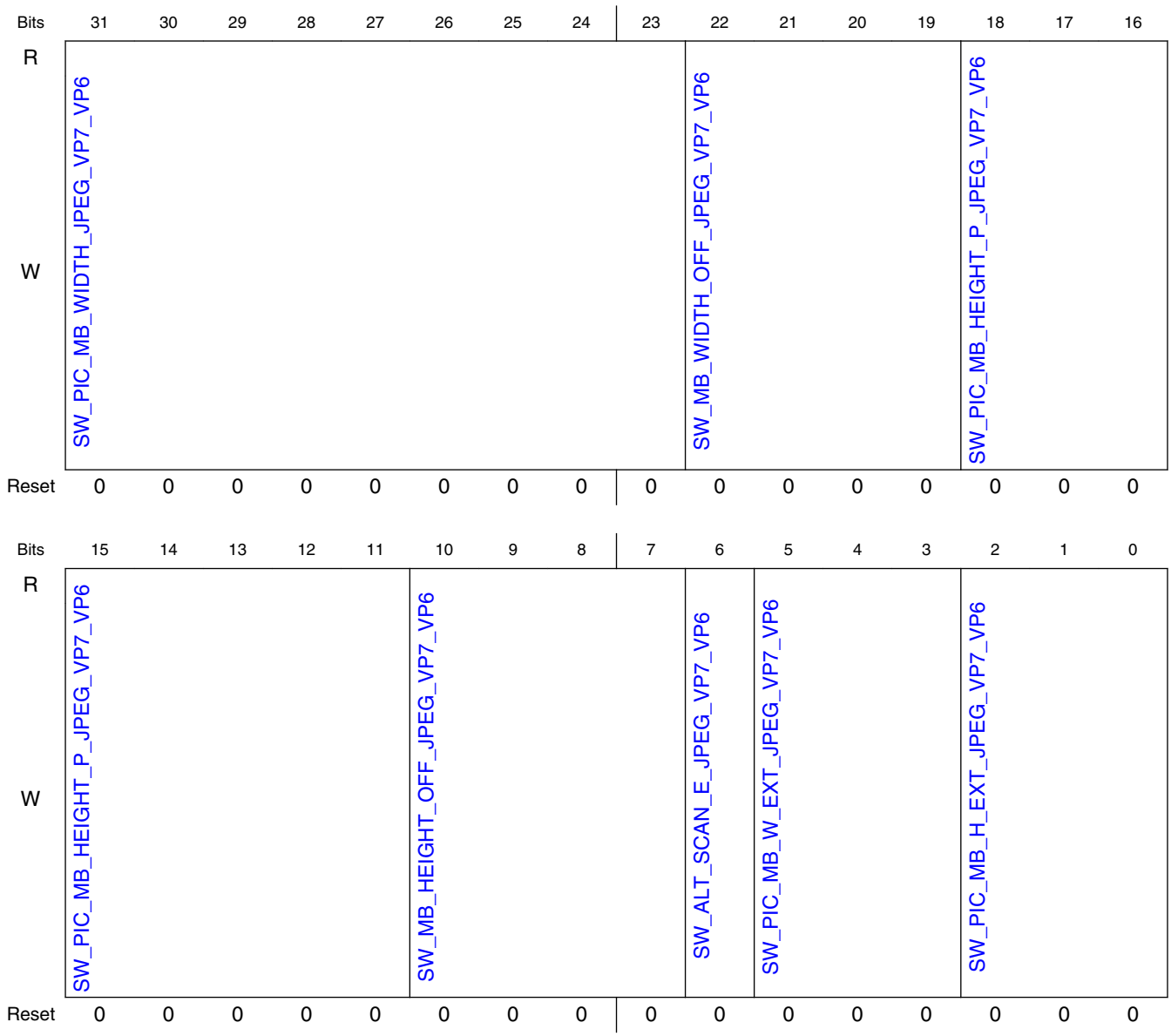
Offset	Register	Width (In bits)	Access	Reset value
7Ch	Reference picture numbers for index 2 and 3 (H264 VLC) / VP6 scan maps / VP7,VP8 loop filter ref pic level adjusts (SWREG31_VP7_VP8)	32	RW	0000_0000h
80h	Reference picture numbers for index 4 and 5 (H264 VLC) / VP6 scan maps / VP7,VP8 loop filter levels (SWREG32_VP7_VP8)	32	RW	0000_0000h
84h	Reference picture numbers for index 6 and 7 (H264 VLC) / VP6 scan maps / VP7,VP8 quantization values (SWREG33_VP7_VP8)	32	RW	0000_0000h
88h	Reference picture numbers for index 8 and 9 (H264 VLC) / MPEG4, VC1, VPx prediction filter taps (SWREG34_H263)	32	RW	0000_0000h
8Ch	Reference picture numbers for index 10 and 11 (H264 VLC) / VC1, VPx prediction filter taps (SWREG35_VC1)	32	RW	0000_0000h
90h	Reference picture numbers for index 12 and 13 (H264 VLC) / VC1, VPx prediction filter taps (SWREG36_VC1)	32	RW	0000_0000h
94h	Reference picture numbers for index 14 and 15 (H264 VLC) / VPx prediction filter taps (SWREG37_VP6_VP7_VP8)	32	RW	0000_0000h
98h	Reference picture long term flags (H264 VLC) / VPx prediction filter taps (SWREG38_VP6_VP7_VP8)	32	RW	0000_0000h
9Ch	Reference picture valid flags (H264 VLC) / VPx prediction filter taps (SWREG39_VP6_VP7_VP8)	32	RW	0000_0000h
A8h	bi_dir initial ref pic list register (0-2) / VP6 prediction filter taps / Progressive JPEG Cb ACDC coefficient base (SWREG42_VP6)	32	RW	0000_0000h
ACh	bi-dir initial ref pic list register (3-5) / VP6 prediction filter taps / Progressive JPEG Cr ACDC coefficient base (SWREG43_VP7_VP8)	32	RW	0000_0000h
B0h	bi-dir initial ref pic list register (6-8) / VP6 prediction filter taps (SWREG44_VP7_VP8)	32	RW	0000_0000h
B4h	bi-dir initial ref pic list register (9-11) / VP6 prediction filter taps (SWREG45_VP7_VP8)	32	RW	0000_0000h
B8h	bi-dir initial ref pic list register (12-14) / VP7,VP8 quantization values (SWREG46_VP7_VP8)	32	RW	0000_0000h
BCh	bi-dir and P fwd initial ref pic list register (15 and P 0-3) / VP7,VP8 quantization values (SWREG47_VP7_VP8)	32	RW	0000_0000h

15.1.5.4.2 Decoder control register 1 (picture parameters) (SWREG4_JPEG_VP7_VP8)

15.1.5.4.2.1 Offset

Register	Offset
SWREG4_JPEG_VP7_VP8	10h

15.1.5.4.2.2 Diagram



15.1.5.4.2.3 Fields

Field	Function
31-23 SW_PIC_MB_WIDTH_JPEG_VP7_VP6	Picture width in macroblocks = ((width in pixels + 15) /16)
22-19	The amount of meaningfull horizontal pixels in last MB (width offset) 0 if exactly 16 pixels multiple picture and all the horizontal pixels in last MB are meaningfull

Table continues on the next page...

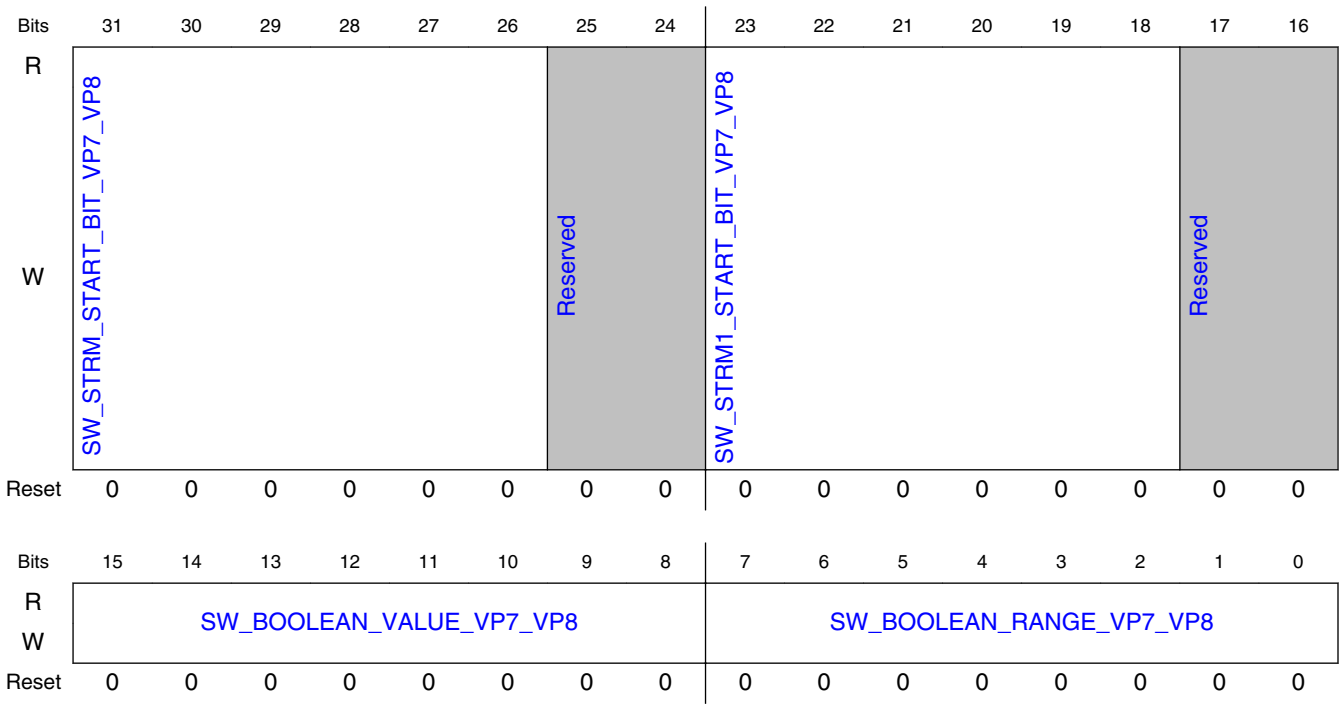
Field	Function
SW_MB_WIDTH_OFF_JPEG_VP7_VP6	
18-11 SW_PIC_MB_HEIGHT_P_JPEG_VP7_VP6	Picture height in macroblocks =((height in pixels+15)/16). Picture height is informed as size of the (progressive) frame also for single field (of interlaced content) is being decoded
10-7 SW_MB_HEIGHT_OFF_JPEG_VP7_VP6	The amount of meaningful vertical pixels in last MB (height offset 0 if exactly 16 pixels multiple picture and all the vertical pixels in last MB are meaningfull
6 SW_ALT_SCAN_E_JPEG_VP7_VP6	indicates alternative vertical scan method used for interlaced frames
5-3 SW_PIC_MB_W_EXT_JPEG_VP7_VP6	Picture mb width extension. If sw_pic_mb_width does not fit to 9 bits then these bits are used to increase the range upto 11 bits (used as 3 msb)
2-0 SW_PIC_MB_H_EXT_JPEG_VP7_VP6	Picture mb height extension. If sw_pic_mb_height_p does not fit to 9 bits then these bits are used to increase the range upto 11 bits (used as 3 msb). For 4k video one bit is used for extension (bit 0)

15.1.5.4.3 Decoder control register 2 (stream decoding table selects) (SWREG5_VP7_VP8)

15.1.5.4.3.1 Offset

Register	Offset
SWREG5_VP7_VP8	14h

15.1.5.4.3.2 Diagram



15.1.5.4.3.3 Fields

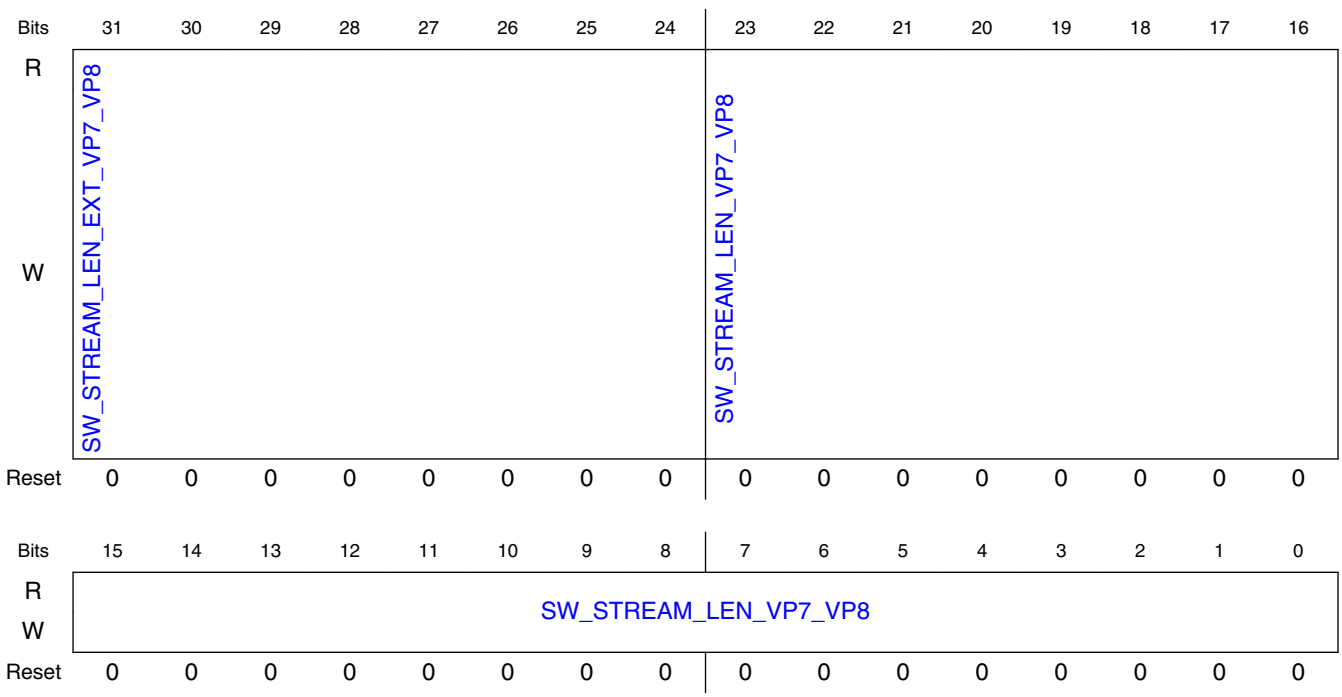
Field	Function
31-26 SW_STRM_START_BIT_VP7_VP8	Exact bit of stream start word where decoding can be started (assosiates with sw_rlc_vlc_base)
25-24 —	Reserved.
23-18 SW_STRM1_START_BIT_VP7_VP8	Start bit for ctrl-stream (needed if multistream is enabled, assosiates with sw_bitpl_ctrl_base)
17-16 —	Reserved.
15-8 SW_BOOLEAN_VALUE_VP7_VP8	Initial value for boolean dec
7-0 SW_BOOLEAN_RANGE_VP7_VP8	Initial range for boolean dec

15.1.5.4.4 Decoder control register 3 (stream buffer information) (SWREG6_VP7_VP8)

15.1.5.4.4.1 Offset

Register	Offset
SWREG6_VP7_VP8	18h

15.1.5.4.4.2 Diagram



15.1.5.4.4.3 Fields

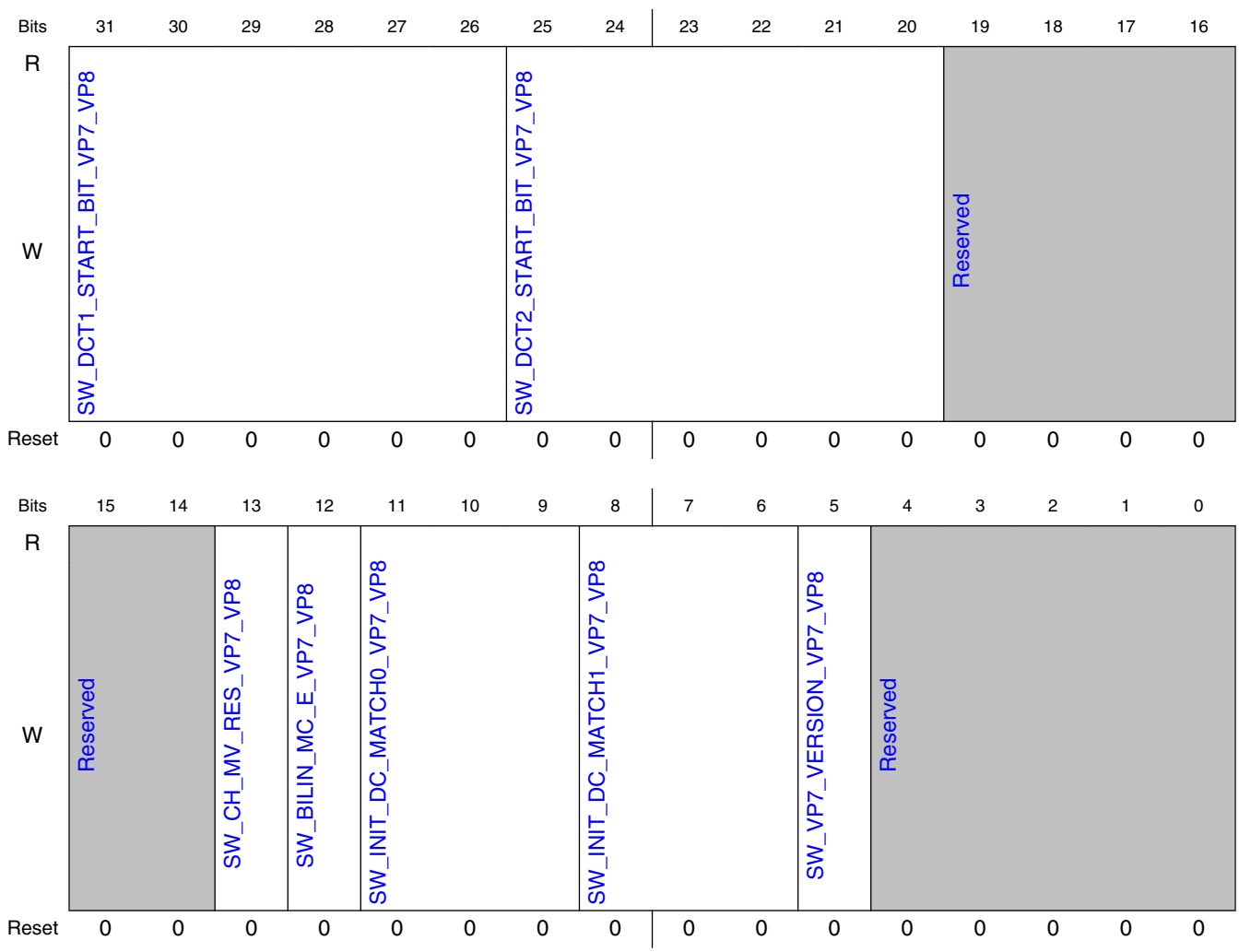
Field	Function
31-24 SW_STREAM_LEN_EXT_VP7_VP8	Extended stream length for WEBP/VP8
23-0 SW_STREAM_LEN_VP7_VP8	Amount of stream data bytes in input buffer. If the given buffer size is not enough for finishing the picture the corresponding interrupt is given and new stream buffer base address and stream buffer size information should be given (assosiates with sw_rlc_vlc_base). For VC-1/VP6 the buffer must include data for one picture/slice of the picture For H264/MPEG4/H263/MPEG2/MPEG1 the buffer must include at least data for one slice/VP of the picture For JPEG the buffer size must be a multiple of 256 bytes or the amount of data for one picture.

15.1.5.4.5 Decoder control register 4 (H264, VC-1, VP6 and progressive JPEG control) (SWREG7_VP7_VP8)

15.1.5.4.5.1 Offset

Register	Offset
SWREG7_VP7_VP8	1Ch

15.1.5.4.5.2 Diagram



15.1.5.4.5.3 Fields

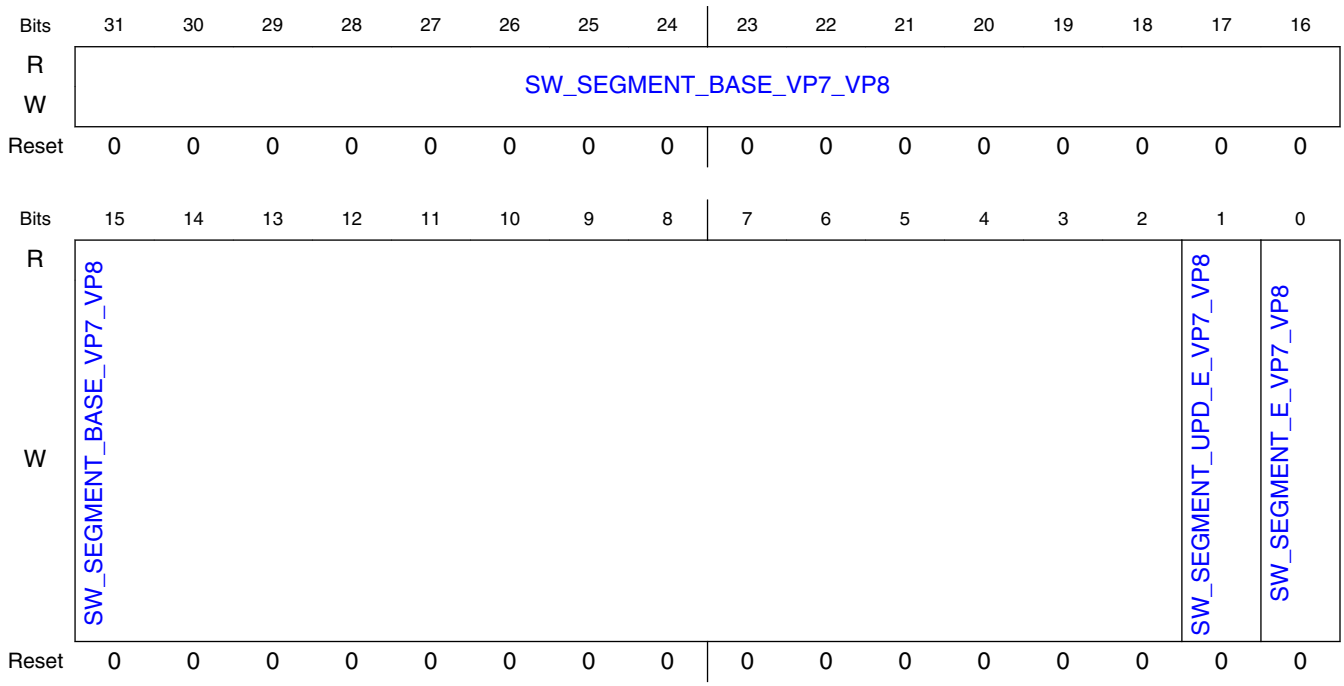
Field	Function
31-26 SW_DCT1_START_BIT_VP7_VP8	Start bit for VP7/VP8 DCT stream partition index 1
25-20 SW_DCT2_START_BIT_VP7_VP8	Start bit for VP7/VP8 DCT stream partition index 2
19-14 —	Reserved.
13 SW_CH_MV_RES_VP7_VP8	VP7/VP8 Chrominance motion vector resolution: 0b - Full pixel 1b - 1/8 pixel
12 SW_BILIN_MC_E_VP7_VP8	Bilinear motion compensation enable: 0b - Bicubic interpolation used 1b - Bilinear interpolation used
11-9 SW_INIT_DC_MATCH0_VP7_VP8	Initial DC prediction mach count 0. After HW has decoded a picture HW returns the final match count0 information which is read by SW
8-6 SW_INIT_DC_MATCH1_VP7_VP8	Initial DC prediction mach count 1. After HW has decoded a picture HW returns the final match count1 information which is read by SW
5 SW_VP7_VERSION_VP7_VP8	VP7 version information to streamd: 0b - VP7 version 7.0 1b - VP7 version 7.1 or better
4-0 —	Reserved.

15.1.5.4.6 Base address for differential motion vector base address (RLC-mode) /H264 P initial fwd ref pic list register (4-9)/ VC-1 intensity control 1/ VP7 and VP8 segmentation base register (SWREG10_VP7_VP8)

15.1.5.4.6.1 Offset

Register	Offset
SWREG10_VP7_VP8	28h

15.1.5.4.6.2 Diagram



15.1.5.4.6.3 Fields

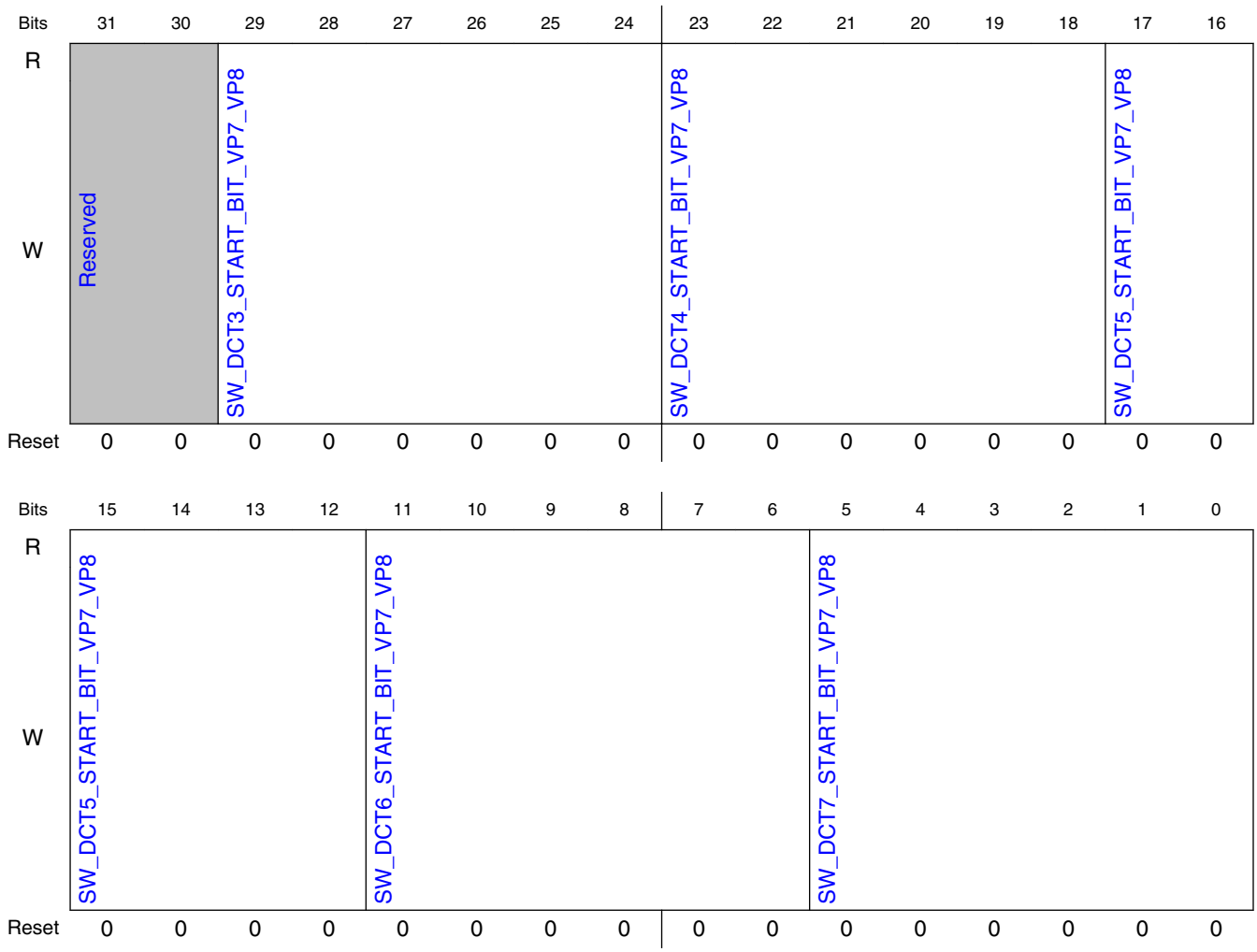
Field	Function
31-2 SW_SEGMENT_BASE_VP7_VP8	VP7/VP8: base address for segmentation map values
1 SW_SEGMENT_UPD_E_VP7_VP8	VP7/VP8 Segmentation map update enable: '0': segmentation values are read from external memory (from segment_base) '1': segmentation update is included in stream
0 SW_SEGMENT_E_VP7_VP8	Segmentation enable: '0': segmentation is not enabled '1': segmentation is enabled (sw_segment_upd_e value is used)

15.1.5.4.7 Decoder control register 7 (VLC) / base address for H.264 intra prediction 4x4 / base address for MPEG-4 DC component (RLC) / H264 P initial fwd ref pic list register (10-15) / VC-1 intensity control 2 (SWREG11_VP7_VP8)

15.1.5.4.7.1 Offset

Register	Offset
SWREG11_VP7_VP8	2Ch

15.1.5.4.7.2 Diagram



15.1.5.4.7.3 Fields

Field	Function
31-30 —	Reserved.
29-24	Start bit for VP7/VP8 DCT stream partition index 3

Table continues on the next page...

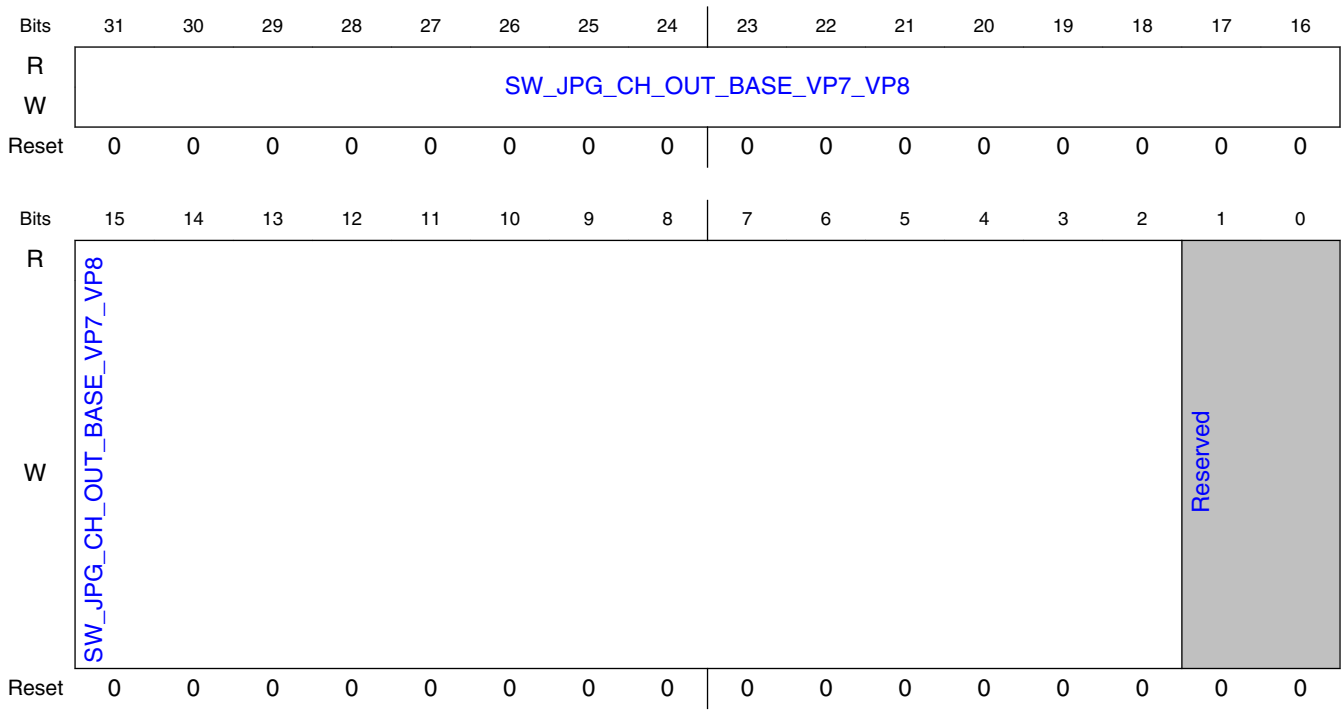
Field	Function
SW_DCT3_START_BIT_VP7_VP8	
23-18 SW_DCT4_START_BIT_VP7_VP8	Start bit for VP7/VP8 DCT stream partition index 4
17-12 SW_DCT5_START_BIT_VP7_VP8	Start bit for VP7/VP8 DCT stream partition index 5
11-6 SW_DCT6_START_BIT_VP7_VP8	Start bit for VP7/VP8 DCT stream partition index 6
5-0 SW_DCT7_START_BIT_VP7_VP8	Start bit for VP7/VP8 DCT stream partition index 7

15.1.5.4.8 Base address for reference picture index 0 / base address for JPEG decoder output chrominance picture (SWREG14_VP7_VP8)

15.1.5.4.8.1 Offset

Register	Offset
SWREG14_VP7_VP8	38h

15.1.5.4.8.2 Diagram



15.1.5.4.8.3 Fields

Field	Function
31-2 SW_JPG_CH_OUT_BASE_VP7_VP8	Base address for decoder output chrominance picture. Used in JPEG and web-p picture mode (not needed if decoder output is not written)
1-0 —	Reserved.

15.1.5.4.9 Base address for reference picture index 1 / JPEG control (SWREG15_VP7_VP8)

15.1.5.4.9.1 Offset

Register	Offset
SWREG15_VP7_VP8	3Ch

15.1.5.4.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								SW_JPEG_SLICE_H_VP7_VP8							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.1.5.4.9.3 Fields

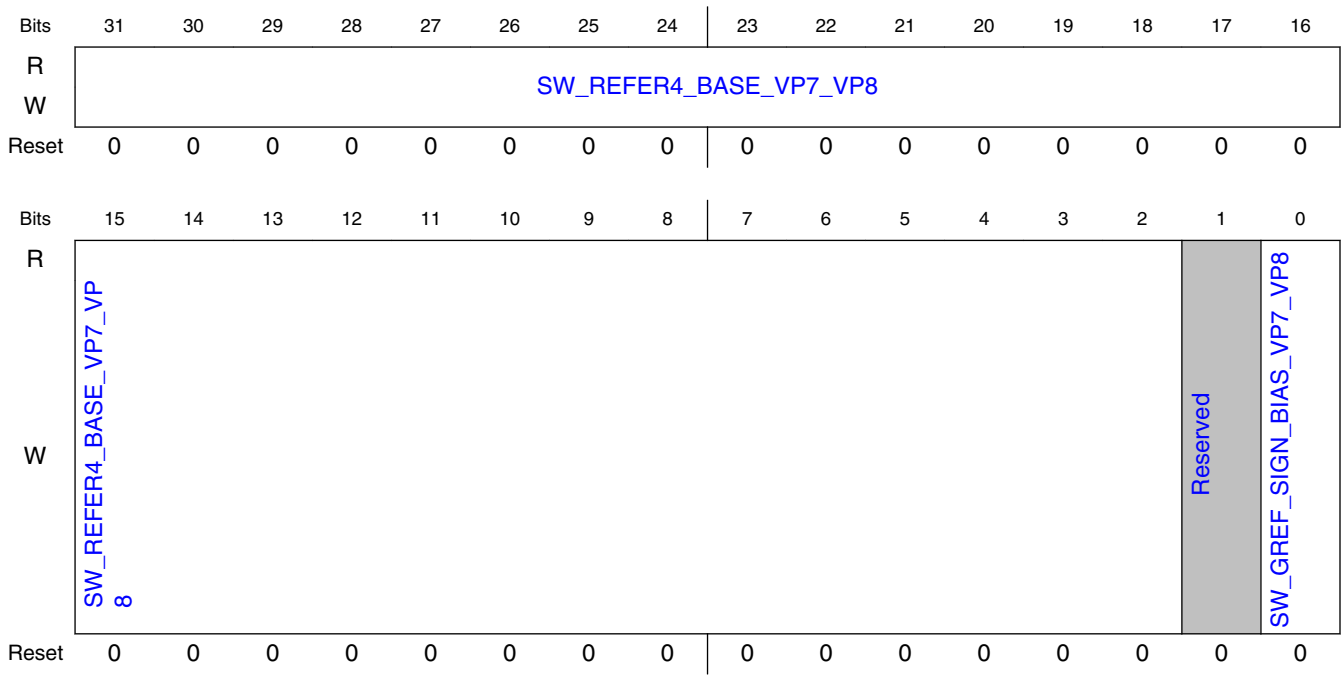
Field	Function
31-8 —	Reserved.
7-0 SW_JPEG_SLICE_H_VP7_VP8	JPEG/Web-p. Height of the slice (multiple of 16 pixels) that HW decodes before interrupt. When slice is decoded HW will rise an interrupt and reset external addresses back to base address. Note, value 0 disables slice mode. Slice mode must be used if picture size is more than 16 Mpixels. However for bigger than 4096 MBs the slice mode usage is recommended.

15.1.5.4.10 Base address for reference picture index 4 / VC1 control / MPEG4 MVD control/ List of VLC code lengths in first JPEG AC table / VC-1 intensity control 4 / VP6/VP7, VP8 Golden refer picture base (SWREG18_VP7_VP8)

15.1.5.4.10.1 Offset

Register	Offset
SWREG18_VP7_VP8	48h

15.1.5.4.10.2 Diagram



15.1.5.4.10.3 Fields

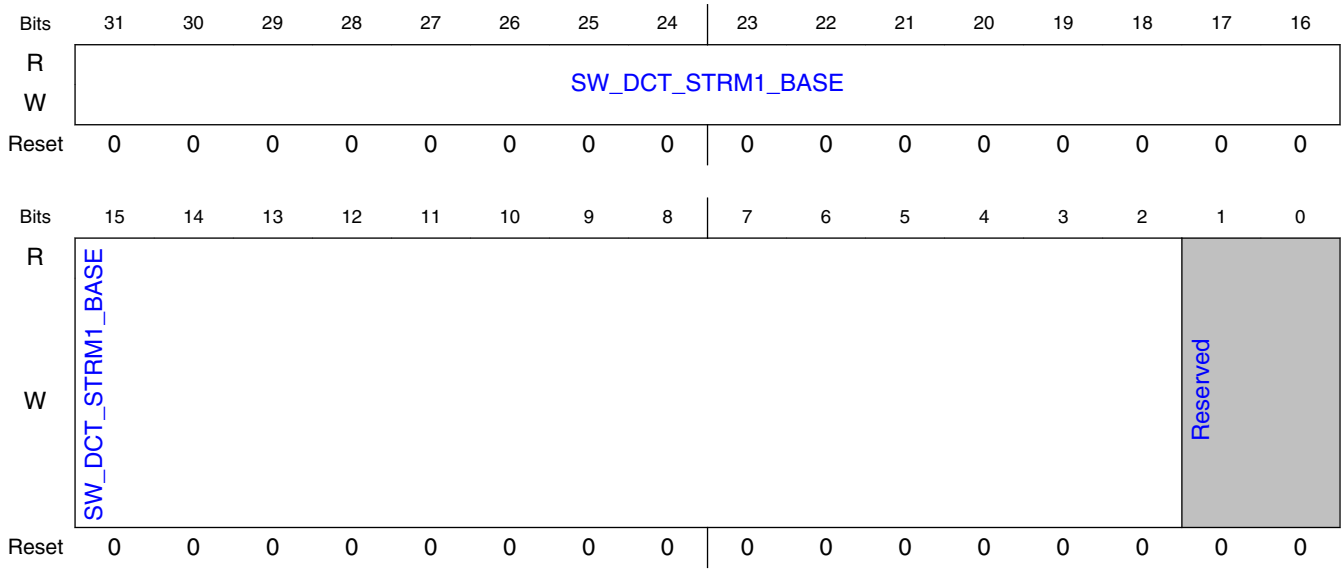
Field	Function
31-2 SW_REFER4_BASE_VP7_VP8	H264: Base address for reference picture index 4 VP6/VP7/VP8: Base address for Golden reference picture (corresponds picid 4)
1 —	Reserved.
0 SW_GREF_SIGN_BIAS_VP7_VP8	Reference picture sign bias for Golden reference frame

15.1.5.4.11 Base address for reference picture index 8 / List of VLC code lengths in second JPEG AC table / VP6 scan maps / VP7,VP8 DCT stream 1 base (SWREG22_VP7_VP8)

15.1.5.4.11.1 Offset

Register	Offset
SWREG22_VP7_VP8	58h

15.1.5.4.11.2 Diagram



15.1.5.4.11.3 Fields

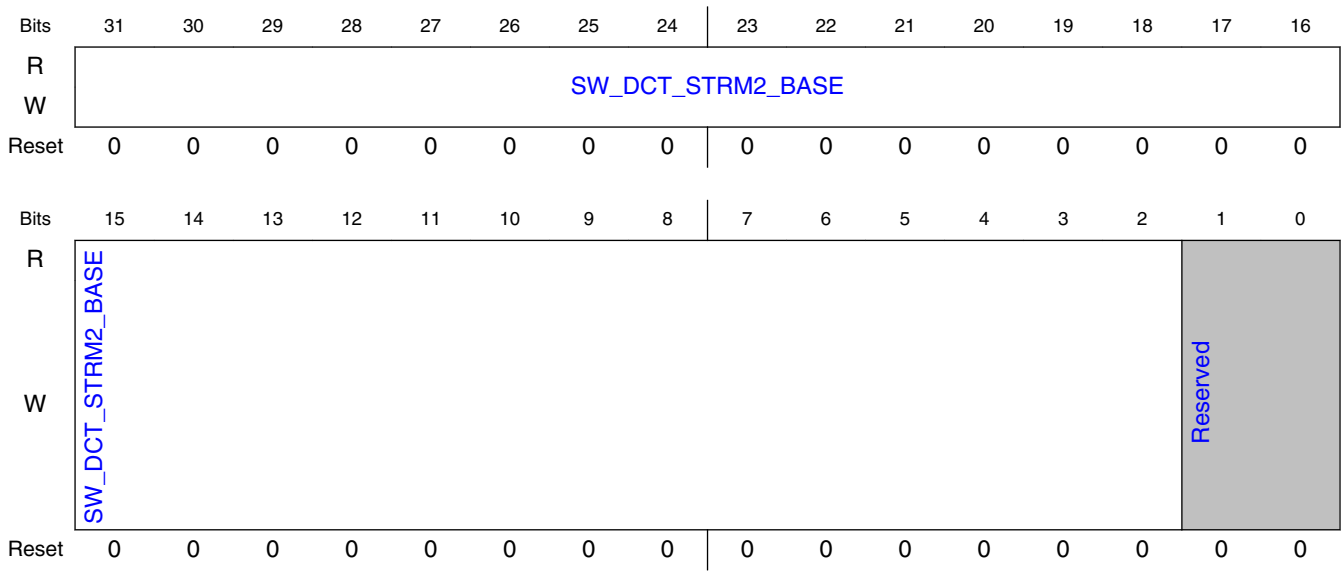
Field	Function
31-2 SW_DCT_STRM1_BASE	Base address for VP7/VP8 DCT stream MB row 1,2n+1
1-0 —	Reserved.

15.1.5.4.12 Base address for reference picture index 9 / List of VLC code lengths in first JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 2 base (SWREG23_VP7_VP8)

15.1.5.4.12.1 Offset

Register	Offset
SWREG23_VP7_VP8	5Ch

15.1.5.4.12.2 Diagram



15.1.5.4.12.3 Fields

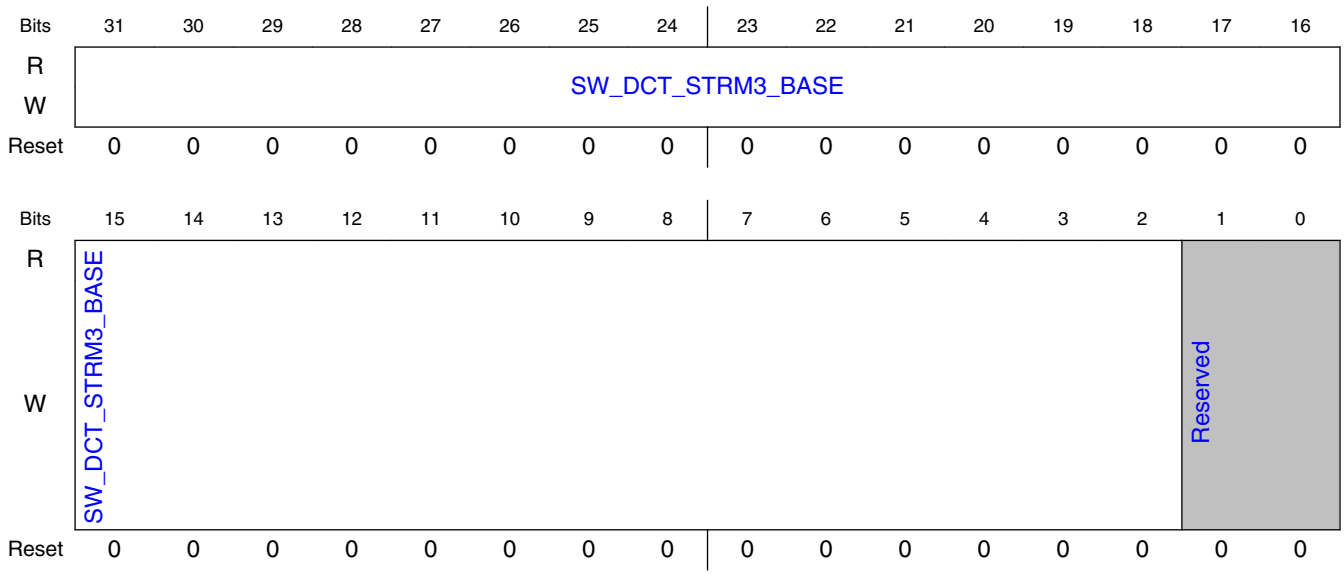
Field	Function
31-2 SW_DCT_STRM2_BASE	Base address for VP7/VP8 DCT stream MB row 2,2n+2
1-0 —	Reserved.

15.1.5.4.13 Base address for reference picture index 10 / List of VLC code lengths in first JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 3 base (SWREG24_VP7_VP8)

15.1.5.4.13.1 Offset

Register	Offset
SWREG24_VP7_VP8	60h

15.1.5.4.13.2 Diagram



15.1.5.4.13.3 Fields

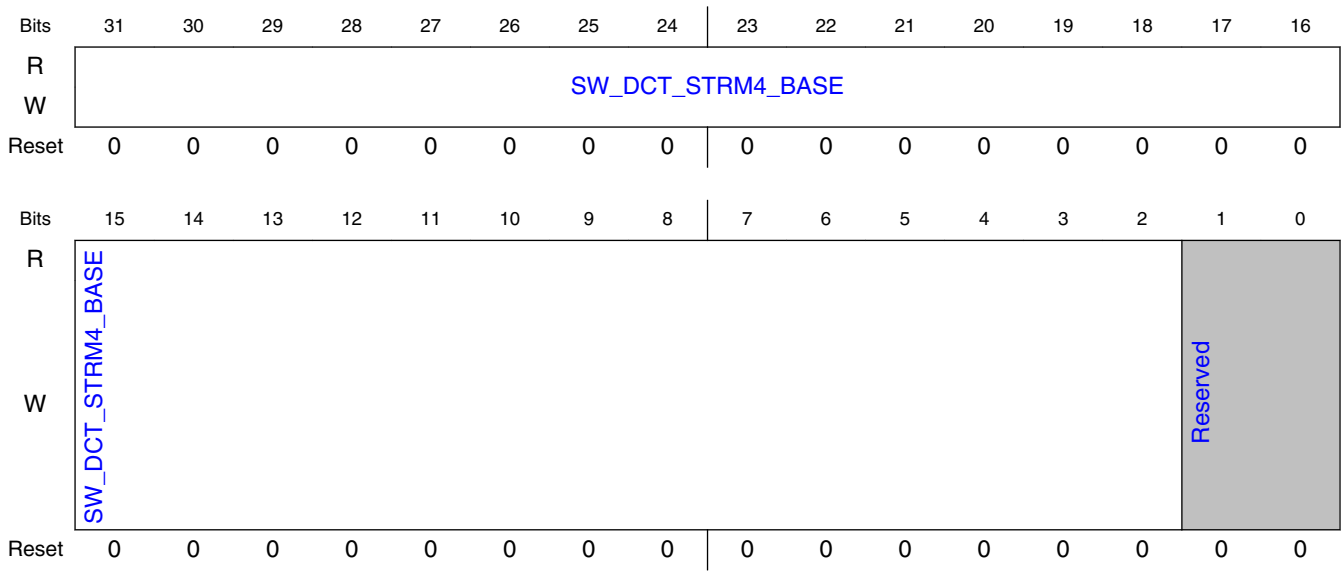
Field	Function
31-2 SW_DCT_STRM3_BASE	Base address for VP7/VP8 DCT stream MB row 3,2n+3
1-0 —	Reserved.

15.1.5.4.14 Base address for reference picture index 11 / List of VLC code lengths in second JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 4 base (SWREG25_VP7_VP8)

15.1.5.4.14.1 Offset

Register	Offset
SWREG25_VP7_VP8	64h

15.1.5.4.14.2 Diagram



15.1.5.4.14.3 Fields

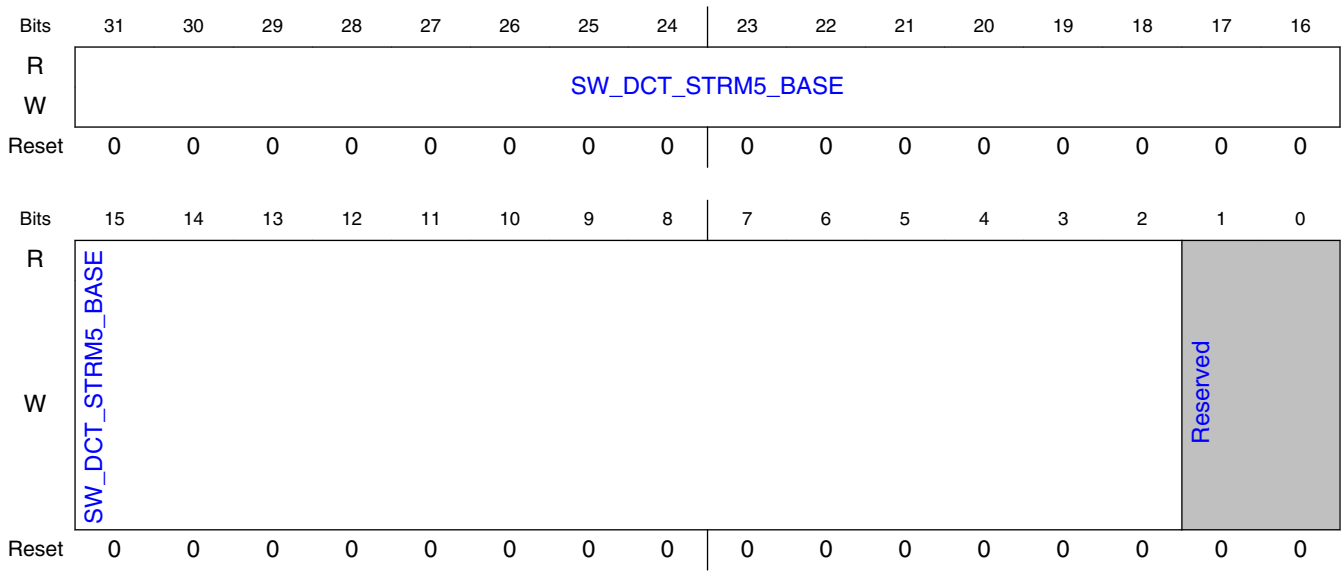
Field	Function
31-2 SW_DCT_STRM4_BASE	Base address for VP7/VP8 DCT stream MB row 4,2n+4
1-0 —	Reserved.

15.1.5.4.15 Base address for reference picture index 12 / List of VLC code lengths in second JPEG DC table / VP6 scan maps / VP7,VP8 DCT stream 5 base (SWREG26_VP7_VP8)

15.1.5.4.15.1 Offset

Register	Offset
SWREG26_VP7_VP8	68h

15.1.5.4.15.2 Diagram



15.1.5.4.15.3 Fields

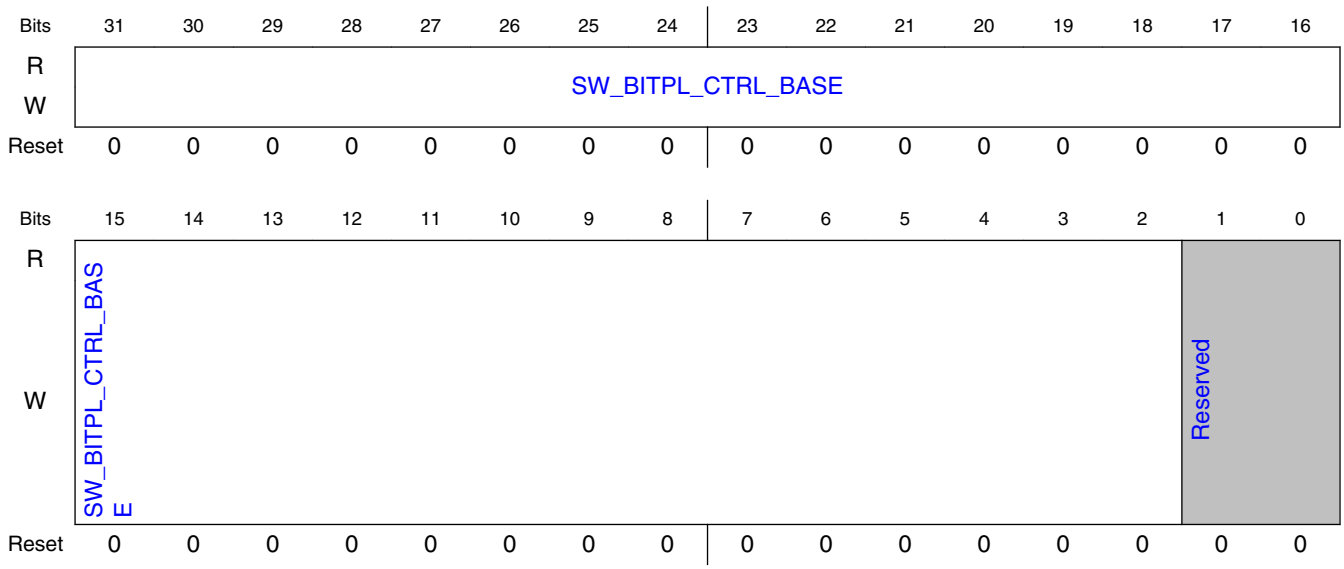
Field	Function
31-2 SW_DCT_STRM5_BASE	Base address for VP7/VP8 DCT stream MB row 5,2n+5
1-0 —	Reserved.

15.1.5.4.16 Base address for reference picture index 13 / VC-1 bitpl mbctrl or VP6,VP7,VP8 ctrl stream base /Progressive JPEG DC table (SWREG27_VC1)

15.1.5.4.16.1 Offset

Register	Offset
SWREG27_VC1	6Ch

15.1.5.4.16.2 Diagram



15.1.5.4.16.3 Fields

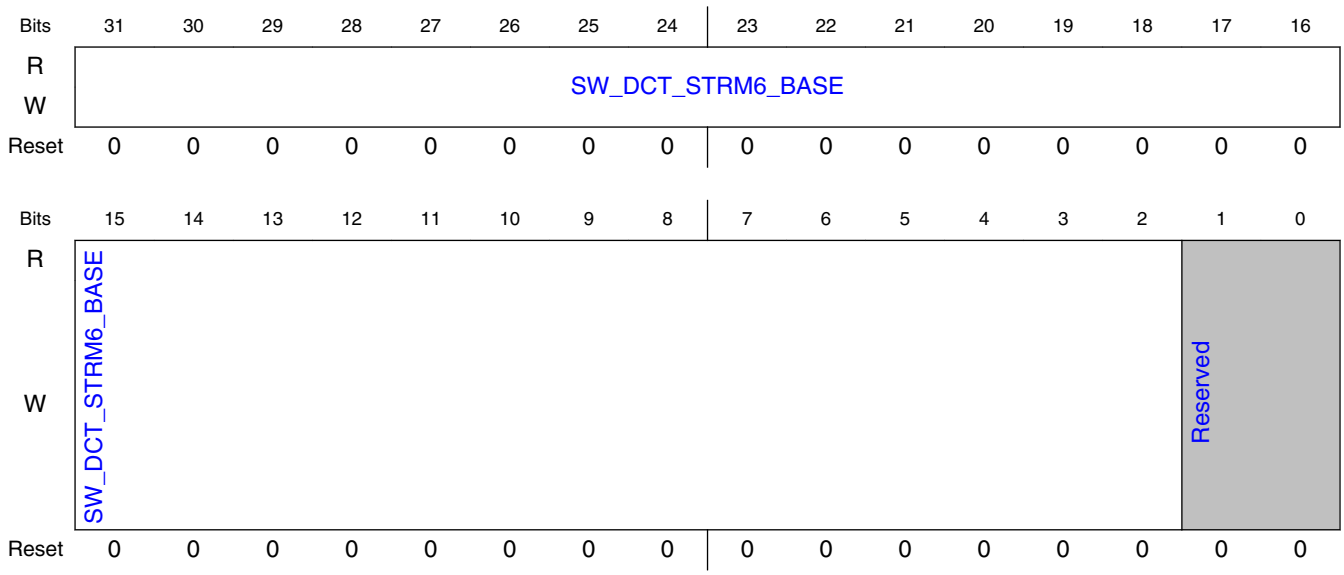
Field	Function
31-2 SW_BITPL_CTRL_BASE	VC-1: Base address for bitplane mb control VP6/VP7/VP8 : Base address for ctrl data stream. Used if multistream is enabled
1-0 —	Reserved.

15.1.5.4.17 Base address for reference picture index 14 / VP6 scan maps / Progressive JPEG DC table / VP7,VP8 DCT stream 6 base (SWREG28_VP7_VP8)

15.1.5.4.17.1 Offset

Register	Offset
SWREG28_VP7_VP8	70h

15.1.5.4.17.2 Diagram



15.1.5.4.17.3 Fields

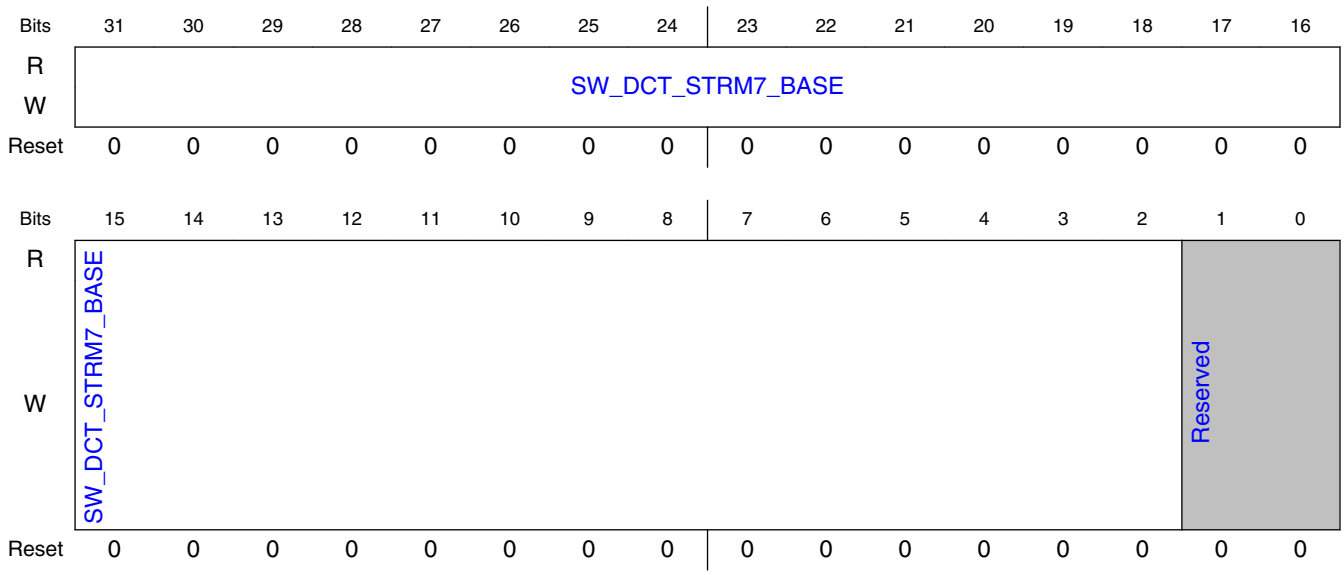
Field	Function
31-2 SW_DCT_STRM6_BASE	Base address for VP7/VP8 DCT stream MB row 6,2n+6
1-0 —	Reserved.

15.1.5.4.18 Base address for reference picture index 15 / VP6 scan maps / VP7,VP8 DCT stream 7 base (SWREG29_VP7_VP8)

15.1.5.4.18.1 Offset

Register	Offset
SWREG29_VP7_VP8	74h

15.1.5.4.18.2 Diagram



15.1.5.4.18.3 Fields

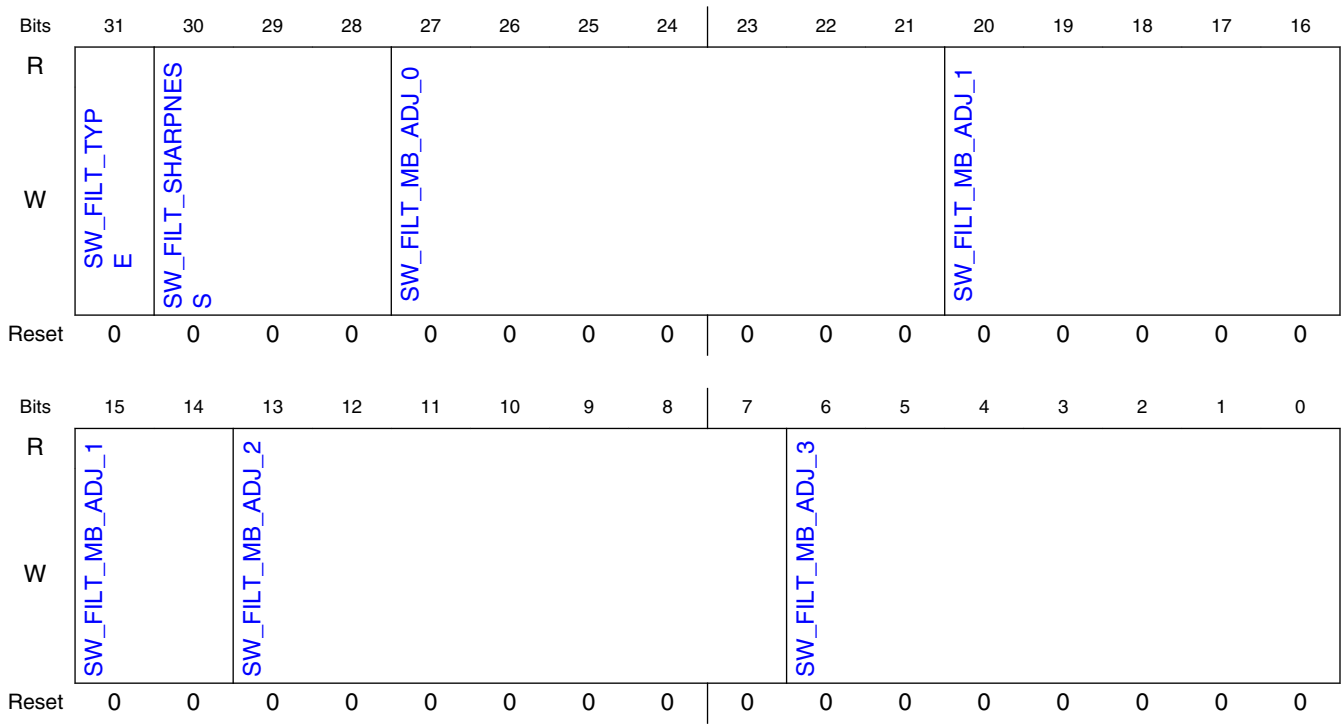
Field	Function
31-2 SW_DCT_STRM7_BASE	Base address for VP7/VP8 DCT stream MB row 7,2n+7
1-0 —	Reserved.

15.1.5.4.19 Reference picture numbers for index 0 and 1 (H264 VLC) / VP6 scan maps / VP7,VP8 loop filter mb level adjusts (SWREG30_VP7_VP8)

15.1.5.4.19.1 Offset

Register	Offset
SWREG30_VP7_VP8	78h

15.1.5.4.19.2 Diagram



15.1.5.4.19.3 Fields

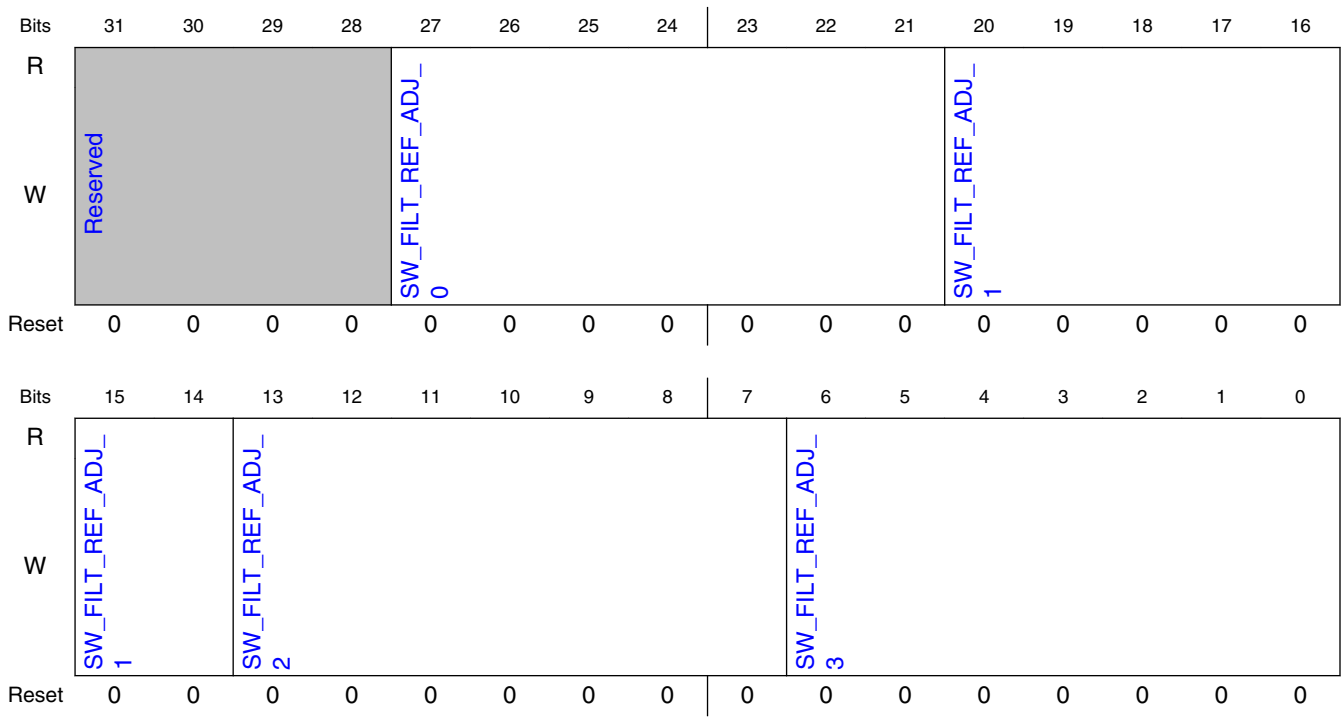
Field	Function
31 SW_FILT_TYPE	VP7/VP8 loop filter type
30-28 SW_FILT_SHARPNESS	VP7/VP8 loop filter sharpness
27-21 SW_FILT_MB_ADJ_0	VP7/VP8 filter level adjustment for MB type 0
20-14 SW_FILT_MB_ADJ_1	VP7/VP8 filter level adjustment for MB type 1
13-7 SW_FILT_MB_ADJ_2	VP7/VP8 filter level adjustment for MB type 2
6-0 SW_FILT_MB_ADJ_3	VP7/VP8 filter level adjustment for MB type 3

15.1.5.4.20 Reference picture numbers for index 2 and 3 (H264 VLC) / VP6 scan maps / VP7,VP8 loop filter ref pic level adjusts (SWREG31_VP7_VP8)

15.1.5.4.20.1 Offset

Register	Offset
SWREG31_VP7_VP8	7Ch

15.1.5.4.20.2 Diagram



15.1.5.4.20.3 Fields

Field	Function
31-28 —	Reserved.
27-21 SW_FILT_REF_ADJ_0	VP7/VP8 filter level adjustment for reference frame type 0
20-14	VP7/VP8 filter level adjustment for reference frame type 1

Table continues on the next page...

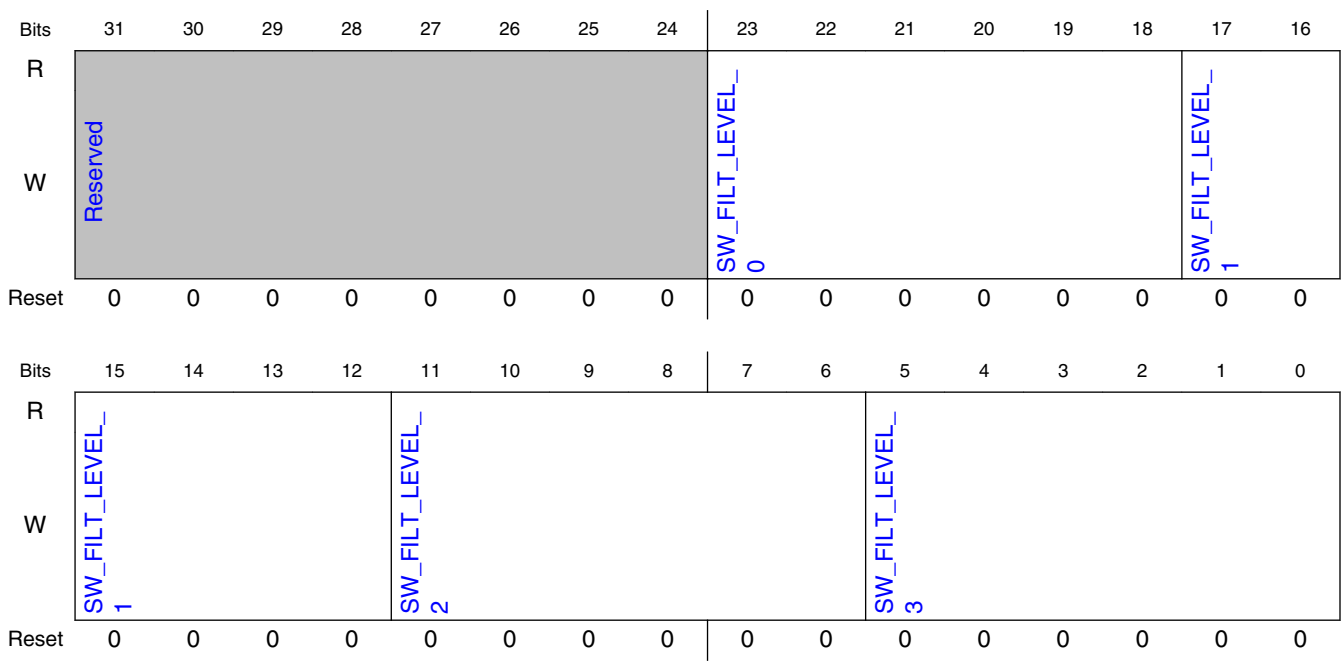
Field	Function
SW_FILT_REF_ADJ_1	
13-7 SW_FILT_REF_ADJ_2	VP7/VP8 filter level adjustment for reference frame type 2
6-0 SW_FILT_REF_ADJ_3	VP7/VP8 filter level adjustment for reference frame type 3

15.1.5.4.21 Reference picture numbers for index 4 and 5 (H264 VLC) / VP6 scan maps / VP7,VP8 loop filter levels (SWREG32_VP7_VP8)

15.1.5.4.21.1 Offset

Register	Offset
SWREG32_VP7_VP8	80h

15.1.5.4.21.2 Diagram



15.1.5.4.21.3 Fields

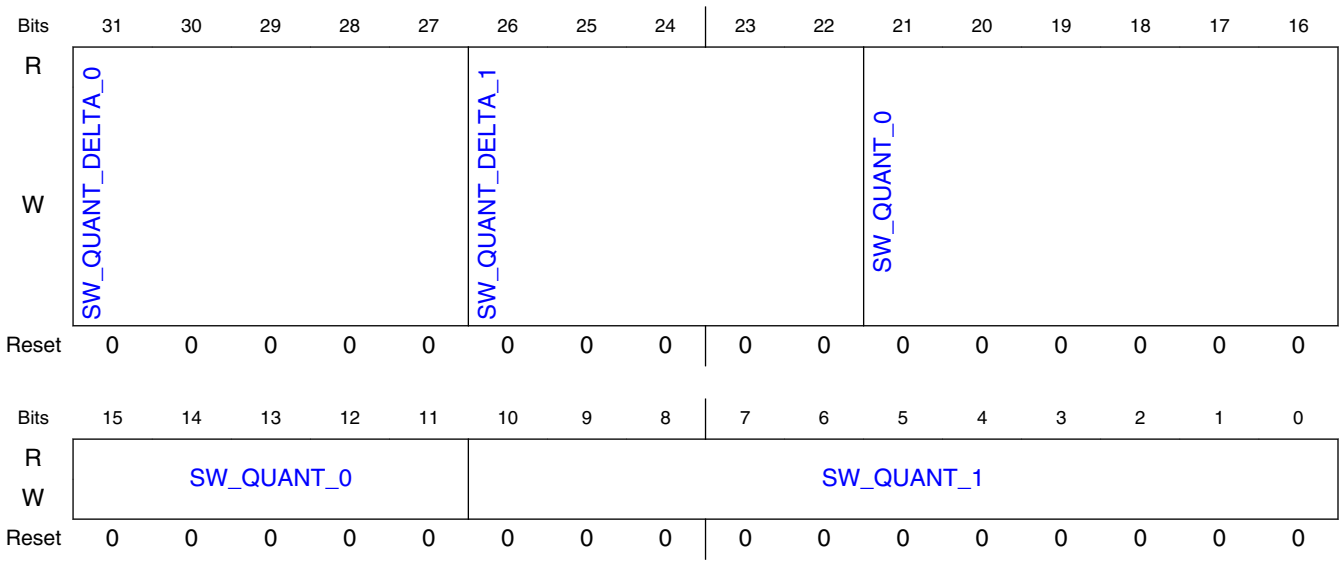
Field	Function
31-24 —	Reserved.
23-18 SW_FILT_LEVEL_0	VP7/VP8 filter level value for reference frame type 0
17-12 SW_FILT_LEVEL_1	VP7/VP8 filter level value for reference frame type 1
11-6 SW_FILT_LEVEL_2	VP7/VP8 filter level value for reference frame type 2
5-0 SW_FILT_LEVEL_3	VP7/VP8 filter level value for reference frame type 3

15.1.5.4.22 Reference picture numbers for index 6 and 7 (H264 VLC) / VP6 scan maps / VP7,VP8 quantization values (SWREG33_VP7_VP8)

15.1.5.4.22.1 Offset

Register	Offset
SWREG33_VP7_VP8	84h

15.1.5.4.22.2 Diagram



15.1.5.4.22.3 Fields

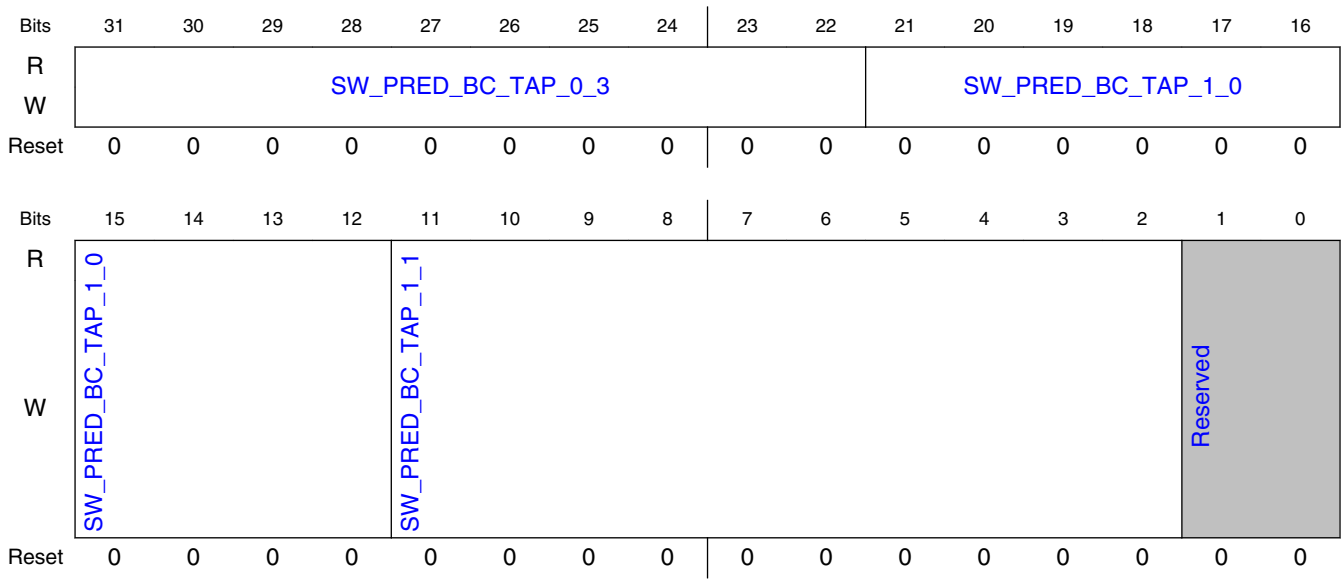
Field	Function
31-27 SW_QUANT_DELTA_0	VP8 quantisizer delta 0
26-22 SW_QUANT_DELTA_1	VP8 quantisizer delta 1
21-11 SW_QUANT_0	VP7: QP (11 bit) VP8: quantisizer value for LUT (7 bit)
10-0 SW_QUANT_1	VP7: QP (11 bit) VP8: quantisizer value for LUT (7 bit)

15.1.5.4.23 Reference picture numbers for index 8 and 9 (H264 VLC) / MPEG4, VC1, VPx prediction filter taps (SWREG34_H263)

15.1.5.4.23.1 Offset

Register	Offset
SWREG34_H263	88h

15.1.5.4.23.2 Diagram



15.1.5.4.23.3 Fields

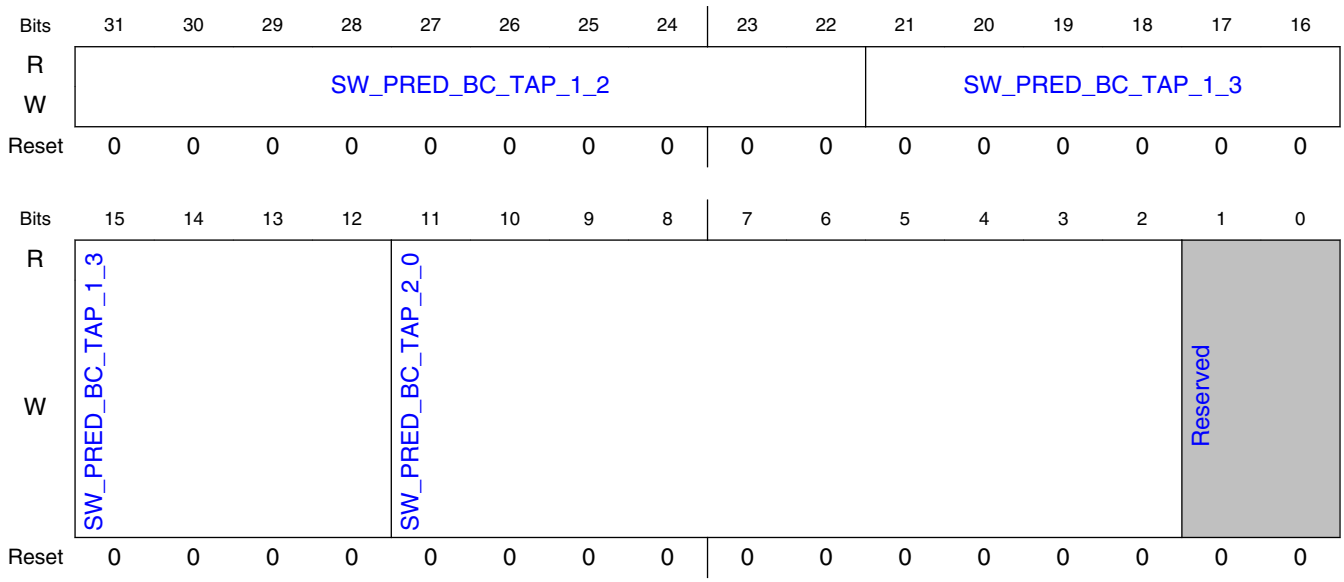
Field	Function
31-22 SW_PRED_BC_TAP_0_3	Prediction filter set 0, tap 3
21-12 SW_PRED_BC_TAP_1_0	Prediction filter set 1, tap 0
11-2 SW_PRED_BC_TAP_1_1	Prediction filter set 1, tap 1
1-0 —	Reserved.

15.1.5.4.24 Reference picture numbers for index 10 and 11 (H264 VLC) / VC1, VPx prediction filter taps (SWREG35_VC1)

15.1.5.4.24.1 Offset

Register	Offset
SWREG35_VC1	8Ch

15.1.5.4.24.2 Diagram



15.1.5.4.24.3 Fields

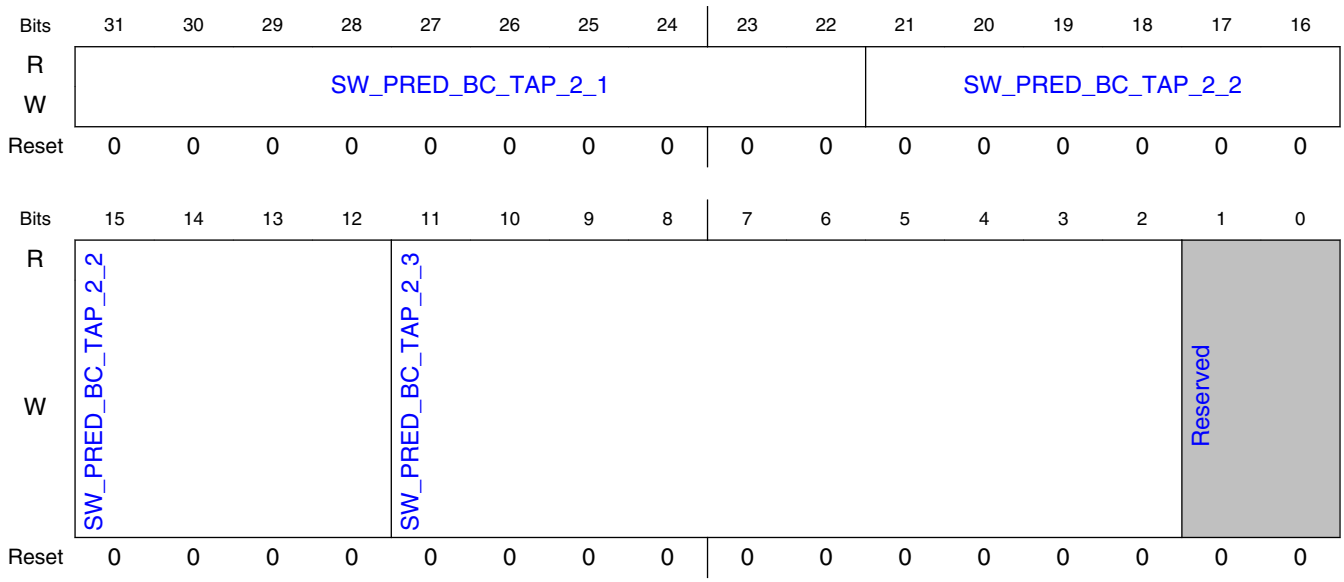
Field	Function
31-22 SW_PRED_BC_TAP_1_2	Prediction filter set 1, tap 2
21-12 SW_PRED_BC_TAP_1_3	Prediction filter set 1, tap 3
11-2 SW_PRED_BC_TAP_2_0	Prediction filter set 2, tap 0
1-0 —	Reserved.

15.1.5.4.25 Reference picture numbers for index 12 and 13 (H264 VLC) / VC1, VPx prediction filter taps (SWREG36_VC1)

15.1.5.4.25.1 Offset

Register	Offset
SWREG36_VC1	90h

15.1.5.4.25.2 Diagram



15.1.5.4.25.3 Fields

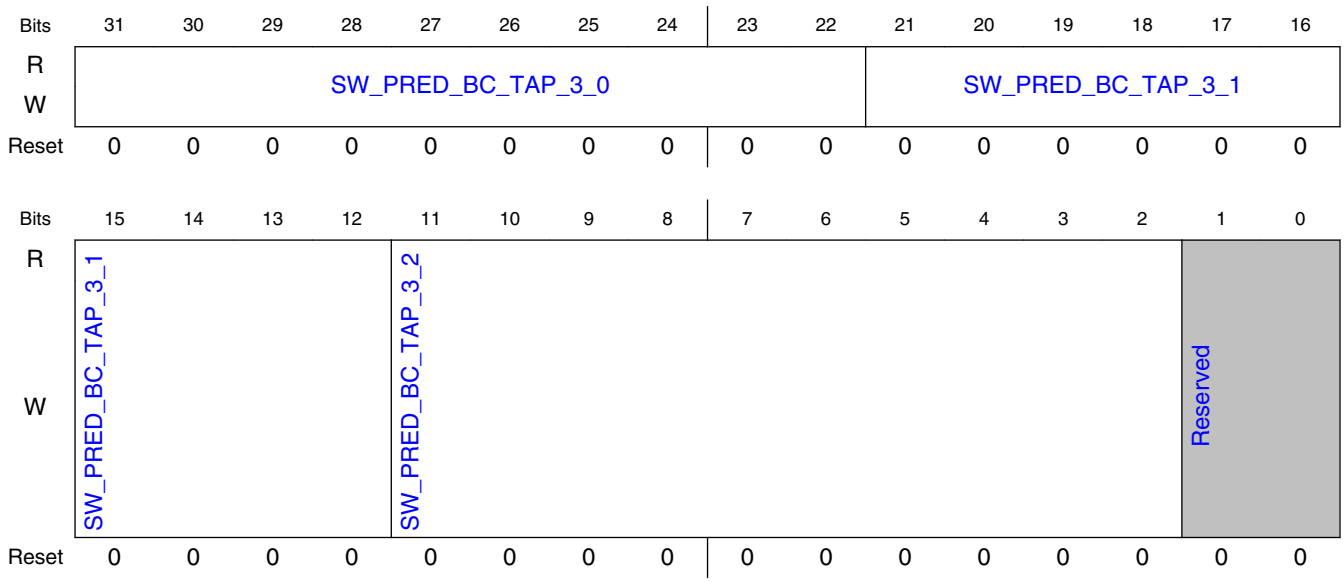
Field	Function
31-22 SW_PRED_BC_TAP_2_1	Prediction filter set 2, tap 1
21-12 SW_PRED_BC_TAP_2_2	Prediction filter set 2, tap 2
11-2 SW_PRED_BC_TAP_2_3	Prediction filter set 2, tap 3
1-0 —	Reserved.

15.1.5.4.26 Reference picture numbers for index 14 and 15 (H264 VLC) / VPx prediction filter taps (SWREG37_VP6_VP7_VP8)

15.1.5.4.26.1 Offset

Register	Offset
SWREG37_VP6_VP7_VP8	94h

15.1.5.4.26.2 Diagram



15.1.5.4.26.3 Fields

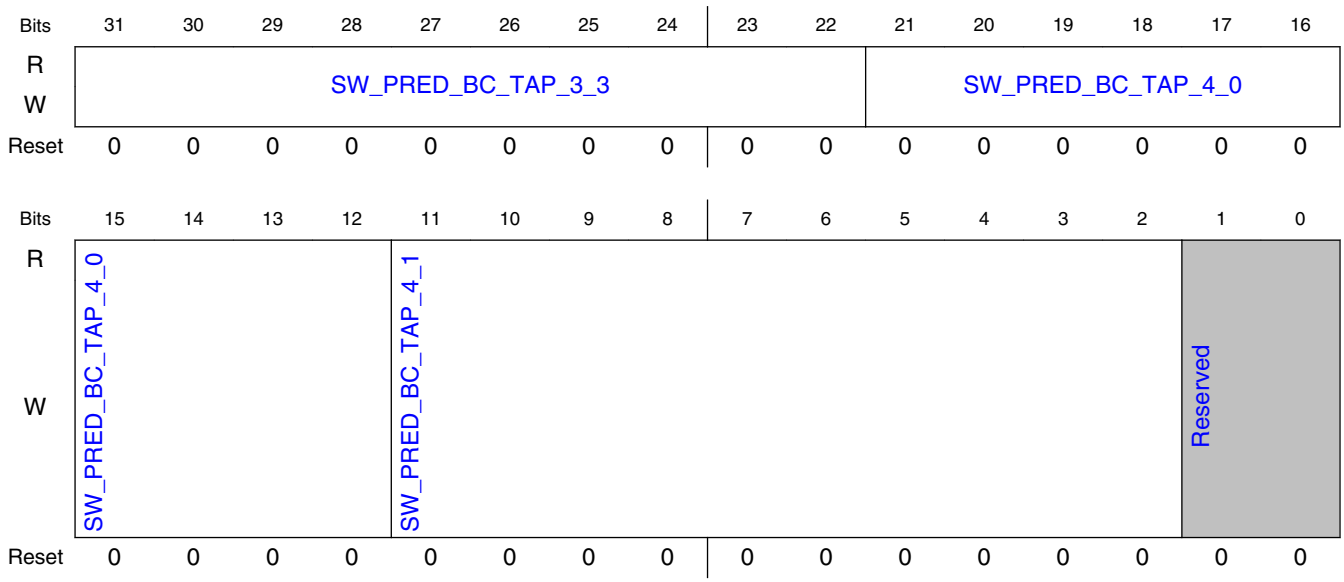
Field	Function
31-22 SW_PRED_BC_TAP_3_0	Prediction filter set 3, tap 0
21-12 SW_PRED_BC_TAP_3_1	Prediction filter set 3, tap 1
11-2 SW_PRED_BC_TAP_3_2	Prediction filter set 3, tap 2
1-0 —	Reserved.

15.1.5.4.27 Reference picture long term flags (H264 VLC) / VPx prediction filter taps (SWREG38_VP6_VP7_VP8)

15.1.5.4.27.1 Offset

Register	Offset
SWREG38_VP6_VP7_VP8	98h

15.1.5.4.27.2 Diagram



15.1.5.4.27.3 Fields

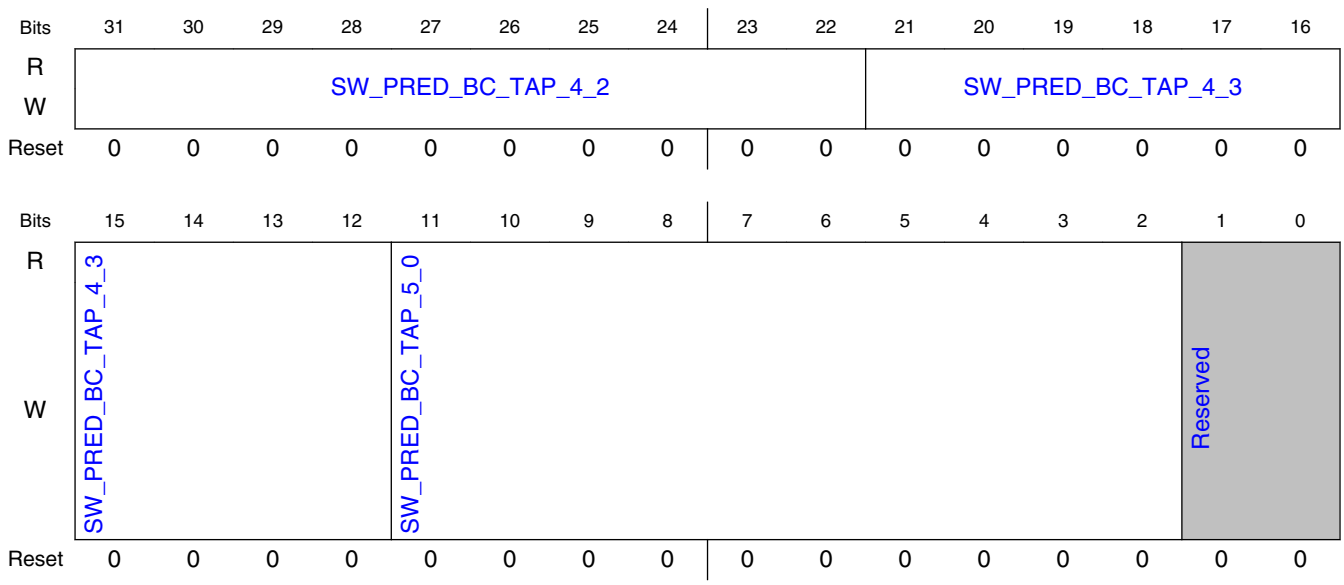
Field	Function
31-22 SW_PRED_BC_TAP_3_3	Prediction filter set 3, tap 3
21-12 SW_PRED_BC_TAP_4_0	Prediction filter set 4, tap 0
11-2 SW_PRED_BC_TAP_4_1	Prediction filter set 4, tap 1
1-0 —	Reserved.

15.1.5.4.28 Reference picture valid flags (H264 VLC) / VPx prediction filter taps (SWREG39_VP6_VP7_VP8)

15.1.5.4.28.1 Offset

Register	Offset
SWREG39_VP6_VP7_VP8	9Ch

15.1.5.4.28.2 Diagram



15.1.5.4.28.3 Fields

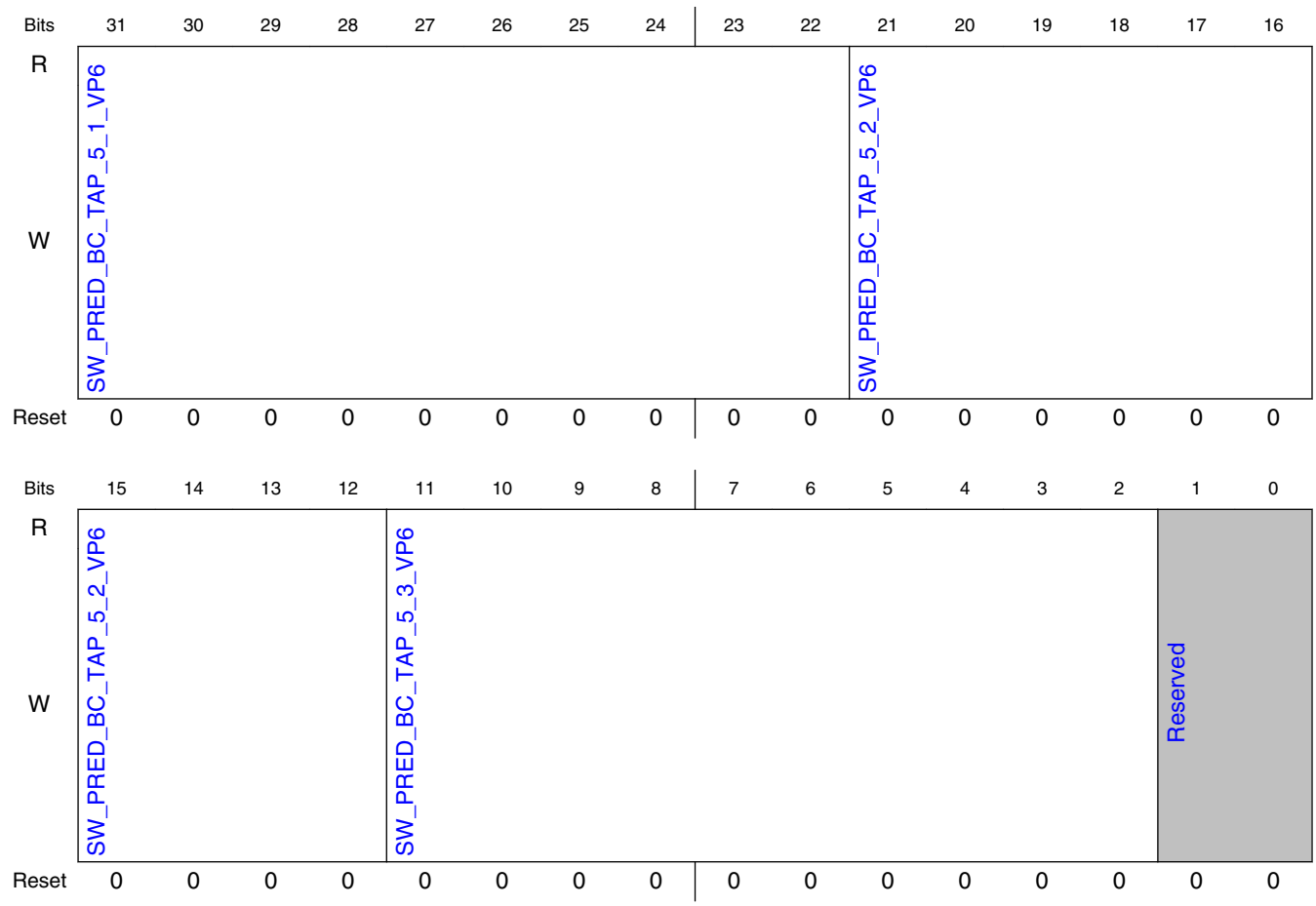
Field	Function
31-22 SW_PRED_BC_TAP_4_2	Prediction filter set 4, tap 2
21-12 SW_PRED_BC_TAP_4_3	Prediction filter set 4, tap 3
11-2 SW_PRED_BC_TAP_5_0	Prediction filter set 5, tap 0
1-0 —	Reserved.

15.1.5.4.29 bi_dir initial ref pic list register (0-2) / VP6 prediction filter taps / Progressive JPEG Cb ACDC coefficient base (SWREG42_VP6)

15.1.5.4.29.1 Offset

Register	Offset
SWREG42_VP6	A8h

15.1.5.4.29.2 Diagram



15.1.5.4.29.3 Fields

Field	Function
31-22	Prediction filter set 5, tap 1

Table continues on the next page...

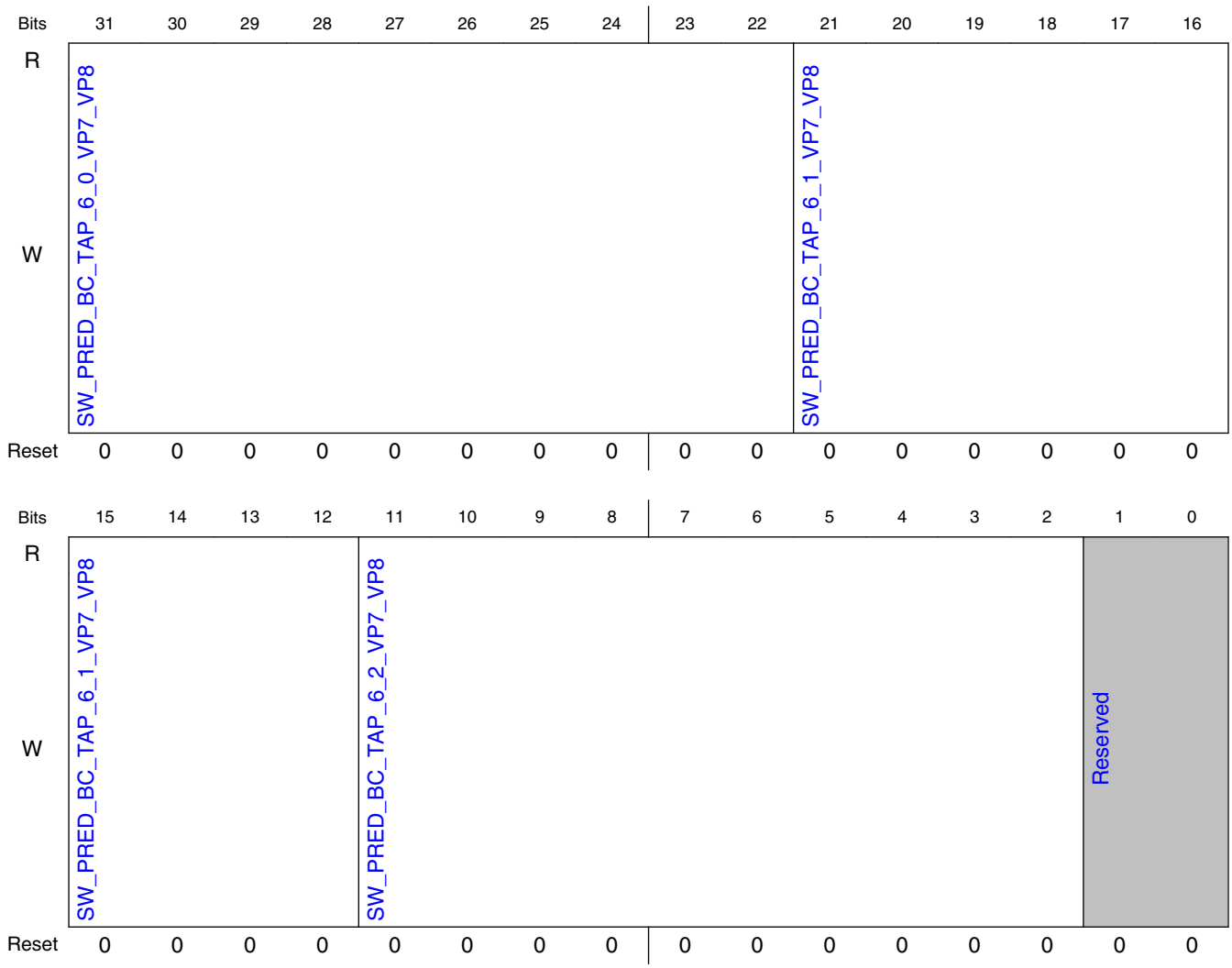
Field	Function
SW_PRED_BC_TAP_5_1_VP6	
21-12 SW_PRED_BC_TAP_5_2_VP6	Prediction filter set 5, tap 2
11-2 SW_PRED_BC_TAP_5_3_VP6	Prediction filter set 5, tap 3
1-0 —	Reserved.

15.1.5.4.30 bi-dir initial ref pic list register (3-5) / VP6 prediction filter taps / Progressive JPEG Cr ACDC coefficient base (SWREG43_VP7_VP8)

15.1.5.4.30.1 Offset

Register	Offset
SWREG43_VP7_VP8	ACh

15.1.5.4.30.2 Diagram



15.1.5.4.30.3 Fields

Field	Function
31-22 SW_PRED_BC_TAP_6_0_VP7_VP8	Prediction filter set 6, tap 0
21-12 SW_PRED_BC_TAP_6_1_VP7_VP8	Prediction filter set 6, tap 1
11-2	Prediction filter set 6, tap 2

Table continues on the next page...

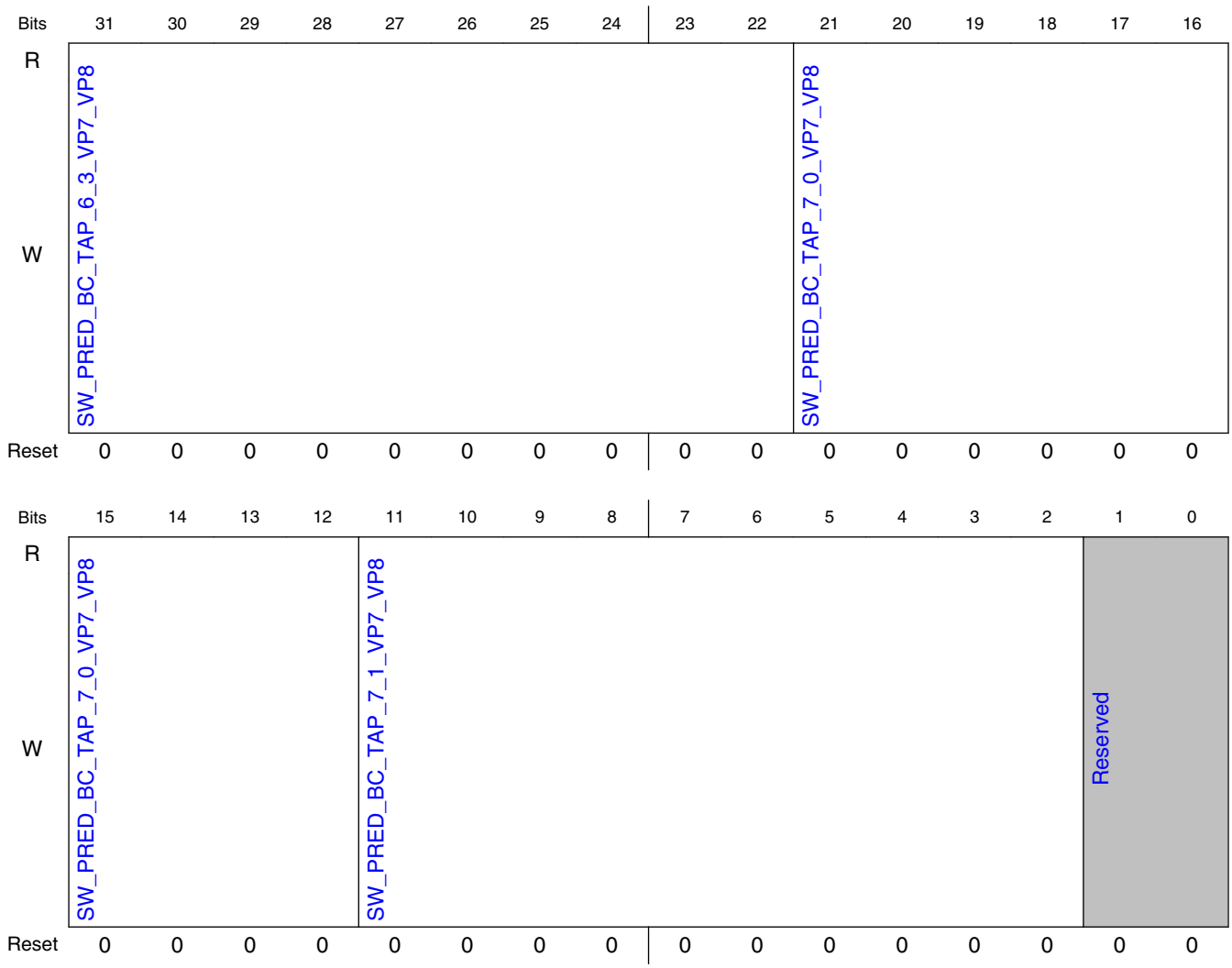
Field	Function
SW_PRED_BC_TAP_6_2_VP7_VP8	
1-0 —	Reserved.

15.1.5.4.31 bi-dir initial ref pic list register (6-8) / VP6 prediction filter taps (SWREG44_VP7_VP8)

15.1.5.4.31.1 Offset

Register	Offset
SWREG44_VP7_VP8	B0h

15.1.5.4.31.2 Diagram



15.1.5.4.31.3 Fields

Field	Function
31-22 SW_PRED_BC_TAP_6_3_VP7_VP8	Prediction filter set 6, tap 3
21-12 SW_PRED_BC_TAP_7_0_VP7_VP8	Prediction filter set 7, tap 0
11-2	Prediction filter set 7, tap 1

Table continues on the next page...

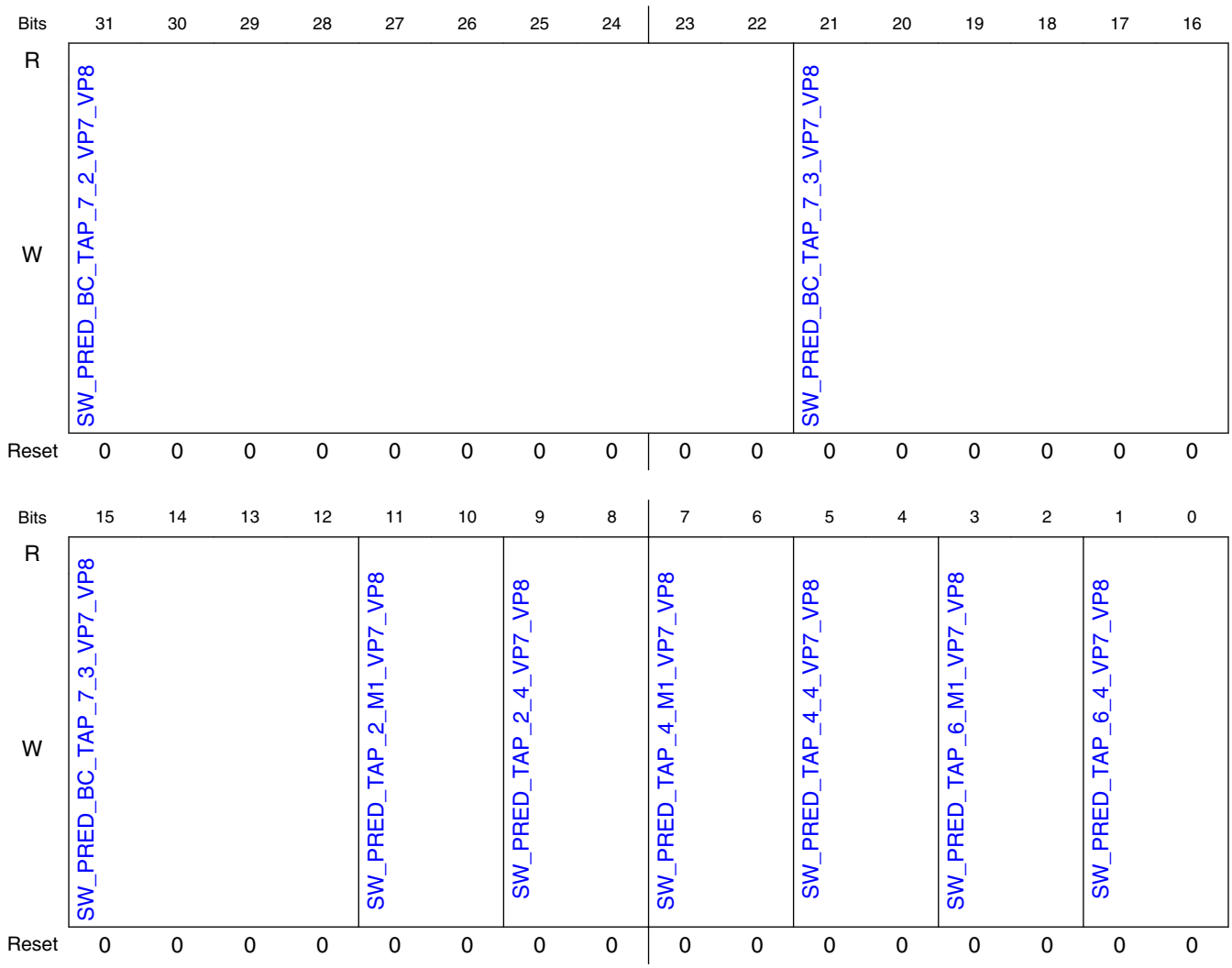
Field	Function
SW_PRED_BC_TAP_7_1_VP7_VP8	
1-0 —	Reserved.

15.1.5.4.32 bi-dir initial ref pic list register (9-11) / VP6 prediction filter taps (SWREG45_VP7_VP8)

15.1.5.4.32.1 Offset

Register	Offset
SWREG45_VP7_VP8	B4h

15.1.5.4.32.2 Diagram



15.1.5.4.32.3 Fields

Field	Function
31-22 SW_PRED_BC_TAP_7_2_VP7_VP8	Prediction filter set 7, tap 2
21-12 SW_PRED_BC_TAP_7_3_VP7_VP8	Prediction filter set 7, tap 3
11-10	Additional Prediction filter tap -1 for set 2

Table continues on the next page...

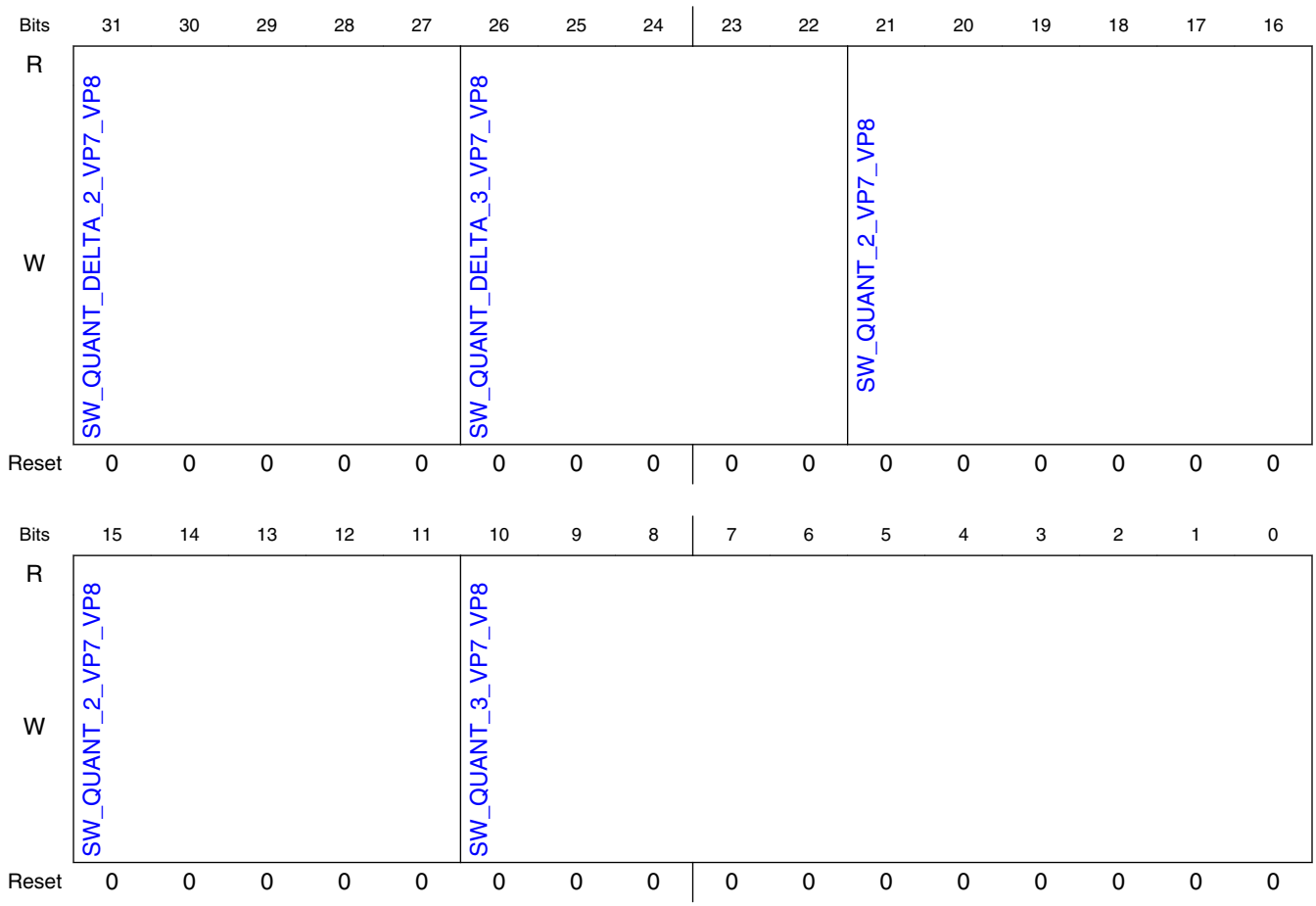
Field	Function
SW_PRED_TAP_2_M1_VP7_VP8	
9-8 SW_PRED_TAP_2_4_VP7_VP8	Additional Prediction filter tap 4 for set 2
7-6 SW_PRED_TAP_4_M1_VP7_VP8	Additional Prediction filter tap -1 for set 4
5-4 SW_PRED_TAP_4_4_VP7_VP8	Additional Prediction filter tap 4 for set 4
3-2 SW_PRED_TAP_6_M1_VP7_VP8	Additional Prediction filter tap -1 for set 6
1-0 SW_PRED_TAP_6_4_VP7_VP8	Additional Prediction filter tap 4 for set 6

15.1.5.4.33 bi-dir initial ref pic list register (12-14) / VP7,VP8 quantization values (SWREG46_VP7_VP8)

15.1.5.4.33.1 Offset

Register	Offset
SWREG46_VP7_VP8	B8h

15.1.5.4.33.2 Diagram



15.1.5.4.33.3 Fields

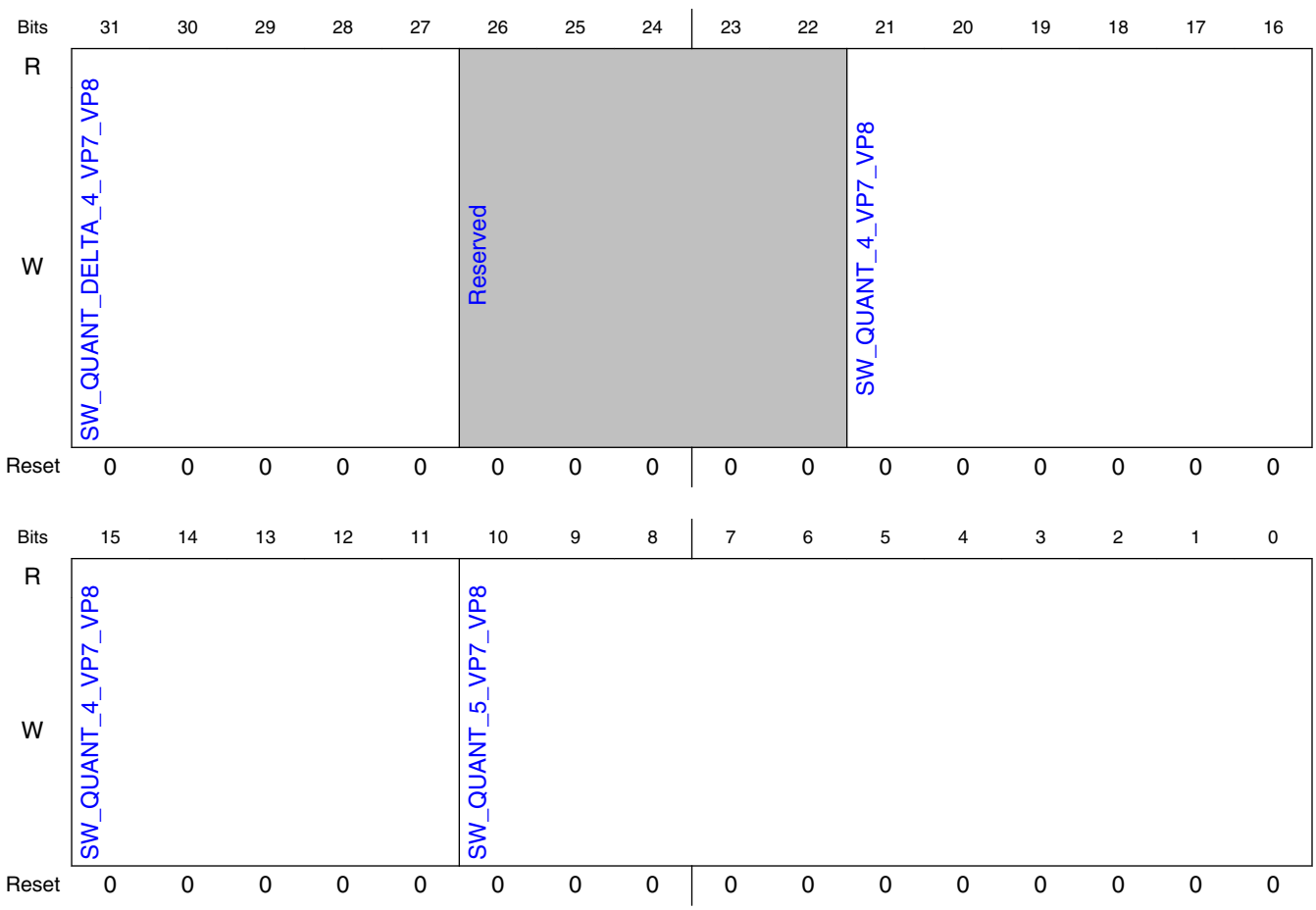
Field	Function
31-27 SW_QUANT_DELTA_2_VP7_VP8	VP8 quantisizer delta 2
26-22 SW_QUANT_DELTA_3_VP7_VP8	VP8 quantisizer delta 3
21-11 SW_QUANT_2_VP7_VP8	VP7: QP (11 bit) VP8: quantisizer value for LUT (7 bit)
10-0 SW_QUANT_3_VP7_VP8	VP7: QP (11 bit) VP8: quantisizer value for LUT (7 bit)

15.1.5.4.34 bi-dir and P fwd initial ref pic list register (15 and P 0-3) / VP7,VP8 quantization values (SWREG47_VP7_VP8)

15.1.5.4.34.1 Offset

Register	Offset
SWREG47_VP7_VP8	BCh

15.1.5.4.34.2 Diagram



15.1.5.4.34.3 Fields

Field	Function
31-27	VP8 quantisizer delta 4

Table continues on the next page...

Field	Function
SW_QUANT_DELTA_4_VP7_VP8	
26-22 —	Reserved.
21-11 SW_QUANT_4_VP7_VP8	VP7 QP (11 bit)
10-0 SW_QUANT_5_VP7_VP8	VP7 QP (11 bit)

15.2 VPU G2 (VPU_G2)

15.2.1 Overview

This block details the G2 hardware-based decoder and API. The decoder is able to decode HEVC standard Main/Main 10 profile compatible video streams, and Google's VP9 Profile 0 compatible video streams. The G2 decoder conforms to the HEVC Main/ Main 10 profiles and can decode streams up to level 5.1.

15.2.2 Video Frame Storage Format

The HW decoder may support two storage formats for frames, raster-scan, tiled without being compressed and compressed output. The application using the decoder may specify upon initialization which format is preferred, depending on HW support. The tiled format generally speaking offers better bus utilization and performance, however the decoder output must be converted back to raster-scan format prior to displaying it, except that GPU supports tiled format displaying. This postprocessing is beyond the scope of G2 decoder.

15.2.2.1 Raster-scan format

The output picture of the decoder is in semi-planar YCbCr 4:2:0 format, i.e. luminance data forms one plane in memory, and chrominance data forms the other. The output picture has to be stored linearly and contiguously in the memory. The interleaved

chrominance block has to be stored right after the luminance block in external memory as shown in Figure . The number of bytes in one luminance row must be divisible by 16, which means there will be padded 0 in the right edge when necessary.

The interleaved chrominance is stored raster-scanned in memory in Cb0Cr0Cb1Cr1Cb2Cr2Cb3… format.

The decoder output picture for interlaced sequence is always stored as an interlaced frame where even lines belongs to top field and odd lines belongs to bottom field of the frame.

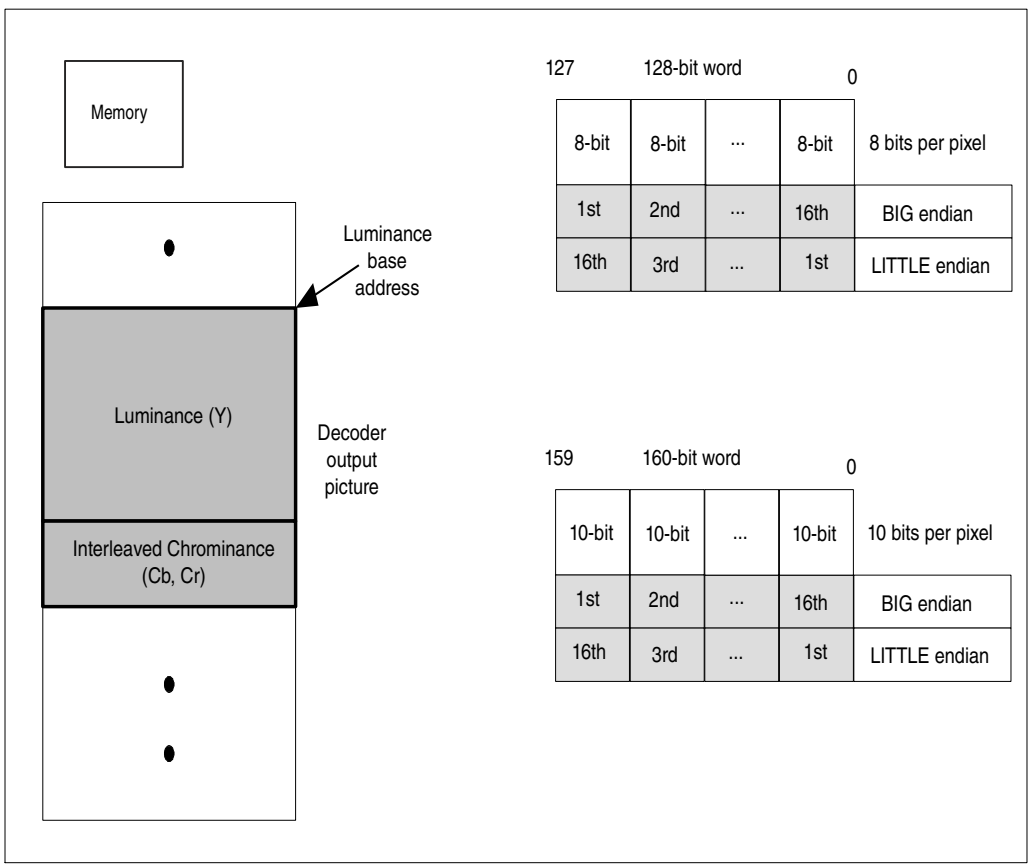


Figure 15-7. External Memory Usage

As an example the luminance pixels of a QCIF sized (176 x 144) frame with each pixel in 8 bits are numbered as presented in Table 1. Table 2 shows the storage of the luminance pixel data of the QCIF sized video frame in raster-scan order (in little endian mode).

Table 15-1. Pixel numbers of the luminance data of the QCIF video frame

0	1	2	3	4	5	6	7	…	172	173	174	175
176	177	178	179	180	181	182	183	…	348	349	350	351
::												::
25200	25201	25202	25203	25204	25205	25206	25207	…	25340	25341	25342	25343

The storage of the luminance pixel data of a QCIF video frame in raster-scan order. All pixels of a pixel row are stored in consecutive memory locations.

Table 15-2. Pixel Memory Locations

Address offset (decimal)	128-bit words(Pixel data of the luminance component. Each byte contains data for one pixel. Number in table are pixel numbers)			
0	15	...	1	0
16	31	...	17	16
32	47	...	33	32
48	63	...	49	48
64	79	...	65	64
:	:			
25296	25311	...	25297	25296
25312	25327	...	25313	25312
25328	25343	...	25329	25328

15.2.2.2 Tiled format

The output picture of the decoder is in semi-planar YCbCr 4:2:0 4x4 tiled format, i.e. luminance data forms one plane in memory, and chrominance data forms another. The distinction from raster scan format is that the output picture is grouped into tiles, which are then stored linearly and contiguously in the memory. The chrominance tiles have to be stored right after the luminance tiles in external memory as shown in Figure . The number of luminance pixels in one row must be divisible by 16.

The chrominance is stored into tiles in interleaved Cb0Cr0Cb1Cr1Cb2Cr2Cb3… format.

Figure contains example external memory usage when using 4x4 tiles, which is used in G2 decoder.

Below is an example code of converting tiled YCbCr 4:2:0 output picture back to raster scan format. The code is in generalized form, i.e. can function on arbitrary tile sizes as long as the picture dimensions are multiple of tile sizes. In the example, chrominance data is interleaved into 4x4 tile so that one 4x4 tile contains one 2x4 block of Cb and Cr each. Output is YCbCr 4:2:0 semiplanar.

```
void ConvertTiledToRaster( DecPicture decPicture, u32 tileWidth,
            u32 tileHeight, u8 *pOutput )
            {
            u32 i, j;
            u32 k, l;
            u32 s, t;
            u8 *pIn;
            pIn = decPicture.pOutputPicture;
            /* loop all tile rows for luminance */
            for( i = 0 ; i < decPicture.frameHeight ; i += tileHeight )
```

```
{
t = 0;
s = 0;
/* loop all tiles in one row */
for( j = 0 ; j < decPicture.frameWidth ; j += tileWidth )
{
/* copy one tile */
for( k = 0 ; k < tileHeight ; ++k )
{
for( l = 0 ; l < tileWidth ; ++l )
{
pOutput[ k*decPicture.frameWidth + l + s ] = pIn[t++];
}
}
/* move to next horizontal tile */
s += tileWidth;
}
/* move to next tile row */
pOutput += decPicture.frameWidth * tileHeight;
pIn += decPicture.frameWidth * tileHeight;
}
/* setup pointers to chrominance data */
pIn = decPicture.pOutputPicture +
decPicture.frameWidth * decPicture.frameHeight;
/* loop all tile rows for chrominance */
for( i = 0 ; i < decPicture.frameHeight/2 ; i += tileHeight )
{
t = 0;
s = 0;
/* loop all tiles in one row */
for( j = 0 ; j < decPicture.frameWidth ; j += tileWidth )
{
/* copy one tile */
for( k = 0 ; k < tileHeight ; ++k )
{
for( l = 0 ; l < tileWidth ; ++l )
{
pOutput[ k*decPicture.frameWidth + l + s ] = pIn[t++];
}
}
/* move to next horizontal tile */
s += tileWidth;
}
/* move to next tile row */
pOutput += decPicture.frameWidth * tileHeight;
pIn += decPicture.frameWidth * tileHeight;
}
}
```

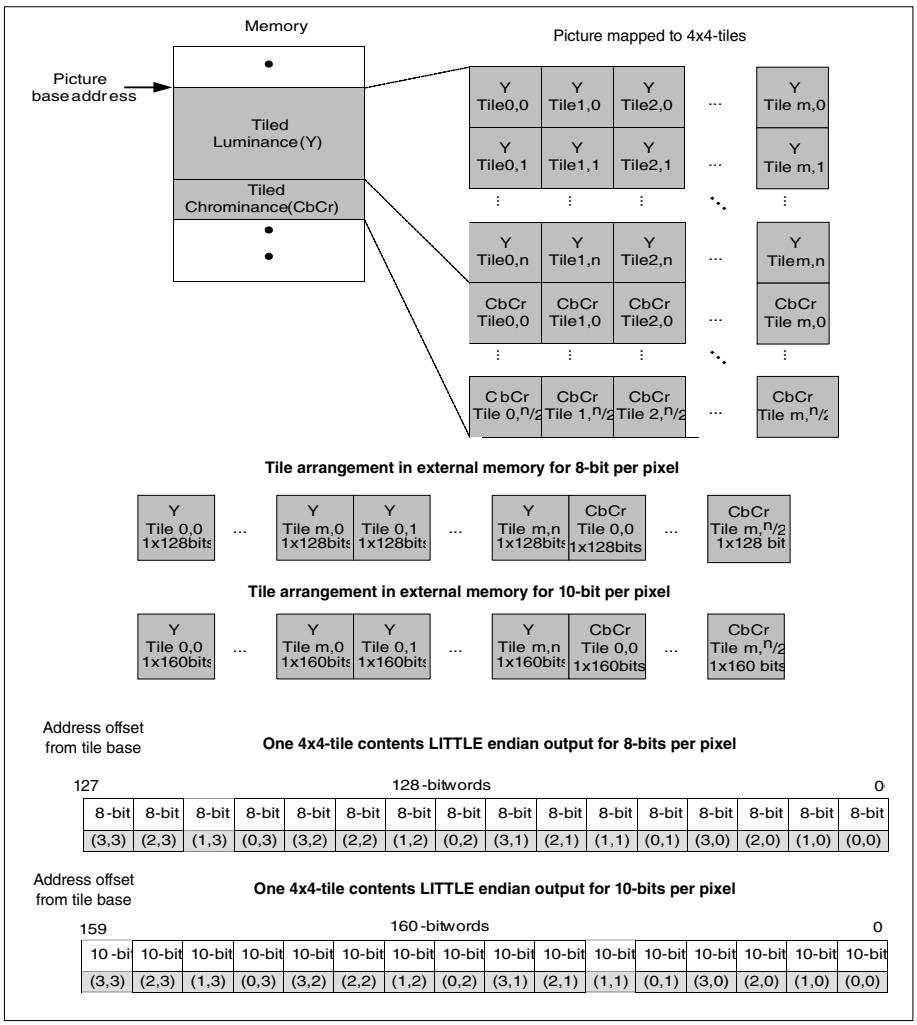


Figure 15-8. External Memory Usage in 4x4 Tiled Reference Picture Mode

15.2.2.3 Compressed format

G2 decoder supports compressed tiled output, when reference buffer compression (RFC) is enabled when synthesis, and user allows compressed reference frame output directly.

Reference frame compression is a feature added in G2 decoder to compress frame buffer so that the bandwidth of storing/loading reference frame can be reduced, especially when the resolution of decoded stream is of high definition.

When compression is enabled, the picture is divided into CBS rows, which is further divided into continuous CBS groups. Each CBS group is composed of 16 CBS. The luminance CBS is composed of 1 8x8 coded block (CB), which one chrominance CBS is composed of two 8x4 coded blocks, with cb CB first then cr CB following. The compression is performed to each CB in the CBS. The first CB's compressed data is

saved from the same offset of that CBS in raster scan buffer. And the compressed output of following CBs in the CBS row is saved continuously. That means there may be gaps between the compressed data of each CBS row.

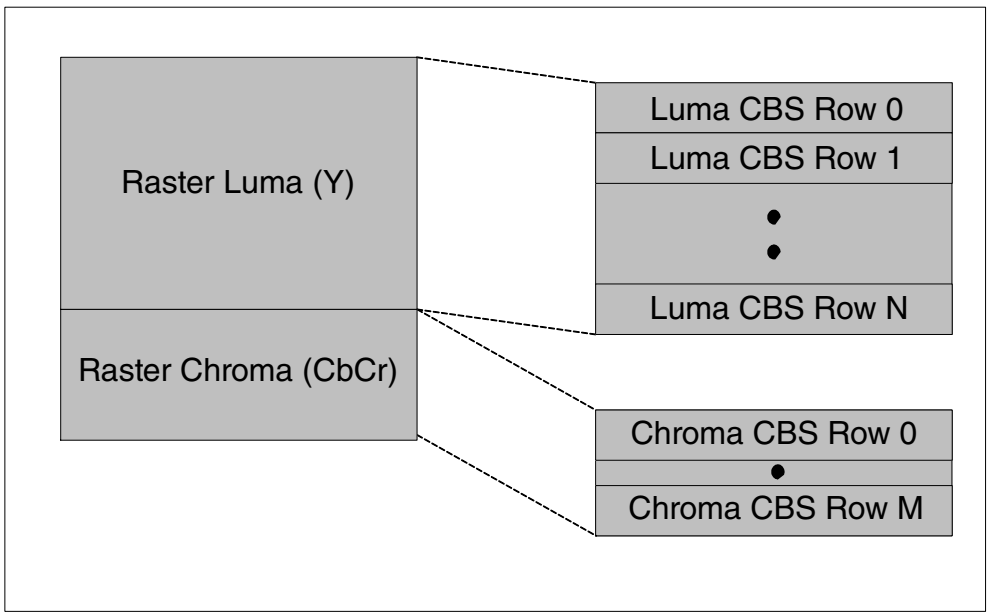


Figure 15-9. Raster scan picture mapped to CBS rows

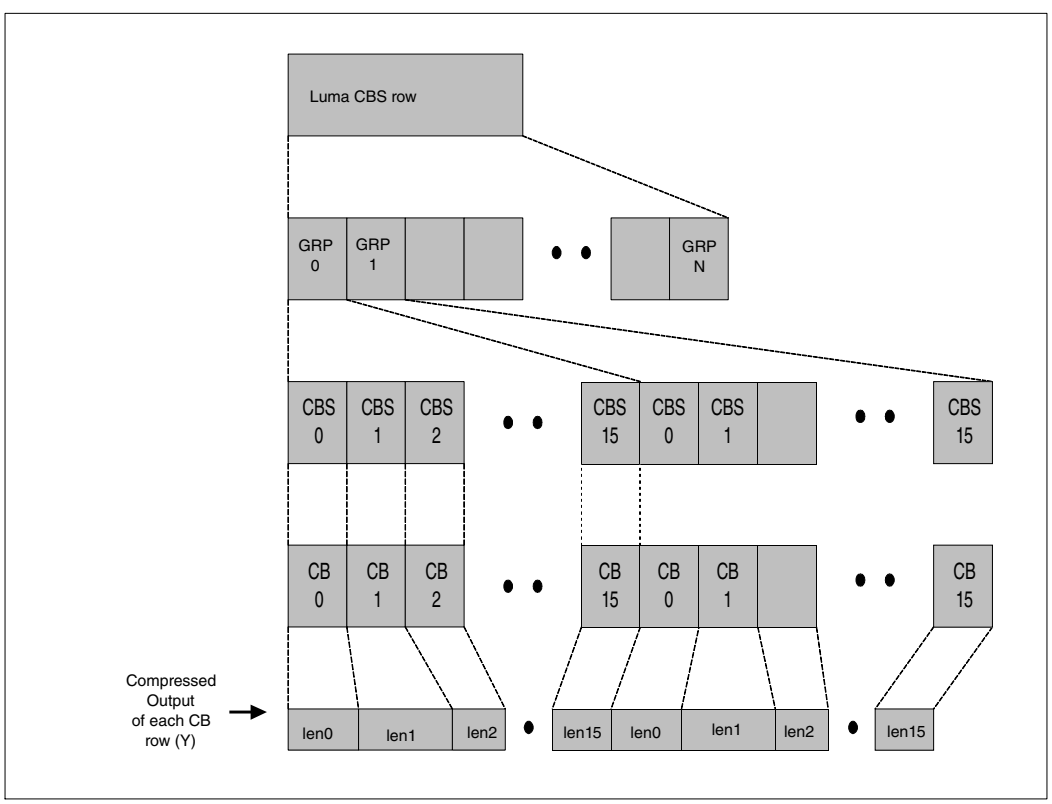


Figure 15-10. Compression performed to each Luma CBS row

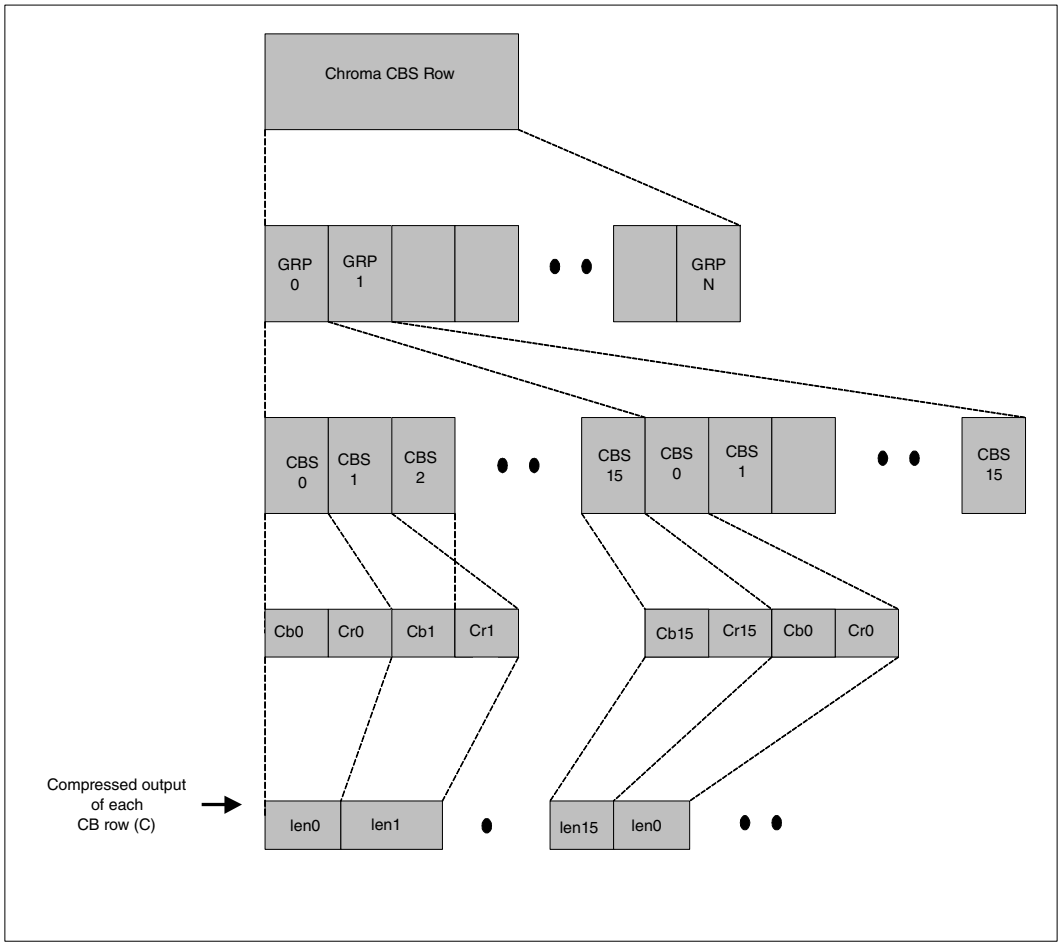


Figure 15-11. Compression performed to each Chroma CBS row

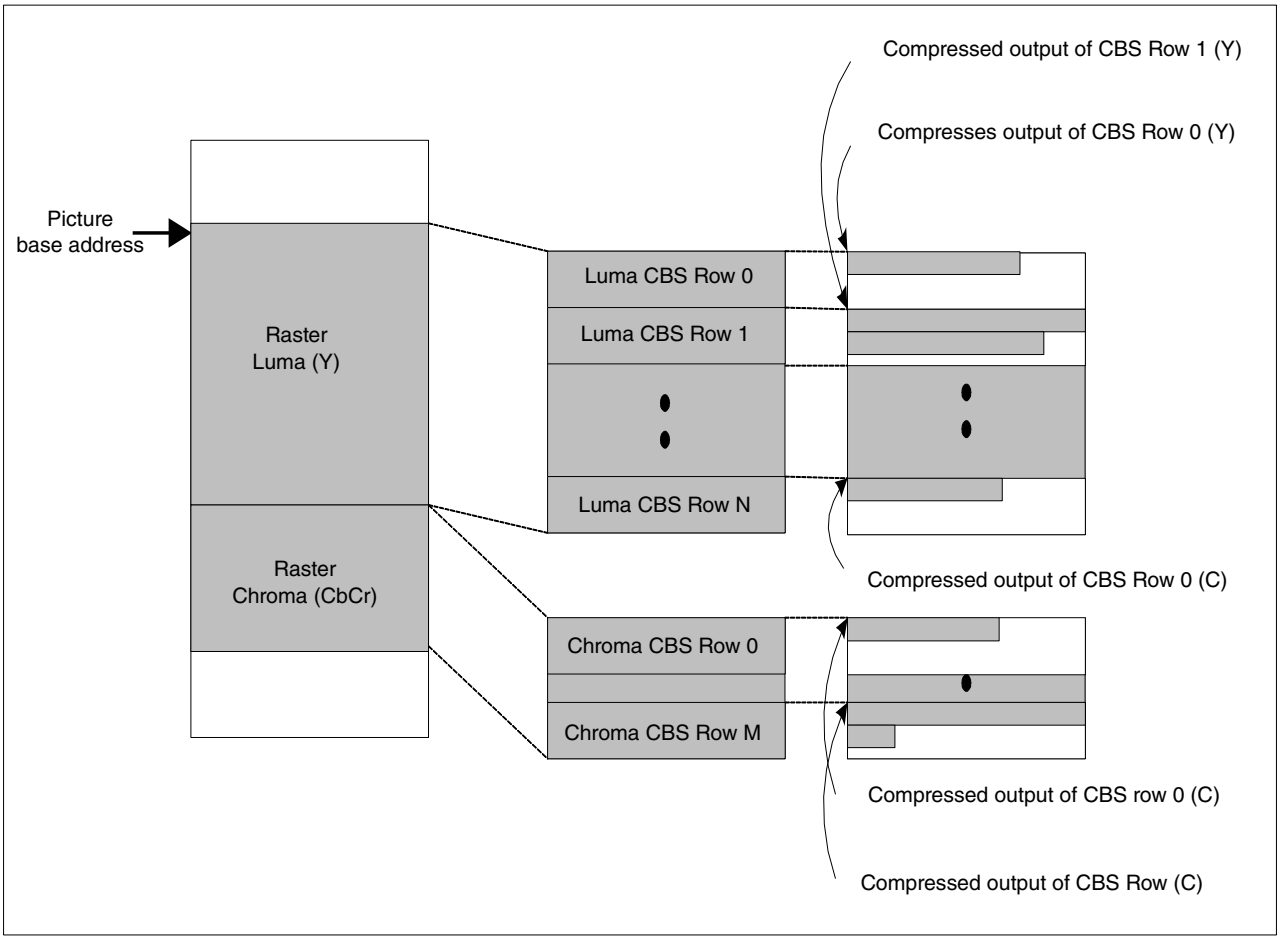


Figure 15-12. Map of external memory in RFC output buffer

Corresponding to each CBS group the starting offset of first compressed CBS output is saved in two bytes in CBS table, followed by the length of each CBS compress output, which is saved in 7 bits. So the table length of each CBS group information is 2+7*16/8=16 bytes.

This 16 bytes for each CBS group in compression table is mapped as shown in the following table. Here, X[n] means the bit-n of the length of CBS X in CBS group.

Table 15-3. Storage of the compression cache table for each CBS group

Byte	Bit7	Bit6	Bit5	Bit4	Bit 3	Bit 2	Bit 1	Bit 0
0	Offset[7:0]							
1	Offset[15:8]							
2	1[0]	0[6]	0[5]	0[4]	0[3]	0[2]	0[1]	0[0]
3	2[1]	2[0]	1[6]	1[5]	1[4]	1[3]	1[2]	1[1]
4	3[2]	3[1]	3[0]	2[6]	2[5]	2[4]	2[3]	2[2]
5	4[3]	4[2]	4[1]	4[0]	3[6]	3[5]	3[4]	3[3]
6	5[4]	5[3]	5[2]	5[1]	5[0]	4[6]	4[5]	4[4]

Table continues on the next page...

Table 15-3. Storage of the compression cache table for each CBS group (continued)

Byte	Bit7	Bit6	Bit5	Bit4	Bit 3	Bit 2	Bit 1	Bit 0
7	6[5]	6[4]	6[3]	6[2]	6[1]	6[0]	5[6]	5[5]
8	7[6]	7[5]	7[4]	7[3]	7[2]	7[1]	7[0]	6[6]
9	9[0]	8[6]	8[5]	8[4]	8[3]	8[2]	8[1]	8[0]
10	10[1]	10[0]	9[6]	9[5]	9[4]	9[3]	9[2]	9[1]
11	11[2]	11[1]	11[0]	10[6]	10[5]	10[4]	10[3]	10[2]
12	12[3]	12[2]	12[1]	12[0]	11[6]	11[5]	11[4]	11[3]
13	13[4]	13[3]	13[2]	13[1]	13[0]	12[6]	12[5]	12[4]
14	14[5]	14[4]	14[3]	14[2]	14[1]	14[0]	13[6]	13[5]
15	15[6]	15[5]	15[4]	15[3]	15[2]	15[1]	15[0]	14[6]

15.2.3 Interface Functions

This chapter describes the API declared in decapi.h, the definition of API in “software \source\common\decapi.c” of the release package.

Common numeric data types are declared in basetype.h as follows:

- **u8** – unsigned 8 bits integer value
- **i8** – signed 8 bits integer value
- **u16** – unsigned 16 bits integer value
- **i16** – signed 16 bits integer value
- **u32** – unsigned 32 bits value
- **i32** – signed 32 bits value

15.2.3.1 DecGetBuild

Syntax

```
struct DecSwHwBuild DecGetBuild(void)
```

Purpose

Returns the hardware and software build information of the decoder. Does not require the creation of a decoder instance.

Parameters

None.

Return value

DecSwHwBuild -- A structure containing the build information of the decoder.

```
struct DecSwHwBuild {
        u32 sw_build;                                   /* Software build ID */
        u32 hw_build;                                   /* Hardware build ID */
        struct DecHwConfig hw_config;          /* Hardware configuration */
};
```

Function	Description
sw_build	The internal release version of the Hantro HW HEVC software.
hw_build	The internal release version of the Hantro HW hardware.

DecHwConfig -- A structure containing the hardware-supported configuration.

```
struct DecHwConfig {
     u32 mpeg4_support;
     u32 custom_mpeg4_support;
     u32 h264_support;
     u32 vc1_support;
     u32 mpeg2_support;
     u32 jpeg_support;
     u32 jpeg_prog_support;
     u32 max_dec_pic_width;
     u32 max_dec_pic_height;
     u32 pp_support;
     u32 pp_config;
     u32 max_pp_out_pic_width;
     u32 sorenson_spark_support;
     u32 ref_buf_support;
     u32 tiled_mode_support;
     u32 vp6_support;
     u32 vp7_support;
     u32 vp8_support;
     u32 vp9_support;
     u32 avs_support;
     u32 jpeg_esupport;
     u32 rv_support;
     u32 mvc_support;
     u32 webp_support;
     u32 ec_support;
     u32 stride_support;
     u32 field_dpb_support;
     u32 hevc_support;
     u32 double_buffer_support;
     u32 hevc_main10_support;
     u32 vp9_10bit_support;
     u32 ds_support;
     u32 rfc_support;
     u32 ring_buffer_support;
};
```

Function	Description
mpeg4_support	Hardware support for MPEG-4 decoding. Possible values: • MPEG4_NOT_SUPPORTED – decoding not supported • MPEG4_SIMPLE_PROFILE – simple profile decoder • MPEG4_ADVANCED_SIMPLE_PROFILE – advanced simple profile decoder (includes • simple profile)
custom_mpeg4_support	Hardware support for DivX®decoding. Possible values:

Table continues on the next page...

Function	Description
	• MPEG4_CUSTOM_NOT_SUPPORTED – decoding not supported • MPEG4_CUSTOM - DivX Home Theater Profile QualifiedTM decoder available
h264_support	Hardware support for H.264 decoding. Possible values: • H264_NOT_SUPPORTED – decoding not supported • H264_BASELINE_PROFILE – baseline profile decoder • H264_MAIN_PROFILE – main profile decoder (includes baseline profile) • H264_HIGH_PROFILE – high profile decoder (includes baseline and main profile)
vc1_support	Hardware support for VC-1 decoding. Possible values: • VC1_NOT_SUPPORTED – decoding not supported. • VC1_SIMPLE_PROFILE – simple profile decoder • VC1_MAIN_PROFILE – main profile decoder (includes simple profile) • VC1_ADVANCED_PROFILE – advanced profile decoder (includes main profile)
mpeg2_support	Hardware support for MPEG-2 (includes MPEG-1) decoding. Possible values: • MPEG2_NOT_SUPPORTED – decoding not supported • MPEG2_MAIN_PROFILE – main profile decoder
jpeg_support	Hardware support for JPEG decoding. Possible values: • JPEG_NOT_SUPPORTED – decoding not supported • JPEG_BASELINE – baseline decoder
jpeg_prog_support	Hardware support for progressive JPEG decoding. Possible values: • JPEG_NOT_SUPPORTED – decoding not supported • JPEG_PROGRESSIVE – progressive decoder available
max_dec_pic_width	Maximum decoded picture's width for video decoders. The value is given in pixel units.
max_dec_pic_height	Maximum decoded picture's height for video decoders. The value is given in pixel units.
pp_support	Hardware support for video post-processing. Possible values: • PP_NOT_SUPPORTED – post-processing not supported. • PP_SUPPORTED – post-processing supported in hardware.
pp_config	Post-processing functions available. This is a bitwise list of the following values: • PP_DITHERING, PP_SCALING, PP_DEINTERLACING, and • PP_ALPHA_BLENDING. The availability of all these post-processor functions is • dependent on the hardware build.
max_pp_out_pic_width	Maximum output picture's width for the post-processor. The value is given in pixel units.

Table continues on the next page...

Function	Description
sorenson_spark_support	Hardware support for Sorenson Spark decoding. Possible values: • SORENSON_SPARK_NOT_SUPPORTED – decoding not supported • SORENSON_SPARK_SUPPORTED – decoder available
ref_buf_support	Hardware includes reference buffer handling speedup logic. Possible values: • REF_BUF_NOT_SUPPORTED – logic not available • REF_BUF_SUPPORTED – logic available for progressive pictures (flag) • REF_BUF_INTERLACED – logic available for interlaced pictures (flag) • REF_BUF_DOUBLE – logic available for buffering two reference pictures (flag) • Multiple values are possible using bitwise OR operation of valid flags.
tiled_mode_support	Hardware includes support for handling reference pictures in tiled mode to optimize bus utilization. Possible values: • TILED_NOT_SUPPORTED – tiled reference picture support not available • TILED_8x4_SUPPORTED – tiled reference picture support for 8x4 tiles available • TILED_8x4_ILACED_SUPPORTED – tiled reference picture support for 8x4 tiles • available also for interlaced pictures.
vp6_support	Hardware support for VP6 decoding. Possible values: • VP6_NOT_SUPPORTED – decoding not supported • VP6_SUPPORTED – decoder available
vp7_support	Hardware support for VP7 decoding. Possible values: • VP7_NOT_SUPPORTED – decoding not supported • VP7_SUPPORTED – decoder available
vp8_support	Hardware support for VP8 decoding. Possible values: • VP8_NOT_SUPPORTED – decoding not supported • VP8_SUPPORTED – decoder available
vp9_support	Hardware support for VP9 decoding. Possible values: • VP9_NOT_SUPPORTED – decoding not supported • VP9_SUPPORTED – decoder available
avs_support	Hardware support for AVS decoding. Possible values: • AVS_NOT_SUPPORTED – decoding not supported • AVS_SUPPORTED – decoder available
jpeg_esupport	Hardware support for baseline JPEG decoding up to 67 Mpixel images, and 4:1:1 and 4:4:4 sampling. Possible values: • JPEG_EXT_NOT_SUPPORTED – decoding not supported • JPEG_EXT_SUPPORTED – decoder available
rv_support	Hardware support for RV decoding. Possible values:

Table continues on the next page...

Function	Description
	• RV_NOT_SUPPORTED – decoding not supported • RV_SUPPORTED – decoder available
mvc_support	Hardware support for MVC decoding. Possible values: • MVC_NOT_SUPPORTED – decoding not supported • MVC_SUPPORTED – decoder available
webp_support	Hardware support for WebP decoding. Possible values: • WEBP_NOT_SUPPORTED – decoding not supported • WEBP_SUPPORTED – decoder available
ec_support	Hardware support for error concealment. Possible values: • EC_NOT_SUPPORTED – error concealment not supported • EC_SUPPORTED – error concealment available
stride_support	Hardware support for separate Y and C strides. Possible values: • STRIDE_NOT_SUPPORTED – strides not supported • STRIDE_SUPPORTED – strides available
field_dpb_support	Hardware support for field-mode DPB. Possible values: • FIELD_DPB_NOT_SUPPORTED – field-mode DPB not supported • FIELD_DPB_SUPPORTED –field-mode DPB available
hevc_support	Hardware support for HEVC decoding. Possible values: • HEVC_NOT_SUPPORTED – decoding not supported • HEVC_SUPPORTED – decoder available
double_buffer_support	Hardware support for decoder internal reference double buffering. Possible values: • DOUBLE_BUFFER_NOT_SUPPORTED – internal reference double buffering not • supported • DOUBLE_BUFFER_SUPPORTED – internal reference double buffering available
hevc_main10_support	Hardware support for HEVC main10 profile decoding. Possible values: • 0 – decoding not supported • 1 – decoder available
vp9_10bit_support	Hardware support for VP9 10 bits decoding. Possible values: • 0 – decoding not supported • 1 – decoder available
ds_support	Hardware support for decoder down scale outputting. Possible values: • 0 – decoding not supported • 1 – decoder available
rfc_support	Hardware support for reference frame compression. Possible values: • 0 – decoding not supported • 1 – decoder available
ring_buffer_support	Hardware support for ring buffer. Possible values:

Function	Description
	• 0 – decoding not supported • 1 – decoder available

15.2.3.2 DecInit

Syntax

```
enum DecRet DecInit(enum DecCodec codec,
                                  DecInst* decoder,
                                  struct DecConfig config,
                                  struct DecClientHandle callbacks)
```

Purpose

DecInit initializes a new decoder instance based on the given codec type. If parameter noOutputReordering is set to a non-zero value the decoder will not provide output pictures in display order but in decoding order (unless decoding order and display order is the same). Outputting pictures in decode order will lower the memory consumption of the decoder in case the number of reference frames used by the stream is less than the maximum number allowed by the profile and level of the stream.

Parameters

```
enum DecCodec codec
                  Codec type to be initialized.
                  enum DecCodec {
                  DEC_VP9,
                  DEC_HEVC
                  };
             DecInst * decoder
                  Pointer to the location where DecInit returns a decoder instance. This
instance is later
                  passed to other decoder API functions.
             struct DecConfig config
                  Decoder initialization parameters.
                    struct DecConfig {
                    u32 disable_picture_reordering;
                    enum DecPictureFormat output_format;
                    struct DWL dwl;
                    const void* dwl_inst;
                    u32 max_num_pics_to_decode;
                    enum DecErrorConcealment concealment_mode;
                    struct DecDownscaleCfg dscale_cfg;
                    u32 use_video_compressor;
                    u32 use_ringbuffer;
                    u32 use_8bits_output;
             }
             u32 disable_picture_reordering
                 Flag to disable the reordering of output frames. When set to a non-zero
value the
                 decoder will not provide output pictures in display order but in decoding
order (unless
                 decoding order and display order is the same).
             enum DecPictureFormat output_format
                  Format of the output picture.
```

```
                    enum DecPictureFormat {
                     DEC_OUT_FRM_TILED_4X4 = 0,
                     DEC_OUT_FRM_RASTER_SCAN = 1,
                     DEC_OUT_FRM_PLANAR_420 = 2
                     };
                    DEC_OUT_FRM_TILED_4X4 - Tiled 4x4 format
                    DEC_OUT_FRM_RASTER_SCAN - raster scan format (a.k.a semi-planar)
                    DEC_OUT_FRM_PLANAR_420 - planar format
               struct DWL dwl
                    Decoder wrapper layer functionality.
               const void* dwl_inst
                    Pointer to a DWL instance, which should be instantiated by calling DWLInit
in
                    advance.
               u32 max_num_pics_to_decode
                    Limits the maximum pictures number to be decoded, 0 for unlimited.
               enum DecErrorConcealment concealment_mode
                    Flag to determine which error concealment method to use. When set to
                    DEC_INTRA_FREEZE the decoder will conceal every frame after an error has
been
                    detected in the bitstream, until the next IDR (key frame) frame is
decoded. When set to
                    DEC_PICTURE_FREEZE, the decoder will conceal only the frames having errors
in the
                    bitstream.

                    enum DecErrorConcealment {
                    DEC_PICTURE_FREEZE = 0,
                    DEC_INTRA_FREEZE = 1
                    };
               struct DecDownscaleCfg dscale_cfg
                    Down scale parameters.
                    struct DecDownscaleCfg {
                      u32 down_scale_x;
                      u32 down_scale_y;
                    };
                    u32 down_scale_x
                        Down scale ratio in horizontal direction. Possible value: 2, 4, 8.
Available
                        value:2, 4, 8, which means down scaled to 1/2, 1/4 or 1/8
respectively. 1
                        means no scaling.
                    u32 down_scale_y
                        Down scale ratio in vertical direction. Possible value: 2, 4, 8.
Available value:2,
                        4, 8, which means down scaled to 1/2, 1/4 or 1/8 respectively. 1 means
no
                        scaling.
```

NOTE

If the down scale ratio in one direction is set to 1, no matter what the value is in the other direction, no down scale will be performed in both directions.

```
               u32 use_video_compressor
                   Flag to enable reference buffer compression. Set 0 to bypass reference
buffer
                   compression, or when reference buffer compression is not supported. When
set to 1,
                   the decoded reference buffer will be saved in compressed format.
               u32 use_ringbuffer
                   Flag to indicate decoder use ring buffer or not. Set 1 to enable ring
buffer. When set to
                   0, ring buffer will be disabled.
                   Stream ring buffer mode is an alternative storage for stream. In this mode,
frame
```

stream data usually doesn’t start from buffer head but follow a previous frame stream
data. When stream data goes to stream buffer end, it must turnaround to buffer head to
save left stream data. It requires a whole frame stream data saved in stream buffer and
stream data start address is byte-aligned.
Below figure shows how stream data is stored in stream buffer if ring buffer mode.

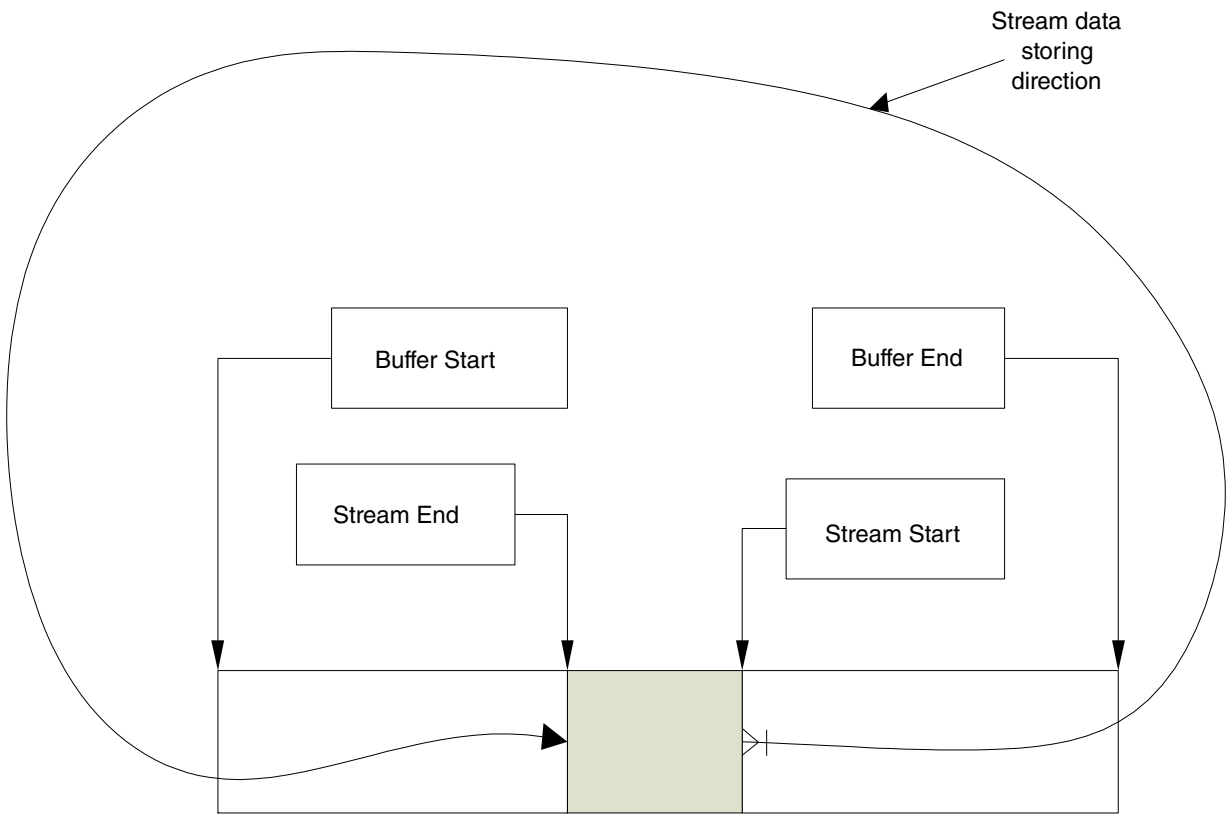


Figure 15-13. Data storage in Stream buffer in Ring Buffer Mode

u32 use_8bits_output
Flag to force 8 bit per pixel output when it's a stream with higher bit depth, e.g., HEVC
Main10 profile stream. The conversion is based on following rule:

Table 15-4. Rule for conversion

Bit depth of original pixel	Original pixel	8bit output
9	XXXXXXXX0	XXXXXXXX
	XXXXXXXX1	XXXXXXXX+1
10	XXXXXXXX00	XXXXXXXX
	XXXXXXXX01	XXXXXXXX
	XXXXXXXX10	XXXXXXXX+1
	XXXXXXXX11	XXXXXXXX+1

struct DecClientHandle callbacks
Callback interfaces that client should provide to facilitate the decoder

```
to inform the client that
                    there is any events or state changing in decoder.
                    struct DecClientHandle {
                           ClientInst client;
                           ClientInitialized* Initialized;
                           ClientHeadersDecoded* HeadersDecoded;
                           ClientBufferDecoded* BufferDecoded;
                           ClientPictureReady* PictureReady;
                           ClientEndOfStream* EndOfStream;
                           ClientReleased* Released;
                           ClientNotifyError* NotifyError;
                    };
                    typedef const void* ClientInst;
                           Opaque pointer to the client.
                    typedef void ClientInitialized(ClientInst inst);
                           Function to notify the client that decoder has successfully
initialized.
                    typedef void ClientHeadersDecoded(ClientInst inst,
                                struct DecSequenceInfo sequence_info);
                           Function to notify about successful decoding of the stream
parameters. Decoder
                           expects client to provide needed buffers through
DecSetPictureBuffers function to
                           continue decoding the actual stream.
                    typedef void ClientBufferDecoded(ClientInst inst,
                                                     struct DecInput* input);
                           Function to notify client that a buffer has been consumed by the
decoder and it can be
                           handled freely by the client.
                    typedef void ClientPictureReady(ClientInst inst,
                                                    struct DecPicture picture);
                           Function to notify about picture that is ready to be outputted.
Client is expected to notify
                           the decoder when it has finished processing the picture, so
decoder can reuse the
                           picture buffer for another picture.
                    typedef void ClientEndOfStream(ClientInst inst);
                            Function to notify the client that all the pending pictures
have been outputted and
                            decoder can be safely shut down.
                    typedef void ClientReleased(ClientInst inst);
                           Function to notify the client that decoder has shut down.
                    typedef void ClientNotifyError(ClientInst inst,
                                                   u32 pic_id,
                                                   enum DecRet rv);
                           Function to notify client about error in the decoding process.
```

Return

```
DEC_OK
                API function returned successfully.
            DEC_PARAM_ERROR
                Bad input parameters.
            DEC_MEMFAIL
                The decoder was not able to allocate memory.
            DEC_FORMAT_NOT_SUPPORTED
                Format is not supported in hardware. See DecGetBuild.
```

15.2.3.3 DecDecode

Syntax

```
enum DecRet DecDecode(DecInst decInst,
        DecInput *pInput)
```

Purpose

This function will push a COMMAND_DECODE command to the decoding thread, along with the input stream parameters, which will inform the decoding thread to decode one or more NAL units from the current stream for HEVC or element stream for VP9. The input buffer should contain one of the following:

1. Exactly one NAL unit and nothing else.
2. One or more NAL units in byte stream format, as defined in Annex B of the standard [1].
3. Or for VP9 at least one whole frame bit stream data.

The decoder automatically detects the format of the stream data and decodes NAL units until the whole buffer is processed or decoding of a picture is finished for HEVC. For VP9 if there is not enough bit stream data in input buffer, HW decoder will return TIMEOUT error.

Parameters

DecInst*decInst* - A decoder instance created earlier with an DecInit call.

DecInput **pInput* - Pointer to the decoder input structure.

```
struct DecInput {
               struct DWLLinearMem buffer; /* Pointer to the input buffer. */
               u32 data_len;
        };
```

struct DWLLinearMem buffer -- Linear buffer descriptor.

```
struct DWLLinearMem {
                  u32 *virtual_address;
                  u32 bus_address;
                  u32 size;
                  u32 logical_size;
          };
```

Function	Description
*virtual_address	Pointer to the virtual address of stream linear buffer.
bus_address	Bus address of the linear buffer. NOTE: The start of the stream buffer must be in a 128-bit aligned position or accessible for reading from the previous 128-bit aligned position.
size	Physical size of the linear buffer, which is rounded to page multiple.
logical_size	Physical size of the linear buffer, which is rounded to page multiple.

Table continues on the next page...

Function	Description
datalen	Number of bytes contained in the buffer stream buffer. The length can be represented on maximum 32 bits. Valid range: [0, (2^32-1)]

Return value

DEC_OK - The COMMAND_DECODE is successfully pushed to decoder thread's input queue.

DEC_PARAM_ERROR - Error in calling parameters.

DEC_NOT_INITIALIZED - Decoder instance is not initialized. Stream decoding is not started. DecInit must be called before decoding can be started.

15.2.3.4 DecSetPictureBuffers

Syntax

```
enum DecRet DecSetPictureBuffers(DecInst dec_inst,
               const struct DWLLinearMem* buffers,
               u32 num_of_buffers)
```

Purpose

DecSetPictureBuffers assigns picture buffers for the decoder. When decoder has finished decoding the stream headers and knows which types of buffers and how many of them it will need, it will inform that through the HeadersDecoded callback. Buffers must not be written into until client has successfully called DecRelease or decoder has requested new set of buffers through HeadersDecoded callback.

Parameters

DecInst*decInst* - A decoder instance created earlier with an DecInit call.

const struct DWLLinearMem* buffers - Linear buffer to be assigned to the decoder.

u32 num_of_buffers - Number of buffers to be assigned to the decoder.

Return value

DEC_OK - Successfully set picture buffers.

DEC_PARAM_ERROR - Error in calling parameters.

15.2.3.5 DecPictureConsumed

Syntax

```
enum DecRet DecPictureConsumed(DecInst decInst,
        struct DecPicture picture)
```

Purpose

DecPictureConsumed informs the decoder that the client has finished processing a specific picture, which was previously sent to client through the PictureReady callback.

Parameters

DecInst*decInst* - A decoder instance.

DecPicture *pOutput - Pointer to the location used to return the picture parameters. The picture parameters are valid only when the return value indicates that an output picture is available.

```
struct DecPicture {
               struct DecSequenceInfo sequence_info;
               struct DWLLinearMem luma;
               struct DWLLinearMem chroma;
               struct DecPictureInfo picture_info;
               struct DWLLinearMem dscale_luma;
               struct DWLLinearMem dscale_chroma;
               u32 dscale_width;
               u32 dscale_height;
               u32 dscale_stride;
          };
          struct DecSequenceInfo sequence_info
                struct DecSequenceInfo {
                   u32 pic_width;
                   u32 pic_height;
                   u32 sar_width;
                   u32 sar_height;
                   struct DecCropParams crop_params;
                   enum DecVideoRange video_range;
                   u32 matrix_coefficients;
                   u32 is_mono_chrome;
                   u32 is_interlaced;
                   u32 num_of_ref_frames;
                   u32 bit_depth_luma;
                   u32 bit_depth_chroma;
                   u32 pic_stride;
                };
```

Function	Description
picture_width	Decoded picture width in pixels.
picture_height	Decoded picture height in pixels.
sar_width	Horizontal size of the sample aspect ratio.
sar_height	Vertical size of the sample aspect ratio.
struct DecCropParams crop_param	Cropping parameters for the picture.

Table continues on the next page...

Function	Description

	struct DecCropParams {
	u32 crop_left_offset;
	u32 crop_out_width;
	u32 crop_top_offset;
	u32 crop_out_height;
	};

crop_left_offset	Left offset for cropping process, i.e. number of luminance samples removed from the left edge of the picture (always a multiple of 2).
crop_out_width	Width of the picture in luminance samples after cropping is performed.
crop_top_offset	Top offset for cropping process, i.e. number of luminance samples removed from the top edge of the picture (always a multiple of 2).
crop_out_height	Height of the picture in luminance samples after cropping is performed.
Enum DecVideoRange video_range	YUV sample video range. Possible value: • DEC_VIDEO_RANGE_NORMAL - sample range [16, 235] • DEC_VIDEO_RANGE_FULL - sample range [0, 255]
matrix_coefficients	Matrix coefficients RGB->YUV conversion.
is_mono_chrome	Flag to indicate whether sequence is of monochrome.
is_interlaced	Flag to indicate whether sequence is interlaced.
num_of_ref_frames	Maximum number of reference frames.
bit_depth_luma	Bit depth of stored pixels for luma plane.
bit_depth_chroma	Bit depth of stored pixels for chroma plane.
pic_stride	Stride of the picture as stored in memory in bytes.
struct DWLLinearMem luma	Linear buffer for luma pixels.
struct DWLLinearMem chroma	Linear buffer for chroma pixels.

struct DecPictureInfo picture_info	Picture specific information.
	struct DecPictureInfo {
	enum DecPicCodingType pic_coding_type;
	u32 is_corrupted;
	enum DecPictureFormat format;
	u32 cycles_per_mb;
	u32 pic_id;
	};

enum DecPicCodingType pic_coding_type	Picture coding type. Possible value: • DEC_PIC_TYPE_I - key frame • DEC_PIC_TYPE_P - P frame • DEC_PIC_TYPE_B - B frame
is_corrupted	Flag indicates whether picture is corrupted.
enum DecPictureFormat format	Color format of the picture. Refer to DecPictureFormat in 4.2.
cycles_per_mb	Average decoding time in cycles per mb.

Table continues on the next page...

Function	Description
pic_id	Identifier for the picture to be decoded.
struct DWLLinearMem dscale_luma	Linear buffer for luma pixels in down scaled output.
struct DWLLinearMem dscale_chroma	Linear buffer for chroma pixels in down scaled output.
dscale_width	Width of down scaled output in pixels.
dscale_height	Height of down scaled output in pixels.
dscale_stride	Stride of a pixel line in down scaled output in bytes.

Return value

DEC_OK - No pictures available for display.

DEC_PARAM_ERROR - Error in calling parameters.

DEC_NOT_INITIALIZED - Decoder instance is not initialized. DecInit must be called before decoding can be started.

15.2.3.6 DecEndOfStream

Syntax

```
enum DecRet DecEndOfStream (DecInst decInst)
```

Purpose

DecEndOfStream sends a COMMAND_END_OF_STREAM command to the decoder thread to inform the decoder that it should not be expecting any more input stream and finish decoding and outputting all the buffers that are currently pending in the component. Once decoder has finished outputting the pending pictures it will notify the client about it by calling the EndOfStream callback.

Parameters

DecInst*decInst* - A decoder instance.

Return value

DEC_OK - API function returned successfully

DEC_PARAM_ERROR - Error in calling parameters.

DEC_INITFAIL - Decoder instance is not initialized.

15.2.3.7 DecRelease

Syntax

```
void DecRelease(DecInst decInst)
```

Purpose

DecRelease sends a COMMAND_RELEASE command to decoder thread, which will finally close the decoder instance decInst and releases all internally allocated resources. This function must not be called twice for the same instance.

Parameters

DecInst*decInst* - A decoder instance to be released.

Return value

None.

15.2.4 Decoder Application Examples

Purpose

The purpose of the following examples is to show how to implement client's callback functions, as long as using G2 decoder API to implement a simple decoder with demuxer to decode both HEVC and VP9 streams. InitializedCb, HeadersDecodedCb, BufferDecodedCb, PictureReadyCb, EndOfStreamCb, ReleasedCb, and NotifyErrorCb are the system/application dependent callback functions, which are provided by client to the decoder wrapper.

The examples are simplified, e.g. they do not take errors into consideration.

NOTE

Program code in the examples should be considered as pseudo code and may not compile as such.

```
            void SetupDefaultParams(struct TestParams* params);
            int ParseParams(int argc, char* argv[], struct TestParams* params);
            void* CreateDemuxer(struct Client* client);
            void ReleaseDemuxer(struct Client* client);
            void PostProcessPicture(struct Client* client, struct DecPicture* picture);

            static void DispatchBufferForDecoding(struct Client* client, struct DecInput*
buffer)
            {
             enum DecRet rv;
             i32 size = buffer->buffer.size;
             memset(buffer->buffer.virtual_address, 0, size);
             i32 len = client->demuxer.ReadPacket(client->demuxer.inst, (u8*)buffer-
>buffer.virtual_address, &size);
             if (len <= 0) {/* EOS or error. */
```

```
              /* If reading was due to insufficient buffer size, try to realloc with
sufficient size. */
              if (size > buffer->buffer.size) {
               i32 i;
               for (i = 0; i < GetStreamBufferCount(client); i++) {
                if (client->buffers[i].buffer.virtual_address == buffer-
>buffer.virtual_address) {
                  DWLFreeLinear(client->dwl, &client->buffers[i].buffer);
                  if (DWLMallocLinear(client->dwl, size, &client->buffers[i].buffer))
                  {
                    DispatchEndOfStream(client);
                    return;
                  }
                  DispatchBufferForDecoding(client, &client->buffers[i]);
                  return;
                 }
                }
               }
               else {
                DispatchEndOfStream(client);
                return;
               }
              }
              buffer->data_len = len;
              if (client->eos) return; /* Don't dispatch new buffers if EOS already done. */
              /* Decode the contents of the input stream buffer. */
              switch (rv = DecDecode(client->decoder, buffer)) {
               case DEC_OK:
                /* Everything is good, keep on going. */
                break;
               default:
                DispatchEndOfStream(client);
                break;
              }
             }

             static void InitializedCb(ClientInst inst)
             {
              struct Client* client = (struct Client*)inst;
              /* Internal testing feature: Override HW configuration parameters */
              HwconfigOverride(client->decoder,  &tb_cfg);
              /* Start the output handling thread, if needed. */
              if (client->test_params.extra_output_thread)
               pthread_create(&client->parallel_output_thread, NULL, ParallelOutput, client);
               for (int i = 0; i < GetStreamBufferCount(client); i++) {
                if (DWLMallocLinear(client->dwl, DEFAULT_STREAM_BUFFER_SIZE,&client-
>buffers[i].buffer)) {
                 DecRelease(client->decoder);
                 return;
                }
                /* Dispatch the first buffers for decoding. When decoder finished
                * decoding each buffer it will be refilled within the callback. */
                DispatchBufferForDecoding(client, &client->buffers[i]);
               }
             }

             static void HeadersDecodedCb(ClientInst inst, struct DecSequenceInfo info)
             {
              struct Client* client = (struct Client*)inst;
              if (client->yuvsink.inst == NULL) {
               if (client->test_params.out_file_name == NULL) {
                client->yuvsink.inst = CreateSink(client);
               }
               DecSetPictureBuffers(client->decoder, NULL, 0);
              }
              static void BufferDecodedCb(ClientInst inst, struct DecInput* buffer) {
               struct Client* client = (struct Client*)inst;
                if (!client->eos) {
                 DispatchBufferForDecoding(client, buffer);
                }
```

```
            }
           }

           static void PictureReadyCb(ClientInst inst, struct DecPicture picture)
           {
            static char* pic_types[] = {" IDR", "Non-IDR (P)", "Non-IDR (B)"};
            struct Client* client = (struct Client*)inst;
            client->num_of_output_pics++;
            DEBUG_PRINT(("PIC %2d/%2d, type %s,", client->num_of_output_pics,
picture.picture_info.pic_id, pic_types[picture.picture_info.pic_coding_type]));
            if (picture.picture_info.cycles_per_mb) {
             client->cycle_count += picture.picture_info.cycles_per_mb;
             DEBUG_PRINT((" %4d cycles / mb,", picture.picture_info.cycles_per_mb));
            }
            DEBUG_PRINT((" %d x %d, Crop: (%d, %d), %d x %d %s\n",
            picture.sequence_info.pic_width,
            picture.sequence_info.pic_height,
            picture.sequence_info.crop_params.crop_left_offset,
            picture.sequence_info.crop_params.crop_top_offset,
            picture.sequence_info.crop_params.crop_out_width,
            picture.sequence_info.crop_params.crop_out_height,
            picture.picture_info.is_corrupted ? "CORRUPT" : ""));
            if (client->test_params.extra_output_thread) {
             struct DecPicture* copy = malloc(sizeof(struct DecPicture));
             *copy = picture;
             FifoPush(client->pic_fifo, copy, FIFO_EXCEPTION_DISABLE);
            } else {
             PostProcessPicture(client, &picture);
            }
           }

           static void EndOfStreamCb(ClientInst inst)
           {
            struct Client* client = (struct Client*)inst;
            client->eos = 1;
            if (client->test_params.extra_output_thread) {
             /* We're done, wait for the output to Finish it's job. */
             FifoPush(client->pic_fifo, NULL, FIFO_EXCEPTION_DISABLE);
             pthread_join(client->parallel_output_thread, NULL);
            }
            DecRelease(client->decoder);
           }

           static void ReleasedCb(ClientInst inst)
           {
            struct Client* client = (struct Client*)inst;
            for (int i = 0; i < NUM_OF_STREAM_BUFFERS; i++) {
             if (client->buffers[i].buffer.virtual_address) {
              DWLFreeLinear(client->dwl, &client->buffers[i].buffer);
             }
            }
            ReleaseSink(client);
            sem_post(&client->dec_done);
           }
           static void DispatchEndOfStream(struct Client* client) {
            if (!client->eos) {
             client->eos = 1;
             DecEndOfStream(client->decoder);
            }
           }

           static void NotifyErrorCb(ClientInst inst, u32 pic_id, enum DecRet rv)
           {
            struct Client* client = (struct Client*)inst;
            /* There's serious decoding error, so we'll consider it as end of stream to
            get the pending pictures out of the decoder. */
            DispatchEndOfStream(client);
           }
```

```
            int main(int argc, char* argv[])
            {
             struct Client client;
             memset(&client, 0, sizeof(struct Client));
             struct DecClientHandle client_if = { &client, InitializedCb, HeadersDecodedCb,
BufferDecodedCb,
             PictureReadyCb, EndOfStreamCb, ReleasedCb, NotifyErrorCb, };
             SetupDefaultParams(&client.test_params);
             ParseParams(argc, argv, &client.test_params);
             struct DecSwHwBuild build = DecGetBuild();
             /* Check whether the decoder feature meets our requirements here. */
             ...
             client.demuxer.inst = CreateDemuxer(&client);
             /* Create struct DWL. */
             struct DWLInitParam dwl_params = {DWL_CLIENT_TYPE_HEVC_DEC};
             client.dwl = DWLInit(&dwl_params);
             if (client.test_params.extra_output_thread)
             /* Create the fifo to enable parallel output processing */
             FifoInit(2, &client.pic_fifo);
             sem_init(&client.dec_done, 0, 0);
             enum DecCodec codec;
             switch (client.demuxer.GetVideoFormat(client.demuxer.inst)) {
              case BITSTREAM_HEVC:
               codec = DEC_HEVC;
               break;
              case BITSTREAM_VP9:
               codec = DEC_VP9;
               break;
              default:
               return -1;
             }
             struct DecConfig config;
             config.disable_picture_reordering= client.test_params.disable_display_order;
             config.concealment_mode = client.test_params.concealment_mode;
             switch (client.test_params.hw_format) {
              case DEC_OUT_FRM_TILED_4X4:
               config.output_format = DEC_OUT_FRM_TILED_4X4;
               break;
              case DEC_OUT_FRM_RASTER_SCAN: /* fallthrough */
              case DEC_OUT_FRM_PLANAR_420:
               if (!build.hw_config.pp_support) {
                 fprintf(stderr, "Cannot do raster output; No PP support.\n");
                 return -1;
               }
               config.output_format = DEC_OUT_FRM_RASTER_SCAN;
               break;
              default:
               return -1;
             }
             config.dwl = dwl;
             config.dwl_inst = client.dwl;
             config.max_num_pics_to_decode = client.test_params.num_of_decoded_pics;
             config.dscale_cfg = client.test_params.dscale_cfg;
             config.use_video_compressor = client.test_params.compress_bypass ? 0 : 1;
             /* Initialize the decoder. */
             DecInit(codec, &client.decoder, config, client_if) != DEC_OK;
             /* The rest is driven by the callbacks and this thread just has to Wait
             * until decoder has finished its job. */
             sem_wait(&client.dec_done);
             ReleaseDemuxer(&client);
             if (client.pic_fifo != NULL)
               FifoRelease(client.pic_fifo);
             if (client.dwl != NULL) DWLRelease(client.dwl);
               return 0;
            }
```

15.2.5 VPU G2 Memory Map/Register Definition

15.2.5.1 VPU_G2 register descriptions

15.2.5.1.1 VPU Memory map

VPU_G2 base address: 3831_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ID register (read only) (SWREG0)	32	RO	Table 15-4
4h	Interrupt register decoder (SWREG1)	32	RW	0000_0000h
8h	Data configuration register decoder (SWREG2)	32	RW	0000_0000h
Ch	Decoder control register 0 (SWREG3)	32	RW	0000_0000h
10h	Decoder control register 1 (SWREG4)	32	RW	0000_0000h
14h	Decoder control register 2 (SWREG5)	32	RW	0000_0000h
18h	Decoder control register 3 (SWREG6)	32	RW	0000_0000h
1Ch	Decoder control register 4 (SWREG7)	32	RW	0000_0000h
20h	Decoder control register 5 (SWREG8)	32	RW	0000_0000h
24h	Decoder control register 6 (SWREG9)	32	RW	0000_0000h
28h	Decoder control register 7 (SWREG10)	32	RW	0000_0000h
2Ch	Decoder control register 8 (SWREG11)	32	RW	0000_0000h
30h	Decoder control register 9 (SWREG12)	32	RW	0000_0000h
34h	Decoder control register 10 (SWREG13)	32	RW	0000_0000h
38h	Initial ref pic list register (0-2) (SWREG14)	32	RW	0000_0000h
3Ch	Initial ref pic list register (3-5) (SWREG15)	32	RW	0000_0000h
40h	Initial ref pic list register (6-8) (SWREG16)	32	RW	0000_0000h
44h	Initial ref pic list register (9-11) (SWREG17)	32	RW	0000_0000h
48h	Initial ref pic list register (12-14) (SWREG18)	32	RW	0000_0000h
4Ch	Initial ref pic list register (15 and P 0-3) (SWREG19)	32	RW	0000_0000h
50h	Decoder control register 11 (SWREG20)	32	RW	0000_0000h
54h	Not used (SWREG21)	32	RU	0000_0000h
58h	Not used (SWREG22)	32	RU	0000_0000h
5Ch	Decoder configure status register (SWREG23)	32	RO	0000_0000h
60h	Not used (SWREG24)	32	RU	0000_0000h
64h	Not used (SWREG25)	32	RU	0000_0000h
68h	Not used (SWREG26)	32	RU	0000_0000h
6Ch	Not used (SWREG27)	32	RU	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
70h	Not used (SWREG28)	32	RU	0000_0000h
74h	Not used (SWREG29)	32	RU	0000_0000h
78h	Not used (SWREG30)	32	RU	0000_0000h
7Ch	VP9 segmentation values (SWREG31)	32	RW	0000_0000h
80h	VP9 segmentation values (SWREG32)	32	RW	0000_0000h
84h	VP9 reference picture scaling register 0 (SWREG33)	32	RW	0000_0000h
88h	VP9 reference picture scaling register 1 (SWREG34)	32	RW	0000_0000h
8Ch	VP9 reference picture scaling register 2 (SWREG35)	32	RW	0000_0000h
90h	VP9 reference picture scaling register 3 (SWREG36)	32	RW	0000_0000h
94h	VP9 reference picture scaling register 4 (SWREG37)	32	RW	0000_0000h
98h	VP9 reference picture scaling register 5 (SWREG38)	32	RW	0000_0000h
9Ch	Not used (SWREG39)	32	RU	0000_0000h
A0h	Not used (SWREG40)	32	RU	0000_0000h
A4h	Not used (SWREG41)	32	RU	0000_0000h
A8h	Not used (SWREG42)	32	RU	0000_0000h
ACh	Not used (SWREG43)	32	RU	0000_0000h
B0h	Not used (SWREG44)	32	RU	0000_0000h
B4h	Timeout control register (SWREG45)	32	RW	0000_0000h
B8h	Picture order count from current pictures for index 0-3 (SWREG46)	32	RW	0000_0000h
BCh	Picture order count from current pictures for index 4-7 (SWREG47)	32	RW	0000_0000h
C0h	Picture order count from current pictures for index 8-11 (SWREG48)	32	RW	0000_0000h
C4h	Picture order count from current pictures for index 12-15 (SWREG49)	32	RW	0000_0000h
C8h	Synthesis configuration register decoder 0 (read only) (SWREG50)	32	RO	Table 15-4
CCh	Reference picture buffer control register (SWREG51)	32	RU	0000_0000h
D0h	Reference picture buffer information register 1 (read only) (SWREG52)	32	RU	0000_0000h
D4h	Reference picture buffer information register 2 (read only) (SWREG53)	32	RU	0000_0000h
D8h	Synthesis configuration register decoder 1 (read only) (SWREG54)	32	RO	0000_0000h
DCh	Advanced prefetch control register (SWREG55)	32	RW	0000_0000h
E0h	Synthesis configuration register decoder 2 (read only) (SWREG56)	32	RO	0000_0000h
E4h	Decoder fuse register (read only) (SWREG57)	32	RU	0000_0000h
E8h	Device configuration register decoder 2 + Multi core control register (SWREG58)	32	RW	0000_0000h
ECh	Device configuration register AXI ID (SWREG59)	32	RW	0000_0000h
F0h	Synthesis configuration register decoder 3 for PP (read only) (SWREG60)	32	RO	0000_0000h
F4h	Not used (SWREG61)	32	RU	0000_0000h
F8h	HW proceed register (CU location) (SWREG62)	32	RW	0000_0000h
FCh	HW performance register (cycles running) (SWREG63)	32	RO	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
100h	Base address MSB (bits 63:32) for decoded luminance picture (SWREG64)	32	RW	0000_0000h
104h	Base address LSB (bits 31:0) for decoded luminance picture (SWREG65)	32	RW	0000_0000h
108h	Base address MSB (bits 63:32) for reference luminance picture index 0 (SWREG66)	32	RW	0000_0000h
10Ch	Base address LSB (bits 31:0) for reference luminance picture index 0 (SWREG67)	32	RW	0000_0000h
110h	Base address MSB (bits 63:32) for reference luminance picture index 1 (SWREG68)	32	RW	0000_0000h
114h	Base address LSB (bits 31:0) for reference luminance picture index 1 (SWREG69)	32	RW	0000_0000h
118h	Base address MSB (bits 63:32) for reference luminance picture index 2 (SWREG70)	32	RW	0000_0000h
11Ch	Base address LSB (bits 31:0) for reference luminance picture index 2 (SWREG71)	32	RW	0000_0000h
120h	Base address MSB (bits 63:32) for reference luminance picture index 3 (SWREG72)	32	RW	0000_0000h
124h	Base address LSB (bits 31:0) for reference luminance picture index 3 (SWREG73)	32	RW	0000_0000h
128h	Base address MSB (bits 63:32) for reference luminance picture index 4 (SWREG74)	32	RW	0000_0000h
12Ch	Base address LSB (bits 31:0) for reference luminance picture index 4 (SWREG75)	32	RW	0000_0000h
130h	Base address MSB (bits 63:32) for reference luminance picture index 5 (SWREG76)	32	RW	0000_0000h
134h	Base address LSB (bits 31:0) for reference luminance picture index 5 (SWREG77)	32	RW	0000_0000h
138h	Base address MSB (bits 63:32) for reference luminance picture index 6 /VP9 segment write base MSB (SWREG78)	32	RW	0000_0000h
13Ch	Base address LSB (bits 31:0) for reference luminance picture index 6 /VP9 segment write base LSB (SWREG79)	32	RW	0000_0000h
140h	Base address MSB (bits 63:32) for reference luminance picture index 7 /VP9 segment read base MSB (SWREG80)	32	RW	0000_0000h
144h	Base address LSB (bits 31:0) for reference luminance picture index 7 /VP9 segment read base LSB (SWREG81)	32	RW	0000_0000h
148h	Base address MSB (bits 63:32) for reference luminance picture index 8 (SWREG82)	32	RW	0000_0000h
14Ch	Base address LSB (bits 31:0) for reference luminance picture index 8 (SWREG83)	32	RW	0000_0000h
150h	Base address MSB (bits 63:32) for reference luminance picture index 9 (SWREG84)	32	RW	0000_0000h
154h	Base address LSB (bits 31:0) for reference luminance picture index 9 (SWREG85)	32	RW	0000_0000h
158h	Base address MSB (bits 63:32) for reference luminance picture index 10 (SWREG86)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
15Ch	Base address LSB (bits 31:0) for reference luminance picture index 10 (SWREG87)	32	RW	0000_0000h
160h	Base address MSB (bits 63:32) for reference luminance picture index 11 (SWREG88)	32	RW	0000_0000h
164h	Base address LSB (bits 31:0) for reference luminance picture index 11 (SWREG89)	32	RW	0000_0000h
168h	Base address MSB (bits 63:32) for reference luminance picture index 12 (SWREG90)	32	RW	0000_0000h
16Ch	Base address LSB (bits 31:0) for reference luminance picture index 12 (SWREG91)	32	RW	0000_0000h
170h	Base address MSB (bits 63:32) for reference luminance picture index 13 (SWREG92)	32	RW	0000_0000h
174h	Base address LSB (bits 31:0) for reference luminance picture index 13 (SWREG93)	32	RW	0000_0000h
178h	Base address MSB (bits 63:32) for reference luminance picture index 14 (SWREG94)	32	RW	0000_0000h
17Ch	Base address LSB (bits 31:0) for reference luminance picture index 14 (SWREG95)	32	RW	0000_0000h
180h	Base address MSB (bits 63:32) for reference luminance picture index 15 (SWREG96)	32	RW	0000_0000h
184h	Base address LSB (bits 31:0) for reference luminance picture index 15 (SWREG97)	32	RW	0000_0000h
188h	Base address MSB (bits 63:32) for decoded chrominance picture (SWREG98)	32	RW	0000_0000h
18Ch	Base address LSB (bits 31:0) for decoded chrominance picture (SWREG99)	32	RW	0000_0000h
190h	Base address MSB (bits 63:32) for reference chrominance picture index 0 (SWREG100)	32	RW	0000_0000h
194h	Base address LSB (bits 31:0) for reference chrominance picture index 0 (SWREG101)	32	RW	0000_0000h
198h	Base address MSB (bits 63:32) for reference chrominance picture index 1 (SWREG102)	32	RW	0000_0000h
19Ch	Base address LSB (bits 31:0) for reference chrominance picture index 1 (SWREG103)	32	RW	0000_0000h
1A0h	Base address MSB (bits 63:32) for reference chrominance picture index 2 (SWREG104)	32	RW	0000_0000h
1A4h	Base address LSB (bits 31:0) for reference chrominance picture index 2 (SWREG105)	32	RW	0000_0000h
1A8h	Base address MSB (bits 63:32) for reference chrominance picture index 3 (SWREG106)	32	RW	0000_0000h
1ACh	Base address LSB (bits 31:0) for reference chrominance picture index 3 (SWREG107)	32	RW	0000_0000h
1B0h	Base address MSB (bits 63:32) for reference chrominance picture index 4 (SWREG108)	32	RW	0000_0000h
1B4h	Base address LSB (bits 31:0) for reference chrominance picture index 4 (SWREG109)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
1B8h	Base address MSB (bits 63:32) for reference chrominance picture index 5 (SWREG110)	32	RW	0000_0000h
1BCh	Base address LSB (bits 31:0) for reference chrominance picture index 5 (SWREG111)	32	RW	0000_0000h
1C0h	Base address MSB (bits 63:32) for reference chrominance picture index 6 (SWREG112)	32	RW	0000_0000h
1C4h	Base address LSB (bits 31:0) for reference chrominance picture index 6 (SWREG113)	32	RW	0000_0000h
1C8h	Base address MSB (bits 63:32) for reference chrominance picture index 7 (SWREG114)	32	RW	0000_0000h
1CCh	Base address LSB (bits 31:0) for reference chrominance picture index 7 (SWREG115)	32	RW	0000_0000h
1D0h	Base address MSB (bits 63:32) for reference chrominance picture index 8 (SWREG116)	32	RW	0000_0000h
1D4h	Base address LSB (bits 31:0) for reference chrominance picture index 8 (SWREG117)	32	RW	0000_0000h
1D8h	Base address MSB (bits 63:32) for reference chrominance picture index 9 (SWREG118)	32	RW	0000_0000h
1DCh	Base address LSB (bits 31:0) for reference chrominance picture index 9 (SWREG119)	32	RW	0000_0000h
1E0h	Base address MSB (bits 63:32) for reference chrominance picture index 10 (SWREG120)	32	RW	0000_0000h
1E4h	Base address LSB (bits 31:0) for reference chrominance picture index 10 (SWREG121)	32	RW	0000_0000h
1E8h	Base address MSB (bits 63:32) for reference chrominance picture index 11 (SWREG122)	32	RW	0000_0000h
1ECh	Base address LSB (bits 31:0) for reference chrominance picture index 11 (SWREG123)	32	RW	0000_0000h
1F0h	Base address MSB (bits 63:32) for reference chrominance picture index 12 (SWREG124)	32	RW	0000_0000h
1F4h	Base address LSB (bits 31:0) for reference chrominance picture index 12 (SWREG125)	32	RW	0000_0000h
1F8h	Base address MSB (bits 63:32) for reference chrominance picture index 13 (SWREG126)	32	RW	0000_0000h
1FCh	Base address LSB (bits 31:0) for reference chrominance picture index 13 (SWREG127)	32	RW	0000_0000h
200h	Base address MSB (bits 63:32) for reference chrominance picture index 14 (SWREG128)	32	RW	0000_0000h
204h	Base address LSB (bits 31:0) for reference chrominance picture index 14 (SWREG129)	32	RW	0000_0000h
208h	Base address MSB (bits 63:32) for reference chrominance picture index 15 (SWREG130)	32	RW	0000_0000h
20Ch	Base address LSB (bits 31:0) for reference chrominance picture index 15 (SWREG131)	32	RW	0000_0000h
210h	Base address MSB (bits 63:32) for decoded direct mode MVS (SWREG132)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
214h	Base address LSB (bits 31:0) for decoded direct mode MVS (SWREG133)	32	RW	0000_0000h
218h	Base address MSB (bits 63:32) for reference direct mode MVS index 0 (SWREG134)	32	RW	0000_0000h
21Ch	Base address LSB (bits 31:0) for reference direct mode MVS index 0 (SWREG135)	32	RW	0000_0000h
220h	Base address MSB (bits 63:32) for reference direct mode MVS index 1 (SWREG136)	32	RW	0000_0000h
224h	Base address LSB (bits 31:0) for reference direct mode MVS index 1 (SWREG137)	32	RW	0000_0000h
228h	Base address MSB (bits 63:32) for reference direct mode MVS index 2 (SWREG138)	32	RW	0000_0000h
22Ch	Base address LSB (bits 31:0) for reference direct mode MVS index 2 (SWREG139)	32	RW	0000_0000h
230h	Base address MSB (bits 63:32) for reference direct mode MVS index 3 (SWREG140)	32	RW	0000_0000h
234h	Base address LSB (bits 31:0) for reference direct mode MVS index 3 (SWREG141)	32	RW	0000_0000h
238h	Base address MSB (bits 63:32) for reference direct mode MVS index 4 (SWREG142)	32	RW	0000_0000h
23Ch	Base address LSB (bits 31:0) for reference direct mode MVS index 4 (SWREG143)	32	RW	0000_0000h
240h	Base address MSB (bits 63:32) for reference direct mode MVS index 5 (SWREG144)	32	RW	0000_0000h
244h	Base address LSB (bits 31:0) for reference direct mode MVS index 5 (SWREG145)	32	RW	0000_0000h
248h	Base address MSB (bits 63:32) for reference direct mode MVS index 6 (SWREG146)	32	RW	0000_0000h
24Ch	Base address LSB (bits 31:0) for reference direct mode MVS index 6 (SWREG147)	32	RW	0000_0000h
250h	Base address MSB (bits 63:32) for reference direct mode MVS index 7 (SWREG148)	32	RW	0000_0000h
254h	Base address LSB (bits 31:0) for reference direct mode MVS index 7 (SWREG149)	32	RW	0000_0000h
258h	Base address MSB (bits 63:32) for reference direct mode MVS index 8 (SWREG150)	32	RW	0000_0000h
25Ch	Base address LSB (bits 31:0) for reference direct mode MVS index 8 (SWREG151)	32	RW	0000_0000h
260h	Base address MSB (bits 63:32) for reference direct mode mode MVS index 9 (SWREG152)	32	RW	0000_0000h
264h	Base address LSB (bits 31:0) for reference direct mode mode MVS index 9 (SWREG153)	32	RW	0000_0000h
268h	Base address MSB (bits 63:32) for reference direct mode MVS index 10 (SWREG154)	32	RW	0000_0000h
26Ch	Base address LSB (bits 31:0) for reference direct mode MVS index 10 (SWREG155)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
270h	Base address MSB (bits 63:32) for reference direct mode MVS index 11 (SWREG156)	32	RW	0000_0000h
274h	Base address LSB (bits 31:0) for reference direct mode MVS index 11 (SWREG157)	32	RW	0000_0000h
278h	Base address MSB (bits 63:32) for reference direct mode MVS index 12 (SWREG158)	32	RW	0000_0000h
27Ch	Base address LSB (bits 31:0) for reference direct mode MVS index 12 (SWREG159)	32	RW	0000_0000h
280h	Base address MSB (bits 63:32) for reference direct mode MVS index 13 (SWREG160)	32	RW	0000_0000h
284h	Base address LSB (bits 31:0) for reference direct mode MVS index 13 (SWREG161)	32	RW	0000_0000h
288h	Base address MSB (bits 63:32) for reference direct mode MVS index 14 (SWREG162)	32	RW	0000_0000h
28Ch	Base address LSB (bits 31:0) for reference direct mode MVS index 14 (SWREG163)	32	RW	0000_0000h
290h	Base address MSB (bits 63:32) for reference direct mode MVS index 15 (SWREG164)	32	RW	0000_0000h
294h	Base address LSB (bits 31:0) for reference direct mode MVS index 15 (SWREG165)	32	RW	0000_0000h
298h	Base address MSB (bits 63:32) for tile sizes (SWREG166)	32	RW	0000_0000h
29Ch	Base address LSB (bits 31:0) for tile sizes (SWREG167)	32	RW	0000_0000h
2A0h	Base address MSB (bits 63:32) for / stream start address/decoded end addr register (SWREG168)	32	RW	0000_0000h
2A4h	Base address LSB (bits 31:0) for / stream start address/decoded end addr register (SWREG169)	32	RW	0000_0000h
2A8h	Base address MSB (bits 63:32) for scaling lists / VP9 CTX counter values (SWREG170)	32	RW	0000_0000h
2ACh	Base address LSB (bits 31:0) for scaling lists / VP9 CTX counter values (SWREG171)	32	RW	0000_0000h
2B0h	Base address MSB (bits 63:32) for stream propability tables (SWREG172)	32	RW	0000_0000h
2B4h	Base address LSB (bits 31:0) for stream propability tables (SWREG173)	32	RW	0000_0000h
2B8h	Base address MSB (bits 63:32) for decoder output raster scan Y picture (SWREG174)	32	RW	0000_0000h
2BCh	Base address LSB (bits 31:0) for decoder output raster scan Y picture (SWREG175)	32	RW	0000_0000h
2C0h	Base address MSB (bits 63:32) for decoder output raster scan C picture (SWREG176)	32	RW	0000_0000h
2C4h	Base address LSB (bits 31:0) for decoder output raster scan C picture (SWREG177)	32	RW	0000_0000h
2C8h	Base address MSB (bits 63:32) for tile border coeffients of filter (SWREG178)	32	RW	0000_0000h
2CCh	Base address LSB (bits 31:0) for tile border coeffients of filter (SWREG179)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
2D0h	Base address MSB (bits 63:32) for tile border coeffients of sao (SWREG180)	32	RW	0000_0000h
2D4h	Base address LSB (bits 31:0) for tile border coeffients of sao (SWREG181)	32	RW	0000_0000h
2D8h	Base address MSB (bits 63:32) for tile border bsd control data (SWREG182)	32	RW	0000_0000h
2DCh	Base address LSB (bits 31:0) for tile border bsd control data (SWREG183)	32	RW	0000_0000h
2E0h	Raster scan down scale control register MSM (SWREG184)	32	RW	0000_0000h
2E4h	Base address MSB (bits 63:32) for decoder output raster scan down scale Y picture (SWREG185)	32	RW	0000_0000h
2E8h	Base address LSB (bits 31:0) for decoder output raster scan down scale Y picture (SWREG186)	32	RW	0000_0000h
2ECh	Base address MSB (bits 63:32) for decoder output raster scan down scale C picture (SWREG187)	32	RW	0000_0000h
2F0h	Base address LSB (bits 31:0) for decoder output raster scan down scale C picture (SWREG188)	32	RW	0000_0000h
2F4h	Base address MSB (bits 63:32) for decoder output compress luminance table (SWREG189)	32	RW	0000_0000h
2F8h	Base address LSB (bits 31:0) for decoder output compress luminance table (SWREG190)	32	RW	0000_0000h
2FCh	Base address MSB (bits 63:32) for reference compress luminance table index 0 (SWREG191)	32	RW	0000_0000h
300h	Base address LSB (bits 31:0) for reference compress luminance table index 0 (SWREG192)	32	RW	0000_0000h
304h	Base address MSB (bits 63:32) for reference compress luminance table index 1 (SWREG193)	32	RW	0000_0000h
308h	Base address LSB (bits 31:0) for reference compress luminance table index 1 (SWREG194)	32	RW	0000_0000h
30Ch	Base address MSB (bits 63:32) for reference compress luminance table index 2 (SWREG195)	32	RW	0000_0000h
310h	Base address LSB (bits 31:0) for reference compress luminance table index 2 (SWREG196)	32	RW	0000_0000h
314h	Base address MSB (bits 63:32) for reference compress luminance table index 3 (SWREG197)	32	RW	0000_0000h
318h	Base address LSB (bits 31:0) for reference compress luminance table index 3 (SWREG198)	32	RW	0000_0000h
31Ch	Base address MSB (bits 63:32) for reference compress luminance table index 4 (SWREG199)	32	RW	0000_0000h
320h	Base address LSB (bits 31:0) for reference compress luminance table index 4 (SWREG200)	32	RW	0000_0000h
324h	Base address MSB (bits 63:32) for reference compress luminance table index 5 (SWREG201)	32	RW	0000_0000h
328h	Base address LSB (bits 31:0) for reference compress luminance table index 5 (SWREG202)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
32Ch	Base address MSB (bits 63:32) for reference compress luminance table index 6 (SWREG203)	32	RW	0000_0000h
330h	Base address LSB (bits 31:0) for reference compress luminance table index 6 (SWREG204)	32	RW	0000_0000h
334h	Base address MSB (bits 63:32) for reference compress luminance table index 7 (SWREG205)	32	RW	0000_0000h
338h	Base address LSB (bits 31:0) for reference compress luminance table index 7 (SWREG206)	32	RW	0000_0000h
33Ch	Base address MSB (bits 63:32) for reference compress luminance table index 8 (SWREG207)	32	RW	0000_0000h
340h	Base address LSB (bits 31:0) for reference compress luminance table index 8 (SWREG208)	32	RW	0000_0000h
344h	Base address MSB (bits 63:32) for reference compress luminance table index 9 (SWREG209)	32	RW	0000_0000h
348h	Base address LSB (bits 31:0) for reference compress luminance table index 9 (SWREG210)	32	RW	0000_0000h
34Ch	Base address MSB (bits 63:32) for reference compress luminance table index 10 (SWREG211)	32	RW	0000_0000h
350h	Base address LSB (bits 31:0) for reference compress luminance table index 10 (SWREG212)	32	RW	0000_0000h
354h	Base address MSB (bits 63:32) for reference compress luminance table index 11 (SWREG213)	32	RW	0000_0000h
358h	Base address LSB (bits 31:0) for reference compress luminance table index 11 (SWREG214)	32	RW	0000_0000h
35Ch	Base address MSB (bits 63:32) for reference compress luminance table index 12 (SWREG215)	32	RW	0000_0000h
360h	Base address LSB (bits 31:0) for reference compress luminance table index 12 (SWREG216)	32	RW	0000_0000h
364h	Base address MSB (bits 63:32) for reference compress luminance table index 13 (SWREG217)	32	RW	0000_0000h
368h	Base address LSB (bits 31:0) for reference compress luminance table index 13 (SWREG218)	32	RW	0000_0000h
36Ch	Base address MSB (bits 63:32) for reference compress luminance table index 14 (SWREG219)	32	RW	0000_0000h
370h	Base address LSB (bits 31:0) for reference compress luminance table index 14 (SWREG220)	32	RW	0000_0000h
374h	Base address MSB (bits 63:32) for reference compress luminance table index 15 (SWREG221)	32	RW	0000_0000h
378h	Base address LSB (bits 31:0) for reference compress luminance table index 15 (SWREG222)	32	RW	0000_0000h
37Ch	Base address MSB (bits 63:32) for decoder output compress chrominance table (SWREG223)	32	RW	0000_0000h
380h	Base address LSB (bits 31:0) for decoder output compress chrominance table (SWREG224)	32	RW	0000_0000h
384h	Base address MSB (bits 63:32) for reference compress chrominance table index 0 (SWREG225)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
388h	Base address LSB (bits 31:0) for reference compress chrominance table index 0 (SWREG226)	32	RW	0000_0000h
38Ch	Base address MSB (bits 63:32) for reference compress chrominance table index 1 (SWREG227)	32	RW	0000_0000h
390h	Base address LSB (bits 31:0) for reference compress chrominance table index 1 (SWREG228)	32	RW	0000_0000h
394h	Base address MSB (bits 63:32) for reference compress chrominance table index 2 (SWREG229)	32	RW	0000_0000h
398h	Base address LSB (bits 31:0) for reference compress chrominance table index 2 (SWREG230)	32	RW	0000_0000h
39Ch	Base address MSB (bits 63:32) for reference compress chrominance table index 3 (SWREG231)	32	RW	0000_0000h
3A0h	Base address LSB (bits 31:0) for reference compress chrominance table index 3 (SWREG232)	32	RW	0000_0000h
3A4h	Base address MSB (bits 63:32) for reference compress chrominance table index 4 (SWREG233)	32	RW	0000_0000h
3A8h	Base address LSB (bits 31:0) for reference compress chrominance table index 4 (SWREG234)	32	RW	0000_0000h
3ACh	Base address MSB (bits 63:32) for reference compress chrominance table index 5 (SWREG235)	32	RW	0000_0000h
3B0h	Base address LSB (bits 31:0) for reference compress chrominance table index 5 (SWREG236)	32	RW	0000_0000h
3B4h	Base address MSB (bits 63:32) for reference compress chrominance table index 6 (SWREG237)	32	RW	0000_0000h
3B8h	Base address LSB (bits 31:0) for reference compress chrominance table index 6 (SWREG238)	32	RW	0000_0000h
3BCh	Base address MSB (bits 63:32) for reference compress chrominance table index 7 (SWREG239)	32	RW	0000_0000h
3C0h	Base address LSB (bits 31:0) for reference compress chrominance table index 7 (SWREG240)	32	RW	0000_0000h
3C4h	Base address MSB (bits 63:32) for reference compress chrominance table index 8 (SWREG241)	32	RW	0000_0000h
3C8h	Base address LSB (bits 31:0) for reference compress chrominance table index 8 (SWREG242)	32	RW	0000_0000h
3CCh	Base address MSB (bits 63:32) for reference compress chrominance table index 9 (SWREG243)	32	RW	0000_0000h
3D0h	Base address LSB (bits 31:0) for reference compress chrominance table index 9 (SWREG244)	32	RW	0000_0000h
3D4h	Base address MSB (bits 63:32) for reference compress chrominance table index 10 (SWREG245)	32	RW	0000_0000h
3D8h	Base address LSB (bits 31:0) for reference compress chrominance table index 10 (SWREG246)	32	RW	0000_0000h
3DCh	Base address MSB (bits 63:32) for reference compress chrominance table index 11 (SWREG247)	32	RW	0000_0000h
3E0h	Base address LSB (bits 31:0) for reference compress chrominance table index 11 (SWREG248)	32	RW	0000_0000h

Table continues on the next page...

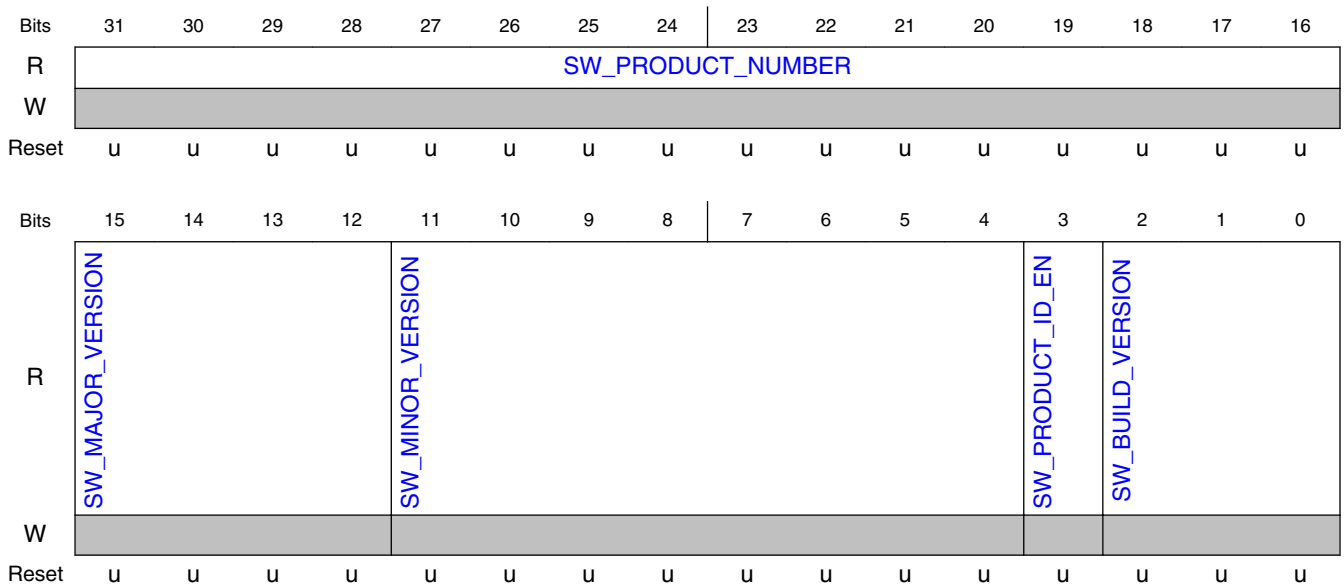
Offset	Register	Width (In bits)	Access	Reset value
3E4h	Base address MSB (bits 63:32) for reference compress chrominance table index 12 (SWREG249)	32	RW	0000_0000h
3E8h	Base address LSB (bits 31:0) for reference compress chrominance table index 12 (SWREG250)	32	RW	0000_0000h
3ECh	Base address MSB (bits 63:32) for reference compress chrominance table index 13 (SWREG251)	32	RW	0000_0000h
3F0h	Base address LSB (bits 31:0) for reference compress chrominance table index 13 (SWREG252)	32	RW	0000_0000h
3F4h	Base address MSB (bits 63:32) for reference compress chrominance table index 14 (SWREG253)	32	RW	0000_0000h
3F8h	Base address LSB (bits 31:0) for reference compress chrominance table index 14 (SWREG254)	32	RW	0000_0000h
3FCh	Base address MSB (bits 63:32) for reference compress chrominance table index 15 (SWREG255)	32	RW	0000_0000h
400h	Base address LSB (bits 31:0) for reference compress chrominance table index 15 (SWREG256)	32	RW	0000_0000h
404h	Not used (SWREG257)	32	RU	0000_0000h
408h	input stream buffer length (SWREG258)	32	RW	0000_0000h
40Ch	input stream buffer start offset (SWREG259)	32	RW	0000_0000h

15.2.5.1.2 ID register (read only) (SWREG0)

15.2.5.1.2.1 Offset

Register	Offset
SWREG0	0h

15.2.5.1.2.2 Diagram



15.2.5.1.2.3 Fields

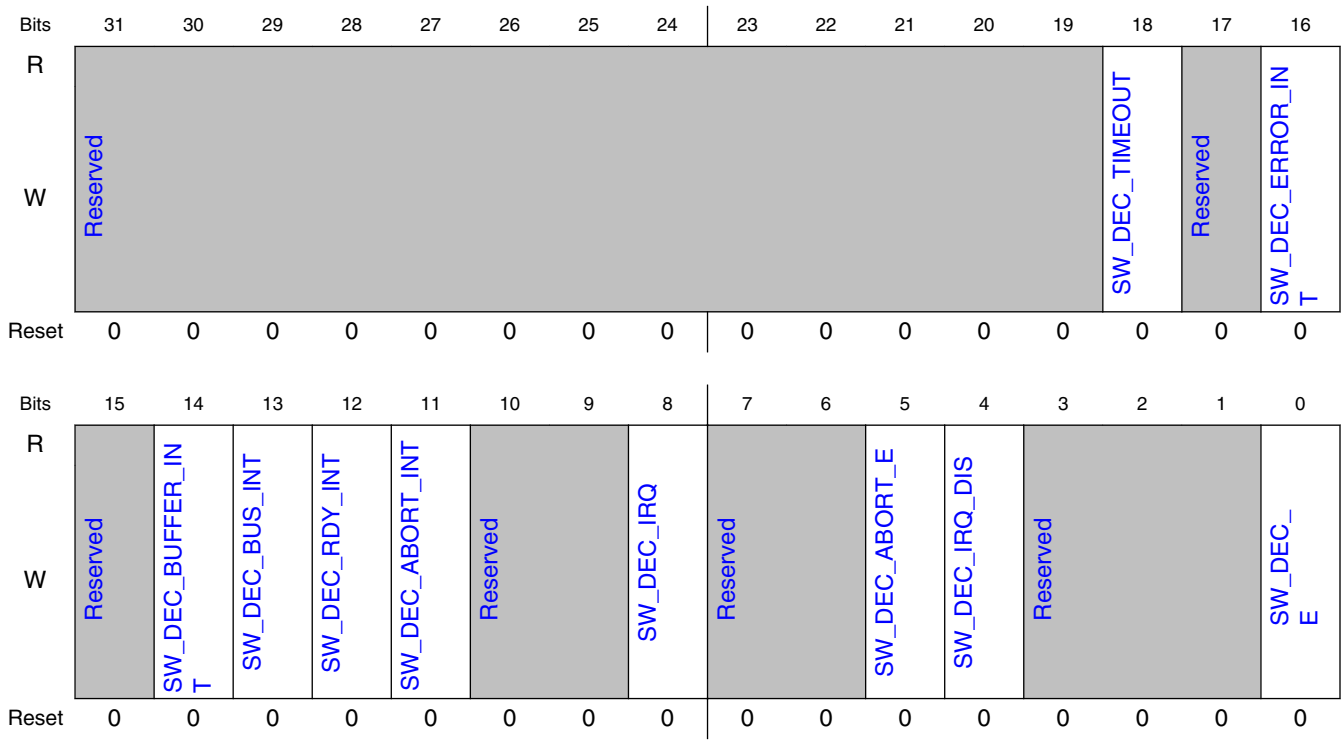
Field	Function
31-16 SW_PRODUCT_NUMBER	Product number (g2)
15-12 SW_MAJOR_VERSION	Major version
11-4 SW_MINOR_VERSION	Minor version
3 SW_PRODUCT_ID_EN	ASCII type product ID enable
2-0 SW_BUILD_VERSION	Build version (core number)

15.2.5.1.3 Interrupt register decoder (SWREG1)

15.2.5.1.3.1 Offset

Register	Offset
SWREG1	4h

15.2.5.1.3.2 Diagram



15.2.5.1.3.3 Fields

Field	Function
31-19 —	Reserved.
18 SW_DEC_TIMEOUT	Interrupt status bit decoder timeout. When high the decoder has been idling for too long. HW will self reset. Possible only if timeout interrupt is enabled
17 —	Reserved.
16 SW_DEC_ERROR_INT	Interrupt status bit input stream error. When high an error is found in input data stream decoding. HW will self reset.
15	Reserved.

Table continues on the next page...

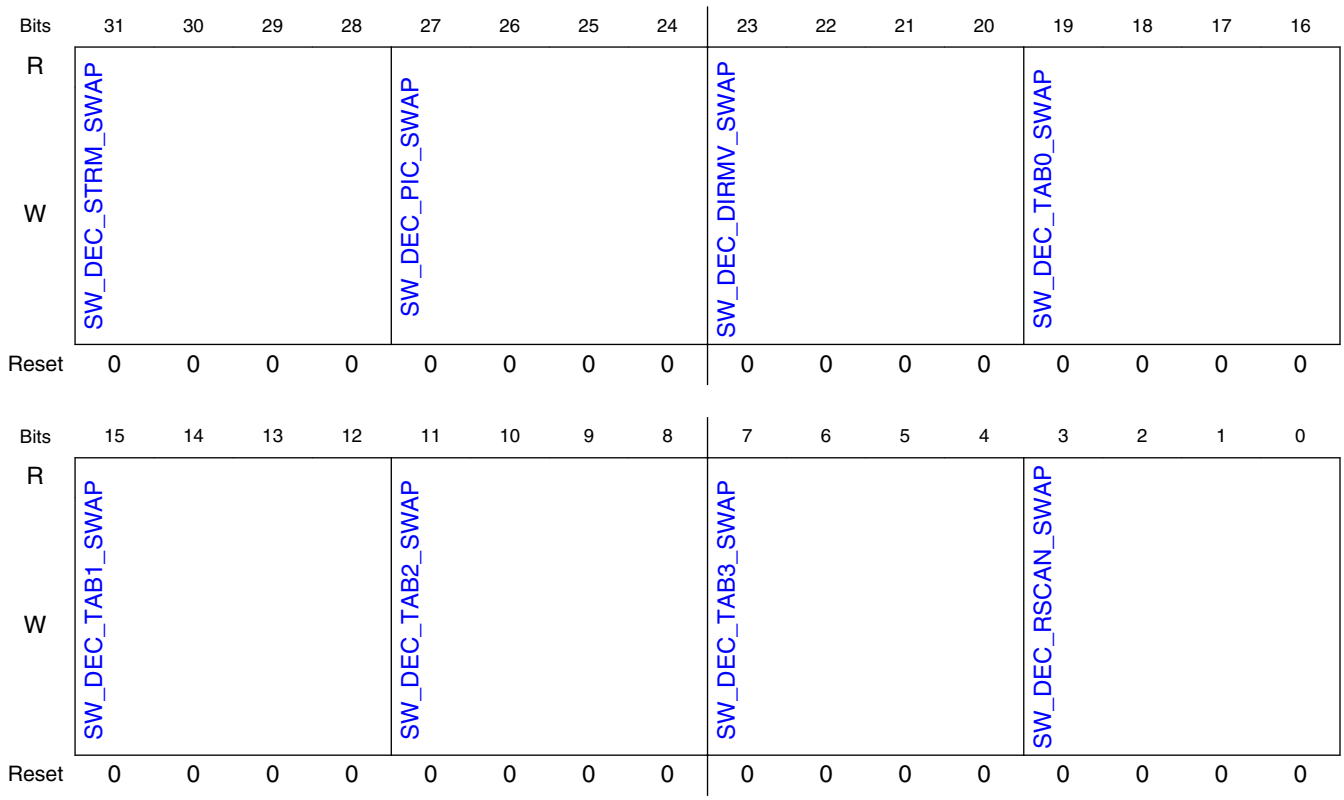
Field	Function
—	
14 SW_DEC_BUFFER_INT	Interrupt status bit input buffer empty. When high input stream buffer is empty but picture is not ready. HW will not self reset.
13 SW_DEC_BUS_INT	Interrupt status bit bus. Error response from bus. HW will self reset.
12 SW_DEC_RDY_INT	Interrupt status bit decoder. When this bit is high decoder has decoded a picture. HW will self reset.
11 SW_DEC_ABORT_INT	Interrupt status bit decoding aborted. When this bit is high decoder has aborted the current picture decoding as SW requested (sw_dec_abort_e). Decoder self reset and sw_dec_abort_e written low
10-9 —	Reserved.
8 SW_DEC_IRQ	Decoder IRQ. When high decoder requests an interrupt. SW will reset this after interrupt is handled.
7-6 —	Reserved.
5 SW_DEC_ABORT_E	Abort decoding enable. Setting this bit high will cause HW to abort decoding and safely to reset itself down. After abort is complete the corresponding interrupt status is set and this bit is set low as well as the decoder enable.
4 SW_DEC_IRQ_DIS	Decoder IRQ disable. When high there are no interrupts concerning decoder from HW. Polling must be used to see the interrupt statuses.
3-1 —	Reserved.
0 SW_DEC_E	Decoder enable. Setting this bit high will start the decoding operation. HW will reset this when picture is processed or ASO or stream error is detected or bus error or timeout interrupt is given.

15.2.5.1.4 Data configuration register decoder (SWREG2)

15.2.5.1.4.1 Offset

Register	Offset
SWREG2	8h

15.2.5.1.4.2 Diagram



15.2.5.1.4.3 Fields

Field	Function
31-28 SW_DEC_STRM_SWAP	Byte swap configuration for stream data 4 Bit byte order vector to control byte locations inside HW internal 128 bit data vector. For 64 and 32 bit external bus widths, the data is first gathered to 128 bit width and then bytes swapped accordingly: Bit 0: 8 bit swap Bit 1: 16 bit swap Bit 2: 32 bit swap Bit 3: 64 bit swap '0000' = 0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15 '0001' = 1-0-3-2-5-4-7-6... '0010' = 2-3-0-1-6-7-4-5... '0011' = 3-2-1-0-7-6-5-4... '0100' = 4-5-6-7-0-1-2-3... ... '0111' = 7-6-5-4-3-2-1-0 '1000' = 8-9-10-11-12-13-14-15-0-1-2-3-4-5-6-7... ...

Table continues on the next page...

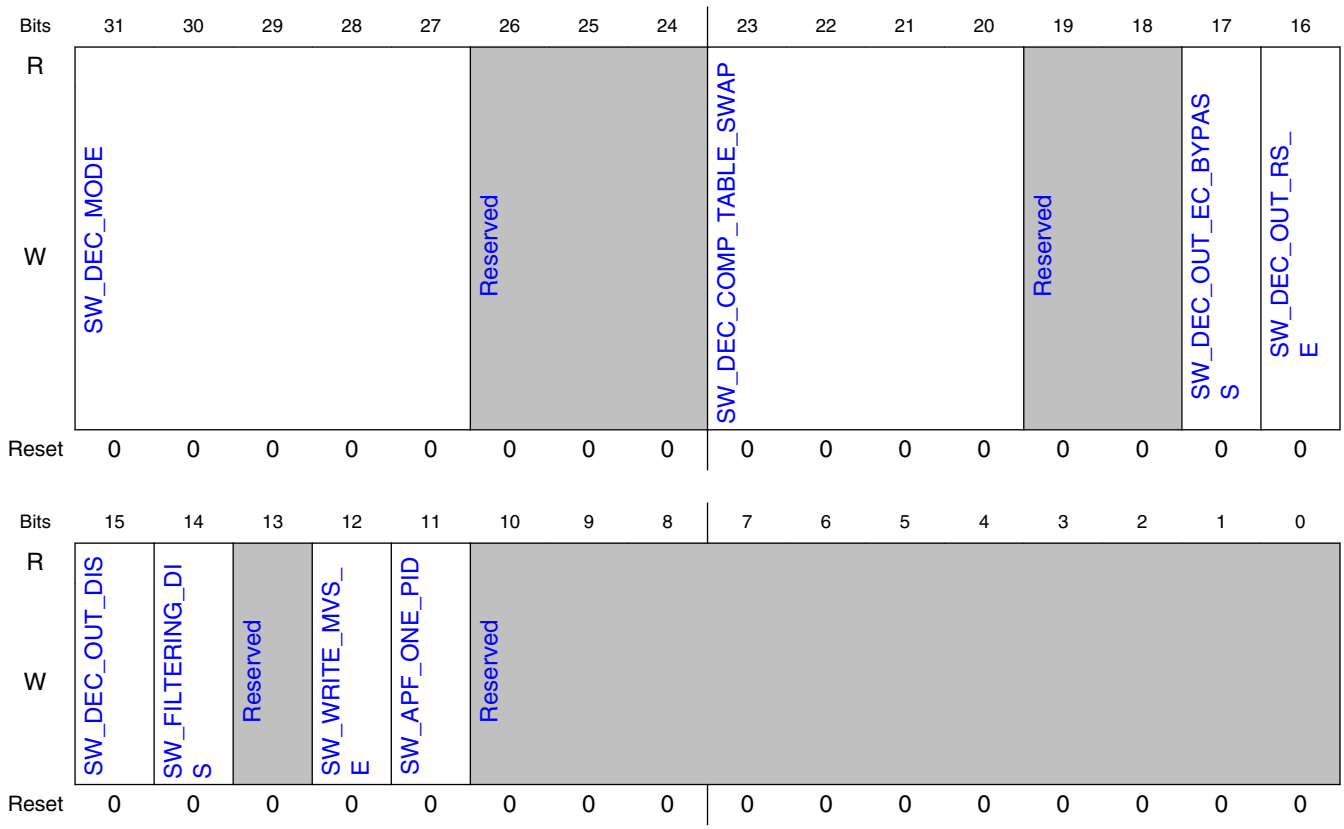
Field	Function
	'1111'=15-14-13-12-11-10-9-8-7-6-5-4-3-2-1-0
27-24 SW_DEC_PIC_SWAP	Byte swap configuration for decoder reference output picture data
23-20 SW_DEC_DIRMV_SWAP	Byte swap configuration for direct mode MV data (read/write)
19-16 SW_DEC_TAB0_SWAP	Byte swap configuration for VP9 stream propability tables
15-12 SW_DEC_TAB1_SWAP	Byte swap configuration for HEVC scaling lists / VP9 segmentation map read/write
11-8 SW_DEC_TAB2_SWAP	Byte swap configuration for VP9 CTX counter values
7-4 SW_DEC_TAB3_SWAP	Byte swap configuration for tile sizes
3-0 SW_DEC_RSCAN_SWAP	Byte swap for raster scan output picture data

15.2.5.1.5 Decoder control register 0 (SWREG3)

15.2.5.1.5.1 Offset

Register	Offset
SWREG3	Ch

15.2.5.1.5.2 Diagram



15.2.5.1.5.3 Fields

Field	Function
31-27 SW_DEC_MODE	Decoding mode: 00000-01011b - Reserved 01100b - HEVC 01101b - VP9 01110-11111b - Reserved
26-24 —	Reserved.
23-20 SW_DEC_COMP_TABLE_SWAP	Byte swap configuration for compress table data
19-18 —	Reserved.
17 SW_DEC_OUT_EC_BYPASS	Compress bypass

Table continues on the next page...

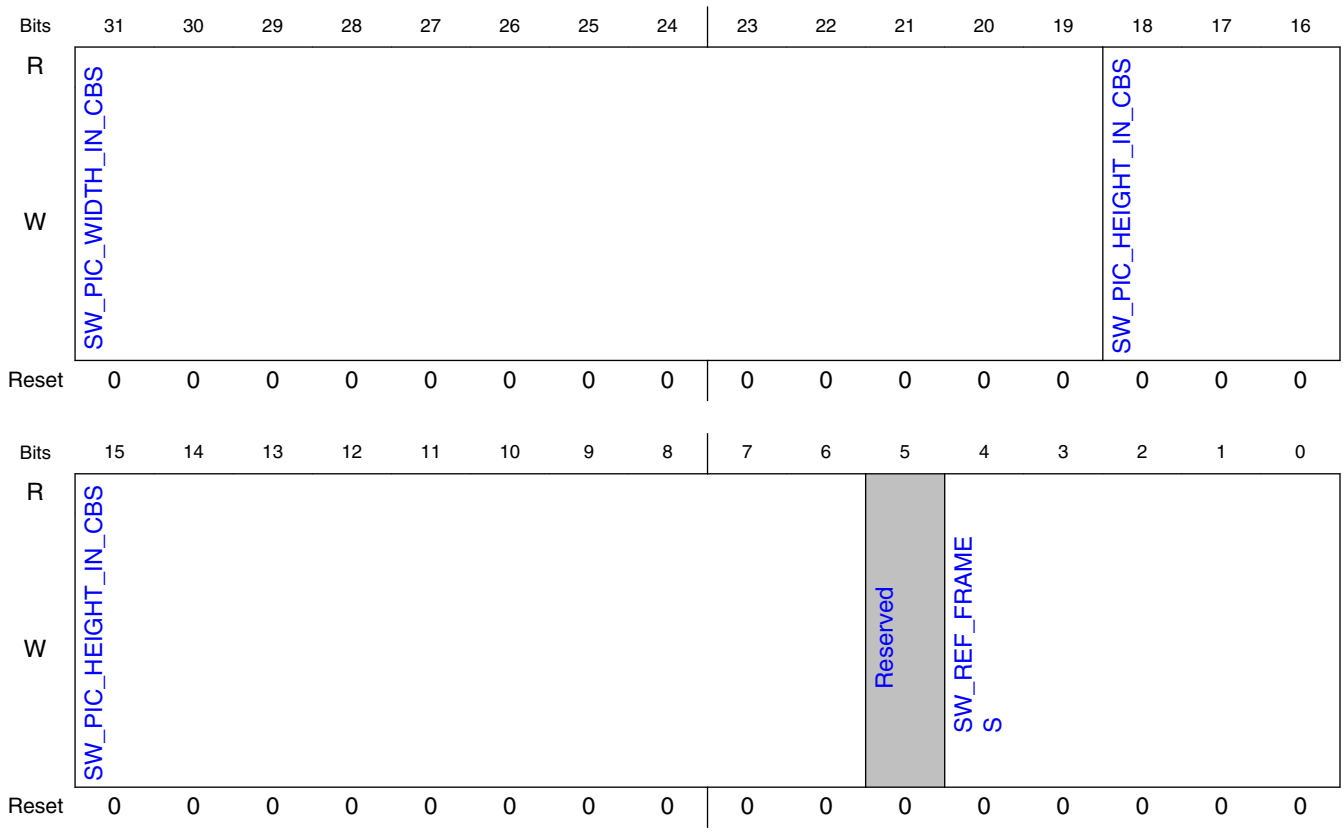
Field	Function
16 SW_DEC_OUT_RS_E	Raster scan output enable. If high decoder writes the raster scan output if the configuration of Decoder includes PP raster scan output
15 SW_DEC_OUT_DIS	Disable decoder output picture writing 0b - Decoder output picture is written to external memory 1b - Decoder output picture is not written to external memory
14 SW_FILTERING_DIS	De-block filtering disable 0b - Filtering is enabled for current picture 1b - Filtering is disabled for current picture
13 —	Reserved.
12 SW_WRITE_MVS_E	Direct mode motion vector write enable for current picture 0b - Writing disabled for current picture. 1b - The direct mode motion vectors are written to external memory. HEVC/VP9 direct mode motion vectors are written to DPB aside with the corresponding reference picture. Other decoding mode dir mode mvs are written to external memory starting from sw_dir_mv_base.
11 SW_APF_ONE_PID	Prefetch partitions that have the same pic_id together
10-0 —	Reserved.

15.2.5.1.6 Decoder control register 1 (SWREG4)

15.2.5.1.6.1 Offset

Register	Offset
SWREG4	10h

15.2.5.1.6.2 Diagram



15.2.5.1.6.3 Fields

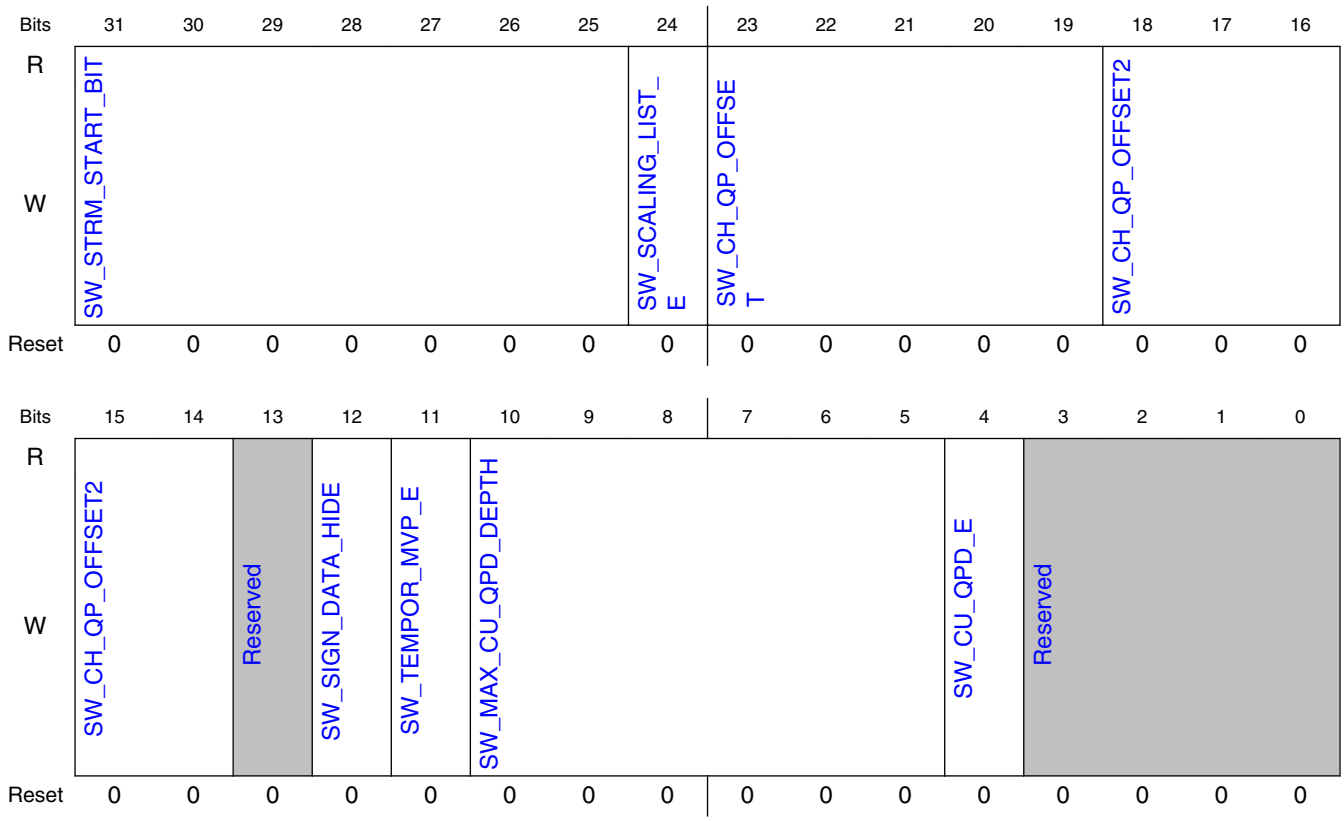
Field	Function
31-19 SW_PIC_WIDTH_IN_CBS	Picture width in min coded blocks (min = 8pix)
18-6 SW_PIC_HEIGHT_IN_CBS	Picture height in min coded blocks (min = 8pix)
5 —	Reserved.
4-0 SW_REF_FRAMES	HEVC: num_ref_frames maximum number of short and long term reference frames in decoded picture buffer

15.2.5.1.7 Decoder control register 2 (SWREG5)

15.2.5.1.7.1 Offset

Register	Offset
SWREG5	14h

15.2.5.1.7.2 Diagram



15.2.5.1.7.3 Fields

Field	Function
31-25 SW_STRM_START_BIT	Exact bit of stream start word where decoding can be started (assosiates with sw_rlc_vlc_base)
24 SW_SCALING_LIST_E	Scaling matrix enable 0b - Normal transform 1b - Use scaling matrix for transform (read from external memory)
23-19 SW_CH_QP_OFFSET	Chroma Qp filter offset. (For HEVC this offset concerns Cb only)

Table continues on the next page...

Field	Function
18-14 SW_CH_QP_OFFSET2	Chroma Qp filter offset for cr type
13 —	Reserved.
12 SW_SIGN_DATA_HIDE	Flag for stream decoding
11 SW_TEMPOR_MVP_E	Temporal mvp enable
10-5 SW_MAX_CU_QPD_DEPTH	Max CU qp delta depth
4 SW_CU_QPD_E	CU qp delta enable
3-0 —	Reserved.

15.2.5.1.8 Decoder control register 3 (SWREG6)

15.2.5.1.8.1 Offset

Register	Offset
SWREG6	18h

15.2.5.1.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_STREAM_LEN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_STREAM_LEN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.8.3 Fields

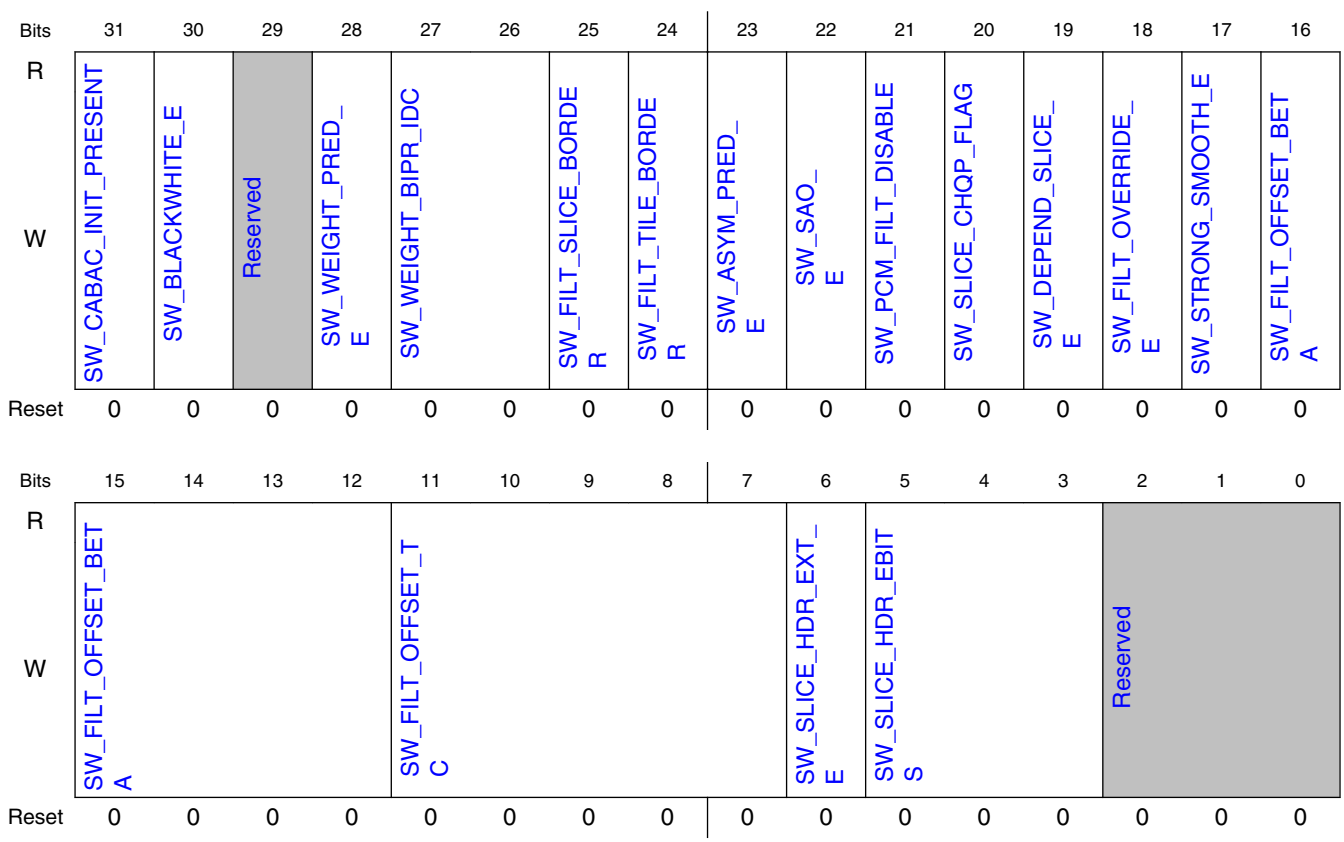
Field	Function
31-0 SW_STREAM_LEN	Amount of stream data bytes in input buffer. If the given buffer size is not enough for finishing the picture the corresponding interrupt is given and new stream buffer base address and stream buffer size information should be given (assosiates with sw_rlc_vlc_base). For HEVC the buffer must include at least data for one slice/VP of the picture

15.2.5.1.9 Decoder control register 4 (SWREG7)

15.2.5.1.9.1 Offset

Register	Offset
SWREG7	1Ch

15.2.5.1.9.2 Diagram



15.2.5.1.9.3 Fields

Field	Function
31 SW_CABAC_INIT_PRESENT	CABAC init present enable for stream decoding
30 SW_BLACKWHITE_E	Sampling 0b - 4:2:0 sampling format 1b - 4:0:0 sampling format (H264 monochroma)
29 —	Reserved.
28 SW_WEIGHT_PRED_E	Weighted prediction enable for P slices
27-26 SW_WEIGHT_BIPR_IDC	Weighted prediction specification 00b - Default weighted prediction is applied to B slices 01b - Explicit weighted prediction shall be applied to B slices 10b - NA 11b - NA
25 SW_FILT_SLICE_BORDER	Filter enable over slice border
24 SW_FILT_TILE_BORDER	Filter enable over tile border
23 SW_ASYM_PRED_E	Asymmetric prediction flag for stream decoding
22 SW_SAO_E	Sample Adaptive Offset enable for stream decoding
21 SW_PCM_FILT_DISABLE	Disable for PCM loop filtering
20 SW_SLICE_CHQP_FLAG	Slice header flag for chroma QP present (if it is included in slice header)
19 SW_DEPEND_SLICE_E	Dependent slice enable
18 SW_FILT_OVERRIDE_E	Filter override enable
17 SW_STRONG_SMOOTH_E	Strong smoothing enable
16-12	Filter beta offset (declared as div2)

Table continues on the next page...

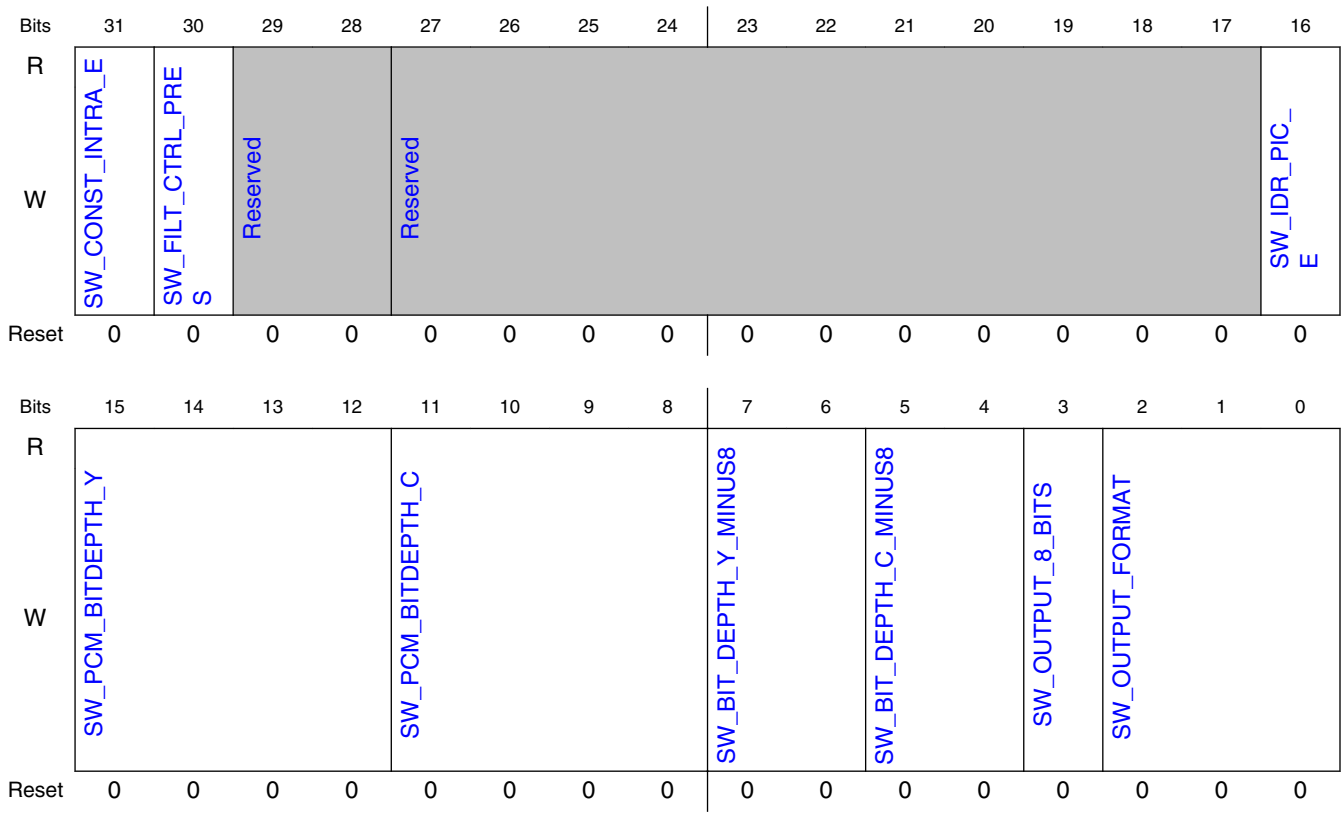
Field	Function
SW_FILT_OFFSET_BETA	
11-7 SW_FILT_OFFSET_TC	Filter tc offset (declared as div2)
6 SW_SLICE_HDR_EXT_E	Slice header extension enable. Reserved for future use
5-3 SW_SLICE_HDR_EBITS	Number of extra slice header bits (if enabled slice header extension)
2-0 —	Reserved.

15.2.5.1.10 Decoder control register 5 (SWREG8)

15.2.5.1.10.1 Offset

Register	Offset
SWREG8	20h

15.2.5.1.10.2 Diagram



15.2.5.1.10.3 Fields

Field	Function
31 SW_CONST_INTRA_E	constrained_intra_pred_flag equal to 1 specifies that intra prediction uses only neighbouring intra macroblocks in prediction. When equal to 0 also neighbouring inter macroblocks are used in intra prediction process.
30 SW_FILT_CTRL_PRES	deblocking_filter_control_present_flag indicates whether extra variables controlling characteristics of the deblocking filter are present in the slice header.
29-28 —	Reserved.
27-17 —	Reserved.
16 SW_IDR_PIC_E	IDR (instantaneous decoding refresh) picture flag.
15-12 SW_PCM_BITDEPTH_Y	Bit depth for PCM Y data
11-8	Bit depth for PCM C data

Table continues on the next page...

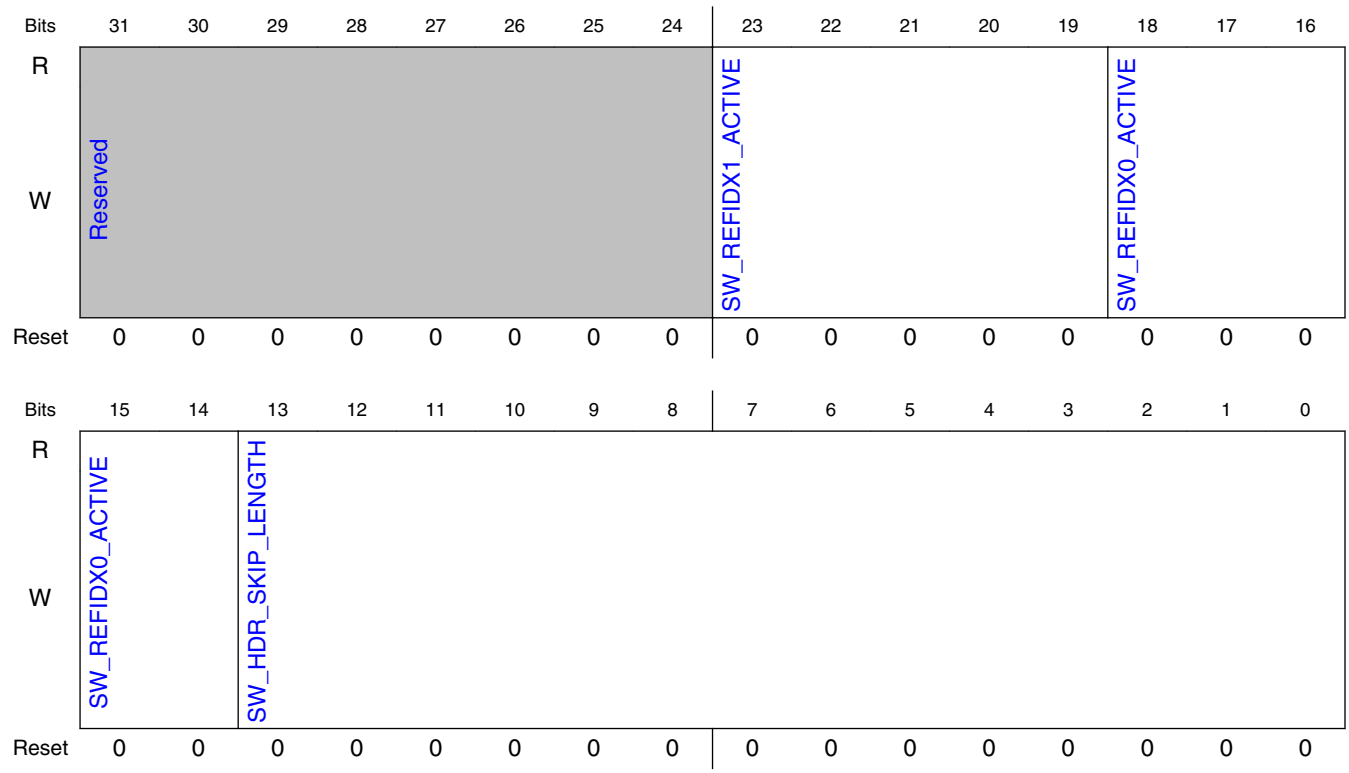
Field	Function
SW_PCM_BITDEPTH_C	
7-6 SW_BIT_DEPTH_Y_MINUS8	Bit depth of luma samples minus 8
5-4 SW_BIT_DEPTH_C_MINUS8	Bit depth of chroma samples minus 8
3 SW_OUTPUT_8_BITS	enable rasterscan output force to 8 bit(only for hevc main10 and vp9 10bit)
2-0 SW_OUTPUT_FORMAT	Raster scan and down scale output data format 000b - Each pixel in 10 bits when luma or chroma pixel bit depth is larger than 8; or 8 bits when both luma and chroma pixel bit depth are 8 bits. (default) 001b - Store in P010 format when luma or chroma pixel bit depth is larger than 8. 010b - A customized format: please refer to register SWREG23[6].

15.2.5.1.11 Decoder control register 6 (SWREG9)

15.2.5.1.11.1 Offset

Register	Offset
SWREG9	24h

15.2.5.1.11.2 Diagram



15.2.5.1.11.3 Fields

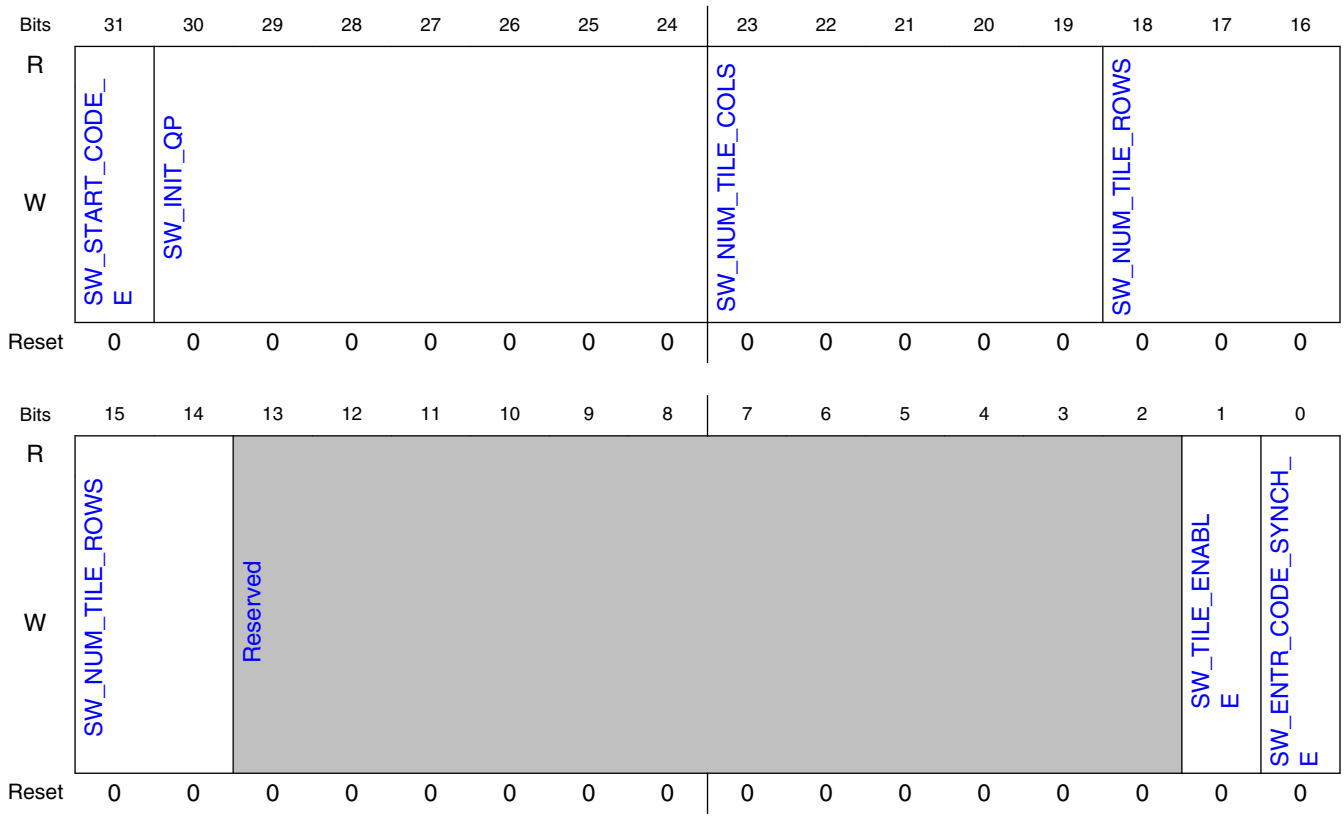
Field	Function
31-24 —	Reserved.
23-19 SW_REFIDX1_ACTIVE	Specifies the maximum reference index that can be used while decoding inter predicted macro blocks.
18-14 SW_REFIDX0_ACTIVE	Specifies the maximum reference index that can be used while decoding inter predicted macro blocks. This is same as in previous decoders (width increased with q bit)
13-0 SW_HDR_SKIP_LENGTH	Length of slice header skip length (bytes used by sw)

15.2.5.1.12 Decoder control register 7 (SWREG10)

15.2.5.1.12.1 Offset

Register	Offset
SWREG10	28h

15.2.5.1.12.2 Diagram



15.2.5.1.12.3 Fields

Field	Function
31 SW_START_CODE_E	Bit for indicating stream start code existence 0b - Stream does not contain start codes 1b - Stream contains start codes
30-24 SW_INIT_QP	Initial value for quantization parameter (picture quantizer).
23-19 SW_NUM_TILE_COLS	Number of tile columns in picture
18-14	Number of tile rows in picture

Table continues on the next page...

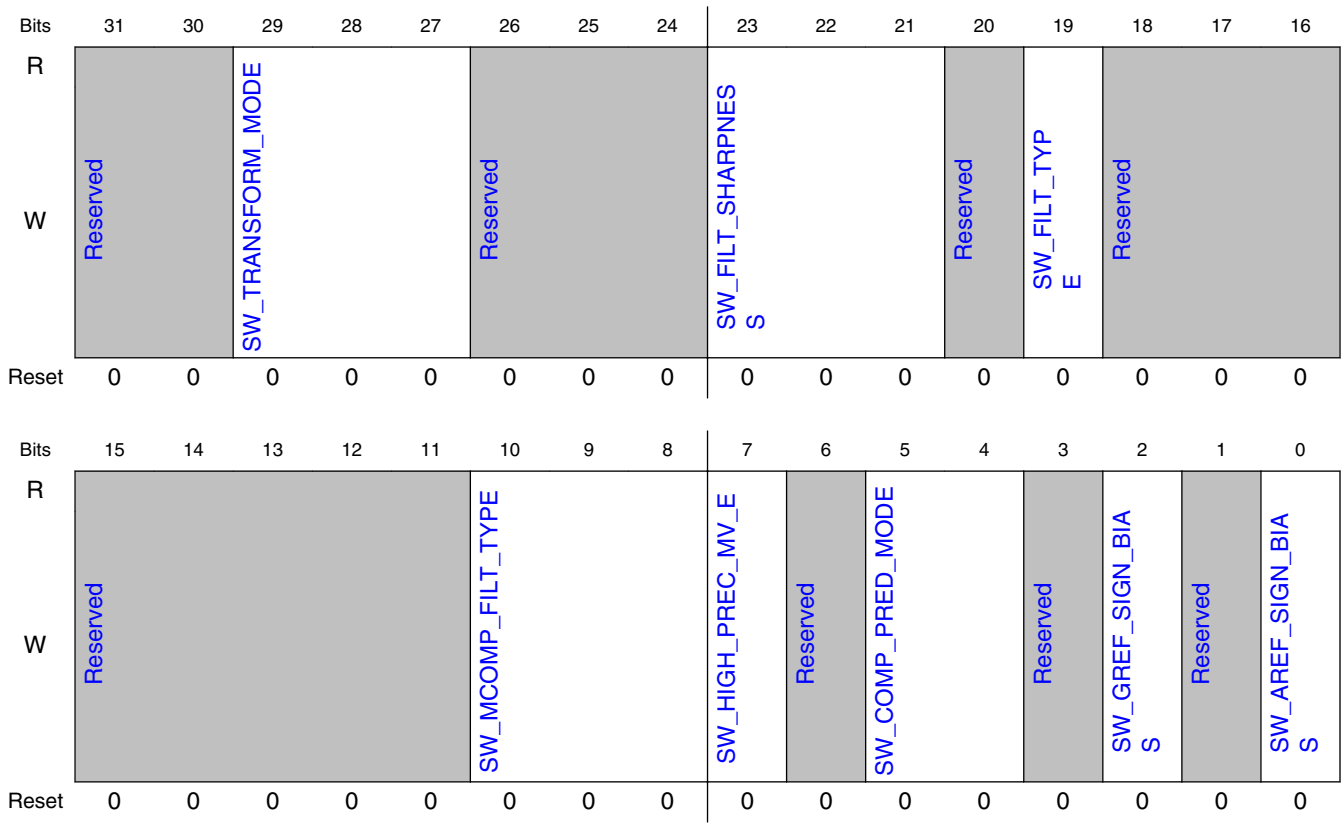
Field	Function
SW_NUM_TILE_ROWS	
13-2 —	Reserved.
1 SW_TILE_ENABLE	Tile enable
0 SW_ENTR_CODE_SYNCH_E	Entropy coding synchronization enable (Possible parallel cabac decoding)

15.2.5.1.13 Decoder control register 8 (SWREG11)

15.2.5.1.13.1 Offset

Register	Offset
SWREG11	2Ch

15.2.5.1.13.2 Diagram



15.2.5.1.13.3 Fields

Field	Function
31-30 —	Reserved.
29-27 SW_TRANSFORM_MODE	Transform modes 000b - 4x4 only 001b - Allow 8x8 010b - Allow 16x16 011b - Allow 32x32 100b - TX mode select
26-24 —	Reserved.
23-21 SW_FILT_SHARPNESS	Filter sharpness value
20 —	Reserved.
19 SW_FILT_TYPE	Filter Type

Table continues on the next page...

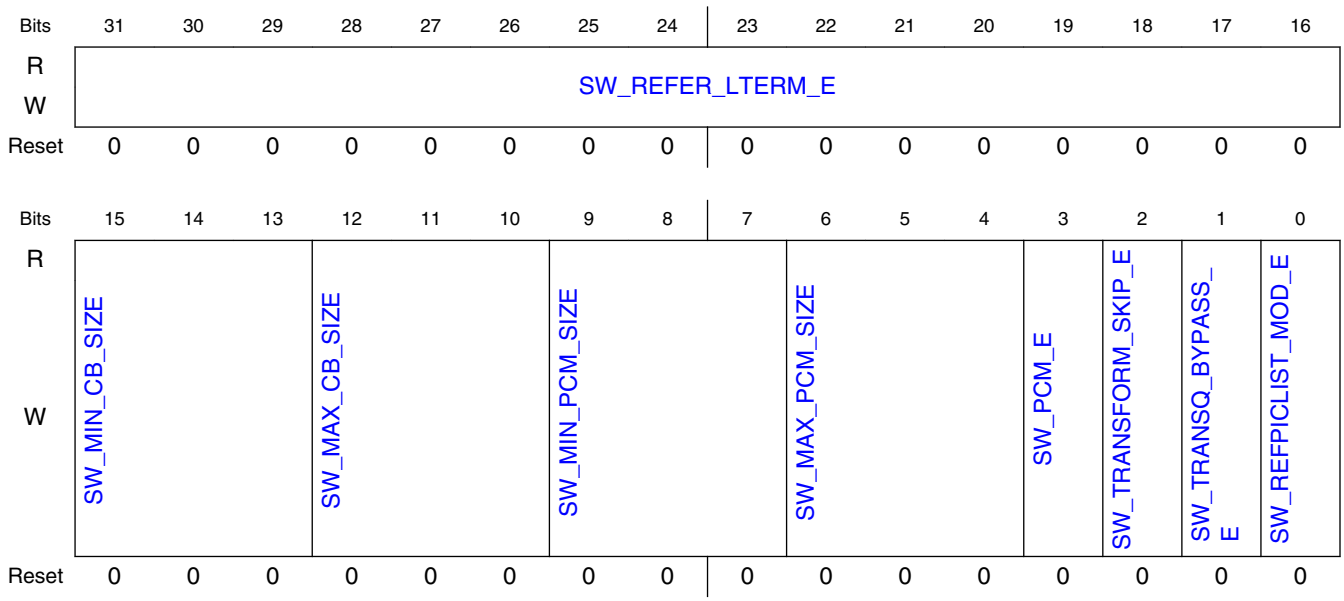
Field	Function
18-11 —	Reserved.
10-8 SW_MCOMP_FILT_TYPE	Inter prediction filter type to stream decoder 000b - Eight tap smooth 001b - Eight tap 010b - Eight tap sharp 011b - Bilinear 100b - Switchable
7 SW_HIGH_PREC_MV_E	High precision MV prediction enable
6 —	Reserved.
5-4 SW_COMP_PRED_MODE	Prediction Comp Type 00b - Single prediction only 01b - COMP prediction only 10b - Hybrid prediction
3 —	Reserved.
2 SW_GREF_SIGN_BIAS	Golden reference picture sign bias used for motion vector decoding
1 —	Reserved.
0 SW_AREF_SIGN_BIAS	Alternate reference picture sign bias used for motion vector decoding

15.2.5.1.14 Decoder control register 9 (SWREG12)

15.2.5.1.14.1 Offset

Register	Offset
SWREG12	30h

15.2.5.1.14.2 Diagram



15.2.5.1.14.3 Fields

Field	Function
31-16 SW_REFER_LTERM_E	Long term flag for reference picture index Definition: Bit 31 for picture index 0 Bit 30 for picture index 1 etc.
15-13 SW_MIN_CB_SIZE	CodedBlock min size (2^N): 011b - 8 pix 100b - 16 pix 101b - 32 pix 110b - 64 pix
12-10 SW_MAX_CB_SIZE	CodedBlock max size (2^N): 011b - 8 pix 100b - 16 pix 101b - 32 pix 110b - 64 pix
9-7 SW_MIN_PCM_SIZE	PCM min size (2^N): 011b - 8 pix 100b - 16 pix 101b - 32 pix 110b - 64 pix
6-4 SW_MAX_PCM_SIZE	PCM max size (2^N): 011b - 8 pix 100b - 16 pix 101b - 32 pix 110b - 64 pix
3 SW_PCM_E	IPCM MBs flag

Table continues on the next page...

Field	Function
2 SW_TRANSFORM_SKIP_E	Transform skipping flag
1 SW_TRANSQ_BYPASS_E	Transform bypass flag (lossless mode)
0 SW_REFPICLIST_MOD_E	Refpic list reordering flag

15.2.5.1.15 Decoder control register 10 (SWREG13)

15.2.5.1.15.1 Offset

Register	Offset
SWREG13	34h

15.2.5.1.15.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			DEC_CTRL_REG10_BF												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DEC_CTRL_REG10_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.15.3 Fields

Field	Function
31-29 —	Reserved.
28-0 DEC_CTRL_REG10_BF	For HEVC: [31:16] - Reserved - Not used [15:13] - sw_min_trb_size - Transform Block min size (2^N): 3 = 8 pix, 4 = 16 pix, 5 = 32 pix, 6 = 64 pix.

Field	Function
	[12:10] - sw_max_trb_size - Transform Block max size (2^N): 3 = 8 pix, 4 = 16 pix, 5 = 32 pix, 6 = 64 pix. [9:7] - sw_max_intra_hierdepth - Intra max hierarchy dept [6:4] - sw_max_inter_hierdepth - Inter max hierarchy depth [3:0] - sw_parallel_merge - Information about differential MV exist inside of coded block. For VP9: [31:29] - Reserved - Not used [28:23] - sw_qp_delta_y_dc - QP delta value for Y DC [22:17] - sw_qp_delta_ch_dc - QP delta value for C DC [16:11] - sw_qp_delta_ch_ac - QP delta value for C AC [10] - sw_last_sign_bias - Previous reference picture sign bias for motion vector decoding [9] - sw_lossless_e - Lossless transform enable [8:7] - sw_comp_pred_var_ref1 - Compaund prediction parameter to select correct sign bias from bin decoding [6:5] - sw_comp_pred_var_ref0 - Compaund prediction parameter to select correct sign bias from bin decoding [4:3] - sw_comp_pred_fixed_ref - Compaund prediction parameter to select last picture sign bias [2] - sw_segment_temp_upd_e - temporal segmentation update enable [1] - sw_segment_upd_e - segmentation update enable. If high new segmentation values are delivered in stream (and will be written out). If 0 the segmentation values are read from external memory [0] - sw_segment_e - segmentation enable

15.2.5.1.16 Initial ref pic list register (0-2) (SWREG14)

15.2.5.1.16.1 Offset

Register	Offset
SWREG14	38h

15.2.5.1.16.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INIT_REF_PIC_0_2_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INIT_REF_PIC_0_2_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.16.3 Fields

Field	Function
31-0 INIT_REF_PIC_0_2_BF	For HEVC: [31:30] - Reserved - Not used [29:25] - sw_init_rlist_b2 - Initial reference picture list for backward picid 2 [24:20] - sw_init_rlist_f2 - Initial reference picture list for forward picid 2 [19:15] - sw_init_rlist_b1 - Initial reference picture list for backward picid 1 [14:10] - sw_init_rlist_f1 - Initial reference picture list for forward picid 1 [9:5] - sw_init_rlist_b0 - Initial reference picture list for backward picid 0 [4:0] - sw_init_rlist_f0 - Initial reference picture list for forward picid 0 For VP9: [31:24] - Reserved - Not used [23:18] - sw_filt_level - Frame filtering level [17:15] - sw_refpic_seg0 - Segment refer picture [14] - sw_skip_seg0 - Segment skip enable [13:8] - sw_filt_level_seg0 - Segment filter level [7:0] - sw_quant_seg0 - Segment quantization parameter

15.2.5.1.17 Initial ref pic list register (3-5) (SWREG15)

15.2.5.1.17.1 Offset

Register	Offset
SWREG15	3Ch

15.2.5.1.17.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INIT_REF_PIC_3_5_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INIT_REF_PIC_3_5_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.17.3 Fields

Field	Function
31-0	For HEVC:

Field	Function
INIT_REF_PIC_3_5_BF	[31:30] - Reserved - Not used [29:25] - sw_init_rlist_b5 - Initial reference picture list for backward picid 5 [24:20] - sw_init_rlist_f5 - Initial reference picture list for forward picid 5 [19:15] - sw_init_rlist_b4 - Initial reference picture list for backward picid 4 [14:10] - sw_init_rlist_f4 - Initial reference picture list for forward picid 4 [9:5] - sw_init_rlist_b3 - Initial reference picture list for backward picid 3 [4:0] - sw_init_rlist_f3 - Initial reference picture list for forward picid 3 For VP9: [31:18] - Reserved - Not used [17:15] - sw_refpic_seg1 - Segment refer picture [14] - sw_skip_seg1 - Segment skip enable [13:8] - sw_filt_level_seg1 - Segment filter level [7:0] - sw_quant_seg1 - Segment quantization parameter

15.2.5.1.18 Initial ref pic list register (6-8) (SWREG16)

15.2.5.1.18.1 Offset

Register	Offset
SWREG16	40h

15.2.5.1.18.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INIT_REF_PIC_6_8_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INIT_REF_PIC_6_8_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.18.3 Fields

Field	Function
31-0 INIT_REF_PIC_6_8_BF	For HEVC: [31:30] - Reserved - Not used [29:25] - sw_init_rlist_b8 - Initial reference picture list for backward picid 8 [24:20] - sw_init_rlist_f8 - Initial reference picture list for forward picid 8 [19:15] - sw_init_rlist_b7 - Initial reference picture list for backward picid 7 [14:10] - sw_init_rlist_f7 - Initial reference picture list for forward picid 7

Field	Function
	[9:5] - sw_init_rlist_b6 - Initial reference picture list for backward picid 6 [4:0] - sw_init_rlist_f6 - Initial reference picture list for forward picid 6 For VP9: [31:18] - Reserved - Not used [17:15] - sw_refpic_seg2 - Segment refer picture [14] - sw_skip_seg2 - Segment skip enable [13:8] - sw_filt_level_seg2 - Segment filter level [7:0] - sw_quant_seg2 - Segment quantization parameter

15.2.5.1.19 Initial ref pic list register (9-11) (SWREG17)

15.2.5.1.19.1 Offset

Register	Offset
SWREG17	44h

15.2.5.1.19.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INIT_REF_PIC_9_11_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INIT_REF_PIC_9_11_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.19.3 Fields

Field	Function
31-0 INIT_REF_PIC_9_11_BF	For HEVC: [31:30] - Reserved - Not used [29:25] - sw_init_rlist_b11 - Initial reference picture list for backward picid 11 [24:20] - sw_init_rlist_f11 - Initial reference picture list for forward picid 11 [19:15] - sw_init_rlist_b10 - Initial reference picture list for backward picid 10 [14:10] - sw_init_rlist_f10 - Initial reference picture list for forward picid 10 [9:5] - sw_init_rlist_b9 - Initial reference picture list for backward picid 9 [4:0] - sw_init_rlist_f9 - Initial reference picture list for forward picid 9 For VP9: [31:18] - Reserved - Not used

Field	Function
	[17:15] - sw_refpic_seg3 - Segment refer picture [14] - sw_skip_seg3 - Segment skip enable [13:8] - sw_filt_level_seg3 - Segment filter level [7:0] - sw_quant_seg3 - Segment quantization parameter

15.2.5.1.20 Initial ref pic list register (12-14) (SWREG18)

15.2.5.1.20.1 Offset

Register	Offset
SWREG18	48h

15.2.5.1.20.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INIT_REF_PIC_12_14_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INIT_REF_PIC_12_14_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.20.3 Fields

Field	Function
31-0 INIT_REF_PIC_12_14_BF	For HEVC: [31:30] - Reserved - Not used [29:25] - sw_init_rlist_b14 - Initial reference picture list for backward picid 14 [24:20] - sw_init_rlist_f14 - Initial reference picture list for forward picid 14 [19:15] - sw_init_rlist_b13 - Initial reference picture list for backward picid 13 [14:10] - sw_init_rlist_f13 - Initial reference picture list for forward picid 13 [9:5] - sw_init_rlist_b12 - Initial reference picture list for backward picid 12 [4:0] - sw_init_rlist_f12 - Initial reference picture list for forward picid 12 For VP9: [31:18] - Reserved - Not used [17:15] - sw_refpic_seg4 - Segment refer picture [14] - sw_skip_seg4 - Segment skip enable [13:8] - sw_filt_level_seg4 - Segment filter level [7:0] - sw_quant_seg4 - Segment quantization parameter

15.2.5.1.21 Initial ref pic list register (15 and P 0-3) (SWREG19)

15.2.5.1.21.1 Offset

Register	Offset
SWREG19	4Ch

15.2.5.1.21.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INIT_REF_PIC_15_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INIT_REF_PIC_15_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.21.3 Fields

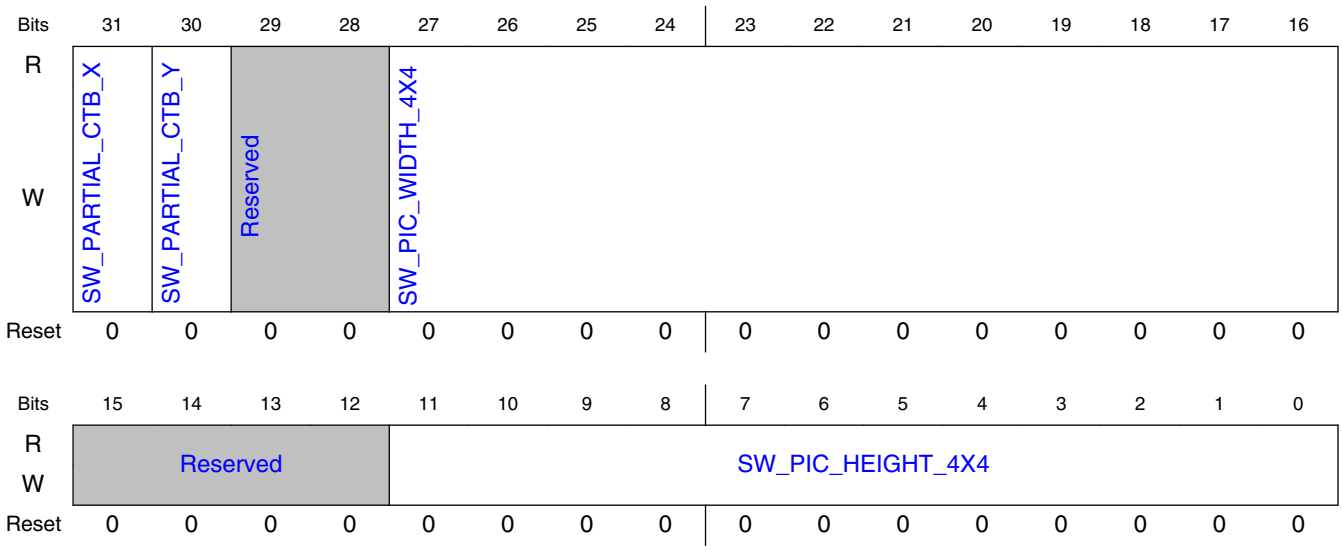
Field	Function
31-0 INIT_REF_PIC_15_BF	For HEVC: [31:10] - Reserved - Not used [9:5] - sw_init_rlist_b15 - Initial reference picture list for backward picid 15 [4:0] - sw_init_rlist_f15 - Initial reference picture list for forward picid 15 For VP9: [31:18] - Reserved - Not used [17:15] - sw_refpic_seg5 - Segment refer picture [14] - sw_skip_seg5 - Segment skip enable [13:8] - sw_filt_level_seg5 - Segment filter level [7:0] - sw_quant_seg5 - Segment quantization parameter

15.2.5.1.22 Decoder control register 11 (SWREG20)

15.2.5.1.22.1 Offset

Register	Offset
SWREG20	50h

15.2.5.1.22.2 Diagram



15.2.5.1.22.3 Fields

Field	Function
31 SW_PARTIAL_CTB_X	Picture width not multiple of CTB size
30 SW_PARTIAL_CTB_Y	Picture height not multiple of CTB size
29-28 —	Reserved.
27-16 SW_PIC_WIDTH_4X4	Current picture width in 4x4 blocks (Needed to reduce overlapping HW conditions in various blocks)
15-12 —	Reserved.
11-0 SW_PIC_HEIGHT_4X4	Current picture height in 4x4 blocks (Needed to reduce overlapping HW conditions in various blocks)

15.2.5.1.23 Not used (SWREG21)

15.2.5.1.23.1 Offset

Register	Offset
SWREG21	54h

15.2.5.1.23.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.23.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.24 Not used (SWREG22)

15.2.5.1.24.1 Offset

Register	Offset
SWREG22	58h

15.2.5.1.24.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.24.3 Fields

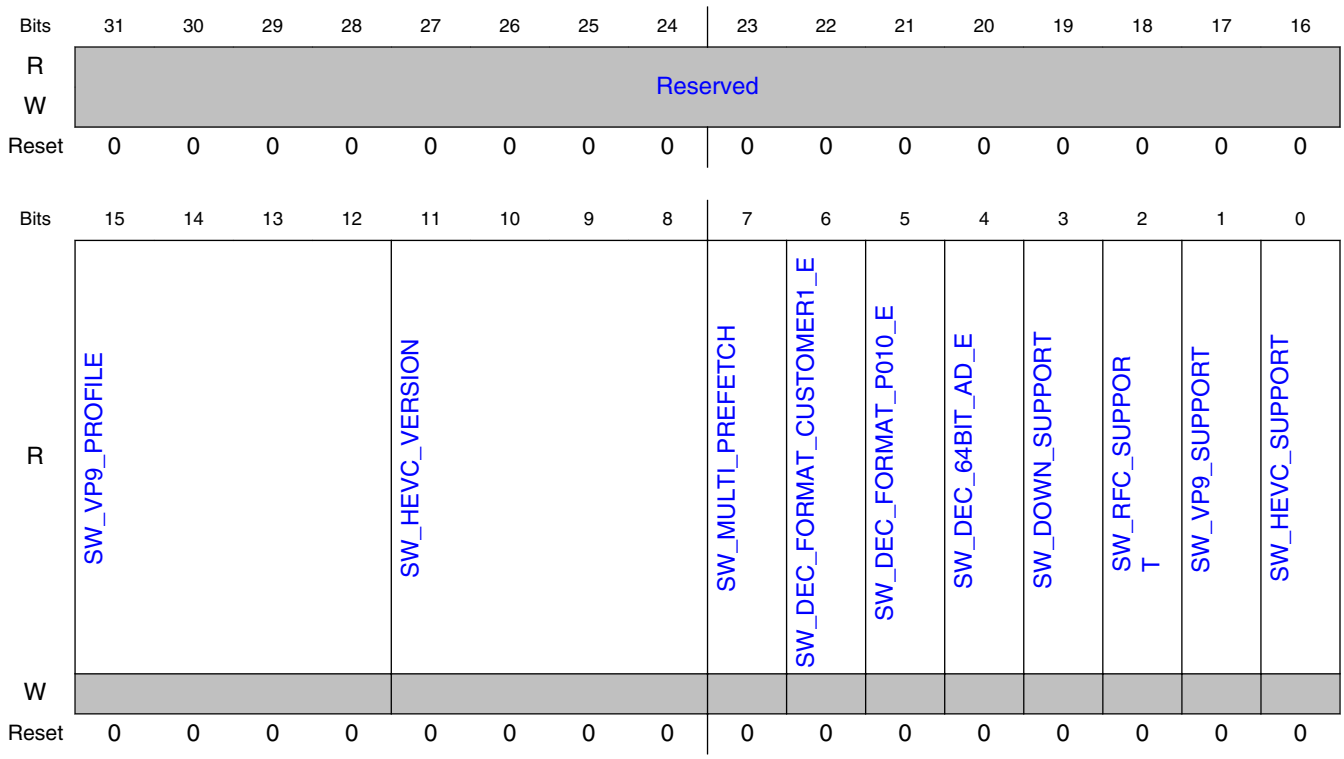
Field	Function
31-0 —	Reserved.

15.2.5.1.25 Decoder configure status register (SWREG23)

15.2.5.1.25.1 Offset

Register	Offset
SWREG23	5Ch

15.2.5.1.25.2 Diagram



15.2.5.1.25.3 Fields

Field	Function
31-16 —	Reserved.
15-12 SW_VP9_PROFILE	VP9 version 0000b - vp9 profile 0 0001b - vp9 profile 2 - 10bits
11-8 SW_HEVC_VERSION	HEVC version 0000b - main8 0001b - main10
7 SW_MULTI_PREFETCH	Multi-Reference Blocks Prefetch 0b - Not supported 1b - Supported
6 SW_DEC_FORMAT_CUSTOMER1_E	Customized output format support Demonstrated as following - from MSB->LSB in a 128-bit burst. • 10-bit pixel • 128-bit burst0: Y0|Y1|Y2|Y3|...|Y11|Y12[9:2] • 128-bit burst1: Y12[1:0]|Y13|Y14|Y15|...|Y24|Y25[9:4] • 128-bit burst2: Y25[3:0]|Y26|Y27|Y28|...|Y37|Y38[9:6]

Table continues on the next page...

Field	Function
	• 128-bit burst3: Y38[5:0]|Y39|Y40|Y41|...|Y50|Y51[9:8] • 128-bit burst4: Y51[7:0]|Y52|Y53|Y54|...|Y62|Y63 • 8-bit pixel • 128-bit burst: Y0|Y1|Y2|...|Y15 • 128-bit burst: Y16|Y17|Y18|...|Y31 0b - Not supported 1b - Supported
5 SW_DEC_FORMAT_P010_E	P010 output format support 0b - Not supported 1b - Supported
4 SW_DEC_64BIT_AD_E	64 bit addressing of master interface support 0b - Not supported (32 bit addressing) 1b - Supported
3 SW_DOWN_SUPPORT	Downscale support 0b - Do not support downscale 1b - Support downscale
2 SW_RFC_SUPPORT	RFC support 0b - Do not support RFC 1b - Support RFC
1 SW_VP9_SUPPORT	VP9 support 0b - Do not support VP9 1b - Support VP9
0 SW_HEVC_SUPPORT	HEVC support 0b - Do not support HEVC 1b - Support HEVC

15.2.5.1.26 Not used (SWREG24)

15.2.5.1.26.1 Offset

Register	Offset
SWREG24	60h

15.2.5.1.26.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.26.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.27 Not used (SWREG25)

15.2.5.1.27.1 Offset

Register	Offset
SWREG25	64h

15.2.5.1.27.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.27.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.28 Not used (SWREG26)

15.2.5.1.28.1 Offset

Register	Offset
SWREG26	68h

15.2.5.1.28.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.28.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.29 Not used (SWREG27)

15.2.5.1.29.1 Offset

Register	Offset
SWREG27	6Ch

15.2.5.1.29.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.29.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.30 Not used (SWREG28)

15.2.5.1.30.1 Offset

Register	Offset
SWREG28	70h

15.2.5.1.30.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.30.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.31 Not used (SWREG29)

15.2.5.1.31.1 Offset

Register	Offset
SWREG29	74h

15.2.5.1.31.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.31.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.32 Not used (SWREG30)

15.2.5.1.32.1 Offset

Register	Offset
SWREG30	78h

15.2.5.1.32.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.32.3 Fields

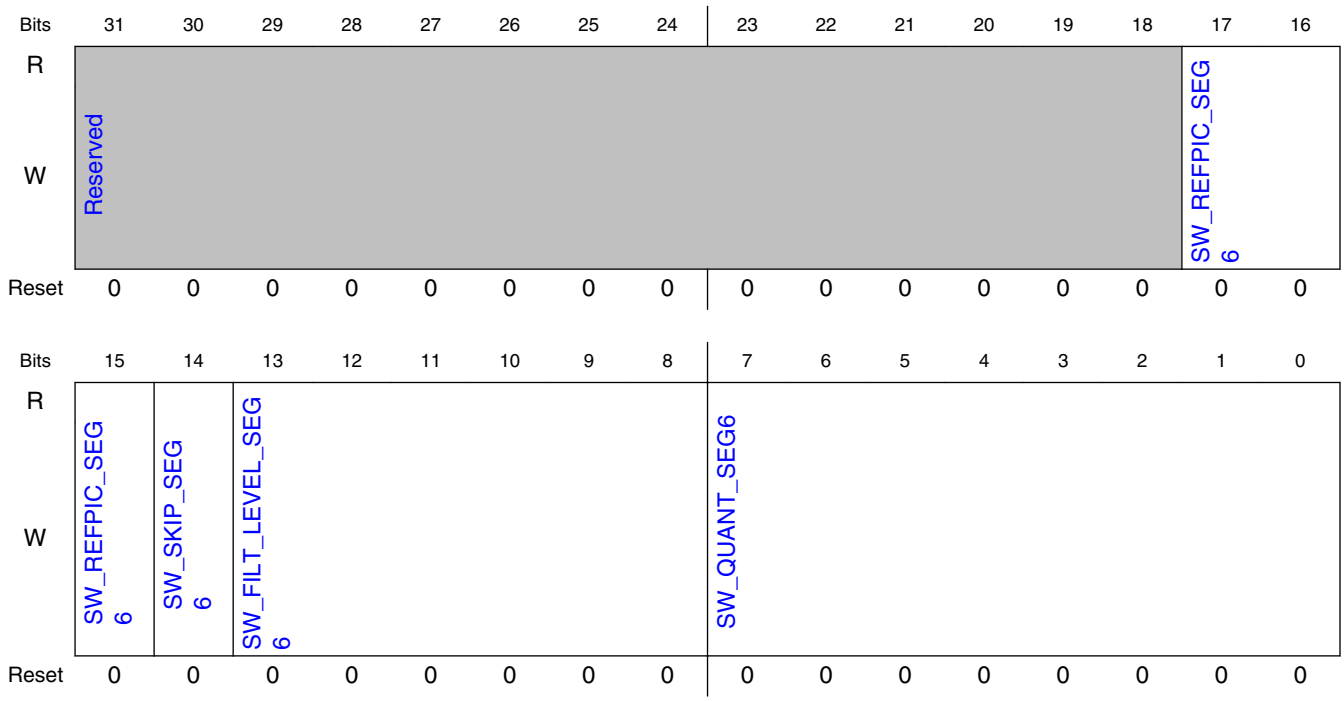
Field	Function
31-0 —	Reserved.

15.2.5.1.33 VP9 segmentation values (SWREG31)

15.2.5.1.33.1 Offset

Register	Offset
SWREG31	7Ch

15.2.5.1.33.2 Diagram



15.2.5.1.33.3 Fields

Field	Function
31-18 —	Reserved.
17-15 SW_REFPIC_SEG6	Segment refer picture
14 SW_SKIP_SEG6	Segment skip enable
13-8 SW_FILT_LEVEL_SEG6	Segment filter level
7-0	Segment quantization parameter

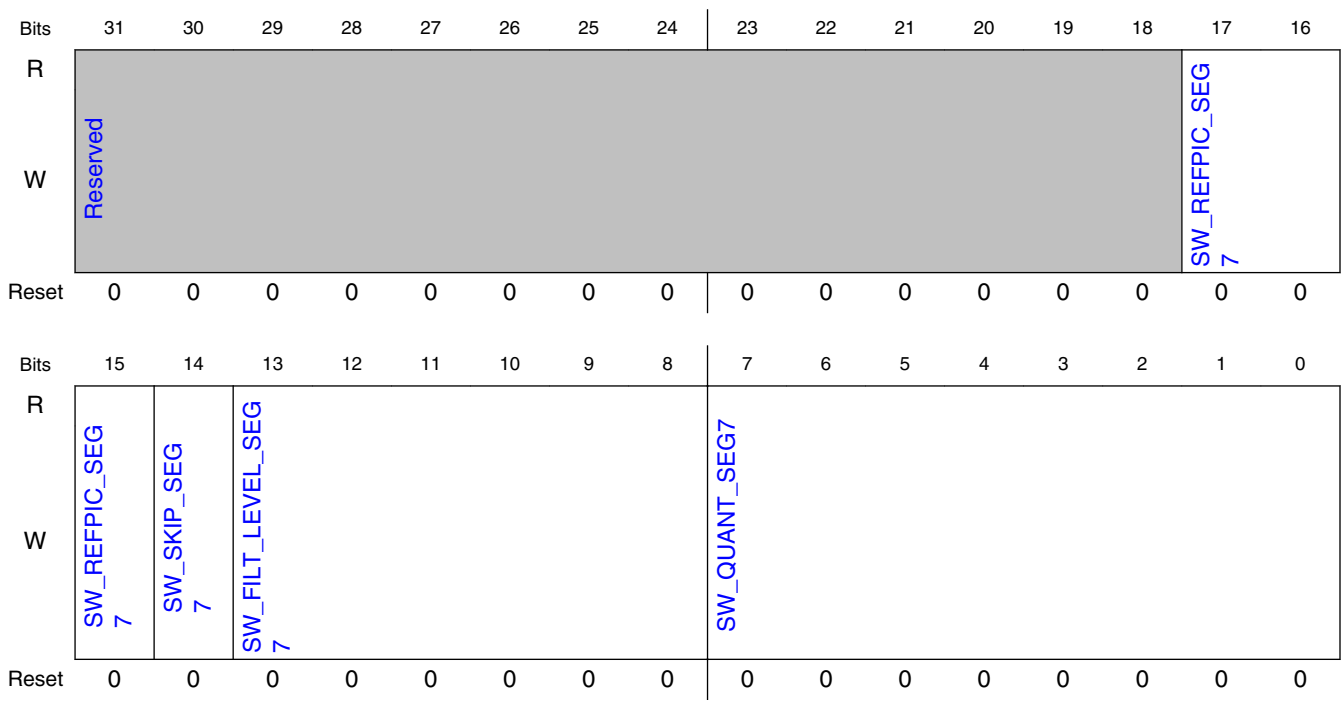
Field	Function
SW_QUANT_SEG6	

15.2.5.1.34 VP9 segmentation values (SWREG32)

15.2.5.1.34.1 Offset

Register	Offset
SWREG32	80h

15.2.5.1.34.2 Diagram



15.2.5.1.34.3 Fields

Field	Function
31-18 —	Reserved.
17-15	Segment refer picture

Table continues on the next page...

Field	Function
SW_REFPIC_SEG7	
14 SW_SKIP_SEG7	Segment skip enable
13-8 SW_FILT_LEVEL_SEG7	Segment filter level
7-0 SW_QUANT_SEG7	Segment quantization parameter

15.2.5.1.35 VP9 reference picture scaling register 0 (SWREG33)

15.2.5.1.35.1 Offset

Register	Offset
SWREG33	84h

15.2.5.1.35.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_LREF_WIDTH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_LREF_HEIGHT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.35.3 Fields

Field	Function
31-16 SW_LREF_WIDTH	Accurate width of last (previous) reference picture in pixels
15-0	Accurate height of last (previous) reference picture in pixels

Field	Function
SW_LREF_HEIGHT	

15.2.5.1.36 VP9 reference picture scaling register 1 (SWREG34)

15.2.5.1.36.1 Offset

Register	Offset
SWREG34	88h

15.2.5.1.36.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_GREF_WIDTH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_GREF_HEIGHT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.36.3 Fields

Field	Function
31-16 SW_GREF_WIDTH	Accurate width of golden reference picture in pixels
15-0 SW_GREF_HEIGHT	Accurate height of golden reference picture in pixels

15.2.5.1.37 VP9 reference picture scaling register 2 (SWREG35)

15.2.5.1.37.1 Offset

Register	Offset
SWREG35	8Ch

15.2.5.1.37.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_AREF_WIDTH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_AREF_HEIGHT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.37.3 Fields

Field	Function
31-16 SW_AREF_WIDTH	Accurate width of alternate reference picture in pixels
15-0 SW_AREF_HEIGHT	Accurate height of alternate reference picture in pixels

15.2.5.1.38 VP9 reference picture scaling register 3 (SWREG36)

15.2.5.1.38.1 Offset

Register	Offset
SWREG36	90h

15.2.5.1.38.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_LREF_HOR_SCALE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_LREF_VER_SCALE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.38.3 Fields

Field	Function
31-16 SW_LREF_HOR_SCALE	Horizontal scaling factor for last (previous) reference picture
15-0 SW_LREF_VER_SCALE	Vertical scaling factor for last (previous) reference picture

15.2.5.1.39 VP9 reference picture scaling register 4 (SWREG37)

15.2.5.1.39.1 Offset

Register	Offset
SWREG37	94h

15.2.5.1.39.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_GREF_HOR_SCALE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_GREF_VER_SCALE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.39.3 Fields

Field	Function
31-16 SW_GREF_HOR_SCALE	Horizontal scaling factor for golden reference picture
15-0 SW_GREF_VER_SCALE	Vertical scaling factor for golden reference picture

15.2.5.1.40 VP9 reference picture scaling register 5 (SWREG38)

15.2.5.1.40.1 Offset

Register	Offset
SWREG38	98h

15.2.5.1.40.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_AREF_HOR_SCALE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_AREF_VER_SCALE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.40.3 Fields

Field	Function
31-16 SW_AREF_HOR_SCALE	Horizontal scaling factor for alternate reference picture
15-0 SW_AREF_VER_SCALE	Vertical scaling factor for alternate reference picture

15.2.5.1.41 Not used (SWREG39)

15.2.5.1.41.1 Offset

Register	Offset
SWREG39	9Ch

15.2.5.1.41.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.41.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.42 Not used (SWREG40)

15.2.5.1.42.1 Offset

Register	Offset
SWREG40	A0h

15.2.5.1.42.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.42.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.43 Not used (SWREG41)

15.2.5.1.43.1 Offset

Register	Offset
SWREG41	A4h

15.2.5.1.43.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.43.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.44 Not used (SWREG42)

15.2.5.1.44.1 Offset

Register	Offset
SWREG42	A8h

15.2.5.1.44.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.44.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.45 Not used (SWREG43)

15.2.5.1.45.1 Offset

Register	Offset
SWREG43	ACh

15.2.5.1.45.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.45.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.46 Not used (SWREG44)

15.2.5.1.46.1 Offset

Register	Offset
SWREG44	B0h

15.2.5.1.46.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.46.3 Fields

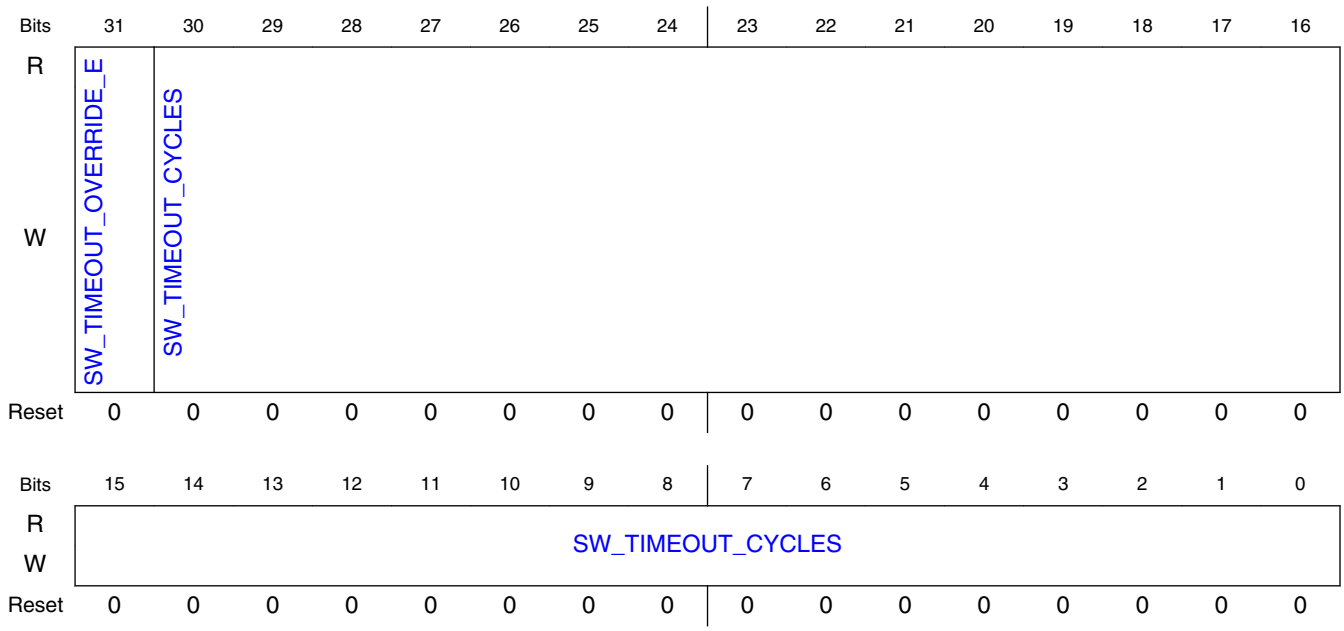
Field	Function
31-0 —	Reserved.

15.2.5.1.47 Timeout control register (SWREG45)

15.2.5.1.47.1 Offset

Register	Offset
SWREG45	B4h

15.2.5.1.47.2 Diagram



15.2.5.1.47.3 Fields

Field	Function
31 SW_TIMEOUT_OVERRIDE_E	Enable for SW controlled timeout. If enabled the sw_timeout_cycles is used to detect HW timeout instead of hard coded HW value
30-0 SW_TIMEOUT_CYCLES	Amount of clock cycles to trigger timeout interrupt if no external master activity acknowledged. Used if sw_timeout_override_e is set

15.2.5.1.48 Picture order count from current pictures for index 0-3 (SWREG46)

15.2.5.1.48.1 Offset

Register	Offset
SWREG46	B8h

15.2.5.1.48.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PIC_ORD_0_3_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PIC_ORD_0_3_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.48.3 Fields

Field	Function
31-0 PIC_ORD_0_3_BF	For HEVC: [31:24] - sw_cur_poc_00 - Picture order count from current picture 0 [23:16] - sw_cur_poc_01 - Picture order count from current picture 1 [15:8] - sw_cur_poc_02 - Picture order count from current picture 2 [7:0] - sw_cur_poc_03 - Picture order count from current picture 3 For VP9: [31] - Reserved - Not used [30:24] - sw_filt_ref_adj_0 - Filter level intra adjustment [23] - Reserved - Not used [22:16] - sw_filt_ref_adj_1 - Filter level last ref pic adjustment [15] - Reserved - Not used [14:8] - sw_filt_ref_adj_2 - Filter level golden pic adjustment [7] - Reserved - Not used [6:0] - sw_filt_ref_adj_3 - Filter level alt ref pic adjustment

15.2.5.1.49 Picture order count from current pictures for index 4-7 (SWREG47)

15.2.5.1.49.1 Offset

Register	Offset
SWREG47	BCh

15.2.5.1.49.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PIC_ORD_4_7_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PIC_ORD_4_7_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.49.3 Fields

Field	Function
31-0 PIC_ORD_4_7_BF	For HEVC: [31:24] - sw_cur_poc_04 - Picture order count from current picture 4 [23:16] - sw_cur_poc_05 - Picture order count from current picture 5 [15:8] - sw_cur_poc_06 - Picture order count from current picture 6 [7:0] - sw_cur_poc_07 - Picture order count from current picture 7 For VP9: [31] - Reserved - Not used [30:24] - sw_filt_mb_adj_0 - Filter level ZERO mv adjustment [23] - Reserved - Not used [22:16] - sw_filt_mb_adj_1 - Filter level adjustment [15] - Reserved - Not used [14:8] - sw_filt_mb_adj_2 - Filter level adjustment (Not used) [7] - Reserved - Not used [6:0] - sw_filt_mb_adj_3 - Filter level adjustment (Not used)

15.2.5.1.50 Picture order count from current pictures for index 8-11 (SWREG48)

15.2.5.1.50.1 Offset

Register	Offset
SWREG48	C0h

15.2.5.1.50.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_CUR_POC_08								SW_CUR_POC_09							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_CUR_POC_10								SW_CUR_POC_11							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.50.3 Fields

Field	Function
31-24 SW_CUR_POC_08	Picture order count from current picture 8
23-16 SW_CUR_POC_09	Picture order count from current picture 9
15-8 SW_CUR_POC_10	Picture order count from current picture 10
7-0 SW_CUR_POC_11	Picture order count from current picture 11

15.2.5.1.51 Picture order count from current pictures for index 12-15 (SWREG49)

15.2.5.1.51.1 Offset

Register	Offset
SWREG49	C4h

15.2.5.1.51.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_CUR_POC_12								SW_CUR_POC_13							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_CUR_POC_14								SW_CUR_POC_15							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.51.3 Fields

Field	Function
31-24 SW_CUR_POC_12	Picture order count from current picture 12
23-16 SW_CUR_POC_13	Picture order count from current picture 13
15-8 SW_CUR_POC_14	Picture order count from current picture 14
7-0 SW_CUR_POC_15	Picture order count from current picture 15

15.2.5.1.52 Synthesis configuration register decoder 0 (read only) (SWREG50)

15.2.5.1.52.1 Offset

Register	Offset
SWREG50	C8h

15.2.5.1.52.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					SW_DEC_MAX_OWIDTH										
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.2.5.1.52.3 Fields

Field	Function
31-11 —	Reserved.
10-0 SW_DEC_MAX_OWIDTH	Max configured decoder video resolution that can be decoded. Informed as width of the picture in pixels

15.2.5.1.53 Reference picture buffer control register (SWREG51)

15.2.5.1.53.1 Offset

Register	Offset
SWREG51	CCh

15.2.5.1.53.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.53.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.54 Reference picture buffer information register 1 (read only) (SWREG52)

15.2.5.1.54.1 Offset

Register	Offset
SWREG52	D0h

15.2.5.1.54.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.54.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.55 Reference picture buffer information register 2 (read only) (SWREG53)

15.2.5.1.55.1 Offset

Register	Offset
SWREG53	D4h

15.2.5.1.55.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.55.3 Fields

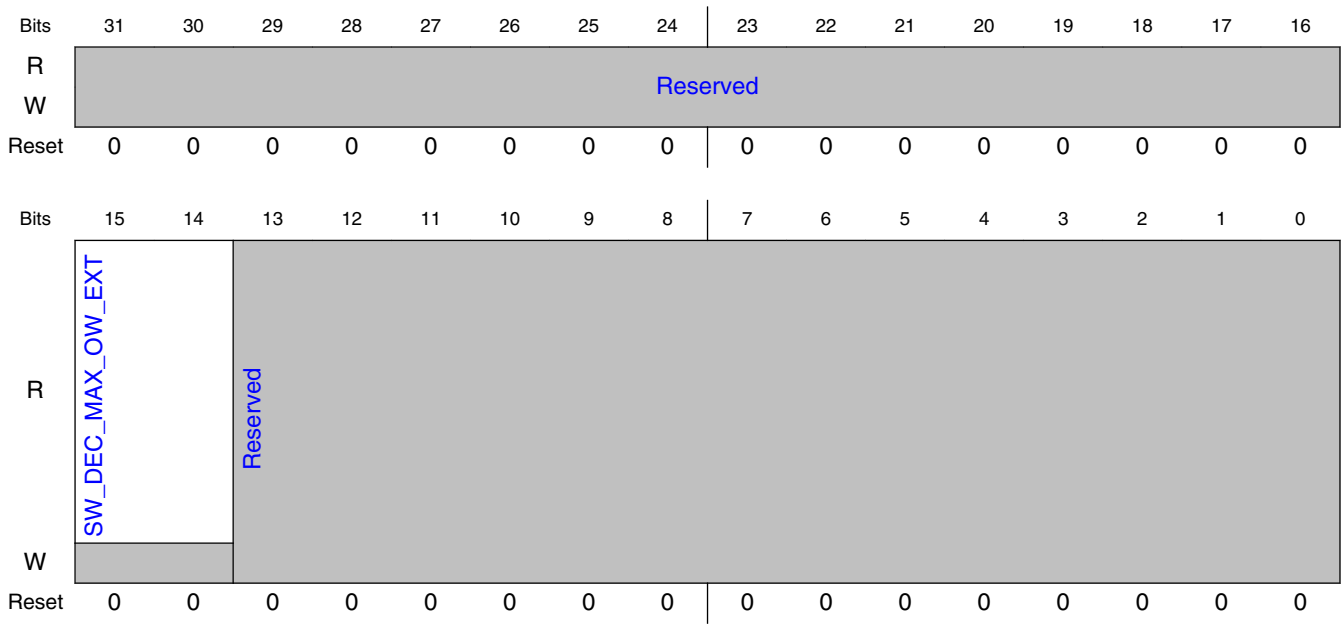
Field	Function
31-0 —	Reserved.

15.2.5.1.56 Synthesis configuration register decoder 1 (read only) (SWREG54)

15.2.5.1.56.1 Offset

Register	Offset
SWREG54	D8h

15.2.5.1.56.2 Diagram



15.2.5.1.56.3 Fields

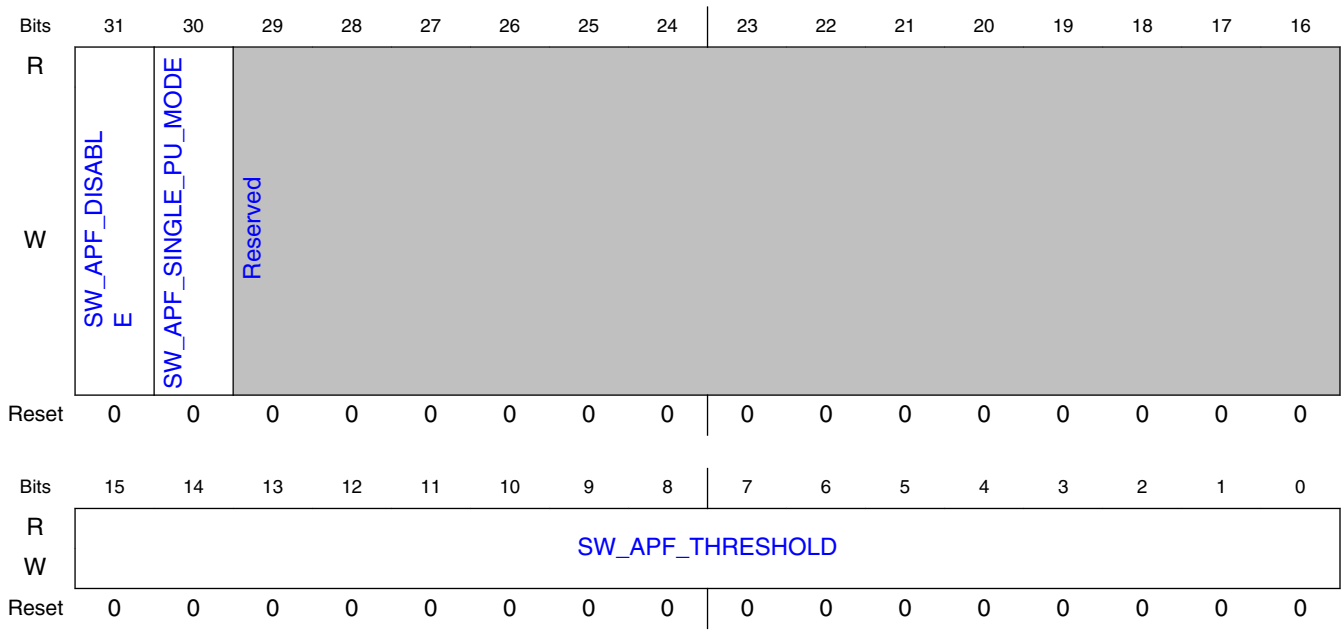
Field	Function
31-16 —	Reserved.
15-14 SW_DEC_MAX_OW_EXT	Max configured decoder video resolution that can be decoded. This is the MSB part of the configuration signal
13-0 —	Reserved.

15.2.5.1.57 Advanced prefetch control register (SWREG55)

15.2.5.1.57.1 Offset

Register	Offset
SWREG55	DCh

15.2.5.1.57.2 Diagram



15.2.5.1.57.3 Fields

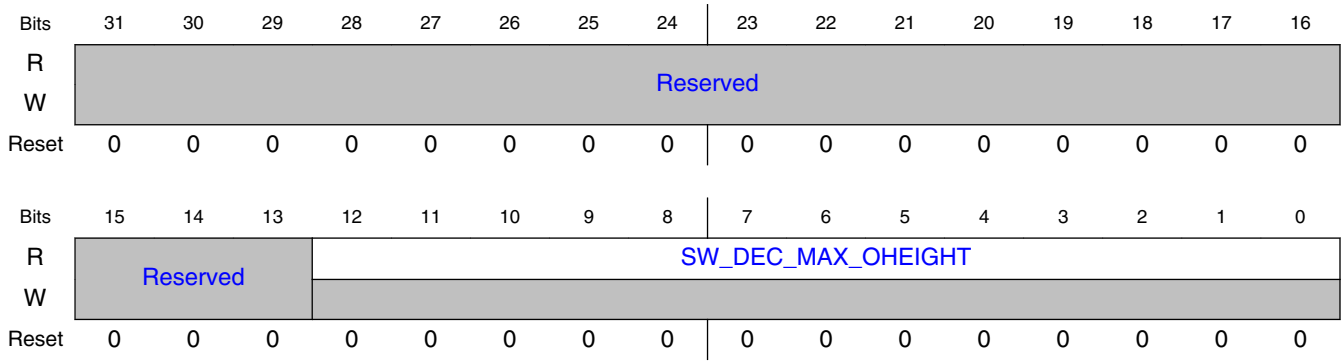
Field	Function
31 SW_APF_DISABLE	Advanced prefetch disable. If hight each partition is read separately
30 SW_APF_SINGLE_PU_MODE	APF amount of buffered Pus: can be restricted to buffer one PU at a time
29-16 —	Reserved.
15-0 SW_APF_THRESHOLD	Advanced prefetch threshold. If current buffered unit exceeds the threshold the advanced mode is not used. Value 0 disables threshold usage and advanced prefetch usage is restricted by internal memory limitation only

15.2.5.1.58 Synthesis configuration register decoder 2 (read only) (SWREG56)

15.2.5.1.58.1 Offset

Register	Offset
SWREG56	E0h

15.2.5.1.58.2 Diagram



15.2.5.1.58.3 Fields

Field	Function
31-13 —	Reserved.
12-0 SW_DEC_MAX_OHEIGHT	Max supported picture height in pixels

15.2.5.1.59 Decoder fuse register (read only) (SWREG57)

15.2.5.1.59.1 Offset

Register	Offset
SWREG57	E4h

15.2.5.1.59.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.59.3 Fields

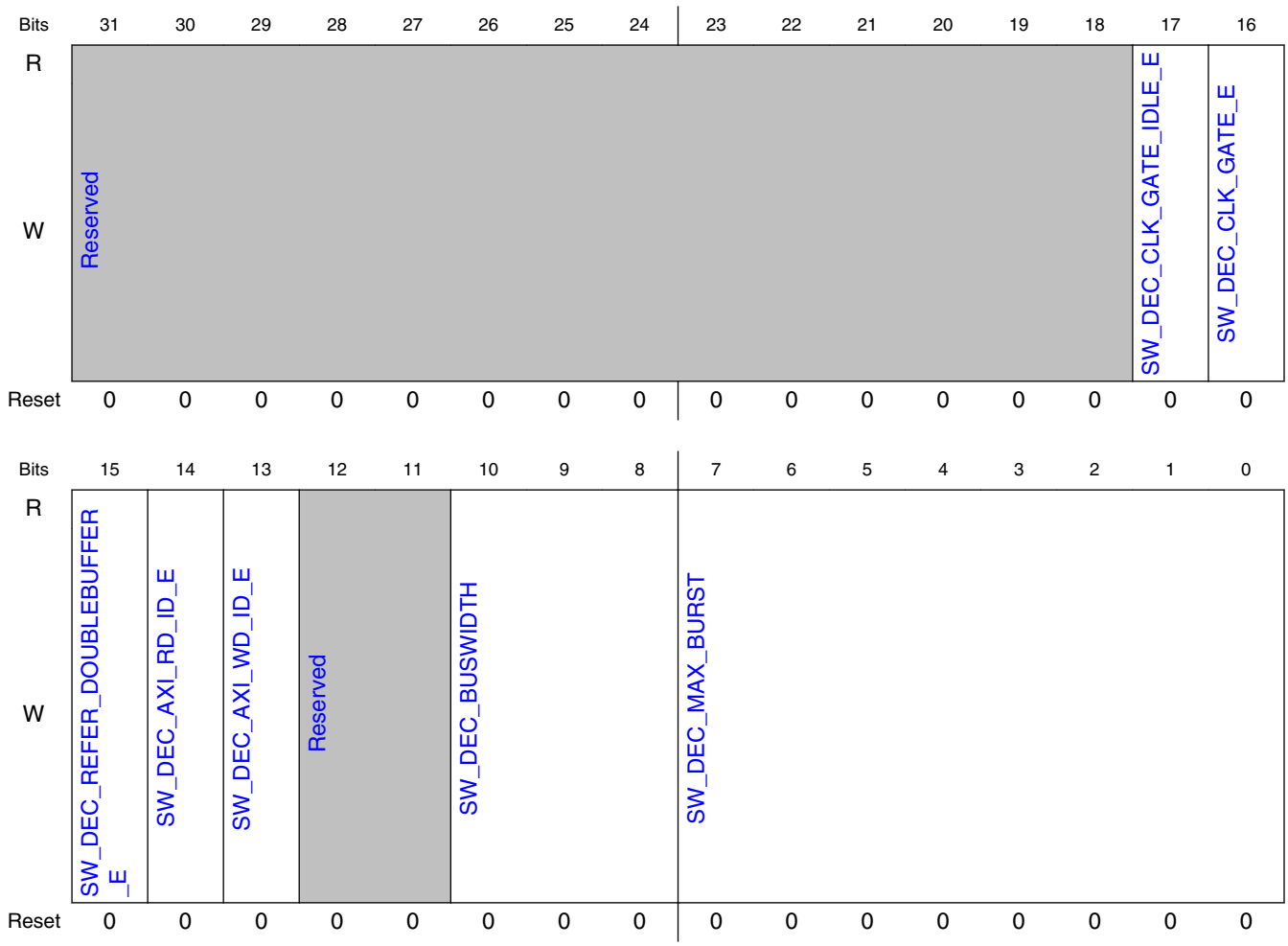
Field	Function
31-0 —	Reserved.

15.2.5.1.60 Device configuration register decoder 2 + Multi core control register (SWREG58)

15.2.5.1.60.1 Offset

Register	Offset
SWREG58	E8h

15.2.5.1.60.2 Diagram



15.2.5.1.60.3 Fields

Field	Function
31-18 —	Reserved.
17 SW_DEC_CLK_GATE_IDLE_E	Clock gating enable for decoder run-time. Generated separate clocks for each block by its own IDLE signal.
16 SW_DEC_CLK_GATE_E	Clock gating enable for picture-wise/decoding format clock gating. Between each picture the clock is gated from HW if this bit is high
15 SW_DEC_REFER_DOUBLEBUFFER_E	HW internal double buffering enable for reference data. This enable requires that there are two buffers available at the configured decoder (see configuration register values)

Table continues on the next page...

Field	Function
14 SW_DEC_AXI_RD_ID_E	SW axi ID enable. When enabled the given sw_dec_axi_rd_id is used as ID base and each sub-block will use offsets 0...max
13 SW_DEC_AXI_WD_ID_E	SW axi ID enable. When enabled the given sw_dec_axi_wd_id is used as ID base and each sub-block will use offsets 0...max
12-11 —	Reserved.
10-8 SW_DEC_BUS WIDTH	Decoder master interface buswidth 000b - 32 bit bus 001b - 64 bit bus 010b - 128 bit bus
7-0 SW_DEC_MAX_BURST	Maximum burst length for decoder bus transactions. Valid values: AXI: 1-256

15.2.5.1.61 Device configuration register AXI ID (SWREG59)

15.2.5.1.61.1 Offset

Register	Offset
SWREG59	ECh

15.2.5.1.61.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_AXI_WR_ID															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_AXI_RD_ID															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.61.3 Fields

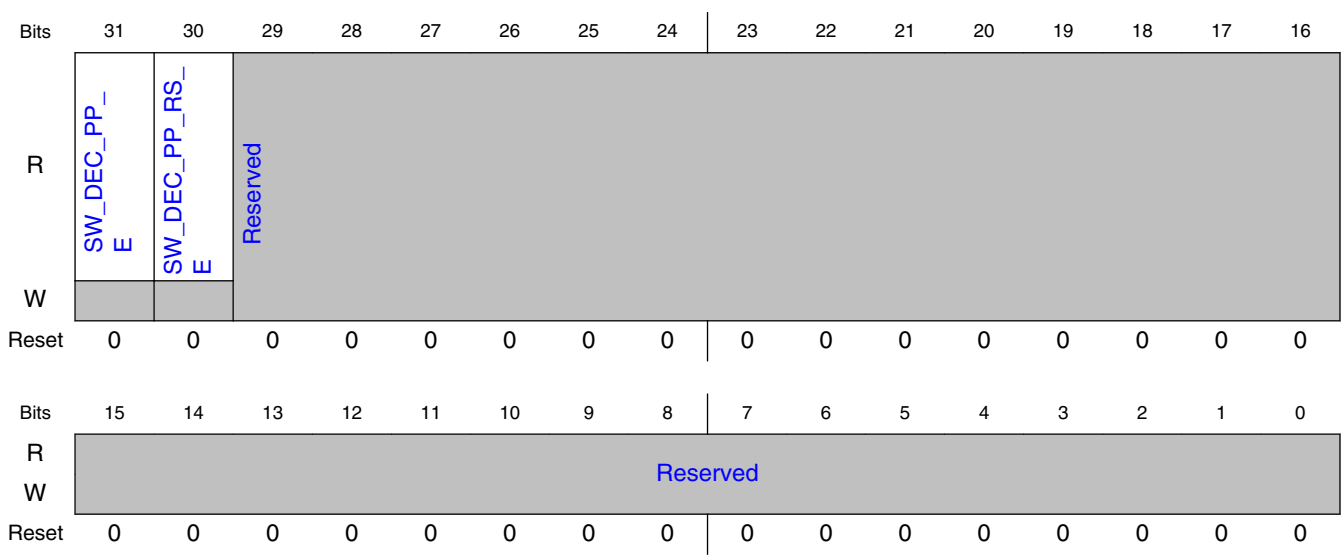
Field	Function
31-16 SW_DEC_AXI_WR_ID	Read ID base for HW write accesses. Each writing device use AXI ID of base+deviceoffset (where device offset is 0 1 2 3...Number of reading sub-blocks)
15-0 SW_DEC_AXI_RD_ID	Write ID base for HW write accesses. Each writing device use AXI ID of base+deviceoffset (where device offset is 0 1 2 3...Number of writing sub-blocks)

15.2.5.1.62 Synthesis configuration register decoder 3 for PP (read only) (SWREG60)

15.2.5.1.62.1 Offset

Register	Offset
SWREG60	F0h

15.2.5.1.62.2 Diagram



15.2.5.1.62.3 Fields

Field	Function
31	Decoder include PP 0b - PP does not exist. None of the PP features can be enabled.

Table continues on the next page...

Field	Function
SW_DEC_PP_E	1b - PP exists
30 SW_DEC_PP_RS_E	Decoder PP raster scan output support 0b - Raster scan output not supported 1b - Raster scan output supported
29-0 —	Reserved.

15.2.5.1.63 Not used (SWREG61)

15.2.5.1.63.1 Offset

Register	Offset
SWREG61	F4h

15.2.5.1.63.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.63.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.64 HW proceed register (CU location) (SWREG62)

15.2.5.1.64.1 Offset

Register	Offset
SWREG62	F8h

15.2.5.1.64.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_CU_LOCATION_X															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_CU_LOCATION_Y															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.64.3 Fields

Field	Function
31-16 SW_CU_LOCATION_X	Cu horizontal start location X in pixels (returned HW internal position during interrupt)
15-0 SW_CU_LOCATION_Y	Cu vertical start location Y in pixels (returned HW internal position during interrupt)

15.2.5.1.65 HW performance register (cycles running) (SWREG63)

15.2.5.1.65.1 Offset

Register	Offset
SWREG63	FCh

15.2.5.1.65.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SW_PERF_CYCLE_COUNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SW_PERF_CYCLE_COUNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.65.3 Fields

Field	Function
31-0 SW_PERF_CYCLE_COUNT	HW clock cycle counter return value. Amount of consumed clock cycles returned to this register when interrupt is being made (any kind of interrupt)

15.2.5.1.66 Base address MSB (bits 63:32) for decoded luminance picture (SWREG64)

15.2.5.1.66.1 Offset

Register	Offset
SWREG64	100h

15.2.5.1.66.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SW_DEC_OUT_YBASE_MSB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SW_DEC_OUT_YBASE_MSB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.66.3 Fields

Field	Function
31-0 SW_DEC_OUT_YBASE_MSB	Base address MSB (bits 63:32) for decoded luminance picture

15.2.5.1.67 Base address LSB (bits 31:0) for decoded luminance picture (SWREG65)

15.2.5.1.67.1 Offset

Register	Offset
SWREG65	104h

15.2.5.1.67.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_OUT_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_OUT_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.67.3 Fields

Field	Function
31-0 SW_DEC_OUT_YBASE_LSB	Base address LSB (bits 31:0) for decoded luminance picture

15.2.5.1.68 Base address MSB (bits 63:32) for reference luminance picture index 0 (SWREG66)

15.2.5.1.68.1 Offset

Register	Offset
SWREG66	108h

15.2.5.1.68.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER0_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER0_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.68.3 Fields

Field	Function
31-0 SW_REFER0_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 0

15.2.5.1.69 Base address LSB (bits 31:0) for reference luminance picture index 0 (SWREG67)

15.2.5.1.69.1 Offset

Register	Offset
SWREG67	10Ch

15.2.5.1.69.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER0_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER0_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.69.3 Fields

Field	Function
31-0 SW_REFER0_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 0

15.2.5.1.70 Base address MSB (bits 63:32) for reference luminance picture index 1 (SWREG68)

15.2.5.1.70.1 Offset

Register	Offset
SWREG68	110h

15.2.5.1.70.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER1_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER1_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.70.3 Fields

Field	Function
31-0 SW_REFER1_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 1

15.2.5.1.71 Base address LSB (bits 31:0) for reference luminance picture index 1 (SWREG69)

15.2.5.1.71.1 Offset

Register	Offset
SWREG69	114h

15.2.5.1.71.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER1_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER1_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.71.3 Fields

Field	Function
31-0 SW_REFER1_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 1

15.2.5.1.72 Base address MSB (bits 63:32) for reference luminance picture index 2 (SWREG70)

15.2.5.1.72.1 Offset

Register	Offset
SWREG70	118h

15.2.5.1.72.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER2_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER2_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.72.3 Fields

Field	Function
31-0 SW_REFER2_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 2

15.2.5.1.73 Base address LSB (bits 31:0) for reference luminance picture index 2 (SWREG71)

15.2.5.1.73.1 Offset

Register	Offset
SWREG71	11Ch

15.2.5.1.73.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER2_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER2_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.73.3 Fields

Field	Function
31-0 SW_REFER2_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 2

15.2.5.1.74 Base address MSB (bits 63:32) for reference luminance picture index 3 (SWREG72)

15.2.5.1.74.1 Offset

Register	Offset
SWREG72	120h

15.2.5.1.74.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER3_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER3_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.74.3 Fields

Field	Function
31-0 SW_REFER3_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 3

15.2.5.1.75 Base address LSB (bits 31:0) for reference luminance picture index 3 (SWREG73)

15.2.5.1.75.1 Offset

Register	Offset
SWREG73	124h

15.2.5.1.75.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER3_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER3_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.75.3 Fields

Field	Function
31-0 SW_REFER3_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 3

15.2.5.1.76 Base address MSB (bits 63:32) for reference luminance picture index 4 (SWREG74)

15.2.5.1.76.1 Offset

Register	Offset
SWREG74	128h

15.2.5.1.76.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER4_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER4_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.76.3 Fields

Field	Function
31-0 SW_REFER4_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 4

15.2.5.1.77 Base address LSB (bits 31:0) for reference luminance picture index 4 (SWREG75)

15.2.5.1.77.1 Offset

Register	Offset
SWREG75	12Ch

15.2.5.1.77.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER4_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER4_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.77.3 Fields

Field	Function
31-0 SW_REFER4_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 4

15.2.5.1.78 Base address MSB (bits 63:32) for reference luminance picture index 5 (SWREG76)

15.2.5.1.78.1 Offset

Register	Offset
SWREG76	130h

15.2.5.1.78.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER5_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER5_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.78.3 Fields

Field	Function
31-0 SW_REFER5_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 5

15.2.5.1.79 Base address LSB (bits 31:0) for reference luminance picture index 5 (SWREG77)

15.2.5.1.79.1 Offset

Register	Offset
SWREG77	134h

15.2.5.1.79.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER5_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER5_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.79.3 Fields

Field	Function
31-0 SW_REFER5_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 5

15.2.5.1.80 Base address MSB (bits 63:32) for reference luminance picture index 6 /VP9 segment write base MSB (SWREG78)

15.2.5.1.80.1 Offset

Register	Offset
SWREG78	138h

15.2.5.1.80.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BASE_ADDR_6_MSB_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BASE_ADDR_6_MSB_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.80.3 Fields

Field	Function
31-0 BASE_ADDR_6_MSB_BF	For HEVC: [31:0] - sw_refer6_ybase_msb - Base address MSB (bits 63:32) for reference luminance picture index 6 For VP9: [31:0] - sw_segment_write_base_msb - VP9 segment write base MSB

15.2.5.1.81 Base address LSB (bits 31:0) for reference luminance picture index 6 /VP9 segment write base LSB (SWREG79)

15.2.5.1.81.1 Offset

Register	Offset
SWREG79	13Ch

15.2.5.1.81.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BASE_ADDR_6_LSB_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BASE_ADDR_6_LSB_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.81.3 Fields

Field	Function
31-0 BASE_ADDR_6_LSB_BF	For HEVC: [31:0] - sw_refer6_ybase_lsb - Base address LSB (bits 31:0) for reference luminance picture index 6 For VP9: [31:0] - sw_segment_write_base_lsb - VP9 segment write base LSB

15.2.5.1.82 Base address MSB (bits 63:32) for reference luminance picture index 7 /VP9 segment read base MSB (SWREG80)

15.2.5.1.82.1 Offset

Register	Offset
SWREG80	140h

15.2.5.1.82.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BASE_ADDR_7_MSB_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BASE_ADDR_7_MSB_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.82.3 Fields

Field	Function
31-0 BASE_ADDR_7_MSB_BF	For HEVC: [31:0] - sw_refer7_ybase_msb - Base address MSB (bits 63:32) for reference luminance picture index 7 For VP9: [31:0] - sw_segment_read_base_msb - VP9 segment read base MSB

15.2.5.1.83 Base address LSB (bits 31:0) for reference luminance picture index 7 /VP9 segment read base LSB (SWREG81)

15.2.5.1.83.1 Offset

Register	Offset
SWREG81	144h

15.2.5.1.83.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BASE_ADDR_7_LSB_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BASE_ADDR_7_LSB_BF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.83.3 Fields

Field	Function
31-0 BASE_ADDR_7_LSB_BF	For HEVC: [31:0] - sw_refer7_ybase_lsb - Base address LSB (bits 31:0) for reference luminance picture index 7 For VP9: [31:0] - sw_segment_read_base_lsb - VP9 segment read base LSB

15.2.5.1.84 Base address MSB (bits 63:32) for reference luminance picture index 8 (SWREG82)

15.2.5.1.84.1 Offset

Register	Offset
SWREG82	148h

15.2.5.1.84.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER8_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER8_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.84.3 Fields

Field	Function
31-0 SW_REFER8_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 8

15.2.5.1.85 Base address LSB (bits 31:0) for reference luminance picture index 8 (SWREG83)

15.2.5.1.85.1 Offset

Register	Offset
SWREG83	14Ch

15.2.5.1.85.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER8_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER8_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.85.3 Fields

Field	Function
31-0 SW_REFER8_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 8

15.2.5.1.86 Base address MSB (bits 63:32) for reference luminance picture index 9 (SWREG84)

15.2.5.1.86.1 Offset

Register	Offset
SWREG84	150h

15.2.5.1.86.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER9_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER9_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.86.3 Fields

Field	Function
31-0 SW_REFER9_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 9

15.2.5.1.87 Base address LSB (bits 31:0) for reference luminance picture index 9 (SWREG85)

15.2.5.1.87.1 Offset

Register	Offset
SWREG85	154h

15.2.5.1.87.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER9_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER9_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.87.3 Fields

Field	Function
31-0 SW_REFER9_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 9

15.2.5.1.88 Base address MSB (bits 63:32) for reference luminance picture index 10 (SWREG86)

15.2.5.1.88.1 Offset

Register	Offset
SWREG86	158h

15.2.5.1.88.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER10_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER10_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.88.3 Fields

Field	Function
31-0 SW_REFER10_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 10

15.2.5.1.89 Base address LSB (bits 31:0) for reference luminance picture index 10 (SWREG87)

15.2.5.1.89.1 Offset

Register	Offset
SWREG87	15Ch

15.2.5.1.89.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER10_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER10_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.89.3 Fields

Field	Function
31-0 SW_REFER10_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 10

15.2.5.1.90 Base address MSB (bits 63:32) for reference luminance picture index 11 (SWREG88)

15.2.5.1.90.1 Offset

Register	Offset
SWREG88	160h

15.2.5.1.90.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER11_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER11_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.90.3 Fields

Field	Function
31-0 SW_REFER11_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 11

15.2.5.1.91 Base address LSB (bits 31:0) for reference luminance picture index 11 (SWREG89)

15.2.5.1.91.1 Offset

Register	Offset
SWREG89	164h

15.2.5.1.91.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER11_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER11_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.91.3 Fields

Field	Function
31-0 SW_REFER11_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 11

15.2.5.1.92 Base address MSB (bits 63:32) for reference luminance picture index 12 (SWREG90)

15.2.5.1.92.1 Offset

Register	Offset
SWREG90	168h

15.2.5.1.92.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER12_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER12_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.92.3 Fields

Field	Function
31-0 SW_REFER12_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 12

15.2.5.1.93 Base address LSB (bits 31:0) for reference luminance picture index 12 (SWREG91)

15.2.5.1.93.1 Offset

Register	Offset
SWREG91	16Ch

15.2.5.1.93.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER12_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER12_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.93.3 Fields

Field	Function
31-0 SW_REFER12_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 12

15.2.5.1.94 Base address MSB (bits 63:32) for reference luminance picture index 13 (SWREG92)

15.2.5.1.94.1 Offset

Register	Offset
SWREG92	170h

15.2.5.1.94.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER13_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER13_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.94.3 Fields

Field	Function
31-0 SW_REFER13_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 13

15.2.5.1.95 Base address LSB (bits 31:0) for reference luminance picture index 13 (SWREG93)

15.2.5.1.95.1 Offset

Register	Offset
SWREG93	174h

15.2.5.1.95.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER13_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER13_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.95.3 Fields

Field	Function
31-0 SW_REFER13_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 13

15.2.5.1.96 Base address MSB (bits 63:32) for reference luminance picture index 14 (SWREG94)

15.2.5.1.96.1 Offset

Register	Offset
SWREG94	178h

15.2.5.1.96.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER14_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER14_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.96.3 Fields

Field	Function
31-0 SW_REFER14_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 14

15.2.5.1.97 Base address LSB (bits 31:0) for reference luminance picture index 14 (SWREG95)

15.2.5.1.97.1 Offset

Register	Offset
SWREG95	17Ch

15.2.5.1.97.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER14_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER14_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.97.3 Fields

Field	Function
31-0 SW_REFER14_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 14

15.2.5.1.98 Base address MSB (bits 63:32) for reference luminance picture index 15 (SWREG96)

15.2.5.1.98.1 Offset

Register	Offset
SWREG96	180h

15.2.5.1.98.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER15_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER15_YBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.98.3 Fields

Field	Function
31-0 SW_REFER15_YBASE_MSB	Base address MSB (bits 63:32) for reference luminance picture index 15

15.2.5.1.99 Base address LSB (bits 31:0) for reference luminance picture index 15 (SWREG97)

15.2.5.1.99.1 Offset

Register	Offset
SWREG97	184h

15.2.5.1.99.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER15_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER15_YBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.99.3 Fields

Field	Function
31-0 SW_REFER15_YBASE_LSB	Base address LSB (bits 31:0) for reference luminance picture index 15

15.2.5.1.100 Base address MSB (bits 63:32) for decoded chrominance picture (SWREG98)

15.2.5.1.100.1 Offset

Register	Offset
SWREG98	188h

15.2.5.1.100.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_OUT_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_OUT_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.100.3 Fields

Field	Function
31-0 SW_DEC_OUT_CBASE_MSB	Base address MSB (bits 64:32) for decoded chrominance picture

15.2.5.1.101 Base address LSB (bits 31:0) for decoded chrominance picture (SWREG99)

15.2.5.1.101.1 Offset

Register	Offset
SWREG99	18Ch

15.2.5.1.101.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_OUT_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_OUT_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.101.3 Fields

Field	Function
31-0 SW_DEC_OUT_CBASE_LSB	Base address LSB (bits 31:0) for decoded chrominance picture

15.2.5.1.102 Base address MSB (bits 63:32) for reference chrominance picture index 0 (SWREG100)

15.2.5.1.102.1 Offset

Register	Offset
SWREG100	190h

15.2.5.1.102.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER0_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER0_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.102.3 Fields

Field	Function
31-0 SW_REFER0_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 0

15.2.5.1.103 Base address LSB (bits 31:0) for reference chrominance picture index 0 (SWREG101)

15.2.5.1.103.1 Offset

Register	Offset
SWREG101	194h

15.2.5.1.103.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER0_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER0_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.103.3 Fields

Field	Function
31-0 SW_REFER0_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 0

15.2.5.1.104 Base address MSB (bits 63:32) for reference chrominance picture index 1 (SWREG102)

15.2.5.1.104.1 Offset

Register	Offset
SWREG102	198h

15.2.5.1.104.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER1_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER1_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.104.3 Fields

Field	Function
31-0 SW_REFER1_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 1

15.2.5.1.105 Base address LSB (bits 31:0) for reference chrominance picture index 1 (SWREG103)

15.2.5.1.105.1 Offset

Register	Offset
SWREG103	19Ch

15.2.5.1.105.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER1_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER1_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.105.3 Fields

Field	Function
31-0 SW_REFER1_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 1

15.2.5.1.106 Base address MSB (bits 63:32) for reference chrominance picture index 2 (SWREG104)

15.2.5.1.106.1 Offset

Register	Offset
SWREG104	1A0h

15.2.5.1.106.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER2_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER2_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.106.3 Fields

Field	Function
31-0 SW_REFER2_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 2

15.2.5.1.107 Base address LSB (bits 31:0) for reference chrominance picture index 2 (SWREG105)

15.2.5.1.107.1 Offset

Register	Offset
SWREG105	1A4h

15.2.5.1.107.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER2_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER2_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.107.3 Fields

Field	Function
31-0 SW_REFER2_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 2

15.2.5.1.108 Base address MSB (bits 63:32) for reference chrominance picture index 3 (SWREG106)

15.2.5.1.108.1 Offset

Register	Offset
SWREG106	1A8h

15.2.5.1.108.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER3_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER3_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.108.3 Fields

Field	Function
31-0 SW_REFER3_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 3

15.2.5.1.109 Base address LSB (bits 31:0) for reference chrominance picture index 3 (SWREG107)

15.2.5.1.109.1 Offset

Register	Offset
SWREG107	1ACh

15.2.5.1.109.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER3_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER3_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.109.3 Fields

Field	Function
31-0 SW_REFER3_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 3

15.2.5.1.110 Base address MSB (bits 63:32) for reference chrominance picture index 4 (SWREG108)

15.2.5.1.110.1 Offset

Register	Offset
SWREG108	1B0h

15.2.5.1.110.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER4_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER4_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.110.3 Fields

Field	Function
31-0 SW_REFER4_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 4

15.2.5.1.111 Base address LSB (bits 31:0) for reference chrominance picture index 4 (SWREG109)

15.2.5.1.111.1 Offset

Register	Offset
SWREG109	1B4h

15.2.5.1.111.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER4_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER4_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.111.3 Fields

Field	Function
31-0 SW_REFER4_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 4

15.2.5.1.112 Base address MSB (bits 63:32) for reference chrominance picture index 5 (SWREG110)

15.2.5.1.112.1 Offset

Register	Offset
SWREG110	1B8h

15.2.5.1.112.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER5_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER5_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.112.3 Fields

Field	Function
31-0 SW_REFER5_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 5

15.2.5.1.113 Base address LSB (bits 31:0) for reference chrominance picture index 5 (SWREG111)

15.2.5.1.113.1 Offset

Register	Offset
SWREG111	1BCh

15.2.5.1.113.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER5_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER5_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.113.3 Fields

Field	Function
31-0 SW_REFER5_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 5

15.2.5.1.114 Base address MSB (bits 63:32) for reference chrominance picture index 6 (SWREG112)

15.2.5.1.114.1 Offset

Register	Offset
SWREG112	1C0h

15.2.5.1.114.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER6_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER6_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.114.3 Fields

Field	Function
31-0 SW_REFER6_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 6

15.2.5.1.115 Base address LSB (bits 31:0) for reference chrominance picture index 6 (SWREG113)

15.2.5.1.115.1 Offset

Register	Offset
SWREG113	1C4h

15.2.5.1.115.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER6_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER6_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.115.3 Fields

Field	Function
31-0 SW_REFER6_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 6

15.2.5.1.116 Base address MSB (bits 63:32) for reference chrominance picture index 7 (SWREG114)

15.2.5.1.116.1 Offset

Register	Offset
SWREG114	1C8h

15.2.5.1.116.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER7_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER7_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.116.3 Fields

Field	Function
31-0 SW_REFER7_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 7

15.2.5.1.117 Base address LSB (bits 31:0) for reference chrominance picture index 7 (SWREG115)

15.2.5.1.117.1 Offset

Register	Offset
SWREG115	1CCh

15.2.5.1.117.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER7_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER7_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.117.3 Fields

Field	Function
31-0 SW_REFER7_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 7

15.2.5.1.118 Base address MSB (bits 63:32) for reference chrominance picture index 8 (SWREG116)

15.2.5.1.118.1 Offset

Register	Offset
SWREG116	1D0h

15.2.5.1.118.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER8_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER8_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.118.3 Fields

Field	Function
31-0 SW_REFER8_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 8

15.2.5.1.119 Base address LSB (bits 31:0) for reference chrominance picture index 8 (SWREG117)

15.2.5.1.119.1 Offset

Register	Offset
SWREG117	1D4h

15.2.5.1.119.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER8_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER8_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.119.3 Fields

Field	Function
31-0 SW_REFER8_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 8

15.2.5.1.120 Base address MSB (bits 63:32) for reference chrominance picture index 9 (SWREG118)

15.2.5.1.120.1 Offset

Register	Offset
SWREG118	1D8h

15.2.5.1.120.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER9_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER9_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.120.3 Fields

Field	Function
31-0 SW_REFER9_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 9

15.2.5.1.121 Base address LSB (bits 31:0) for reference chrominance picture index 9 (SWREG119)

15.2.5.1.121.1 Offset

Register	Offset
SWREG119	1DCh

15.2.5.1.121.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER9_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER9_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.121.3 Fields

Field	Function
31-0 SW_REFER9_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 9

15.2.5.1.122 Base address MSB (bits 63:32) for reference chrominance picture index 10 (SWREG120)

15.2.5.1.122.1 Offset

Register	Offset
SWREG120	1E0h

15.2.5.1.122.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER10_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER10_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.122.3 Fields

Field	Function
31-0 SW_REFER10_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 10

15.2.5.1.123 Base address LSB (bits 31:0) for reference chrominance picture index 10 (SWREG121)

15.2.5.1.123.1 Offset

Register	Offset
SWREG121	1E4h

15.2.5.1.123.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER10_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER10_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.123.3 Fields

Field	Function
31-0 SW_REFER10_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 10

15.2.5.1.124 Base address MSB (bits 63:32) for reference chrominance picture index 11 (SWREG122)

15.2.5.1.124.1 Offset

Register	Offset
SWREG122	1E8h

15.2.5.1.124.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER11_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER11_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.124.3 Fields

Field	Function
31-0 SW_REFER11_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 11

15.2.5.1.125 Base address LSB (bits 31:0) for reference chrominance picture index 11 (SWREG123)

15.2.5.1.125.1 Offset

Register	Offset
SWREG123	1ECh

15.2.5.1.125.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER11_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER11_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.125.3 Fields

Field	Function
31-0 SW_REFER11_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 11

15.2.5.1.126 Base address MSB (bits 63:32) for reference chrominance picture index 12 (SWREG124)

15.2.5.1.126.1 Offset

Register	Offset
SWREG124	1F0h

15.2.5.1.126.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER12_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER12_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.126.3 Fields

Field	Function
31-0 SW_REFER12_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 12

15.2.5.1.127 Base address LSB (bits 31:0) for reference chrominance picture index 12 (SWREG125)

15.2.5.1.127.1 Offset

Register	Offset
SWREG125	1F4h

15.2.5.1.127.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER12_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER12_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.127.3 Fields

Field	Function
31-0 SW_REFER12_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 12

15.2.5.1.128 Base address MSB (bits 63:32) for reference chrominance picture index 13 (SWREG126)

15.2.5.1.128.1 Offset

Register	Offset
SWREG126	1F8h

15.2.5.1.128.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER13_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER13_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.128.3 Fields

Field	Function
31-0 SW_REFER13_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 13

15.2.5.1.129 Base address LSB (bits 31:0) for reference chrominance picture index 13 (SWREG127)

15.2.5.1.129.1 Offset

Register	Offset
SWREG127	1FCh

15.2.5.1.129.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER13_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER13_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.129.3 Fields

Field	Function
31-0 SW_REFER13_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 13

15.2.5.1.130 Base address MSB (bits 63:32) for reference chrominance picture index 14 (SWREG128)

15.2.5.1.130.1 Offset

Register	Offset
SWREG128	200h

15.2.5.1.130.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER14_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER14_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.130.3 Fields

Field	Function
31-0 SW_REFER14_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 14

15.2.5.1.131 Base address LSB (bits 31:0) for reference chrominance picture index 14 (SWREG129)

15.2.5.1.131.1 Offset

Register	Offset
SWREG129	204h

15.2.5.1.131.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER14_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER14_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.131.3 Fields

Field	Function
31-0 SW_REFER14_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 14

15.2.5.1.132 Base address MSB (bits 63:32) for reference chrominance picture index 15 (SWREG130)

15.2.5.1.132.1 Offset

Register	Offset
SWREG130	208h

15.2.5.1.132.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER15_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER15_CBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.132.3 Fields

Field	Function
31-0 SW_REFER15_CBASE_MSB	Base address MSB (bits 63:32) for reference chrominance picture index 15

15.2.5.1.133 Base address LSB (bits 31:0) for reference chrominance picture index 15 (SWREG131)

15.2.5.1.133.1 Offset

Register	Offset
SWREG131	20Ch

15.2.5.1.133.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER15_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER15_CBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.133.3 Fields

Field	Function
31-0 SW_REFER15_CBASE_LSB	Base address LSB (bits 31:0) for reference chrominance picture index 15

15.2.5.1.134 Base address MSB (bits 63:32) for decoded direct mode MVS (SWREG132)

15.2.5.1.134.1 Offset

Register	Offset
SWREG132	210h

15.2.5.1.134.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_OUT_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_OUT_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.134.3 Fields

Field	Function
31-0 SW_DEC_OUT_DBASE_MSB	Base address MSB (bits 63:32) for decoded direct mode MVS

15.2.5.1.135 Base address LSB (bits 31:0) for decoded direct mode MVS (SWREG133)

15.2.5.1.135.1 Offset

Register	Offset
SWREG133	214h

15.2.5.1.135.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_OUT_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_OUT_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.135.3 Fields

Field	Function
31-0 SW_DEC_OUT_DBASE_LSB	Base address LSB (bits 31:0) for decoded direct mode MVS

15.2.5.1.136 Base address MSB (bits 63:32) for reference direct mode MVS index 0 (SWREG134)

15.2.5.1.136.1 Offset

Register	Offset
SWREG134	218h

15.2.5.1.136.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER0_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER0_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.136.3 Fields

Field	Function
31-0 SW_REFER0_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 0

15.2.5.1.137 Base address LSB (bits 31:0) for reference direct mode MVS index 0 (SWREG135)

15.2.5.1.137.1 Offset

Register	Offset
SWREG135	21Ch

15.2.5.1.137.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER0_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER0_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.137.3 Fields

Field	Function
31-0 SW_REFER0_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 0

15.2.5.1.138 Base address MSB (bits 63:32) for reference direct mode MVS index 1 (SWREG136)

15.2.5.1.138.1 Offset

Register	Offset
SWREG136	220h

15.2.5.1.138.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER1_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER1_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.138.3 Fields

Field	Function
31-0 SW_REFER1_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 1

15.2.5.1.139 Base address LSB (bits 31:0) for reference direct mode MVS index 1 (SWREG137)

15.2.5.1.139.1 Offset

Register	Offset
SWREG137	224h

15.2.5.1.139.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER1_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER1_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.139.3 Fields

Field	Function
31-0 SW_REFER1_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 1

15.2.5.1.140 Base address MSB (bits 63:32) for reference direct mode MVS index 2 (SWREG138)

15.2.5.1.140.1 Offset

Register	Offset
SWREG138	228h

15.2.5.1.140.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER2_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER2_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.140.3 Fields

Field	Function
31-0 SW_REFER2_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 2

15.2.5.1.141 Base address LSB (bits 31:0) for reference direct mode MVS index 2 (SWREG139)

15.2.5.1.141.1 Offset

Register	Offset
SWREG139	22Ch

15.2.5.1.141.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER2_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER2_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.141.3 Fields

Field	Function
31-0 SW_REFER2_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 2

15.2.5.1.142 Base address MSB (bits 63:32) for reference direct mode MVS index 3 (SWREG140)

15.2.5.1.142.1 Offset

Register	Offset
SWREG140	230h

15.2.5.1.142.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER3_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER3_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.142.3 Fields

Field	Function
31-0 SW_REFER3_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 3

15.2.5.1.143 Base address LSB (bits 31:0) for reference direct mode MVS index 3 (SWREG141)

15.2.5.1.143.1 Offset

Register	Offset
SWREG141	234h

15.2.5.1.143.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER3_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER3_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.143.3 Fields

Field	Function
31-0 SW_REFER3_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 3

15.2.5.1.144 Base address MSB (bits 63:32) for reference direct mode MVS index 4 (SWREG142)

15.2.5.1.144.1 Offset

Register	Offset
SWREG142	238h

15.2.5.1.144.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER4_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER4_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.144.3 Fields

Field	Function
31-0 SW_REFER4_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 4

15.2.5.1.145 Base address LSB (bits 31:0) for reference direct mode MVS index 4 (SWREG143)

15.2.5.1.145.1 Offset

Register	Offset
SWREG143	23Ch

15.2.5.1.145.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER4_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER4_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.145.3 Fields

Field	Function
31-0 SW_REFER4_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 4

15.2.5.1.146 Base address MSB (bits 63:32) for reference direct mode MVS index 5 (SWREG144)

15.2.5.1.146.1 Offset

Register	Offset
SWREG144	240h

15.2.5.1.146.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER5_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER5_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.146.3 Fields

Field	Function
31-0 SW_REFER5_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 5

15.2.5.1.147 Base address LSB (bits 31:0) for reference direct mode MVS index 5 (SWREG145)

15.2.5.1.147.1 Offset

Register	Offset
SWREG145	244h

15.2.5.1.147.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER5_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER5_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.147.3 Fields

Field	Function
31-0 SW_REFER5_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 5

15.2.5.1.148 Base address MSB (bits 63:32) for reference direct mode MVS index 6 (SWREG146)

15.2.5.1.148.1 Offset

Register	Offset
SWREG146	248h

15.2.5.1.148.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER6_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER6_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.148.3 Fields

Field	Function
31-0 SW_REFER6_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 6

15.2.5.1.149 Base address LSB (bits 31:0) for reference direct mode MVS index 6 (SWREG147)

15.2.5.1.149.1 Offset

Register	Offset
SWREG147	24Ch

15.2.5.1.149.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER6_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER6_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.149.3 Fields

Field	Function
31-0 SW_REFER6_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 6

15.2.5.1.150 Base address MSB (bits 63:32) for reference direct mode MVS index 7 (SWREG148)

15.2.5.1.150.1 Offset

Register	Offset
SWREG148	250h

15.2.5.1.150.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER7_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER7_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.150.3 Fields

Field	Function
31-0 SW_REFER7_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 7

15.2.5.1.151 Base address LSB (bits 31:0) for reference direct mode MVS index 7 (SWREG149)

15.2.5.1.151.1 Offset

Register	Offset
SWREG149	254h

15.2.5.1.151.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER7_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER7_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.151.3 Fields

Field	Function
31-0 SW_REFER7_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 7

15.2.5.1.152 Base address MSB (bits 63:32) for reference direct mode MVS index 8 (SWREG150)

15.2.5.1.152.1 Offset

Register	Offset
SWREG150	258h

15.2.5.1.152.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER8_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER8_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.152.3 Fields

Field	Function
31-0 SW_REFER8_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 8

15.2.5.1.153 Base address LSB (bits 31:0) for reference direct mode MVS index 8 (SWREG151)

15.2.5.1.153.1 Offset

Register	Offset
SWREG151	25Ch

15.2.5.1.153.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER8_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER8_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.153.3 Fields

Field	Function
31-0 SW_REFER8_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode mode MVS index 8

15.2.5.1.154 Base address MSB (bits 63:32) for reference direct mode mode MVS index 9 (SWREG152)

15.2.5.1.154.1 Offset

Register	Offset
SWREG152	260h

15.2.5.1.154.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER9_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER9_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.154.3 Fields

Field	Function
31-0 SW_REFER9_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 9

15.2.5.1.155 Base address LSB (bits 31:0) for reference direct mode mode MVS index 9 (SWREG153)

15.2.5.1.155.1 Offset

Register	Offset
SWREG153	264h

15.2.5.1.155.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER9_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER9_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.155.3 Fields

Field	Function
31-0 SW_REFER9_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode mode MVS index 9

15.2.5.1.156 Base address MSB (bits 63:32) for reference direct mode MVS index 10 (SWREG154)

15.2.5.1.156.1 Offset

Register	Offset
SWREG154	268h

15.2.5.1.156.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER10_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER10_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.156.3 Fields

Field	Function
31-0 SW_REFER10_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 10

15.2.5.1.157 Base address LSB (bits 31:0) for reference direct mode MVS index 10 (SWREG155)

15.2.5.1.157.1 Offset

Register	Offset
SWREG155	26Ch

15.2.5.1.157.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER10_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER10_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.157.3 Fields

Field	Function
31-0 SW_REFER10_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 10

15.2.5.1.158 Base address MSB (bits 63:32) for reference direct mode MVS index 11 (SWREG156)

15.2.5.1.158.1 Offset

Register	Offset
SWREG156	270h

15.2.5.1.158.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER11_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER11_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.158.3 Fields

Field	Function
31-0 SW_REFER11_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 11

15.2.5.1.159 Base address LSB (bits 31:0) for reference direct mode MVS index 11 (SWREG157)

15.2.5.1.159.1 Offset

Register	Offset
SWREG157	274h

15.2.5.1.159.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER11_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER11_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.159.3 Fields

Field	Function
31-0 SW_REFER11_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 11

15.2.5.1.160 Base address MSB (bits 63:32) for reference direct mode MVS index 12 (SWREG158)

15.2.5.1.160.1 Offset

Register	Offset
SWREG158	278h

15.2.5.1.160.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER12_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER12_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.160.3 Fields

Field	Function
31-0 SW_REFER12_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 12

15.2.5.1.161 Base address LSB (bits 31:0) for reference direct mode MVS index 12 (SWREG159)

15.2.5.1.161.1 Offset

Register	Offset
SWREG159	27Ch

15.2.5.1.161.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER12_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER12_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.161.3 Fields

Field	Function
31-0 SW_REFER12_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 12

15.2.5.1.162 Base address MSB (bits 63:32) for reference direct mode MVS index 13 (SWREG160)

15.2.5.1.162.1 Offset

Register	Offset
SWREG160	280h

15.2.5.1.162.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER13_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER13_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.162.3 Fields

Field	Function
31-0 SW_REFER13_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 13

15.2.5.1.163 Base address LSB (bits 31:0) for reference direct mode MVS index 13 (SWREG161)

15.2.5.1.163.1 Offset

Register	Offset
SWREG161	284h

15.2.5.1.163.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER13_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER13_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.163.3 Fields

Field	Function
31-0 SW_REFER13_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 13

15.2.5.1.164 Base address MSB (bits 63:32) for reference direct mode MVS index 14 (SWREG162)

15.2.5.1.164.1 Offset

Register	Offset
SWREG162	288h

15.2.5.1.164.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER14_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER14_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.164.3 Fields

Field	Function
31-0 SW_REFER14_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 14

15.2.5.1.165 Base address LSB (bits 31:0) for reference direct mode MVS index 14 (SWREG163)

15.2.5.1.165.1 Offset

Register	Offset
SWREG163	28Ch

15.2.5.1.165.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER14_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER14_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.165.3 Fields

Field	Function
31-0 SW_REFER14_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 14

15.2.5.1.166 Base address MSB (bits 63:32) for reference direct mode MVS index 15 (SWREG164)

15.2.5.1.166.1 Offset

Register	Offset
SWREG164	290h

15.2.5.1.166.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER15_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER15_DBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.166.3 Fields

Field	Function
31-0 SW_REFER15_DBASE_MSB	Base address MSB (bits 63:32) for reference direct mode MVS index 15

15.2.5.1.167 Base address LSB (bits 31:0) for reference direct mode MVS index 15 (SWREG165)

15.2.5.1.167.1 Offset

Register	Offset
SWREG165	294h

15.2.5.1.167.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER15_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER15_DBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.167.3 Fields

Field	Function
31-0 SW_REFER15_DBASE_LSB	Base address LSB (bits 31:0) for reference direct mode MVS index 15

15.2.5.1.168 Base address MSB (bits 63:32) for tile sizes (SWREG166)

15.2.5.1.168.1 Offset

Register	Offset
SWREG166	298h

15.2.5.1.168.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_TILE_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_TILE_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.168.3 Fields

Field	Function
31-0 SW_TILE_BASE_MSB	Base address MSB (bits 63:32) for tile sizes

15.2.5.1.169 Base address LSB (bits 31:0) for tile sizes (SWREG167)

15.2.5.1.169.1 Offset

Register	Offset
SWREG167	29Ch

15.2.5.1.169.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_TILE_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_TILE_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.169.3 Fields

Field	Function
31-0 SW_TILE_BASE_LSB	Base address LSB (bits 31:0) for tile sizes

15.2.5.1.170 Base address MSB (bits 63:32) for / stream start address/ decoded end addr register (SWREG168)

15.2.5.1.170.1 Offset

Register	Offset
SWREG168	2A0h

15.2.5.1.170.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_STREAM_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_STREAM_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.170.3 Fields

Field	Function
31-0 SW_STREAM_BASE_MSB	Base address MSB (bits 63:32) for / stream start address/decoded end addr register

15.2.5.1.171 Base address LSB (bits 31:0) for / stream start address/ decoded end addr register (SWREG169)

15.2.5.1.171.1 Offset

Register	Offset
SWREG169	2A4h

15.2.5.1.171.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_STREAM_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_STREAM_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.171.3 Fields

Field	Function
31-0 SW_STREAM_BASE_LSB	Base address LSB (bits 31:0) for / stream start address/decoded end addr register

15.2.5.1.172 Base address MSB (bits 63:32) for scaling lists / VP9 CTX counter values (SWREG170)

15.2.5.1.172.1 Offset

Register	Offset
SWREG170	2A8h

15.2.5.1.172.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_SCALE_LIST_CTX_COUNTER_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_SCALE_LIST_CTX_COUNTER_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.172.3 Fields

Field	Function
31-0 SW_SCALE_LIST_CTX_COUNTER_BASE_MSB	HEVC: Base address MSB (bits 63:32) for scaling lists VP9: CTX counter values

15.2.5.1.173 Base address LSB (bits 31:0) for scaling lists / VP9 CTX counter values (SWREG171)

15.2.5.1.173.1 Offset

Register	Offset
SWREG171	2ACh

15.2.5.1.173.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_SCALE_LISTT_CTX_COUNTER_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_SCALE_LISTT_CTX_COUNTER_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.173.3 Fields

Field	Function
31-0 SW_SCALE_LISTT_CTX_COUNTER_BASE_LSB	HEVC: Base address LSB (bits 31:0) for scaling lists VP9: CTX counter values

15.2.5.1.174 Base address MSB (bits 63:32) for stream propability tables (SWREG172)

15.2.5.1.174.1 Offset

Register	Offset
SWREG172	2B0h

15.2.5.1.174.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_PROB_TAB_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_PROB_TAB_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.174.3 Fields

Field	Function
31-0 SW_PROB_TAB_BASE_MSB	Base address MSB (bits 63:32) for stream propability tables

15.2.5.1.175 Base address LSB (bits 31:0) for stream propability tables (SWREG173)

15.2.5.1.175.1 Offset

Register	Offset
SWREG173	2B4h

15.2.5.1.175.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_PROB_TAB_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_PROB_TAB_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.175.3 Fields

Field	Function
31-0 SW_PROB_TAB_BASE_LSB	Base address LSB (bits 31:0) for stream propability tables

15.2.5.1.176 Base address MSB (bits 63:32) for decoder output raster scan Y picture (SWREG174)

15.2.5.1.176.1 Offset

Register	Offset
SWREG174	2B8h

15.2.5.1.176.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_RSY_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_RSY_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.176.3 Fields

Field	Function
31-0 SW_DEC_RSY_BASE_MSB	Base address MSB (bits 63:32) for decoder output raster scan Y picture

15.2.5.1.177 Base address LSB (bits 31:0) for decoder output raster scan Y picture (SWREG175)

15.2.5.1.177.1 Offset

Register	Offset
SWREG175	2BCh

15.2.5.1.177.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_RSY_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_RSY_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.177.3 Fields

Field	Function
31-0 SW_DEC_RSY_BASE_LSB	Base address LSB (bits 31:0) for decoder output raster scan Y picture

15.2.5.1.178 Base address MSB (bits 63:32) for decoder output raster scan C picture (SWREG176)

15.2.5.1.178.1 Offset

Register	Offset
SWREG176	2C0h

15.2.5.1.178.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_RSC_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_RSC_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.178.3 Fields

Field	Function
31-0 SW_DEC_RSC_BASE_MSB	Base address MSB (bits 63:32) for decoder output raster scan C picture

15.2.5.1.179 Base address LSB (bits 31:0) for decoder output raster scan C picture (SWREG177)

15.2.5.1.179.1 Offset

Register	Offset
SWREG177	2C4h

15.2.5.1.179.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_RSC_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_RSC_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.179.3 Fields

Field	Function
31-0 SW_DEC_RSC_BASE_LSB	Base address LSB (bits 31:0) for decoder output raster scan C picture

15.2.5.1.180 Base address MSB (bits 63:32) for tile border coeffients of filter (SWREG178)

15.2.5.1.180.1 Offset

Register	Offset
SWREG178	2C8h

15.2.5.1.180.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_VERT_FILT_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_VERT_FILT_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.180.3 Fields

Field	Function
31-0 SW_DEC_VERT_FILT_BASE_MSB	Base address MSB to store/read filtering coeffients of current picture at tile border.

15.2.5.1.181 Base address LSB (bits 31:0) for tile border coeffients of filter (SWREG179)

15.2.5.1.181.1 Offset

Register	Offset
SWREG179	2CCh

15.2.5.1.181.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_VERT_FILT_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_VERT_FILT_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.181.3 Fields

Field	Function
31-0 SW_DEC_VERT_FILT_BASE_LSB	Base address LSB to store/read filtering coeffients of current picture at tile border.

15.2.5.1.182 Base address MSB (bits 63:32) for tile border coeffients of sao (SWREG180)

15.2.5.1.182.1 Offset

Register	Offset
SWREG180	2D0h

15.2.5.1.182.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_VERT_SAO_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_VERT_SAO_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.182.3 Fields

Field	Function
31-0 SW_DEC_VERT_SAO_BASE_MSB	Base address MSB to store/read sao coeffients of current picture at tile border.

15.2.5.1.183 Base address LSB (bits 31:0) for tile border coeffients of sao (SWREG181)

15.2.5.1.183.1 Offset

Register	Offset
SWREG181	2D4h

15.2.5.1.183.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_VERT_SAO_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_VERT_SAO_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.183.3 Fields

Field	Function
31-0 SW_DEC_VERT_SAO_BASE_LSB	Base address LSB to store/read sao coeffients of current picture at tile border.

15.2.5.1.184 Base address MSB (bits 63:32) for tile border bsd control data (SWREG182)

15.2.5.1.184.1 Offset

Register	Offset
SWREG182	2D8h

15.2.5.1.184.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_BSD_CTRL_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_BSD_CTRL_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.184.3 Fields

Field	Function
31-0 SW_DEC_BSD_CTRL_BASE_MSB	Base address MSB to store/read BSD control data of current picture at tile border.

15.2.5.1.185 Base address LSB (bits 31:0) for tile border bsd control data (SWREG183)

15.2.5.1.185.1 Offset

Register	Offset
SWREG183	2DCh

15.2.5.1.185.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_BSD_CTRL_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_BSD_CTRL_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.185.3 Fields

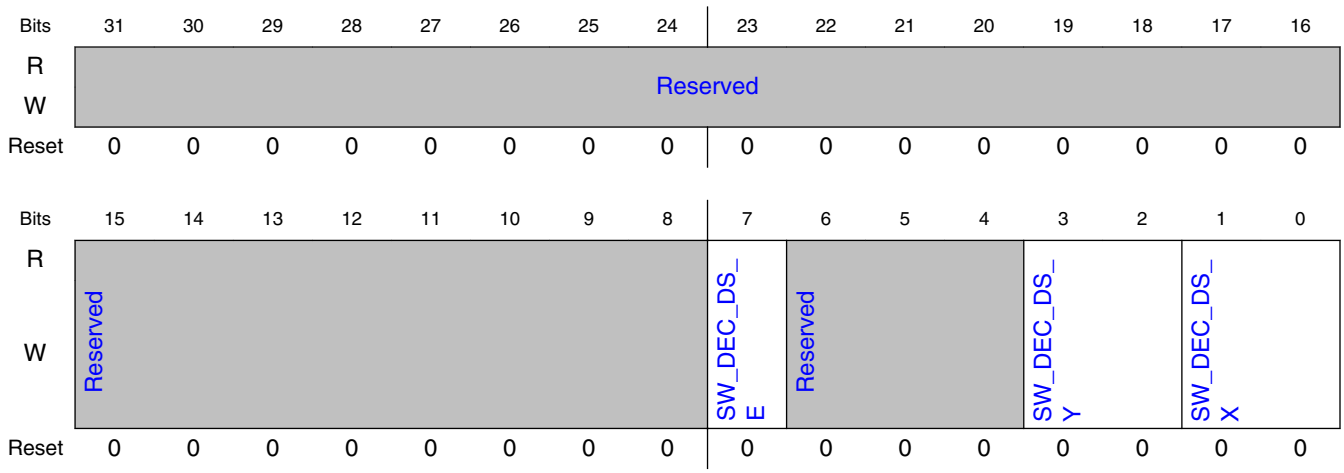
Field	Function
31-0 SW_DEC_BSD_CTRL_BASE_LSB	Base address LSB to store/read BSD control data of current picture at tile border.

15.2.5.1.186 Raster scan down scale control register MSM (SWREG184)

15.2.5.1.186.1 Offset

Register	Offset
SWREG184	2E0h

15.2.5.1.186.2 Diagram



15.2.5.1.186.3 Fields

Field	Function
31-8 —	Reserved.
7 SW_DEC_DS_E	Raster scan down scale enable 0b - Disable 1b - Enable
6-4 —	Reserved.
3-2 SW_DEC_DS_Y	Y coordinate down scale times for raster scan output picture data 00b - 1/2 01b - 1/4 10b - 1/8
1-0 SW_DEC_DS_X	X coordinate down scale times for raster scan output picture data 00b - 1/2 01b - 1/4 10b - 1/8

15.2.5.1.187 Base address MSB (bits 63:32) for decoder output raster scan down scale Y picture (SWREG185)

15.2.5.1.187.1 Offset

Register	Offset
SWREG185	2E4h

15.2.5.1.187.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_DSY_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_DSY_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.187.3 Fields

Field	Function
31-0 SW_DEC_DSY_BASE_MSB	Base address MSB (bits 63:32) for decoder output raster scan down scale Y picture

15.2.5.1.188 Base address LSB (bits 31:0) for decoder output raster scan down scale Y picture (SWREG186)

15.2.5.1.188.1 Offset

Register	Offset
SWREG186	2E8h

15.2.5.1.188.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_DSY_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_DSY_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.188.3 Fields

Field	Function
31-0 SW_DEC_DSY_BASE_LSB	Base address LSB (bits 31:0) for decoder output raster scan down scale Y picture

15.2.5.1.189 Base address MSB (bits 63:32) for decoder output raster scan down scale C picture (SWREG187)

15.2.5.1.189.1 Offset

Register	Offset
SWREG187	2ECh

15.2.5.1.189.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_DSC_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_DSC_BASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.189.3 Fields

Field	Function
31-0 SW_DEC_DSC_BASE_MSB	Base address MSB (bits 63:32) for decoder output raster scan down scale C picture

15.2.5.1.190 Base address LSB (bits 31:0) for decoder output raster scan down scale C picture (SWREG188)

15.2.5.1.190.1 Offset

Register	Offset
SWREG188	2F0h

15.2.5.1.190.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_DSC_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_DSC_BASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.190.3 Fields

Field	Function
31-0 SW_DEC_DSC_BASE_LSB	Base address LSB (bits 31:0) for decoder output raster scan down scale C picture

15.2.5.1.191 Base address MSB (bits 63:32) for decoder output compress luminance table (SWREG189)

15.2.5.1.191.1 Offset

Register	Offset
SWREG189	2F4h

15.2.5.1.191.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_OUT_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_OUT_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.191.3 Fields

Field	Function
31-0 SW_DEC_OUT_TYBASE_MSB	Base address MSB (bits 63:32) for decoder output compress luminance table

15.2.5.1.192 Base address LSB (bits 31:0) for decoder output compress luminance table (SWREG190)

15.2.5.1.192.1 Offset

Register	Offset
SWREG190	2F8h

15.2.5.1.192.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_OUT_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_OUT_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.192.3 Fields

Field	Function
31-0 SW_DEC_OUT_TYBASE_LSB	Base address LSB (bits 31:0) for decoder output compress luminance table

15.2.5.1.193 Base address MSB (bits 63:32) for reference compress luminance table index 0 (SWREG191)

15.2.5.1.193.1 Offset

Register	Offset
SWREG191	2FCh

15.2.5.1.193.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER0_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER0_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.193.3 Fields

Field	Function
31-0 SW_REFER0_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 0

15.2.5.1.194 Base address LSB (bits 31:0) for reference compress luminance table index 0 (SWREG192)

15.2.5.1.194.1 Offset

Register	Offset
SWREG192	300h

15.2.5.1.194.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER0_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER0_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.194.3 Fields

Field	Function
31-0 SW_REFER0_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 0

15.2.5.1.195 Base address MSB (bits 63:32) for reference compress luminance table index 1 (SWREG193)

15.2.5.1.195.1 Offset

Register	Offset
SWREG193	304h

15.2.5.1.195.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER1_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER1_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.195.3 Fields

Field	Function
31-0 SW_REFER1_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 1

15.2.5.1.196 Base address LSB (bits 31:0) for reference compress luminance table index 1 (SWREG194)

15.2.5.1.196.1 Offset

Register	Offset
SWREG194	308h

15.2.5.1.196.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER1_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER1_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.196.3 Fields

Field	Function
31-0 SW_REFER1_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 1

15.2.5.1.197 Base address MSB (bits 63:32) for reference compress luminance table index 2 (SWREG195)

15.2.5.1.197.1 Offset

Register	Offset
SWREG195	30Ch

15.2.5.1.197.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER2_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER2_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.197.3 Fields

Field	Function
31-0 SW_REFER2_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 2

15.2.5.1.198 Base address LSB (bits 31:0) for reference compress luminance table index 2 (SWREG196)

15.2.5.1.198.1 Offset

Register	Offset
SWREG196	310h

15.2.5.1.198.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER2_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER2_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.198.3 Fields

Field	Function
31-0 SW_REFER2_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 2

15.2.5.1.199 Base address MSB (bits 63:32) for reference compress luminance table index 3 (SWREG197)

15.2.5.1.199.1 Offset

Register	Offset
SWREG197	314h

15.2.5.1.199.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER3_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER3_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.199.3 Fields

Field	Function
31-0 SW_REFER3_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 3

15.2.5.1.200 Base address LSB (bits 31:0) for reference compress luminance table index 3 (SWREG198)

15.2.5.1.200.1 Offset

Register	Offset
SWREG198	318h

15.2.5.1.200.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER3_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER3_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.200.3 Fields

Field	Function
31-0 SW_REFER3_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 3

15.2.5.1.201 Base address MSB (bits 63:32) for reference compress luminance table index 4 (SWREG199)

15.2.5.1.201.1 Offset

Register	Offset
SWREG199	31Ch

15.2.5.1.201.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER4_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER4_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.201.3 Fields

Field	Function
31-0 SW_REFER4_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 4

15.2.5.1.202 Base address LSB (bits 31:0) for reference compress luminance table index 4 (SWREG200)

15.2.5.1.202.1 Offset

Register	Offset
SWREG200	320h

15.2.5.1.202.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER4_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER4_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.202.3 Fields

Field	Function
31-0 SW_REFER4_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 4

15.2.5.1.203 Base address MSB (bits 63:32) for reference compress luminance table index 5 (SWREG201)

15.2.5.1.203.1 Offset

Register	Offset
SWREG201	324h

15.2.5.1.203.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER5_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER5_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.203.3 Fields

Field	Function
31-0 SW_REFER5_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 5

15.2.5.1.204 Base address LSB (bits 31:0) for reference compress luminance table index 5 (SWREG202)

15.2.5.1.204.1 Offset

Register	Offset
SWREG202	328h

15.2.5.1.204.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER5_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER5_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.204.3 Fields

Field	Function
31-0 SW_REFER5_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 5

15.2.5.1.205 Base address MSB (bits 63:32) for reference compress luminance table index 6 (SWREG203)

15.2.5.1.205.1 Offset

Register	Offset
SWREG203	32Ch

15.2.5.1.205.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER6_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER6_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.205.3 Fields

Field	Function
31-0 SW_REFER6_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 6

15.2.5.1.206 Base address LSB (bits 31:0) for reference compress luminance table index 6 (SWREG204)

15.2.5.1.206.1 Offset

Register	Offset
SWREG204	330h

15.2.5.1.206.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER6_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER6_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.206.3 Fields

Field	Function
31-0 SW_REFER6_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 6

15.2.5.1.207 Base address MSB (bits 63:32) for reference compress luminance table index 7 (SWREG205)

15.2.5.1.207.1 Offset

Register	Offset
SWREG205	334h

15.2.5.1.207.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER7_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER7_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.207.3 Fields

Field	Function
31-0 SW_REFER7_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 7

15.2.5.1.208 Base address LSB (bits 31:0) for reference compress luminance table index 7 (SWREG206)

15.2.5.1.208.1 Offset

Register	Offset
SWREG206	338h

15.2.5.1.208.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER7_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER7_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.208.3 Fields

Field	Function
31-0 SW_REFER7_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 7

15.2.5.1.209 Base address MSB (bits 63:32) for reference compress luminance table index 8 (SWREG207)

15.2.5.1.209.1 Offset

Register	Offset
SWREG207	33Ch

15.2.5.1.209.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER8_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER8_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.209.3 Fields

Field	Function
31-0 SW_REFER8_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 8

15.2.5.1.210 Base address LSB (bits 31:0) for reference compress luminance table index 8 (SWREG208)

15.2.5.1.210.1 Offset

Register	Offset
SWREG208	340h

15.2.5.1.210.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER8_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER8_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.210.3 Fields

Field	Function
31-0 SW_REFER8_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 8

15.2.5.1.211 Base address MSB (bits 63:32) for reference compress luminance table index 9 (SWREG209)

15.2.5.1.211.1 Offset

Register	Offset
SWREG209	344h

15.2.5.1.211.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER9_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER9_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.211.3 Fields

Field	Function
31-0 SW_REFER9_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 9

15.2.5.1.212 Base address LSB (bits 31:0) for reference compress luminance table index 9 (SWREG210)

15.2.5.1.212.1 Offset

Register	Offset
SWREG210	348h

15.2.5.1.212.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER9_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER9_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.212.3 Fields

Field	Function
31-0 SW_REFER9_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 9

15.2.5.1.213 Base address MSB (bits 63:32) for reference compress luminance table index 10 (SWREG211)

15.2.5.1.213.1 Offset

Register	Offset
SWREG211	34Ch

15.2.5.1.213.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER10_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER10_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.213.3 Fields

Field	Function
31-0 SW_REFER10_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 10

15.2.5.1.214 Base address LSB (bits 31:0) for reference compress luminance table index 10 (SWREG212)

15.2.5.1.214.1 Offset

Register	Offset
SWREG212	350h

15.2.5.1.214.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER10_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER10_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.214.3 Fields

Field	Function
31-0 SW_REFER10_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 10

15.2.5.1.215 Base address MSB (bits 63:32) for reference compress luminance table index 11 (SWREG213)

15.2.5.1.215.1 Offset

Register	Offset
SWREG213	354h

15.2.5.1.215.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER11_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER11_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.215.3 Fields

Field	Function
31-0 SW_REFER11_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 11

15.2.5.1.216 Base address LSB (bits 31:0) for reference compress luminance table index 11 (SWREG214)

15.2.5.1.216.1 Offset

Register	Offset
SWREG214	358h

15.2.5.1.216.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER11_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER11_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.216.3 Fields

Field	Function
31-0 SW_REFER11_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 11

15.2.5.1.217 Base address MSB (bits 63:32) for reference compress luminance table index 12 (SWREG215)

15.2.5.1.217.1 Offset

Register	Offset
SWREG215	35Ch

15.2.5.1.217.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER12_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER12_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.217.3 Fields

Field	Function
31-0 SW_REFER12_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 12

15.2.5.1.218 Base address LSB (bits 31:0) for reference compress luminance table index 12 (SWREG216)

15.2.5.1.218.1 Offset

Register	Offset
SWREG216	360h

15.2.5.1.218.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER12_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER12_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.218.3 Fields

Field	Function
31-0 SW_REFER12_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 12

15.2.5.1.219 Base address MSB (bits 63:32) for reference compress luminance table index 13 (SWREG217)

15.2.5.1.219.1 Offset

Register	Offset
SWREG217	364h

15.2.5.1.219.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER13_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER13_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.219.3 Fields

Field	Function
31-0 SW_REFER13_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 13

15.2.5.1.220 Base address LSB (bits 31:0) for reference compress luminance table index 13 (SWREG218)

15.2.5.1.220.1 Offset

Register	Offset
SWREG218	368h

15.2.5.1.220.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER13_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER13_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.220.3 Fields

Field	Function
31-0 SW_REFER13_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 13

15.2.5.1.221 Base address MSB (bits 63:32) for reference compress luminance table index 14 (SWREG219)

15.2.5.1.221.1 Offset

Register	Offset
SWREG219	36Ch

15.2.5.1.221.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER14_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER14_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.221.3 Fields

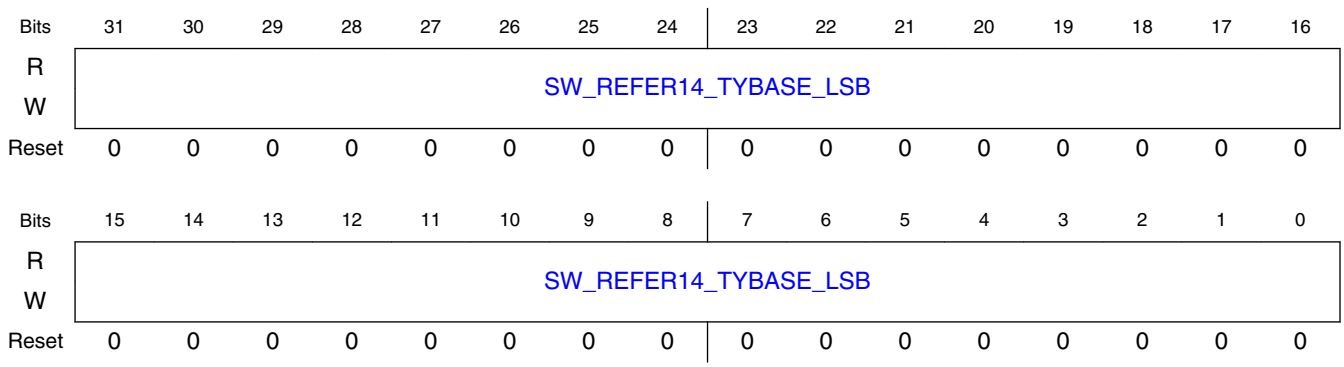
Field	Function
31-0 SW_REFER14_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 14

15.2.5.1.222 Base address LSB (bits 31:0) for reference compress luminance table index 14 (SWREG220)

15.2.5.1.222.1 Offset

Register	Offset
SWREG220	370h

15.2.5.1.222.2 Diagram



15.2.5.1.222.3 Fields

Field	Function
31-0 SW_REFER14_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 14

15.2.5.1.223 Base address MSB (bits 63:32) for reference compress luminance table index 15 (SWREG221)

15.2.5.1.223.1 Offset

Register	Offset
SWREG221	374h

15.2.5.1.223.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER15_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER15_TYBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.223.3 Fields

Field	Function
31-0 SW_REFER15_TYBASE_MSB	Base address MSB (bits 63:32) for reference compress luminance table index 15

15.2.5.1.224 Base address LSB (bits 31:0) for reference compress luminance table index 15 (SWREG222)

15.2.5.1.224.1 Offset

Register	Offset
SWREG222	378h

15.2.5.1.224.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER15_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER15_TYBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.224.3 Fields

Field	Function
31-0 SW_REFER15_TYBASE_LSB	Base address LSB (bits 31:0) for reference compress luminance table index 15

15.2.5.1.225 Base address MSB (bits 63:32) for decoder output compress chrominance table (SWREG223)

15.2.5.1.225.1 Offset

Register	Offset
SWREG223	37Ch

15.2.5.1.225.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_OUT_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_OUT_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.225.3 Fields

Field	Function
31-0 SW_DEC_OUT_TCBASE_MSB	Base address MSB (bits 63:32) for decoder output compress chrominance table

15.2.5.1.226 Base address LSB (bits 31:0) for decoder output compress chrominance table (SWREG224)

15.2.5.1.226.1 Offset

Register	Offset
SWREG224	380h

15.2.5.1.226.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_DEC_OUT_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_DEC_OUT_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.226.3 Fields

Field	Function
31-0 SW_DEC_OUT_TCBASE_LSB	Base address LSB (bits 31:0) for decoder output compress chrominance table

15.2.5.1.227 Base address MSB (bits 63:32) for reference compress chrominance table index 0 (SWREG225)

15.2.5.1.227.1 Offset

Register	Offset
SWREG225	384h

15.2.5.1.227.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER0_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER0_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.227.3 Fields

Field	Function
31-0 SW_REFER0_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 0

15.2.5.1.228 Base address LSB (bits 31:0) for reference compress chrominance table index 0 (SWREG226)

15.2.5.1.228.1 Offset

Register	Offset
SWREG226	388h

15.2.5.1.228.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER0_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER0_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.228.3 Fields

Field	Function
31-0 SW_REFER0_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 0

15.2.5.1.229 Base address MSB (bits 63:32) for reference compress chrominance table index 1 (SWREG227)

15.2.5.1.229.1 Offset

Register	Offset
SWREG227	38Ch

15.2.5.1.229.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER1_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER1_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.229.3 Fields

Field	Function
31-0 SW_REFER1_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 1

15.2.5.1.230 Base address LSB (bits 31:0) for reference compress chrominance table index 1 (SWREG228)

15.2.5.1.230.1 Offset

Register	Offset
SWREG228	390h

15.2.5.1.230.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER1_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER1_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.230.3 Fields

Field	Function
31-0 SW_REFER1_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 1

15.2.5.1.231 Base address MSB (bits 63:32) for reference compress chrominance table index 2 (SWREG229)

15.2.5.1.231.1 Offset

Register	Offset
SWREG229	394h

15.2.5.1.231.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER2_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER2_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.231.3 Fields

Field	Function
31-0 SW_REFER2_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 2

15.2.5.1.232 Base address LSB (bits 31:0) for reference compress chrominance table index 2 (SWREG230)

15.2.5.1.232.1 Offset

Register	Offset
SWREG230	398h

15.2.5.1.232.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER2_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER2_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.232.3 Fields

Field	Function
31-0 SW_REFER2_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 2

15.2.5.1.233 Base address MSB (bits 63:32) for reference compress chrominance table index 3 (SWREG231)

15.2.5.1.233.1 Offset

Register	Offset
SWREG231	39Ch

15.2.5.1.233.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER3_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER3_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.233.3 Fields

Field	Function
31-0 SW_REFER3_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 3

15.2.5.1.234 Base address LSB (bits 31:0) for reference compress chrominance table index 3 (SWREG232)

15.2.5.1.234.1 Offset

Register	Offset
SWREG232	3A0h

15.2.5.1.234.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER3_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER3_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.234.3 Fields

Field	Function
31-0 SW_REFER3_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 3

15.2.5.1.235 Base address MSB (bits 63:32) for reference compress chrominance table index 4 (SWREG233)

15.2.5.1.235.1 Offset

Register	Offset
SWREG233	3A4h

15.2.5.1.235.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER4_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER4_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.235.3 Fields

Field	Function
31-0 SW_REFER4_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 4

15.2.5.1.236 Base address LSB (bits 31:0) for reference compress chrominance table index 4 (SWREG234)

15.2.5.1.236.1 Offset

Register	Offset
SWREG234	3A8h

15.2.5.1.236.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER4_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER4_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.236.3 Fields

Field	Function
31-0 SW_REFER4_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 4

15.2.5.1.237 Base address MSB (bits 63:32) for reference compress chrominance table index 5 (SWREG235)

15.2.5.1.237.1 Offset

Register	Offset
SWREG235	3ACh

15.2.5.1.237.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER5_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER5_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.237.3 Fields

Field	Function
31-0 SW_REFER5_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 5

15.2.5.1.238 Base address LSB (bits 31:0) for reference compress chrominance table index 5 (SWREG236)

15.2.5.1.238.1 Offset

Register	Offset
SWREG236	3B0h

15.2.5.1.238.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER5_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER5_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.238.3 Fields

Field	Function
31-0 SW_REFER5_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 5

15.2.5.1.239 Base address MSB (bits 63:32) for reference compress chrominance table index 6 (SWREG237)

15.2.5.1.239.1 Offset

Register	Offset
SWREG237	3B4h

15.2.5.1.239.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER6_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER6_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.239.3 Fields

Field	Function
31-0 SW_REFER6_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 6

15.2.5.1.240 Base address LSB (bits 31:0) for reference compress chrominance table index 6 (SWREG238)

15.2.5.1.240.1 Offset

Register	Offset
SWREG238	3B8h

15.2.5.1.240.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER6_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER6_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.240.3 Fields

Field	Function
31-0 SW_REFER6_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 6

15.2.5.1.241 Base address MSB (bits 63:32) for reference compress chrominance table index 7 (SWREG239)

15.2.5.1.241.1 Offset

Register	Offset
SWREG239	3BCh

15.2.5.1.241.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER7_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER7_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.241.3 Fields

Field	Function
31-0 SW_REFER7_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 7

15.2.5.1.242 Base address LSB (bits 31:0) for reference compress chrominance table index 7 (SWREG240)

15.2.5.1.242.1 Offset

Register	Offset
SWREG240	3C0h

15.2.5.1.242.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER7_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER7_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.242.3 Fields

Field	Function
31-0 SW_REFER7_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 7

15.2.5.1.243 Base address MSB (bits 63:32) for reference compress chrominance table index 8 (SWREG241)

15.2.5.1.243.1 Offset

Register	Offset
SWREG241	3C4h

15.2.5.1.243.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER8_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER8_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.243.3 Fields

Field	Function
31-0 SW_REFER8_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 8

15.2.5.1.244 Base address LSB (bits 31:0) for reference compress chrominance table index 8 (SWREG242)

15.2.5.1.244.1 Offset

Register	Offset
SWREG242	3C8h

15.2.5.1.244.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER8_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER8_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.244.3 Fields

Field	Function
31-0 SW_REFER8_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 8

15.2.5.1.245 Base address MSB (bits 63:32) for reference compress chrominance table index 9 (SWREG243)

15.2.5.1.245.1 Offset

Register	Offset
SWREG243	3CCh

15.2.5.1.245.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER9_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER9_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.245.3 Fields

Field	Function
31-0 SW_REFER9_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 9

15.2.5.1.246 Base address LSB (bits 31:0) for reference compress chrominance table index 9 (SWREG244)

15.2.5.1.246.1 Offset

Register	Offset
SWREG244	3D0h

15.2.5.1.246.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER9_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER9_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.246.3 Fields

Field	Function
31-0 SW_REFER9_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 9

15.2.5.1.247 Base address MSB (bits 63:32) for reference compress chrominance table index 10 (SWREG245)

15.2.5.1.247.1 Offset

Register	Offset
SWREG245	3D4h

15.2.5.1.247.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER10_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER10_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.247.3 Fields

Field	Function
31-0 SW_REFER10_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 10

15.2.5.1.248 Base address LSB (bits 31:0) for reference compress chrominance table index 10 (SWREG246)

15.2.5.1.248.1 Offset

Register	Offset
SWREG246	3D8h

15.2.5.1.248.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER10_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER10_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.248.3 Fields

Field	Function
31-0 SW_REFER10_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 10

15.2.5.1.249 Base address MSB (bits 63:32) for reference compress chrominance table index 11 (SWREG247)

15.2.5.1.249.1 Offset

Register	Offset
SWREG247	3DCh

15.2.5.1.249.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER11_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER11_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.249.3 Fields

Field	Function
31-0 SW_REFER11_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 11

15.2.5.1.250 Base address LSB (bits 31:0) for reference compress chrominance table index 11 (SWREG248)

15.2.5.1.250.1 Offset

Register	Offset
SWREG248	3E0h

15.2.5.1.250.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER11_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER11_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.250.3 Fields

Field	Function
31-0 SW_REFER11_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 11

15.2.5.1.251 Base address MSB (bits 63:32) for reference compress chrominance table index 12 (SWREG249)

15.2.5.1.251.1 Offset

Register	Offset
SWREG249	3E4h

15.2.5.1.251.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER12_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER12_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.251.3 Fields

Field	Function
31-0 SW_REFER12_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 12

15.2.5.1.252 Base address LSB (bits 31:0) for reference compress chrominance table index 12 (SWREG250)

15.2.5.1.252.1 Offset

Register	Offset
SWREG250	3E8h

15.2.5.1.252.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER12_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER12_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.252.3 Fields

Field	Function
31-0 SW_REFER12_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 12

15.2.5.1.253 Base address MSB (bits 63:32) for reference compress chrominance table index 13 (SWREG251)

15.2.5.1.253.1 Offset

Register	Offset
SWREG251	3ECh

15.2.5.1.253.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER13_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER13_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.253.3 Fields

Field	Function
31-0 SW_REFER13_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 13

15.2.5.1.254 Base address LSB (bits 31:0) for reference compress chrominance table index 13 (SWREG252)

15.2.5.1.254.1 Offset

Register	Offset
SWREG252	3F0h

15.2.5.1.254.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER13_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER13_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.254.3 Fields

Field	Function
31-0 SW_REFER13_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 13

15.2.5.1.255 Base address MSB (bits 63:32) for reference compress chrominance table index 14 (SWREG253)

15.2.5.1.255.1 Offset

Register	Offset
SWREG253	3F4h

15.2.5.1.255.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER14_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER14_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.255.3 Fields

Field	Function
31-0 SW_REFER14_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 14

15.2.5.1.256 Base address LSB (bits 31:0) for reference compress chrominance table index 14 (SWREG254)

15.2.5.1.256.1 Offset

Register	Offset
SWREG254	3F8h

15.2.5.1.256.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER14_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER14_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.256.3 Fields

Field	Function
31-0 SW_REFER14_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 14

15.2.5.1.257 Base address MSB (bits 63:32) for reference compress chrominance table index 15 (SWREG255)

15.2.5.1.257.1 Offset

Register	Offset
SWREG255	3FCh

15.2.5.1.257.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER15_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER15_TCBASE_MSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.257.3 Fields

Field	Function
31-0 SW_REFER15_TCBASE_MSB	Base address MSB (bits 63:32) for reference compress chrominance table index 15

15.2.5.1.258 Base address LSB (bits 31:0) for reference compress chrominance table index 15 (SWREG256)

15.2.5.1.258.1 Offset

Register	Offset
SWREG256	400h

15.2.5.1.258.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_REFER15_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_REFER15_TCBASE_LSB															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.258.3 Fields

Field	Function
31-0 SW_REFER15_TCBASE_LSB	Base address LSB (bits 31:0) for reference compress chrominance table index 15

15.2.5.1.259 Not used (SWREG257)

15.2.5.1.259.1 Offset

Register	Offset
SWREG257	404h

15.2.5.1.259.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.259.3 Fields

Field	Function
31-0 —	Reserved.

15.2.5.1.260 input stream buffer length (SWREG258)

15.2.5.1.260.1 Offset

Register	Offset
SWREG258	408h

15.2.5.1.260.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_STRM_BUFFER_LEN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_STRM_BUFFER_LEN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.260.3 Fields

Field	Function
31-0 SW_STRM_BUFFER_LEN	input stream buffer length

15.2.5.1.261 input stream buffer start offset (SWREG259)

15.2.5.1.261.1 Offset

Register	Offset
SWREG259	40Ch

15.2.5.1.261.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SW_STRM_START_OFFSET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SW_STRM_START_OFFSET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.2.5.1.261.3 Fields

Field	Function
31-0 SW_STRM_START_OFFSET	input stream buffer start offset

15.3 VPU H1 (VPU_H1)

15.3.1 Overview

This block describes the features and functionality of the VPU H1 VP8/H.264/MVC video encoder.

15.3.1.1 Block Diagram

The block diagram of the VPU H1 encoder is shown below.

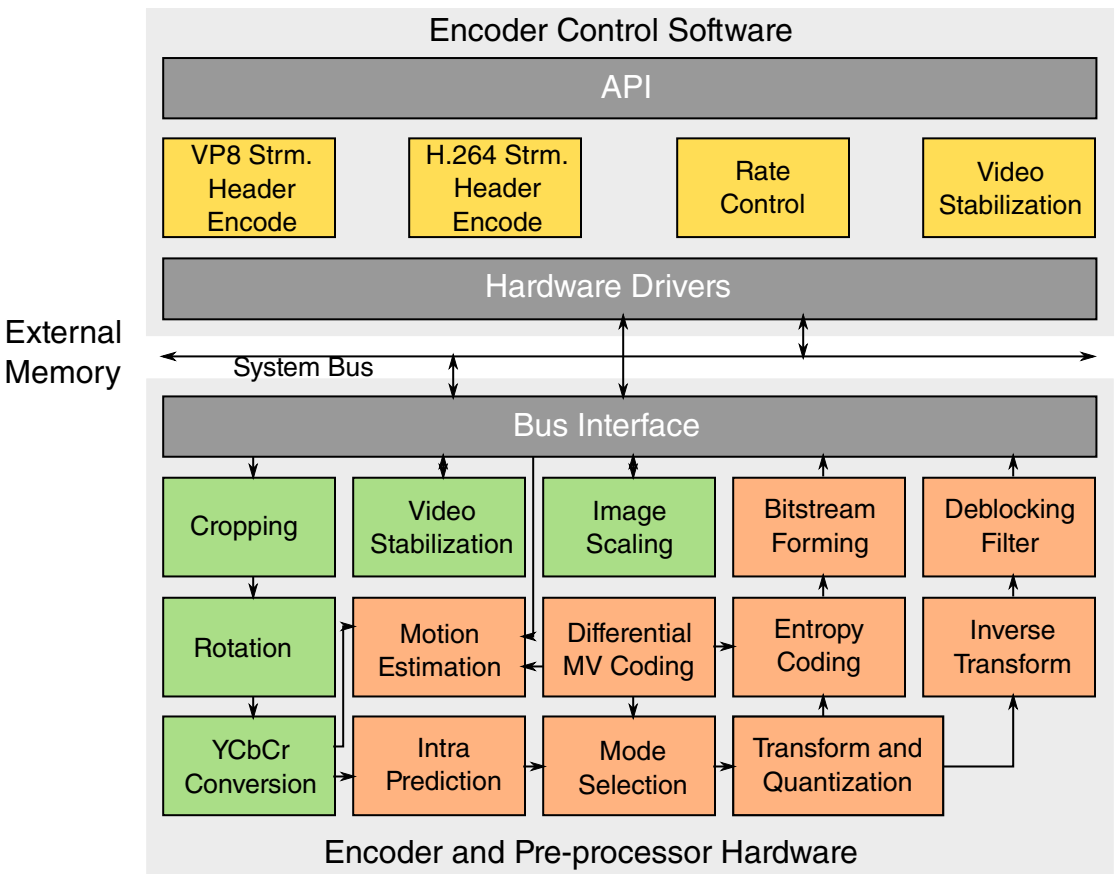


Figure 15-14. VPU H1 Encoder Block Diagram

15.3.1.2 Features

The supported standards, profiles and levels are presented in the table below.

Table 15-5. Supported Standards, Profiles and Levels

Standard	Encoder support
VP8	VP8
H.264 Profile and level	Baseline Profile, levels 1 – 5.1 Main Profile, levels 1 - 5.1 High Profile, levels 1 - 5.1
MVC extension to H.264	Stereo High

The supported VP8 and H.264 video tools are shown in the tables below.

Table 15-6. Supported VP8 Tools

Tool	Encoder support
Frame Types	I, P, Golden and droppable frames
Basic coding tools	14 intra prediction modes 4x4 transform Loop filter: normal, simple and off Error resilience: o Stream mode with no cumulative probability updates
Motion estimation	Configurable search area : - vertical ±30 pixels, horizontal ±126 pixels or - vertical ±14 pixels, horizontal ±126 pixels Pixel accuracy: ½ or ¼ MB and Sub-MB partitions: 16x16, 16x8, 8x16, 8x8 and 4x4 Algorithm: Indexed motion estimation Reconstruction filter: bicubic and bilinear
Number of reference frames	Up to 2
Maximum number of segments	Up to 4 different quantization parameters per frame
Maximum number of residual partitions	5 (control + up to 4 residual partitions)

Table 15-7. Supported H.264 / MVC Tools

Tool	Encoder support
Slices	I and P slices Programmable slice type (mixed I and P slices in one frame)
Fr types	Progressive Interlaced fields
Entropy encoding	CAVLC CABAC Mixed CAVLC (for intra frames) / CABAC (for inter frames

Table continues on the next page...

Table 15-7. Supported H.264 / MVC Tools (continued)

Tool	Encoder support
Basic	Error resilience: o Constrained intra prediction o Slices, multiple of macroblocks rows Maximum motion vector length: o Vertical +-126 pixels o Horizontal +-30 pixels Motion vector pixel accuracy: o ¼ or ½ pixels Macroblock and sub-macroblock partitions: o 16x16, 8x16, 16x8, 8x8, 4x8, 8x4 and 4x4 12 intra prediction modes o 4x4 and 8x8 transforms
Number of reference frames	Up to 2
Maximum number of segments	Up to 4 different quantization parameters per frame
Maximum number of slice groups	1

Table 15-8. VP8 / H.264 / MVC Features

Feature	Encoder support
Input data format	YCbCr formats: o YCbCr 4:2:0 planar o YCbCr 4:2:0 semi-planar o YCrCb 4:2:0 semi-planar o YCbYCr 4:2:2 raster-scan 1) o CbYCrY 4:2:2 raster-scan 1) RGB formats:1) o RGB444 and BGR444 o RGB555 and BGR555 o RGB565 and BGR565 o RGB888 and BRG888 o RGB101010 and BRG101010
Output data format	VP8: o VP8 bitstream H.264/MVC: o Byte unit stream o NAL unit stream
Supported image size	144 x 96 to 4080 x 4080 Step size 4 pixels
Maximum frame rate	Unlimited by the hardware 30 fps at 1920 x 1080 for VP8 60 fps at 1920 x 1080 for H.264
Bit rate	Minimum: 10kbps Maximum: 40 Mbps at 1080p 60 fps
Motion detection	Encoder outputs the best matching motion vector, the motion estimation quality information (a SAD value) and other coding information for each macroblock
Region-of-interest	Two user definable rectangular areas with separate quantizer setting
Region-of-intra	One user definable rectangular area coded as intra
Cyclic Intra Refresh	Supported

Preprocessing is pipelined with the encoder and it can be used only with the VPU H1 encoder. Preprocessing features are presented in the table below.

Table 15-9. Supported Standards, Profiles and Levels

Feature	Encoder support
RGB to YCbCr 4:2:0 color space conversion	BT.601, BT.709 or user defined coefficients conversion for RGB: o RGB444 and BGR444 o RGB555 and BGR555 o RGB565 and BGR565 o RGB888 and BRG888 o RGB101010 and BRG101010
YCbCr 4:2:2 to YCbCr 4:2:0 color space conversion	YCbCr formats: o YCbCr 4:2:0 planar o YCbCr 4:2:0 semi-planar o YCrCb 4:2:0 semi-planar o YCbYCr 4:2:2 raster-scan o CbYCrY 4:2:2 raster-scan
Cropping	Video - from 8192 x 8192 to any supported encoding size
Rotation	90 or 270 degrees
Image down-scaling	Proprietary averaging filter Arbitrary, non-integer scaling ratio separately for both dimensions Unlimited down-scaling ratio (e.g. from 1080p to QCIF)

15.3.2 VPU H1 Memory Map/Register Definition

15.3.2.1 VPU_H1 common registers descriptions

15.3.2.1.1 VPU H1 common registers memory map

VPU_H1_COMMON_REGISTERS base address: 3832_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	VPU H1 Register 0 (SWREG0)	32	RO	Table 15-9
4h	VPU H1 Register 1 (SWREG1)	32	RW	Table 15-9
8h	VPU H1 Register 2 (SWREG2)	32	RW	Table 15-9
Ch	VPU H1 Register 3 (SWREG3)	32	RW	Table 15-9
14h	VPU H1 Register 5 (SWREG5)	32	RW	Table 15-9
18h	VPU H1 Register 6 (SWREG6)	32	RW	Table 15-9
1Ch	VPU H1 Register 7 (SWREG7)	32	RW	Table 15-9
20h	VPU H1 Register 8 (SWREG8)	32	RW	Table 15-9
24h	VPU H1 Register 9 (SWREG9)	32	RW	Table 15-9
28h	VPU H1 Register 10 (SWREG10)	32	RW	Table 15-9

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
2Ch	VPU H1 Register 11 (SWREG11)	32	RW	Table 15-9
30h	VPU H1 Register 12 (SWREG12)	32	RW	Table 15-9
34h	VPU H1 Register 13 (SWREG13)	32	RW	Table 15-9
38h	VPU H1 Register 14 (SWREG14)	32	RW	Table 15-9
3Ch	VPU H1 Register 15 (SWREG15)	32	RW	Table 15-9
48h	VPU H1 Register 18 (SWREG18)	32	RW	Table 15-9
4Ch	VPU H1 Register 19 (SWREG19)	32	RW	Table 15-9
50h	VPU H1 Register 20 (SWREG20)	32	RW	Table 15-9
54h	VPU H1 Register 21 (SWREG21)	32	RW	Table 15-9
58h	VPU H1 Register 22 (SWREG22)	32	RW	Table 15-9
5Ch	VPU H1 Register 23 (SWREG23)	32	RW	Table 15-9
60h	VPU H1 Register 24 (SWREG24)	32	RW	Table 15-9
64h	VPU H1 Register 25 (SWREG25)	32	RW	Table 15-9
94h	VPU H1 Register 37 (SWREG37)	32	RW	Table 15-9
98h	VPU H1 Register 38 (SWREG38)	32	RW	Table 15-9
9Ch	VPU H1 Register 39 (SWREG39)	32	RW	Table 15-9
A0h	VPU H1 Register 40 (SWREG40)	32	RW	Table 15-9
A4h	VPU H1 Register 41 (SWREG41)	32	RW	Table 15-9
A8h	VPU H1 Register 42 (SWREG42)	32	RW	Table 15-9
ACh	VPU H1 Register 43 (SWREG43)	32	RW	Table 15-9
B0h	VPU H1 Register 44 (SWREG44)	32	RW	Table 15-9
B4h	VPU H1 Register 45 (SWREG45)	32	RW	Table 15-9
B8h	VPU H1 Register 46 (SWREG46)	32	RW	Table 15-9
BCh	VPU H1 Register 47 (SWREG47)	32	RW	Table 15-9
C0h	VPU H1 Register 48 (SWREG48)	32	RW	Table 15-9
C4h	VPU H1 Register 49 (SWREG49)	32	RW	Table 15-9
C8h	VPU H1 Register 50 (SWREG50)	32	RW	Table 15-9
CCh	VPU H1 Register 51 (SWREG51)	32	RW	Table 15-9
D0h	VPU H1 Register 52 (SWREG52)	32	RW	Table 15-9
D4h	VPU H1 Register 53 (SWREG53)	32	RW	Table 15-9
D8h	VPU H1 Register 54 (SWREG54)	32	RW	Table 15-9
DCh	VPU H1 Register 55 (SWREG55)	32	RW	Table 15-9
E0h	VPU H1 Register 56 (SWREG56)	32	RW	Table 15-9
E4h	VPU H1 Register 57 (SWREG57)	32	RW	Table 15-9
F0h	VPU H1 Register 60 (SWREG60)	32	RW	Table 15-9
F4h	VPU H1 Register 61 (SWREG61)	32	RW	Table 15-9
F8h	VPU H1 Register 62 (SWREG62)	32	RW	Table 15-9
FCh	VPU H1 Register 63 (SWREG63)	32	RO	Table 15-9
180h	VPU H1 Register 96 (SWREG96)	32	WO	Table 15-9
184h	VPU H1 Register 97 (SWREG97)	32	WO	Table 15-9

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
188h	VPU H1 Register 98 (SWREG98)	32	WO	Table 15-9
18Ch	VPU H1 Register 99 (SWREG99)	32	WO	Table 15-9
190h	VPU H1 Register 100 (SWREG100)	32	WO	Table 15-9
194h	VPU H1 Register 101 (SWREG101)	32	WO	Table 15-9
198h	VPU H1 Register 102 (SWREG102)	32	WO	Table 15-9
19Ch	VPU H1 Register 103 (SWREG103)	32	WO	Table 15-9
1A0h	VPU H1 Register 104 (SWREG104)	32	WO	Table 15-9
1A4h	VPU H1 Register 105 (SWREG105)	32	WO	Table 15-9
1A8h	VPU H1 Register 106 (SWREG106)	32	WO	Table 15-9
1ACh	VPU H1 Register 107 (SWREG107)	32	WO	Table 15-9
1B0h	VPU H1 Register 108 (SWREG108)	32	WO	Table 15-9
1B4h	VPU H1 Register 109 (SWREG109)	32	WO	Table 15-9
1B8h	VPU H1 Register 110 (SWREG110)	32	WO	Table 15-9
1BCh	VPU H1 Register 111 (SWREG111)	32	WO	Table 15-9
1C0h	VPU H1 Register 112 (SWREG112)	32	WO	Table 15-9
1C4h	VPU H1 Register 113 (SWREG113)	32	WO	Table 15-9
1C8h	VPU H1 Register 114 (SWREG114)	32	WO	Table 15-9
1CCh	VPU H1 Register 115 (SWREG115)	32	WO	Table 15-9
1D0h	VPU H1 Register 116 (SWREG116)	32	WO	Table 15-9
1D4h	VPU H1 Register 117 (SWREG117)	32	WO	Table 15-9
1D8h	VPU H1 Register 118 (SWREG118)	32	WO	Table 15-9
1DCh	VPU H1 Register 119 (SWREG119)	32	WO	Table 15-9
1E0h	VPU H1 Register 120 (SWREG120)	32	WO	Table 15-9
1E4h	VPU H1 Register 121 (SWREG121)	32	WO	Table 15-9
1E8h	VPU H1 Register 122 (SWREG122)	32	WO	Table 15-9
1ECh	VPU H1 Register 123 (SWREG123)	32	WO	Table 15-9
1F0h	VPU H1 Register 124 (SWREG124)	32	WO	Table 15-9
1F4h	VPU H1 Register 125 (SWREG125)	32	WO	Table 15-9
1F8h	VPU H1 Register 126 (SWREG126)	32	WO	Table 15-9
1FCh	VPU H1 Register 127 (SWREG127)	32	WO	Table 15-9
200h	VPU H1 Register 128 (SWREG128)	32	WO	Table 15-9
204h	VPU H1 Register 129 (SWREG129)	32	WO	Table 15-9
208h	VPU H1 Register 130 (SWREG130)	32	WO	Table 15-9
20Ch	VPU H1 Register 131 (SWREG131)	32	WO	Table 15-9
210h	VPU H1 Register 132 (SWREG132)	32	WO	Table 15-9
214h	VPU H1 Register 133 (SWREG133)	32	WO	Table 15-9
218h	VPU H1 Register 134 (SWREG134)	32	WO	Table 15-9
21Ch	VPU H1 Register 135 (SWREG135)	32	WO	Table 15-9
220h	VPU H1 Register 136 (SWREG136)	32	WO	Table 15-9
224h	VPU H1 Register 137 (SWREG137)	32	WO	Table 15-9

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
228h	VPU H1 Register 138 (SWREG138)	32	WO	Table 15-9
22Ch	VPU H1 Register 139 (SWREG139)	32	WO	Table 15-9
230h	VPU H1 Register 140 (SWREG140)	32	WO	Table 15-9
234h	VPU H1 Register 141 (SWREG141)	32	WO	Table 15-9
238h	VPU H1 Register 142 (SWREG142)	32	WO	Table 15-9
23Ch	VPU H1 Register 143 (SWREG143)	32	WO	Table 15-9
240h	VPU H1 Register 144 (SWREG144)	32	WO	Table 15-9
244h	VPU H1 Register 145 (SWREG145)	32	WO	Table 15-9
248h	VPU H1 Register 146 (SWREG146)	32	WO	Table 15-9
24Ch	VPU H1 Register 147 (SWREG147)	32	WO	Table 15-9
250h	VPU H1 Register 148 (SWREG148)	32	WO	Table 15-9
254h	VPU H1 Register 149 (SWREG149)	32	WO	Table 15-9
258h	VPU H1 Register 150 (SWREG150)	32	WO	Table 15-9
25Ch	VPU H1 Register 151 (SWREG151)	32	WO	Table 15-9
260h	VPU H1 Register 152 (SWREG152)	32	WO	Table 15-9
264h	VPU H1 Register 153 (SWREG153)	32	WO	Table 15-9
268h	VPU H1 Register 154 (SWREG154)	32	WO	Table 15-9
26Ch	VPU H1 Register 155 (SWREG155)	32	WO	Table 15-9
270h	VPU H1 Register 156 (SWREG156)	32	WO	Table 15-9
274h	VPU H1 Register 157 (SWREG157)	32	WO	Table 15-9
278h	VPU H1 Register 158 (SWREG158)	32	WO	Table 15-9
27Ch	VPU H1 Register 159 (SWREG159)	32	WO	Table 15-9
280h	VPU H1 Register 160 (SWREG160)	32	RW	Table 15-9
284h	VPU H1 Register 161 (SWREG161)	32	RW	Table 15-9
288h	VPU H1 Register 162 (SWREG162)	32	RW	Table 15-9
28Ch	VPU H1 Register 163 (SWREG163)	32	RW	Table 15-9
290h	VPU H1 Register 164 (SWREG164)	32	RW	Table 15-9
294h	VPU H1 Register 165 (SWREG165)	32	RW	Table 15-9
298h	VPU H1 Register 166 (SWREG166)	32	RW	Table 15-9
29Ch	VPU H1 Register 167 (SWREG167)	32	RW	Table 15-9
2A0h	VPU H1 Register 168 (SWREG168)	32	RW	Table 15-9
2A4h	VPU H1 Register 169 (SWREG169)	32	RW	Table 15-9
2A8h	VPU H1 Register 170 (SWREG170)	32	RW	Table 15-9
2ACh	VPU H1 Register 171 (SWREG171)	32	RW	Table 15-9
2B0h	VPU H1 Register 172 (SWREG172)	32	RW	Table 15-9
2B4h	VPU H1 Register 173 (SWREG173)	32	RW	Table 15-9
2B8h	VPU H1 Register 174 (SWREG174)	32	RW	Table 15-9
2BCh	VPU H1 Register 175 (SWREG175)	32	RW	Table 15-9
2C0h	VPU H1 Register 176 (SWREG176)	32	RW	Table 15-9
2C4h	VPU H1 Register 177 (SWREG177)	32	RW	Table 15-9

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
2C8h	VPU H1 Register 178 (SWREG178)	32	RW	Table 15-9
2CCh	VPU H1 Register 179 (SWREG179)	32	RW	Table 15-9
2D0h	VPU H1 Register 180 (SWREG180)	32	RW	Table 15-9
2D4h	VPU H1 Register 181 (SWREG181)	32	RW	Table 15-9
2D8h	VPU H1 Register 182 (SWREG182)	32	RW	Table 15-9
2DCh	VPU H1 Register 183 (SWREG183)	32	RW	Table 15-9
2E0h	VPU H1 Register 184 (SWREG184)	32	RW	Table 15-9
2E4h	VPU H1 Register 185 (SWREG185)	32	RW	Table 15-9
2E8h	VPU H1 Register 186 (SWREG186)	32	RW	Table 15-9
2ECh	VPU H1 Register 187 (SWREG187)	32	RW	Table 15-9
2F0h	VPU H1 Register 188 (SWREG188)	32	RW	Table 15-9
2F4h	VPU H1 Register 189 (SWREG189)	32	RW	Table 15-9
2F8h	VPU H1 Register 190 (SWREG190)	32	RW	Table 15-9
2FCh	VPU H1 Register 191 (SWREG191)	32	RW	Table 15-9
300h	VPU H1 Register 192 (SWREG192)	32	RW	Table 15-9
304h	VPU H1 Register 193 (SWREG193)	32	RW	Table 15-9
308h	VPU H1 Register 194 (SWREG194)	32	RW	Table 15-9
30Ch	VPU H1 Register 195 (SWREG195)	32	RW	Table 15-9
310h	VPU H1 Register 196 (SWREG196)	32	RW	Table 15-9
314h	VPU H1 Register 197 (SWREG197)	32	RW	Table 15-9
318h	VPU H1 Register 198 (SWREG198)	32	RW	Table 15-9
31Ch	VPU H1 Register 199 (SWREG199)	32	RW	Table 15-9
320h	VPU H1 Register 200 (SWREG200)	32	RW	Table 15-9
324h	VPU H1 Register 201 (SWREG201)	32	RW	Table 15-9
328h	VPU H1 Register 202 (SWREG202)	32	RW	Table 15-9
32Ch	VPU H1 Register 203 (SWREG203)	32	RW	Table 15-9
330h	VPU H1 Register 204 (SWREG204)	32	RW	Table 15-9
334h	VPU H1 Register 205 (SWREG205)	32	RW	Table 15-9
338h	VPU H1 Register 206 (SWREG206)	32	RW	Table 15-9
33Ch	VPU H1 Register 207 (SWREG207)	32	RW	Table 15-9
340h	VPU H1 Register 208 (SWREG208)	32	RW	Table 15-9
344h	VPU H1 Register 209 (SWREG209)	32	RW	Table 15-9
348h	VPU H1 Register 210 (SWREG210)	32	RW	Table 15-9
34Ch	VPU H1 Register 211 (SWREG211)	32	RW	Table 15-9
350h	VPU H1 Register 212 (SWREG212)	32	RW	Table 15-9
354h	VPU H1 Register 213 (SWREG213)	32	RW	Table 15-9
358h	VPU H1 Register 214 (SWREG214)	32	RW	Table 15-9
35Ch	VPU H1 Register 215 (SWREG215)	32	RW	Table 15-9
360h	VPU H1 Register 216 (SWREG216)	32	RW	Table 15-9
364h	VPU H1 Register 217 (SWREG217)	32	RW	Table 15-9

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
368h	VPU H1 Register 218 (SWREG218)	32	RW	Table 15-9
36Ch	VPU H1 Register 219 (SWREG219)	32	RW	Table 15-9
370h	VPU H1 Register 220 (SWREG220)	32	RW	Table 15-9
374h	VPU H1 Register 221 (SWREG221)	32	RW	Table 15-9
378h	VPU H1 Register 222 (SWREG222)	32	RW	Table 15-9
37Ch	VPU H1 Register 223 (SWREG223)	32	RW	Table 15-9
380h	VPU H1 Register 224 (SWREG224)	32	RW	Table 15-9
384h	VPU H1 Register 225 (SWREG225)	32	RW	Table 15-9
388h	VPU H1 Register 226 (SWREG226)	32	RW	Table 15-9
38Ch	VPU H1 Register 227 (SWREG227)	32	RW	Table 15-9
390h	VPU H1 Register 228 (SWREG228)	32	RW	Table 15-9
394h	VPU H1 Register 229 (SWREG229)	32	RW	Table 15-9
398h	VPU H1 Register 230 (SWREG230)	32	RW	Table 15-9
39Ch	VPU H1 Register 231 (SWREG231)	32	RW	Table 15-9
3A0h	VPU H1 Register 232 (SWREG232)	32	RW	Table 15-9
3A4h	VPU H1 Register 233 (SWREG233)	32	RW	Table 15-9
3B0h	VPU H1 Register 236 (SWREG236)	32	RW	Table 15-9
3B4h	VPU H1 Register 237 (SWREG237)	32	RW	Table 15-9
3B8h	VPU H1 Register 238 (SWREG238)	32	RW	Table 15-9
3BCh	VPU H1 Register 239 (SWREG239)	32	RW	Table 15-9
3C0h	VPU H1 Register 240 (SWREG240)	32	RW	Table 15-9
3C4h	VPU H1 Register 241 (SWREG241)	32	RW	Table 15-9
3C8h	VPU H1 Register 242 (SWREG242)	32	RW	Table 15-9
3CCh	VPU H1 Register 243 (SWREG243)	32	RW	Table 15-9
3D0h	VPU H1 Register 244 (SWREG244)	32	RW	Table 15-9
3D4h	VPU H1 Register 245 (SWREG245)	32	RW	Table 15-9
400h	VPU H1 Register 256 (SWREG256)	32	RW	Table 15-9
404h	VPU H1 Register 257 (SWREG257)	32	RW	Table 15-9
408h	VPU H1 Register 258 (SWREG258)	32	RW	Table 15-9
40Ch	VPU H1 Register 259 (SWREG259)	32	RW	Table 15-9
410h	VPU H1 Register 260 (SWREG260)	32	RW	Table 15-9
414h	VPU H1 Register 261 (SWREG261)	32	RW	Table 15-9
418h	VPU H1 Register 262 (SWREG262)	32	RW	Table 15-9
41Ch	VPU H1 Register 263 (SWREG263)	32	RW	Table 15-9
420h	VPU H1 Register 264 (SWREG264)	32	RW	Table 15-9
424h	VPU H1 Register 265 (SWREG265)	32	RW	Table 15-9
428h	VPU H1 Register 266 (SWREG266)	32	RW	Table 15-9
42Ch	VPU H1 Register 267 (SWREG267)	32	RW	Table 15-9
430h	VPU H1 Register 268 (SWREG268)	32	RW	Table 15-9
434h	VPU H1 Register 269 (SWREG269)	32	RW	Table 15-9

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
438h	VPU H1 Register 270 (SWREG270)	32	RW	Table 15-9
43Ch	VPU H1 Register 271 (SWREG271)	32	RW	Table 15-9
440h	VPU H1 Register 272 (SWREG272)	32	RW	Table 15-9
444h	VPU H1 Register 273 (SWREG273)	32	RW	Table 15-9
448h	VPU H1 Register 274 (SWREG274)	32	RW	Table 15-9
44Ch	VPU H1 Register 275 (SWREG275)	32	RW	Table 15-9
450h	VPU H1 Register 276 (SWREG276)	32	RW	Table 15-9
454h	VPU H1 Register 277 (SWREG277)	32	RW	Table 15-9
458h	VPU H1 Register 278 (SWREG278)	32	RW	Table 15-9
45Ch	VPU H1 Register 279 (SWREG279)	32	RW	Table 15-9
460h	VPU H1 Register 280 (SWREG280)	32	RW	Table 15-9
464h	VPU H1 Register 281 (SWREG281)	32	RW	Table 15-9
468h	VPU H1 Register 282 (SWREG282)	32	RW	Table 15-9
46Ch	VPU H1 Register 283 (SWREG283)	32	RW	Table 15-9
470h	VPU H1 Register 284 (SWREG284)	32	RW	Table 15-9
474h	VPU H1 Register 285 (SWREG285)	32	RW	Table 15-9
478h	VPU H1 Register 286 (SWREG286)	32	RW	Table 15-9
47Ch	VPU H1 Register 287 (SWREG287)	32	RW	Table 15-9
480h	VPU H1 Register 288 (SWREG288)	32	RW	Table 15-9
484h	VPU H1 Register 289 (SWREG289)	32	RW	Table 15-9
488h	VPU H1 Register 290 (SWREG290)	32	RW	Table 15-9
48Ch	VPU H1 Register 291 (SWREG291)	32	RW	Table 15-9
490h	VPU H1 Register 292 (SWREG292)	32	RW	Table 15-9
494h	VPU H1 Register 293 (SWREG293)	32	RW	Table 15-9
498h	VPU H1 Register 294 (SWREG294)	32	RW	Table 15-9
49Ch	VPU H1 Register 295 (SWREG295)	32	RW	Table 15-9
4A0h	VPU H1 Register 296 (SWREG296)	32	RO	Table 15-9
4A4h	VPU H1 Register 297 (SWREG297)	32	RW	Table 15-9
4A8h	VPU H1 Register 298 (SWREG298)	32	RW	Table 15-9
4ACh	VPU H1 Register 299 (SWREG299)	32	RW	Table 15-9
4B0h	VPU H1 Register 300 (SWREG300)	32	RW	Table 15-9
4B4h	VPU H1 Register 301 (SWREG301)	32	RW	Table 15-9
4B8h	VPU H1 Register 302 (SWREG302)	32	RW	Table 15-9
4BCh	VPU H1 Register 303 (SWREG303)	32	RW	Table 15-9
4C0h	VPU H1 Register 304 (SWREG304)	32	RW	Table 15-9
4C4h	VPU H1 Register 305 (SWREG305)	32	RW	Table 15-9
4C8h	VPU H1 Register 306 (SWREG306)	32	RW	Table 15-9
4CCh	VPU H1 Register 307 (SWREG307)	32	RW	Table 15-9
4D0h	VPU H1 Register 308 (SWREG308)	32	RW	Table 15-9
4D4h	VPU H1 Register 309 (SWREG309)	32	RW	Table 15-9

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
4D8h	VPU H1 Register 310 (SWREG310)	32	RW	Table 15-9
4DCh	VPU H1 Register 311 (SWREG311)	32	RW	Table 15-9
4E0h	VPU H1 Register 312 (SWREG312)	32	RW	Table 15-9
4E4h	VPU H1 Register 313 (SWREG313)	32	RW	Table 15-9
4E8h	VPU H1 Register 314 (SWREG314)	32	RW	Table 15-9
4ECh	VPU H1 Register 315 (SWREG315)	32	RW	Table 15-9
4F0h	VPU H1 Register 316 (SWREG316)	32	RW	Table 15-9
4F4h	VPU H1 Register 317 (SWREG317)	32	RW	Table 15-9
4F8h	VPU H1 Register 318 (SWREG318)	32	RW	Table 15-9
4FCh	VPU H1 Register 319 (SWREG319)	32	RW	Table 15-9
500h	VPU H1 Register 320 (SWREG320)	32	RW	Table 15-9
504h	VPU H1 Register 321 (SWREG321)	32	RW	Table 15-9
508h	VPU H1 Register 322 (SWREG322)	32	RW	Table 15-9
50Ch	VPU H1 Register 323 (SWREG323)	32	RW	Table 15-9
510h	VPU H1 Register 324 (SWREG324)	32	RW	Table 15-9
514h	VPU H1 Register 325 (SWREG325)	32	RW	Table 15-9
518h	VPU H1 Register 326 (SWREG326)	32	RW	Table 15-9
51Ch	VPU H1 Register 327 (SWREG327)	32	RW	Table 15-9
520h	VPU H1 Register 328 (SWREG328)	32	RW	Table 15-9
524h	VPU H1 Register 329 (SWREG329)	32	RW	Table 15-9
528h	VPU H1 Register 330 (SWREG330)	32	RW	Table 15-9
52Ch	VPU H1 Register 331 (SWREG331)	32	RW	Table 15-9
530h	VPU H1 Register 332 (SWREG332)	32	RW	Table 15-9
534h	VPU H1 Register 333 (SWREG333)	32	RW	Table 15-9
538h	VPU H1 Register 334 (SWREG334)	32	RW	Table 15-9
53Ch	VPU H1 Register 335 (SWREG335)	32	RW	Table 15-9
540h	VPU H1 Register 336 (SWREG336)	32	RW	Table 15-9
544h	VPU H1 Register 337 (SWREG337)	32	RW	Table 15-9
548h	VPU H1 Register 338 (SWREG338)	32	RW	Table 15-9
54Ch	VPU H1 Register 339 (SWREG339)	32	RW	Table 15-9
550h	VPU H1 Register 340 (SWREG340)	32	RW	Table 15-9
554h	VPU H1 Register 341 (SWREG341)	32	RW	Table 15-9
558h	VPU H1 Register 342 (SWREG342)	32	RW	Table 15-9
55Ch	VPU H1 Register 343 (SWREG343)	32	RW	Table 15-9
560h	VPU H1 Register 344 (SWREG344)	32	RW	Table 15-9
564h	VPU H1 Register 345 (SWREG345)	32	RW	Table 15-9
568h	VPU H1 Register 346 (SWREG346)	32	RW	Table 15-9
56Ch	VPU H1 Register 347 (SWREG347)	32	RW	Table 15-9
570h	VPU H1 Register 348 (SWREG348)	32	RW	Table 15-9
574h	VPU H1 Register 349 (SWREG349)	32	RW	Table 15-9

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
578h	VPU H1 Register 350 (SWREG350)	32	RW	Table 15-9
57Ch	VPU H1 Register 351 (SWREG351)	32	RW	Table 15-9
580h	VPU H1 Register 352 (SWREG352)	32	RW	Table 15-9
584h	VPU H1 Register 353 (SWREG353)	32	RW	Table 15-9
588h	VPU H1 Register 354 (SWREG354)	32	RW	Table 15-9
58Ch	VPU H1 Register 355 (SWREG355)	32	RW	Table 15-9
590h	VPU H1 Register 356 (SWREG356)	32	RW	Table 15-9
594h	VPU H1 Register 357 (SWREG357)	32	RW	Table 15-9
598h	VPU H1 Register 358 (SWREG358)	32	RW	Table 15-9
59Ch	VPU H1 Register 359 (SWREG359)	32	RW	Table 15-9
5A0h	VPU H1 Register 360 (SWREG360)	32	RW	Table 15-9
5A4h	VPU H1 Register 361 (SWREG361)	32	RW	Table 15-9
5A8h	VPU H1 Register 362 (SWREG362)	32	RW	Table 15-9
5ACh	VPU H1 Register 363 (SWREG363)	32	RW	Table 15-9
5B0h	VPU H1 Register 364 (SWREG364)	32	RW	Table 15-9
5B4h	VPU H1 Register 365 (SWREG365)	32	RW	Table 15-9
5B8h	VPU H1 Register 366 (SWREG366)	32	RW	Table 15-9
5BCh	VPU H1 Register 367 (SWREG367)	32	RW	Table 15-9
5C0h	VPU H1 Register 368 (SWREG368)	32	RW	Table 15-9
5C4h	VPU H1 Register 369 (SWREG369)	32	RW	Table 15-9
5C8h	VPU H1 Register 370 (SWREG370)	32	RW	Table 15-9
5CCh	VPU H1 Register 371 (SWREG371)	32	RW	Table 15-9
5D0h	VPU H1 Register 372 (SWREG372)	32	RW	Table 15-9
5D4h	VPU H1 Register 373 (SWREG373)	32	RW	Table 15-9
5D8h	VPU H1 Register 374 (SWREG374)	32	RW	Table 15-9
5DCh	VPU H1 Register 375 (SWREG375)	32	RW	Table 15-9
5E0h	VPU H1 Register 376 (SWREG376)	32	RW	Table 15-9
5E4h	VPU H1 Register 377 (SWREG377)	32	RW	Table 15-9
5E8h	VPU H1 Register 378 (SWREG378)	32	RW	Table 15-9
5ECh	VPU H1 Register 379 (SWREG379)	32	RW	Table 15-9
5F0h	VPU H1 Register 380 (SWREG380)	32	RW	Table 15-9
5F4h	VPU H1 Register 381 (SWREG381)	32	RW	Table 15-9
5F8h	VPU H1 Register 382 (SWREG382)	32	RW	Table 15-9
5FCh	VPU H1 Register 383 (SWREG383)	32	RW	Table 15-9
600h	VPU H1 Register 384 (SWREG384)	32	RW	Table 15-9
604h	VPU H1 Register 385 (SWREG385)	32	RW	Table 15-9
608h	VPU H1 Register 386 (SWREG386)	32	RW	Table 15-9
60Ch	VPU H1 Register 387 (SWREG387)	32	RW	Table 15-9
610h	VPU H1 Register 388 (SWREG388)	32	RW	Table 15-9
614h	VPU H1 Register 389 (SWREG389)	32	RW	Table 15-9

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
618h	VPU H1 Register 390 (SWREG390)	32	RW	Table 15-9
61Ch	VPU H1 Register 391 (SWREG391)	32	RW	Table 15-9
620h	VPU H1 Register 392 (SWREG392)	32	RW	Table 15-9
624h	VPU H1 Register 393 (SWREG393)	32	RW	Table 15-9
628h	VPU H1 Register 394 (SWREG394)	32	RW	Table 15-9
62Ch	VPU H1 Register 395 (SWREG395)	32	RW	Table 15-9
630h	VPU H1 Register 396 (SWREG396)	32	RW	Table 15-9
634h	VPU H1 Register 397 (SWREG397)	32	RW	Table 15-9
638h	VPU H1 Register 398 (SWREG398)	32	RW	Table 15-9
63Ch	VPU H1 Register 399 (SWREG399)	32	RW	Table 15-9
640h	VPU H1 Register 400 (SWREG400)	32	RW	Table 15-9
644h	VPU H1 Register 401 (SWREG401)	32	RW	Table 15-9
648h	VPU H1 Register 402 (SWREG402)	32	RW	Table 15-9
64Ch	VPU H1 Register 403 (SWREG403)	32	RW	Table 15-9
650h	VPU H1 Register 404 (SWREG404)	32	RW	Table 15-9
654h	VPU H1 Register 405 (SWREG405)	32	RW	Table 15-9
658h	VPU H1 Register 406 (SWREG406)	32	RW	Table 15-9
65Ch	VPU H1 Register 407 (SWREG407)	32	RW	Table 15-9
660h	VPU H1 Register 408 (SWREG408)	32	RW	Table 15-9
664h	VPU H1 Register 409 (SWREG409)	32	RW	Table 15-9
668h	VPU H1 Register 410 (SWREG410)	32	RW	Table 15-9
66Ch	VPU H1 Register 411 (SWREG411)	32	RW	Table 15-9
670h	VPU H1 Register 412 (SWREG412)	32	RW	Table 15-9
674h	VPU H1 Register 413 (SWREG413)	32	RW	Table 15-9
678h	VPU H1 Register 414 (SWREG414)	32	RW	Table 15-9
67Ch	VPU H1 Register 415 (SWREG415)	32	RW	Table 15-9
680h	VPU H1 Register 416 (SWREG416)	32	RW	Table 15-9
684h	VPU H1 Register 417 (SWREG417)	32	RW	Table 15-9
688h	VPU H1 Register 418 (SWREG418)	32	RW	Table 15-9
68Ch	VPU H1 Register 419 (SWREG419)	32	RW	Table 15-9
690h	VPU H1 Register 420 (SWREG420)	32	RW	Table 15-9
694h	VPU H1 Register 421 (SWREG421)	32	RW	Table 15-9
698h	VPU H1 Register 422 (SWREG422)	32	RW	Table 15-9
69Ch	VPU H1 Register 423 (SWREG423)	32	RW	Table 15-9
6A0h	VPU H1 Register 424 (SWREG424)	32	RW	Table 15-9
6A4h	VPU H1 Register 425 (SWREG425)	32	RW	Table 15-9
6A8h	VPU H1 Register 426 (SWREG426)	32	RW	Table 15-9
6ACh	VPU H1 Register 427 (SWREG427)	32	RW	Table 15-9
6B0h	VPU H1 Register 428 (SWREG428)	32	RW	Table 15-9
6B4h	VPU H1 Register 429 (SWREG429)	32	RW	Table 15-9

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
6B8h	VPU H1 Register 430 (SWREG430)	32	RW	Table 15-9
6BCh	VPU H1 Register 431 (SWREG431)	32	RW	Table 15-9
6C0h	VPU H1 Register 432 (SWREG432)	32	RW	Table 15-9
6C4h	VPU H1 Register 433 (SWREG433)	32	RW	Table 15-9
6C8h	VPU H1 Register 434 (SWREG434)	32	RW	Table 15-9
6CCh	VPU H1 Register 435 (SWREG435)	32	RW	Table 15-9
6D0h	VPU H1 Register 436 (SWREG436)	32	RW	Table 15-9
6D4h	VPU H1 Register 437 (SWREG437)	32	RW	Table 15-9
6D8h	VPU H1 Register 438 (SWREG438)	32	RW	Table 15-9
6DCh	VPU H1 Register 439 (SWREG439)	32	RW	Table 15-9
6E0h	VPU H1 Register 440 (SWREG440)	32	RW	Table 15-9
6E4h	VPU H1 Register 441 (SWREG441)	32	RW	Table 15-9
6E8h	VPU H1 Register 442 (SWREG442)	32	RW	Table 15-9
6ECh	VPU H1 Register 443 (SWREG443)	32	RW	Table 15-9
6F0h	VPU H1 Register 444 (SWREG444)	32	RW	Table 15-9
6F4h	VPU H1 Register 445 (SWREG445)	32	RW	Table 15-9
6F8h	VPU H1 Register 446 (SWREG446)	32	RW	Table 15-9
6FCh	VPU H1 Register 447 (SWREG447)	32	RW	Table 15-9
700h	VPU H1 Register 448 (SWREG448)	32	RW	Table 15-9
704h	VPU H1 Register 449 (SWREG449)	32	RW	Table 15-9
708h	VPU H1 Register 450 (SWREG450)	32	RW	Table 15-9
70Ch	VPU H1 Register 451 (SWREG451)	32	RW	Table 15-9
710h	VPU H1 Register 452 (SWREG452)	32	RW	Table 15-9
714h	VPU H1 Register 453 (SWREG453)	32	RW	Table 15-9
718h	VPU H1 Register 454 (SWREG454)	32	RW	Table 15-9
71Ch	VPU H1 Register 455 (SWREG455)	32	RW	Table 15-9
720h	VPU H1 Register 456 (SWREG456)	32	RW	Table 15-9
724h	VPU H1 Register 457 (SWREG457)	32	WO	Table 15-9
728h	VPU H1 Register 458 (SWREG458)	32	WO	Table 15-9
72Ch	VPU H1 Register 459 (SWREG459)	32	WO	Table 15-9
730h	VPU H1 Register 460 (SWREG460)	32	WO	Table 15-9
734h	VPU H1 Register 461 (SWREG461)	32	WO	Table 15-9
738h	VPU H1 Register 462 (SWREG462)	32	WO	Table 15-9
73Ch	VPU H1 Register 463 (SWREG463)	32	WO	Table 15-9
740h	VPU H1 Register 464 (SWREG464)	32	WO	Table 15-9
744h	VPU H1 Register 465 (SWREG465)	32	WO	Table 15-9
748h	VPU H1 Register 466 (SWREG466)	32	WO	Table 15-9
74Ch	VPU H1 Register 467 (SWREG467)	32	WO	Table 15-9
750h	VPU H1 Register 468 (SWREG468)	32	WO	Table 15-9
754h	VPU H1 Register 469 (SWREG469)	32	WO	Table 15-9

Table continues on the next page...

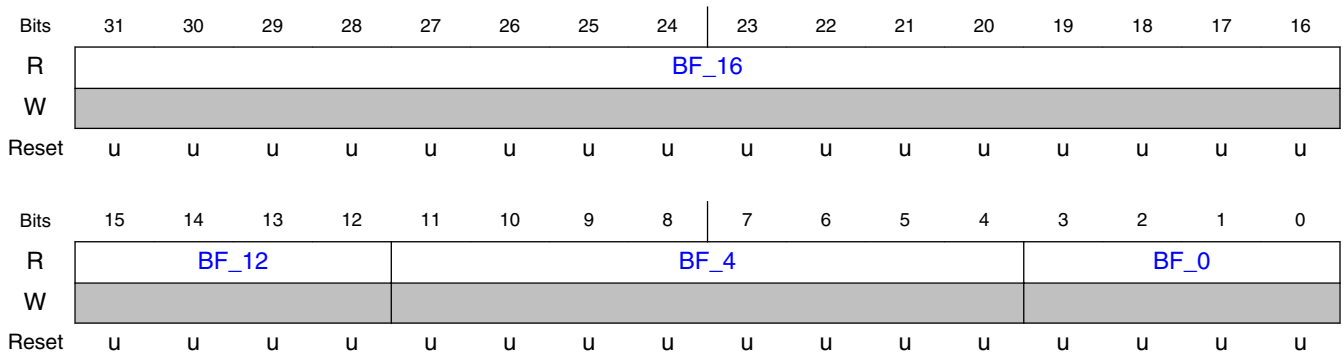
Offset	Register	Width (In bits)	Access	Reset value
758h	VPU H1 Register 470 (SWREG470)	32	WO	Table 15-9
75Ch	VPU H1 Register 471 (SWREG471)	32	WO	Table 15-9
760h	VPU H1 Register 472 (SWREG472)	32	WO	Table 15-9
764h	VPU H1 Register 473 (SWREG473)	32	WO	Table 15-9
768h	VPU H1 Register 474 (SWREG474)	32	WO	Table 15-9
76Ch	VPU H1 Register 475 (SWREG475)	32	WO	Table 15-9
770h	VPU H1 Register 476 (SWREG476)	32	WO	Table 15-9
774h	VPU H1 Register 477 (SWREG477)	32	WO	Table 15-9
778h	VPU H1 Register 478 (SWREG478)	32	WO	Table 15-9
77Ch	VPU H1 Register 479 (SWREG479)	32	WO	Table 15-9
780h	VPU H1 Register 480 (SWREG480)	32	WO	Table 15-9
784h	VPU H1 Register 481 (SWREG481)	32	WO	Table 15-9
788h	VPU H1 Register 482 (SWREG482)	32	WO	Table 15-9
78Ch	VPU H1 Register 483 (SWREG483)	32	WO	Table 15-9
790h	VPU H1 Register 484 (SWREG484)	32	WO	Table 15-9
794h	VPU H1 Register 485 (SWREG485)	32	WO	Table 15-9
798h	VPU H1 Register 486 (SWREG486)	32	WO	Table 15-9
79Ch	VPU H1 Register 487 (SWREG487)	32	WO	Table 15-9
7A0h	VPU H1 Register 488 (SWREG488)	32	WO	Table 15-9
7A4h	VPU H1 Register 489 (SWREG489)	32	WO	Table 15-9
7A8h	VPU H1 Register 490 (SWREG490)	32	WO	Table 15-9
7ACh	VPU H1 Register 491 (SWREG491)	32	WO	Table 15-9
7B0h	VPU H1 Register 492 (SWREG492)	32	WO	Table 15-9
7B4h	VPU H1 Register 493 (SWREG493)	32	WO	Table 15-9
7B8h	VPU H1 Register 494 (SWREG494)	32	WO	Table 15-9
7BCh	VPU H1 Register 495 (SWREG495)	32	WO	Table 15-9
7C0h	VPU H1 Register 496 (SWREG496)	32	WO	Table 15-9
7C4h	VPU H1 Register 497 (SWREG497)	32	RW	Table 15-9

15.3.2.1.2 VPU H1 Register 0 (SWREG0)

15.3.2.1.2.1 Offset

Register	Offset
SWREG0	0h

15.3.2.1.2.2 Diagram



15.3.2.1.2.3 Fields

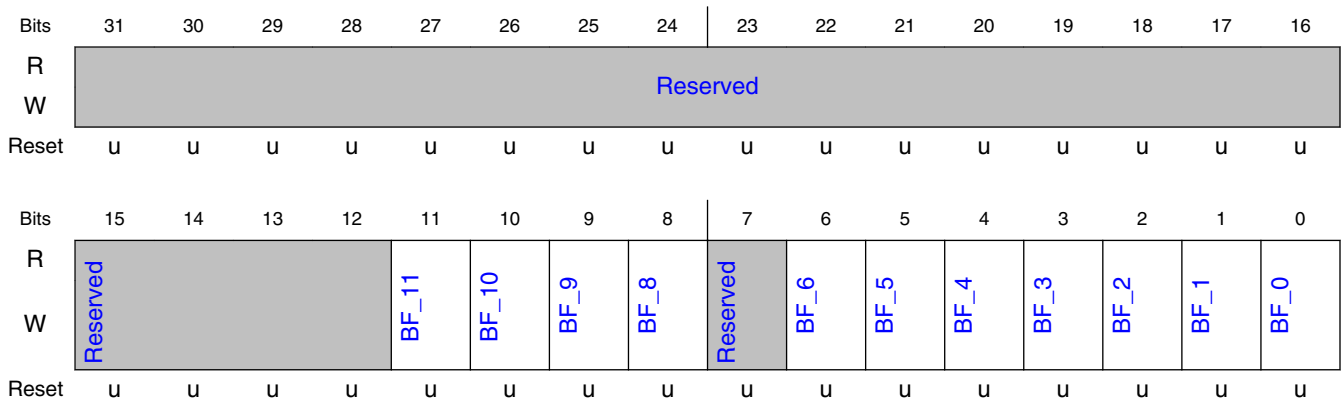
Field	Function
31-16 BF_16	Product ID
15-12 BF_12	Major number
11-4 BF_4	Minor number
3-0 BF_0	Build number defined in synthesis.

15.3.2.1.3 VPU H1 Register 1 (SWREG1)

15.3.2.1.3.1 Offset

Register	Offset
SWREG1	4h

15.3.2.1.3.2 Diagram



15.3.2.1.3.3 Fields

Field	Function
31-12 —	Reserved.
11 BF_11	IRQ Compressed reference buffer overflow status bit.
10 BF_10	IRQ line buffer status bit. One line buffer reading has been done.
9 BF_9	IRQ stream format fuse status bit. Format disabled by fuse.
8 BF_8	IRQ slice ready status bit. Slice is ready but frame encoding continues.
7 —	Reserved.
6 BF_6	IRQ HW timeout status bit. HW encoding failed.
5 BF_5	IRQ buffer full status bit. bufferFullInterrupt
4 BF_4	IRQ SW reset status bit.
3 BF_3	IRQ bus error status bit.
2 BF_2	IRQ frame ready status bit. Encoder has finished a frame.
1 BF_1	IRQ disable. No interrupts from HW. SW must use polling.

Table continues on the next page...

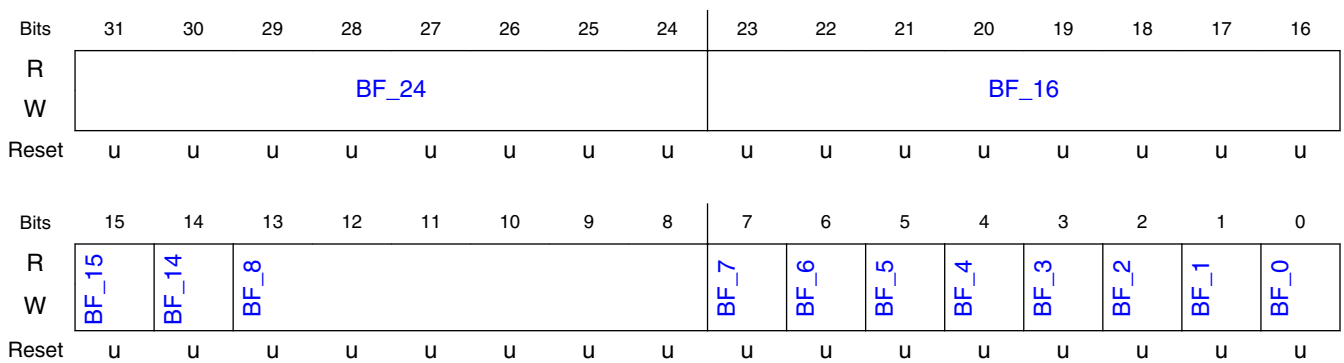
Field	Function
0 BF_0	HINTenc Interrupt from HW. SW resets at IRQ handler.

15.3.2.1.4 VPU H1 Register 2 (SWREG2)

15.3.2.1.4.1 Offset

Register	Offset
SWREG2	8h

15.3.2.1.4.2 Diagram



15.3.2.1.4.3 Fields

Field	Function
31-24 BF_24	AXI Write ID
23-16 BF_16	AXI Read ID
15 BF_15	Enable output stream swap 16-bits
14 BF_14	Enable input swap 16-bits
13-8 BF_8	Burst length. 0=incremental. 4=max BURST4.

Table continues on the next page...

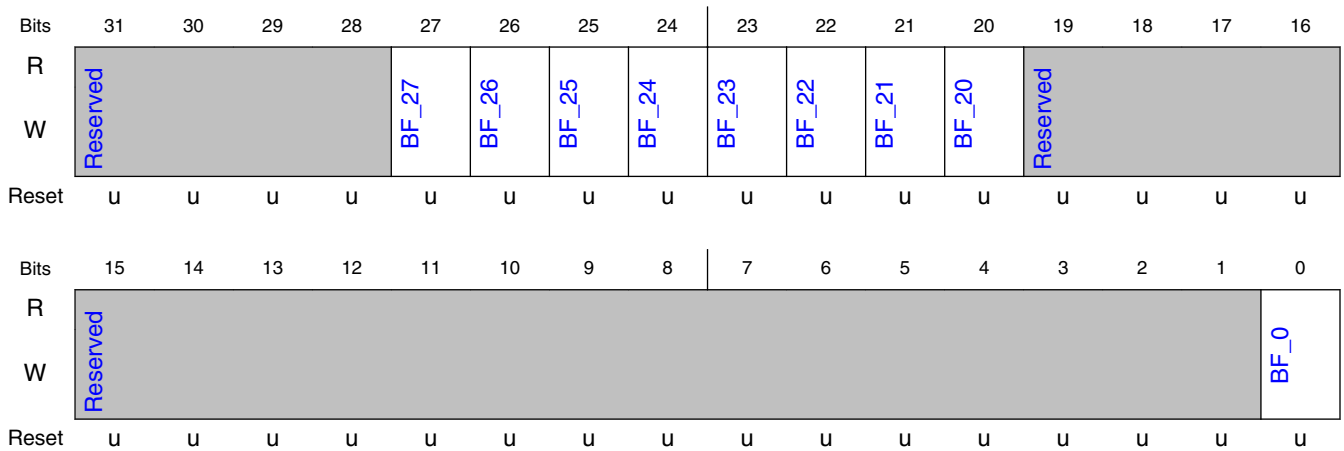
Field	Function
	8=max BURST8. 16=max BURST16.
7 BF_7	Disable burst mode for AXI
6 BF_6	Burst incremental. 1=INCR burst allowed. 0=use SINGLE burst.
5 BF_5	Enable burst data discard. 2 or 3 long reads are using BURST4
4 BF_4	Enable clock gating
3 BF_3	Enable output stream swap 32-bits
2 BF_2	Enable input swap 32-bits
1 BF_1	Enable output stream swap 8-bits
0 BF_0	Enable input swap 8-bits

15.3.2.1.5 VPU H1 Register 3 (SWREG3)

15.3.2.1.5.1 Offset

Register	Offset
SWREG3	Ch

15.3.2.1.5.2 Diagram



15.3.2.1.5.3 Fields

Field	Function
31-28 —	Reserved.
27 BF_27	Enable output scaled image swap for 8-bits
26 BF_26	Enable output scaled image swap for 16-bits
25 BF_25	Enable output scaled image swap for 32-bits
24 BF_24	Enable output MV (MB info) swap for 8-bits
23 BF_23	Enable output MV (MB info) swap for 16-bits
22 BF_22	Enable output MV (MB info) swap for 32-bits
21 BF_21	Chunk size for input picture read. 0=4 MBs. 1=1 MB.
20 BF_20	Disable dual channel AXI. 0=use two channels. 1=use single channel.
19-1 —	Reserved.
0 BF_0	Enable MB bus timing info output.

15.3.2.1.6 VPU H1 Register 5 (SWREG5)

15.3.2.1.6.1 Offset

Register	Offset
SWREG5	14h

15.3.2.1.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.6.3 Fields

Field	Function
31-0 BF_0	Base address for output stream data

15.3.2.1.7 VPU H1 Register 6 (SWREG6)

15.3.2.1.7.1 Offset

Register	Offset
SWREG6	18h

15.3.2.1.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.7.3 Fields

Field	Function
31-0 BF_0	Base address for output control data

15.3.2.1.8 VPU H1 Register 7 (SWREG7)

15.3.2.1.8.1 Offset

Register	Offset
SWREG7	1Ch

15.3.2.1.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.8.3 Fields

Field	Function
31-0 BF_0	Base address for reference luma

15.3.2.1.9 VPU H1 Register 8 (SWREG8)

15.3.2.1.9.1 Offset

Register	Offset
SWREG8	20h

15.3.2.1.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.9.3 Fields

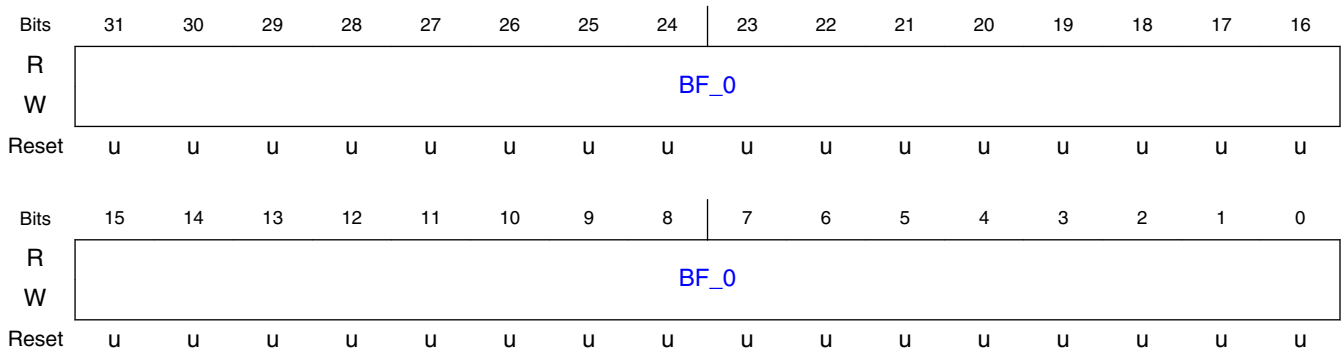
Field	Function
31-0 BF_0	Base address for reference chroma

15.3.2.1.10 VPU H1 Register 9 (SWREG9)

15.3.2.1.10.1 Offset

Register	Offset
SWREG9	24h

15.3.2.1.10.2 Diagram



15.3.2.1.10.3 Fields

Field	Function
31-0 BF_0	Base address for reconstructed luma

15.3.2.1.11 VPU H1 Register 10 (SWREG10)

15.3.2.1.11.1 Offset

Register	Offset
SWREG10	28h

15.3.2.1.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.11.3 Fields

Field	Function
31-0 BF_0	Base address for reconstructed chroma

15.3.2.1.12 VPU H1 Register 11 (SWREG11)

15.3.2.1.12.1 Offset

Register	Offset
SWREG11	2Ch

15.3.2.1.12.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.12.3 Fields

Field	Function
31-0 BF_0	Base address for input picture luma

15.3.2.1.13 VPU H1 Register 12 (SWREG12)

15.3.2.1.13.1 Offset

Register	Offset
SWREG12	30h

15.3.2.1.13.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.13.3 Fields

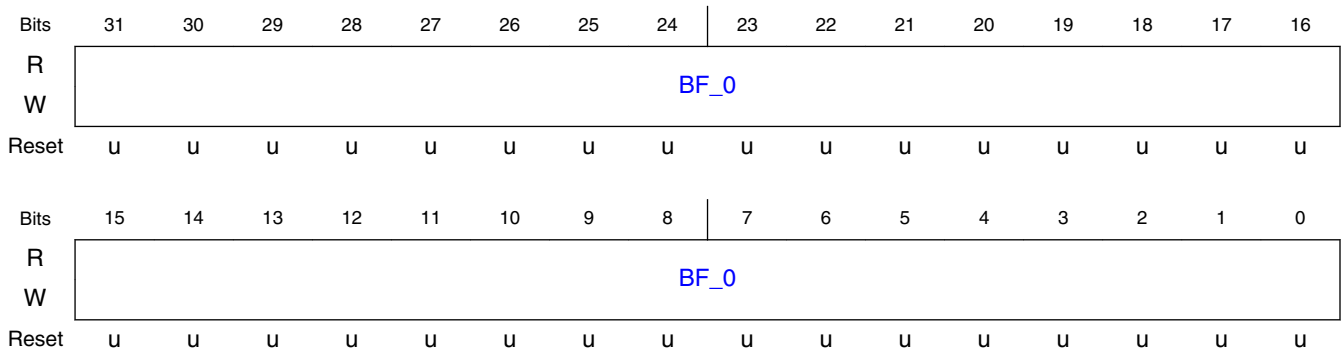
Field	Function
31-0 BF_0	Base address for input picture cb

15.3.2.1.14 VPU H1 Register 13 (SWREG13)

15.3.2.1.14.1 Offset

Register	Offset
SWREG13	34h

15.3.2.1.14.2 Diagram



15.3.2.1.14.3 Fields

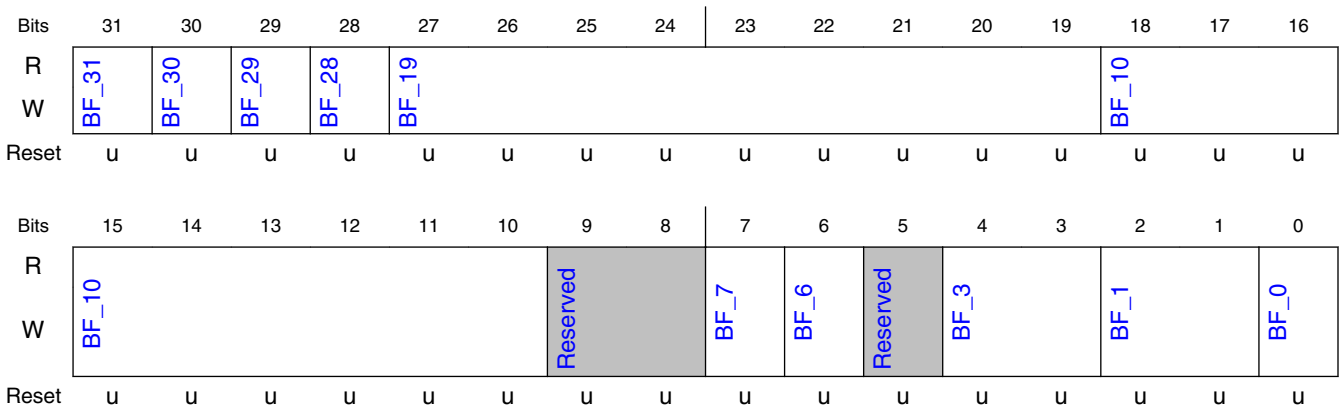
Field	Function
31-0 BF_0	Base address for input picture cr

15.3.2.1.15 VPU H1 Register 14 (SWREG14)

15.3.2.1.15.1 Offset

Register	Offset
SWREG14	38h

15.3.2.1.15.2 Diagram



15.3.2.1.15.3 Fields

Field	Function
31 BF_31	Enable interrupt for timeout
30 BF_30	Enable writing MV and SAD of each MB to BaseMvWrite
29 BF_29	Enable writing size of each NAL unit to BaseControl, nalSizeWriteOut
28 BF_28	Enable interrupt for slice ready
27-19 BF_19	Encoded width. lumWidth (macroblocks) H264:[9..255] JPEG:[6..511]
18-10 BF_10	Encoded height. lumHeight (macroblocks) H264:[6..255] JPEG:[2..511]
9-8 —	Reserved.
7 BF_7	Size of buffer used when writing reconstructed image. 0=1 MB buffered (write burst for every MB). 1=4 MBs buffered (write burst for every fourth MB).
6 BF_6	Disable writing of reconstructed image. recWriteDisable
5 —	Reserved.
4-3 BF_3	Encoded picture type. frameType. 0=INTER. 1=INTRA(IDR).

Table continues on the next page...

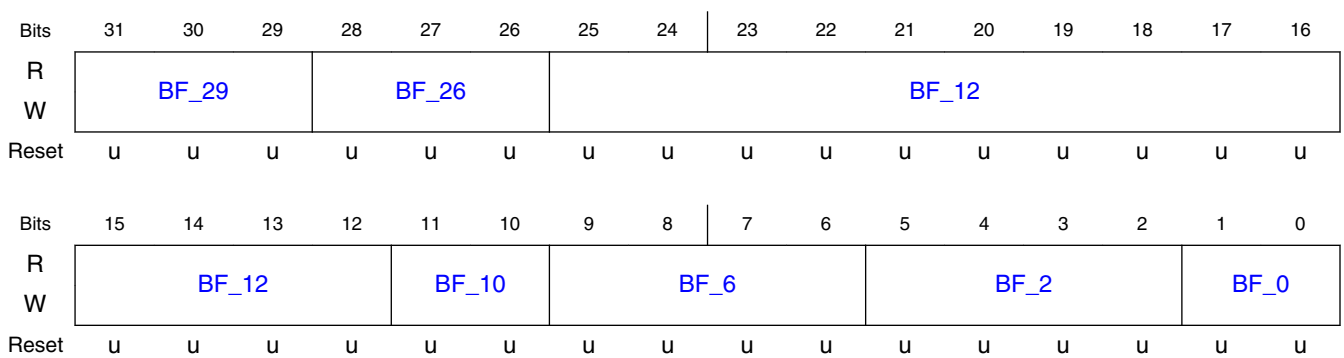
Field	Function
	2=MVC-INTER. 3=MVC-INTER(ref mod).
2-1 BF_1	Encoding mode. streamType. 1=VP8. 2=JPEG. 3=H264.
0 BF_0	Encoder enable

15.3.2.1.16 VPU H1 Register 15 (SWREG15)

15.3.2.1.16.1 Offset

Register	Offset
SWREG15	3Ch

15.3.2.1.16.2 Diagram



15.3.2.1.16.3 Fields

Field	Function
31-29 BF_29	Input chrominance offset (bytes) [0..7]
28-26 BF_26	Input luminance offset (bytes) [0..7]
25-12	Input luminance row length. lumWidthSrc (bytes) [96..8192]

Table continues on the next page...

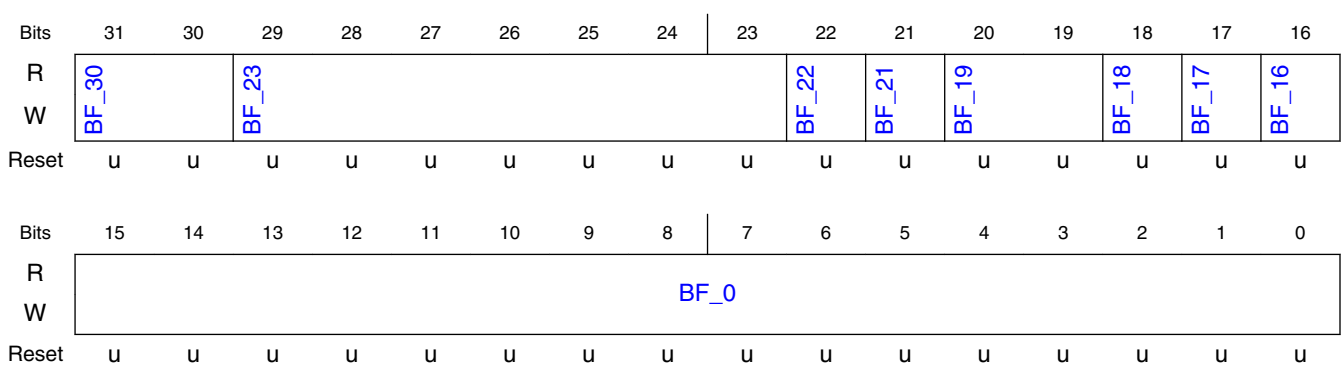
Field	Function
BF_12	
11-10 BF_10	Overfill pixels on right edge of image div4 [0..3]
9-6 BF_6	Overfill pixels on bottom edge of image. YFill. [0..15]
5-2 BF_2	Input image format. inputFormat. YUV420P/YUV420SP/YUYV422/UYVY422/RGB565/RGB555/RGB444/RGB888/RGB101010
1-0 BF_0	Input image rotation. 0=disabled. 1=90 degrees right. 2=90 degrees left.

15.3.2.1.17 VPU H1 Register 18 (SWREG18)

15.3.2.1.17.1 Offset

Register	Offset
SWREG18	48h

15.3.2.1.17.2 Diagram



15.3.2.1.17.3 Fields

Field	Function
31-30 BF_30	Deblocking filter mode. 0=enabled. 1=disabled (vp8=simple).

Table continues on the next page...

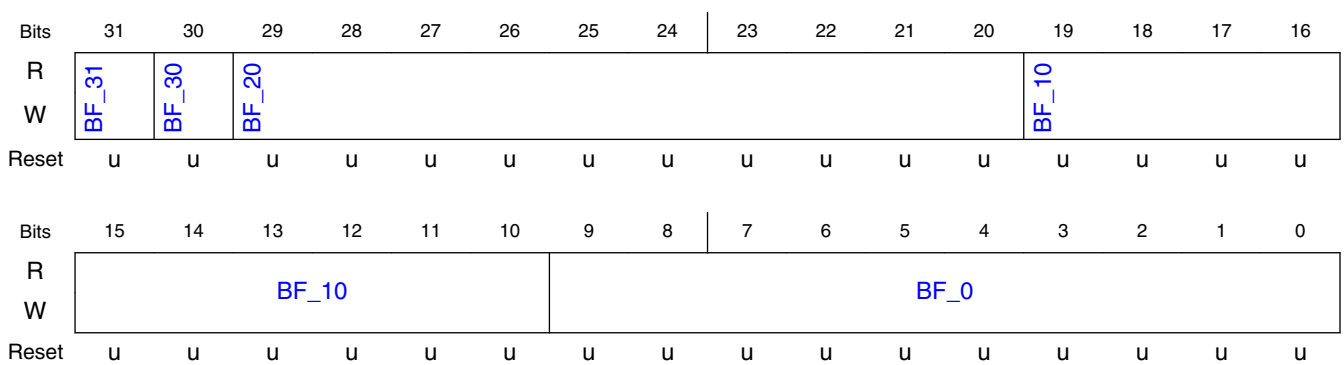
Field	Function
	2=disabled on slice borders.
29-23 BF_23	H.264 Slice size. mbRowPerSlice (mb rows) [0..127] 0=one slice per picture
22 BF_22	H.264 Disable quarter pixel MVs. disableQuarterPixelMv
21 BF_21	H.264 Transform 8x8 enable. High Profile H.264. transform8x8Mode
20-19 BF_19	H.264 CABAC initial IDC. [0..2]
18 BF_18	H.264 CABAC / VP8 boolenc enable. entropyCodingMode. 0=CAVLC (Baseline Profile H.264). 1=CABAC (Main Profile H.264).
17 BF_17	H.264 Inter 4x4 mode restriction. restricted4x4Mode
16 BF_16	H.264 Stream mode. byteStream. 0=NAL unit stream. 1=Byte stream.
15-0 BF_0	Intra prediction intra 16x16 mode favor

15.3.2.1.18 VPU H1 Register 19 (SWREG19)

15.3.2.1.18.1 Offset

Register	Offset
SWREG19	4Ch

15.3.2.1.18.2 Diagram



15.3.2.1.18.3 Fields

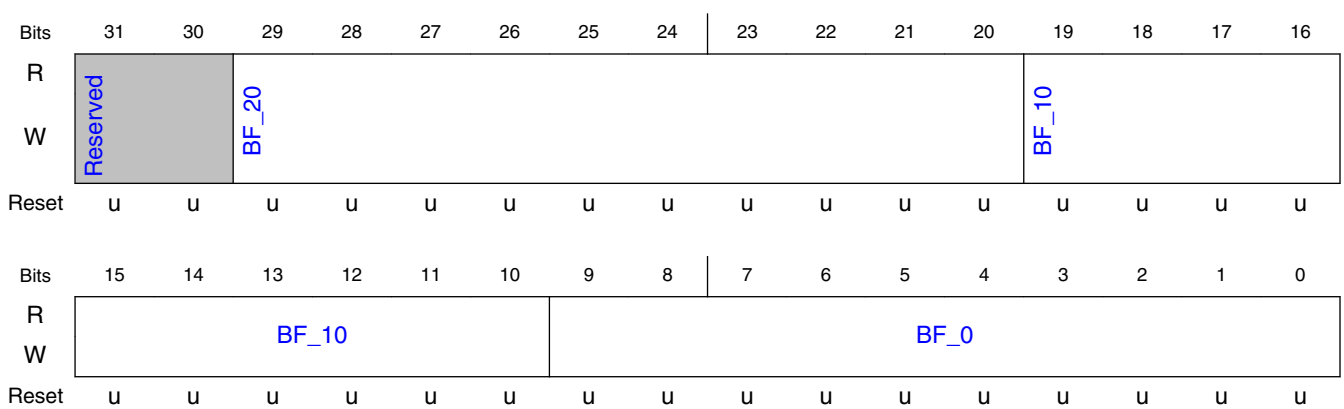
Field	Function
31 BF_31	Swap order of chroma bytes in semiplanar input format.
30 BF_30	Enable using more than 1 MV per macroblock.
29-20 BF_20	Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-10 BF_10	Differential MV penalty for 4p ME. DMVPenalty4p
9-0 BF_0	Differential MV penalty for 1p ME. DMVPenalty1p

15.3.2.1.19 VPU H1 Register 20 (SWREG20)

15.3.2.1.19.1 Offset

Register	Offset
SWREG20	50h

15.3.2.1.19.2 Diagram



15.3.2.1.19.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Penalty for using 8x8 MV.
9-0 BF_0	Penalty for using 8x4 or 4x8 MV.

15.3.2.1.20 VPU H1 Register 21 (SWREG21)

15.3.2.1.20.1 Offset

Register	Offset
SWREG21	54h

15.3.2.1.20.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.20.3 Fields

Field	Function
31-24 BF_24	H.264 SKIP macroblock mode / VP8 zero/nearest/near mode penalty
23-16	H.264 amount of completed slices.

Table continues on the next page...

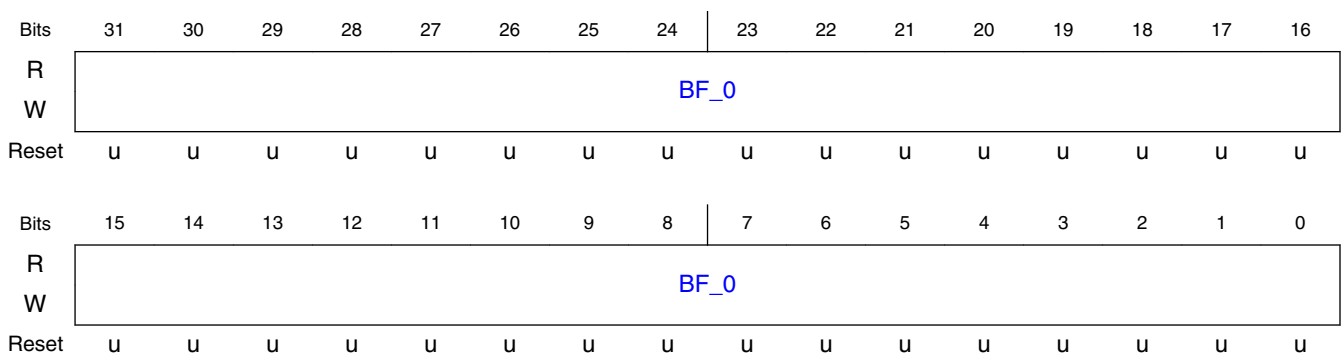
Field	Function
BF_16	
15-0 BF_0	Inter MB mode favor in intra/inter selection

15.3.2.1.21 VPU H1 Register 22 (SWREG22)

15.3.2.1.21.1 Offset

Register	Offset
SWREG22	58h

15.3.2.1.21.2 Diagram



15.3.2.1.21.3 Fields

Field	Function
31-0 BF_0	Stream header remainder bits MSB (MSB aligned)

15.3.2.1.22 VPU H1 Register 23 (SWREG23)

15.3.2.1.22.1 Offset

Register	Offset
SWREG23	5Ch

15.3.2.1.22.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.22.3 Fields

Field	Function
31-0 BF_0	Stream header remainder bits LSB (MSB aligned)

15.3.2.1.23 VPU H1 Register 24 (SWREG24)

15.3.2.1.23.1 Offset

Register	Offset
SWREG24	60h

15.3.2.1.23.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.23.3 Fields

Field	Function
31-0 BF_0	Stream buffer limit (64bit addresses) / output stream size (bits). HWStreamDataCount. If limit is reached buffer full IRQ is given.

15.3.2.1.24 VPU H1 Register 25 (SWREG25)

15.3.2.1.24.1 Offset

Register	Offset
SWREG25	64h

15.3.2.1.24.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_28				BF_22						BF_0					
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.24.3 Fields

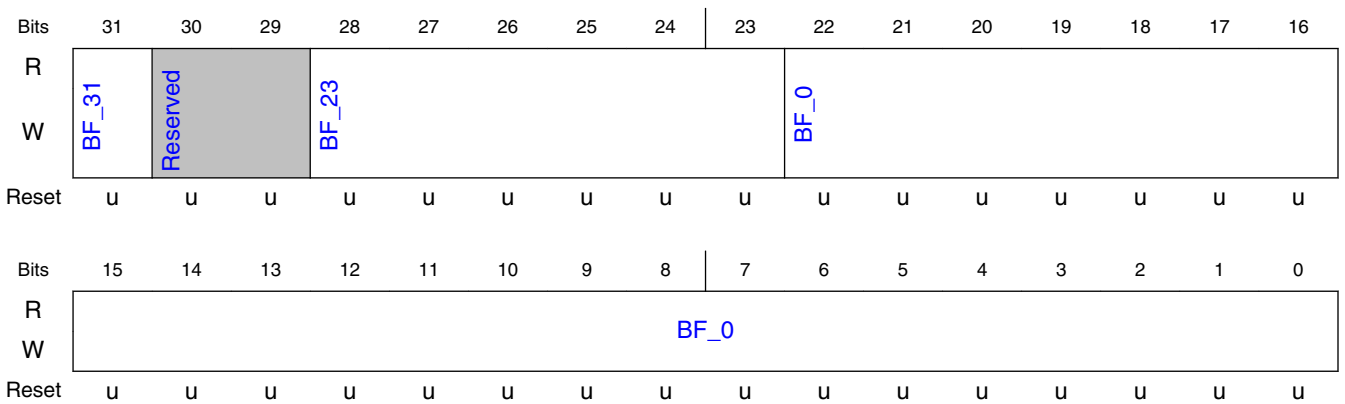
Field	Function
31-28 BF_28	MAD based QP adjustment. madQpChange [-8..7]
27-22 BF_22	MAD threshold div256
21-0 BF_0	QP Sum div2 output

15.3.2.1.25 VPU H1 Register 37 (SWREG37)

15.3.2.1.25.1 Offset

Register	Offset
SWREG37	94h

15.3.2.1.25.2 Diagram



15.3.2.1.25.3 Fields

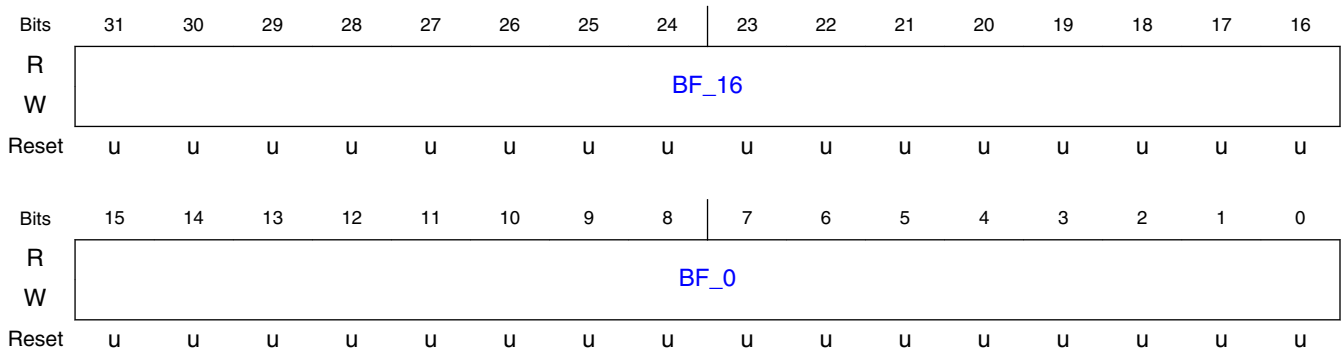
Field	Function
31 BF_31	Mode for VP8 interpolation filter. 0=bicubic. 1=bilinear.
30-29 —	Reserved.
28-23 BF_23	Stream start offset = amount of StrmHdrRem (bits) [0..63]
22-0 BF_0	RLC codeword count div4 output. max 255*255*384/4

15.3.2.1.26 VPU H1 Register 38 (SWREG38)

15.3.2.1.26.1 Offset

Register	Offset
SWREG38	98h

15.3.2.1.26.2 Diagram



15.3.2.1.26.3 Fields

Field	Function
31-16 BF_16	Macroblock count with MAD value under threshold output
15-0 BF_0	MB count output. max 255*255

15.3.2.1.27 VPU H1 Register 39 (SWREG39)

15.3.2.1.27.1 Offset

Register	Offset
SWREG39	9Ch

15.3.2.1.27.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.27.3 Fields

Field	Function
31-0 BF_0	Base address for next pic luminance

15.3.2.1.28 VPU H1 Register 40 (SWREG40)

15.3.2.1.28.1 Offset

Register	Offset
SWREG40	A0h

15.3.2.1.28.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_30		Reserved						BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.28.3 Fields

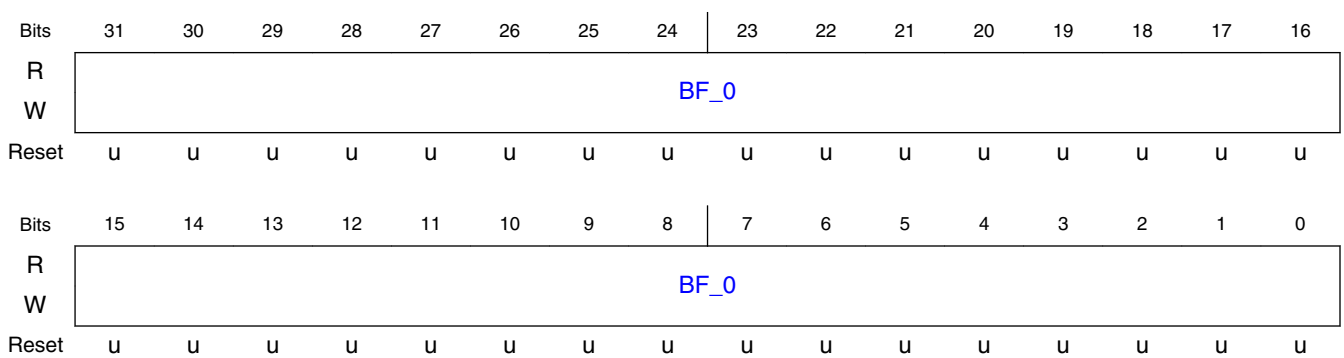
Field	Function
31-30 BF_30	Stabilization mode. 0=disabled. 1=stab only. 2=stab+encode.
29-24 —	Reserved.
23-0 BF_0	Stabilization minimum value output. max 253*253*255

15.3.2.1.29 VPU H1 Register 41 (SWREG41)

15.3.2.1.29.1 Offset

Register	Offset
SWREG41	A4h

15.3.2.1.29.2 Diagram



15.3.2.1.29.3 Fields

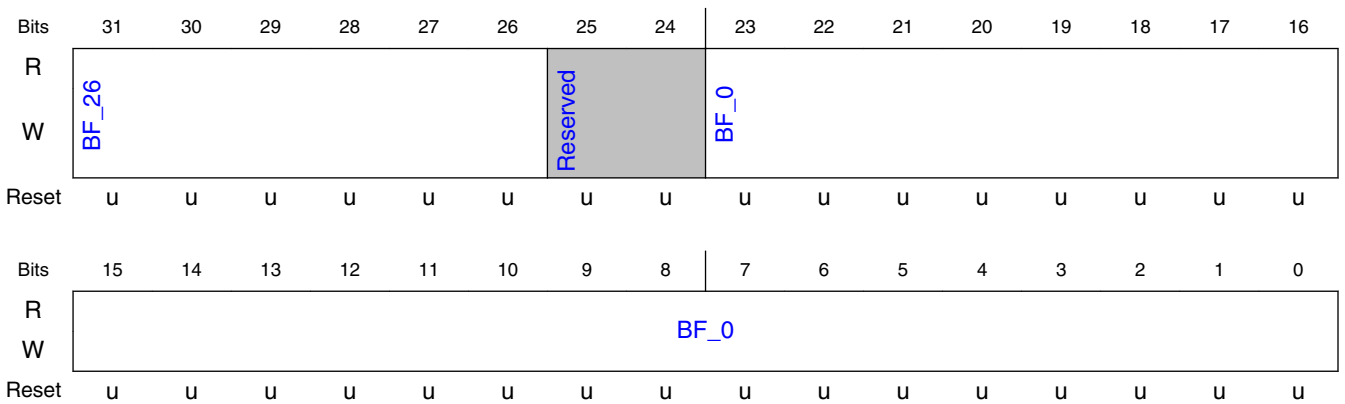
Field	Function
31-0 BF_0	Stabilization motion sum div8 output. max 253*253*255*1089/8

15.3.2.1.30 VPU H1 Register 42 (SWREG42)

15.3.2.1.30.1 Offset

Register	Offset
SWREG42	A8h

15.3.2.1.30.2 Diagram



15.3.2.1.30.3 Fields

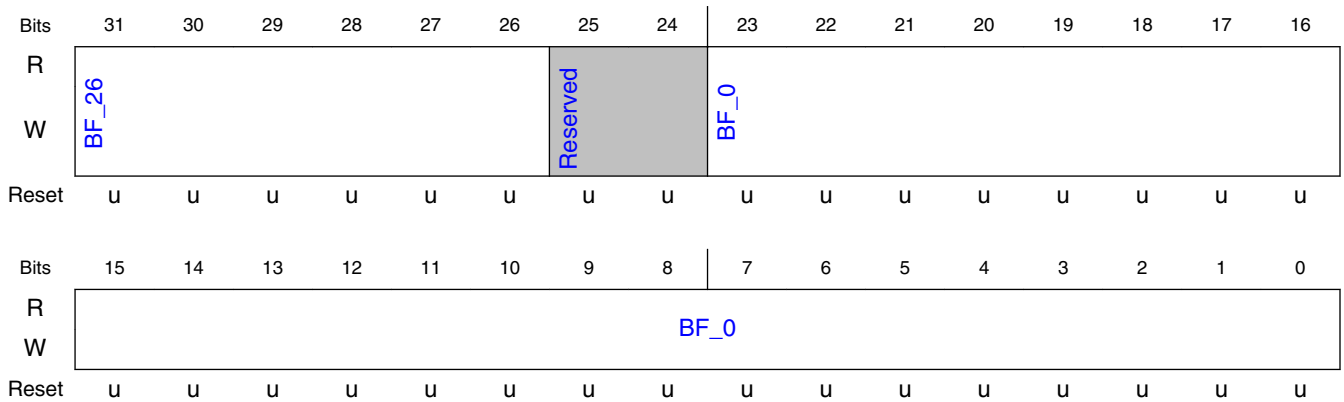
Field	Function
31-26 BF_26	Stabilization GMV horizontal output [-16..16]
25-24 —	Reserved.
23-0 BF_0	Stabilization matrix 1 (up-left position) output

15.3.2.1.31 VPU H1 Register 43 (SWREG43)

15.3.2.1.31.1 Offset

Register	Offset
SWREG43	ACh

15.3.2.1.31.2 Diagram



15.3.2.1.31.3 Fields

Field	Function
31-26 BF_26	Stabilization GMV vertical output [-16..16]
25-24 —	Reserved.
23-0 BF_0	Stabilization matrix 2 (up position) output

15.3.2.1.32 VPU H1 Register 44 (SWREG44)

15.3.2.1.32.1 Offset

Register	Offset
SWREG44	B0h

15.3.2.1.32.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.32.3 Fields

Field	Function
31-24 —	Reserved.
23-0 BF_0	Stabilization matrix 3 (up-right position) output

15.3.2.1.33 VPU H1 Register 45 (SWREG45)

15.3.2.1.33.1 Offset

Register	Offset
SWREG45	B4h

15.3.2.1.33.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.33.3 Fields

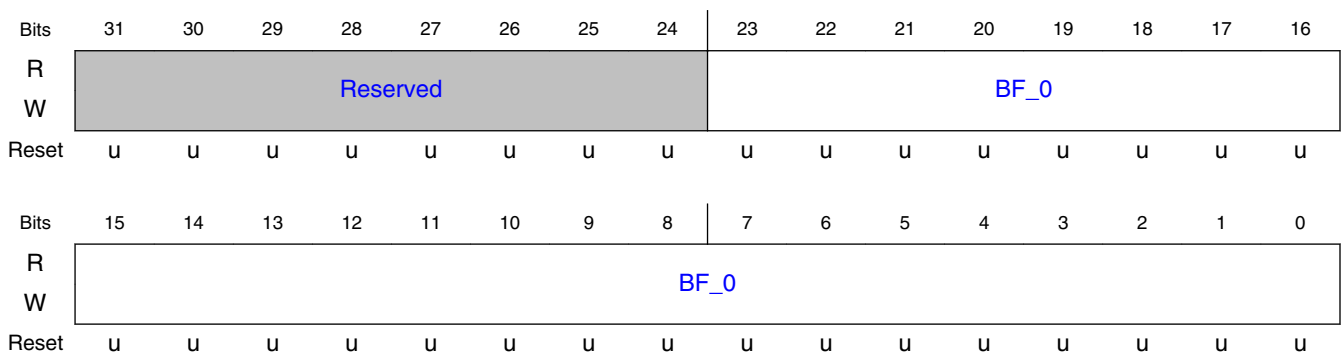
Field	Function
31-24 —	Reserved.
23-0 BF_0	Stabilization matrix 4 (left position) output

15.3.2.1.34 VPU H1 Register 46 (SWREG46)

15.3.2.1.34.1 Offset

Register	Offset
SWREG46	B8h

15.3.2.1.34.2 Diagram



15.3.2.1.34.3 Fields

Field	Function
31-24 —	Reserved.
23-0 BF_0	Stabilization matrix 5 (GMV position) output

15.3.2.1.35 VPU H1 Register 47 (SWREG47)

15.3.2.1.35.1 Offset

Register	Offset
SWREG47	BCh

15.3.2.1.35.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								BF_0							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF_0															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.35.3 Fields

Field	Function
31-24 —	Reserved.
23-0 BF_0	Stabilization matrix 6 (right position) output

15.3.2.1.36 VPU H1 Register 48 (SWREG48)

15.3.2.1.36.1 Offset

Register	Offset
SWREG48	C0h

15.3.2.1.36.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.36.3 Fields

Field	Function
31-24 —	Reserved.
23-0 BF_0	Stabilization matrix 7 (down-left position) output

15.3.2.1.37 VPU H1 Register 49 (SWREG49)

15.3.2.1.37.1 Offset

Register	Offset
SWREG49	C4h

15.3.2.1.37.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.37.3 Fields

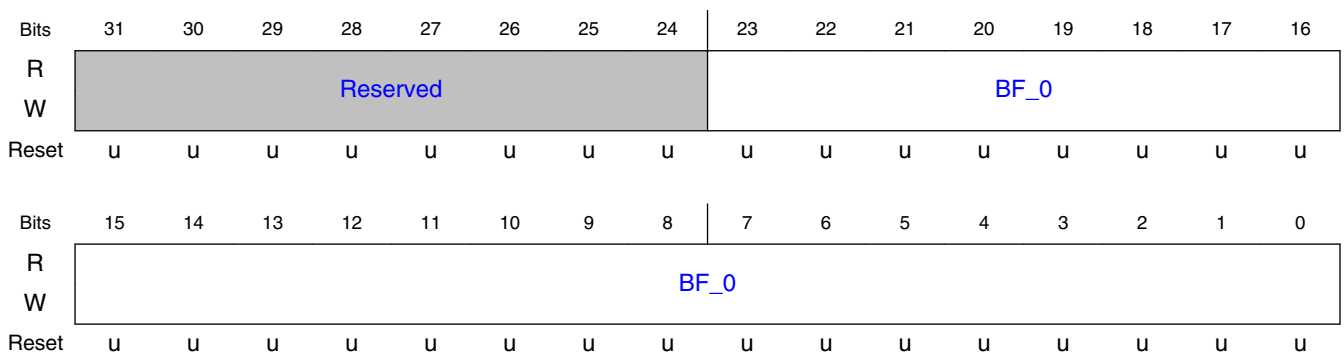
Field	Function
31-24 —	Reserved.
23-0 BF_0	Stabilization matrix 8 (down position) output

15.3.2.1.38 VPU H1 Register 50 (SWREG50)

15.3.2.1.38.1 Offset

Register	Offset
SWREG50	C8h

15.3.2.1.38.2 Diagram



15.3.2.1.38.3 Fields

Field	Function
31-24 —	Reserved.
23-0 BF_0	Stabilization matrix 9 (down-right position) output

15.3.2.1.39 VPU H1 Register 51 (SWREG51)

15.3.2.1.39.1 Offset

Register	Offset
SWREG51	CCh

15.3.2.1.39.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.39.3 Fields

Field	Function
31-0 BF_0	Base address for cabac context tables (H264) or probability tables (VP8)

15.3.2.1.40 VPU H1 Register 52 (SWREG52)

15.3.2.1.40.1 Offset

Register	Offset
SWREG52	D0h

15.3.2.1.40.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.40.3 Fields

Field	Function
31-0 BF_0	Base address for MV output writing

15.3.2.1.41 VPU H1 Register 53 (SWREG53)

15.3.2.1.41.1 Offset

Register	Offset
SWREG53	D4h

15.3.2.1.41.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_16															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.41.3 Fields

Field	Function
31-16 BF_16	RGB to YUV conversion coefficient B
15-0 BF_0	RGB to YUV conversion coefficient A

15.3.2.1.42 VPU H1 Register 54 (SWREG54)

15.3.2.1.42.1 Offset

Register	Offset
SWREG54	D8h

15.3.2.1.42.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_16															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.42.3 Fields

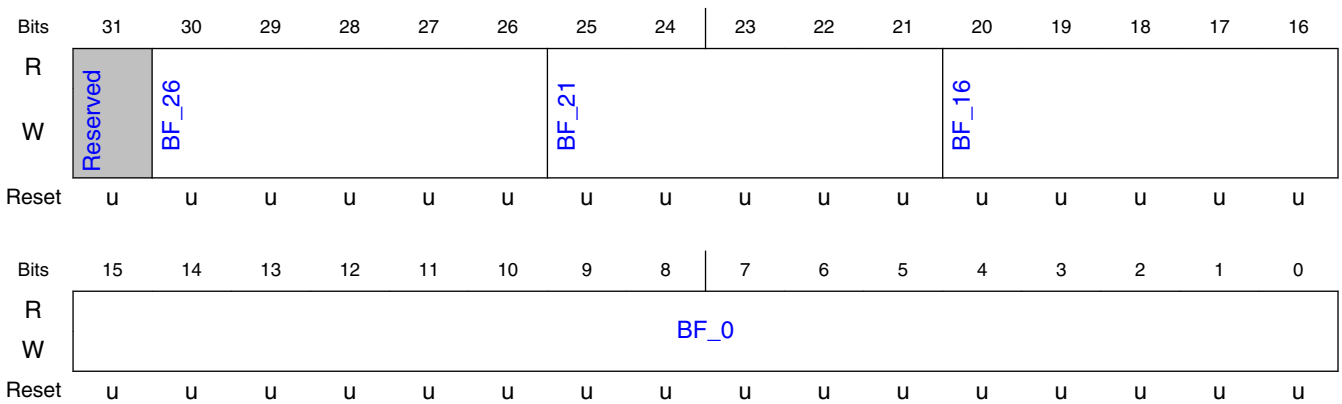
Field	Function
31-16 BF_16	RGB to YUV conversion coefficient E
15-0 BF_0	RGB to YUV conversion coefficient C

15.3.2.1.43 VPU H1 Register 55 (SWREG55)

15.3.2.1.43.1 Offset

Register	Offset
SWREG55	DCh

15.3.2.1.43.2 Diagram



15.3.2.1.43.3 Fields

Field	Function
31 —	Reserved.
30-26 BF_26	RGB B-component mask MSB bit position [0..31]
25-21 BF_21	RGB G-component mask MSB bit position [0..31]
20-16 BF_16	RGB R-component mask MSB bit position [0..31]
15-0 BF_0	RGB to YUV conversion coefficient F

15.3.2.1.44 VPU H1 Register 56 (SWREG56)

15.3.2.1.44.1 Offset

Register	Offset
SWREG56	E0h

15.3.2.1.44.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.44.3 Fields

Field	Function
31-24 BF_24	Intra area left mb column (inside area) [0..255]
23-16 BF_16	Intra area right mb column (outside area) [0..255]
15-8 BF_8	Intra area top mb row (inside area) [0..255]
7-0 BF_0	Intra area bottom mb row (outside area) [0..255]

15.3.2.1.45 VPU H1 Register 57 (SWREG57)

15.3.2.1.45.1 Offset

Register	Offset
SWREG57	E4h

15.3.2.1.45.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_16															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.45.3 Fields

Field	Function
31-16 BF_16	CIR first intra mb. 0=disabled [0..65535]
15-0 BF_0	CIR intra mb interval. 0=disabled [0..65535]

15.3.2.1.46 VPU H1 Register 60 (SWREG60)

15.3.2.1.46.1 Offset

Register	Offset
SWREG60	F0h

15.3.2.1.46.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.46.3 Fields

Field	Function
31-24 BF_24	1st ROI area left mb column (inside area) qp+=Roi1DeltaQp
23-16 BF_16	1st ROI area right mb column (outside area) qp-=Roi1DeltaQp
15-8 BF_8	1st ROI area top mb row (inside area)
7-0 BF_0	1st ROI area bottom mb row (outside area)

15.3.2.1.47 VPU H1 Register 61 (SWREG61)

15.3.2.1.47.1 Offset

Register	Offset
SWREG61	F4h

15.3.2.1.47.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.47.3 Fields

Field	Function
31-24 BF_24	2nd ROI area left mb column (inside area) qp+=Roi2DeltaQp

Table continues on the next page...

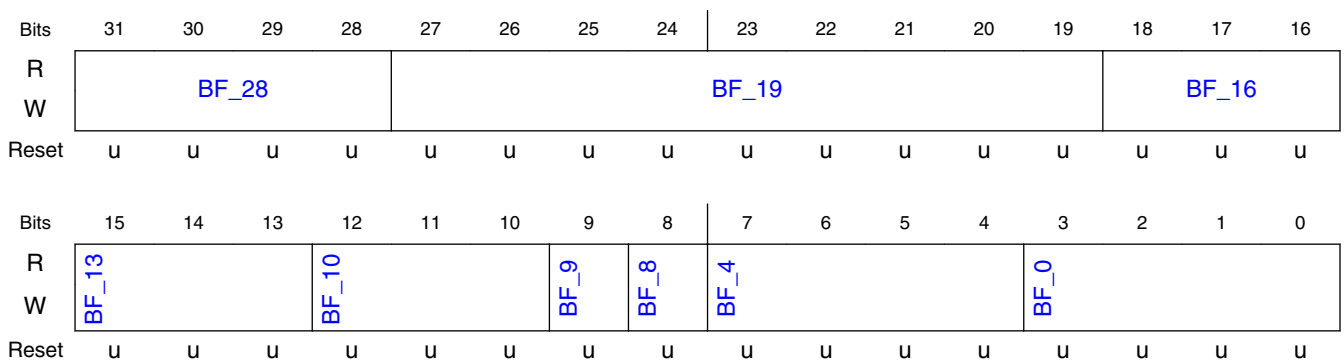
Field	Function
23-16 BF_16	2nd ROI area right mb column (outside area) qp-=Roi2DeltaQp
15-8 BF_8	2nd ROI area top mb row (inside area)
7-0 BF_0	2nd ROI area bottom mb row (outside area)

15.3.2.1.48 VPU H1 Register 62 (SWREG62)

15.3.2.1.48.1 Offset

Register	Offset
SWREG62	F8h

15.3.2.1.48.2 Diagram



15.3.2.1.48.3 Fields

Field	Function
31-28 BF_28	Zero 16x16 MV favor div2.
27-19 BF_19	Penalty for using 4x4 MV.
18-16 BF_16	MVC priority_id [0..7]
15-13	MVC view_id [0..7]

Table continues on the next page...

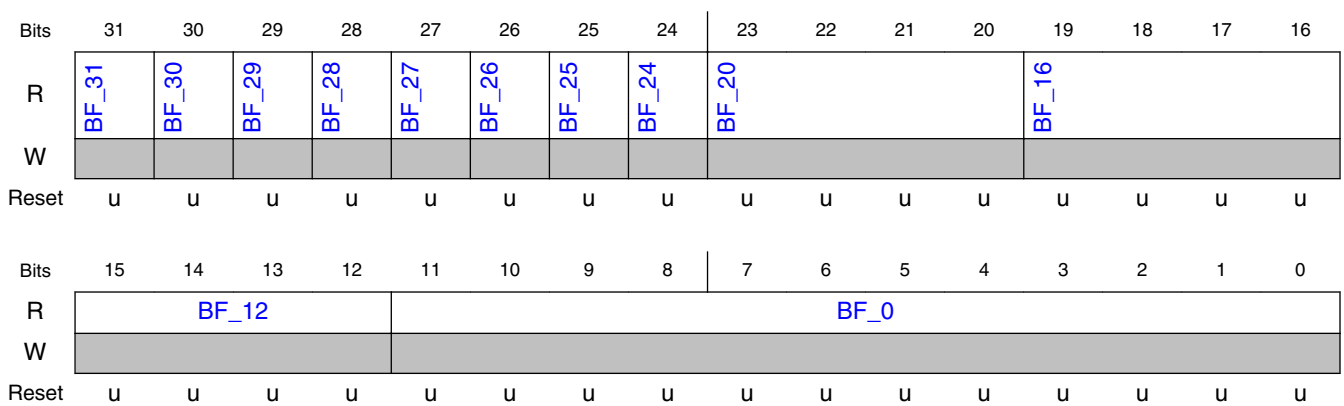
Field	Function
BF_13	
12-10 BF_10	MVC temporal_id [0..7]
9 BF_9	MVC anchor_pic_flag. Specifies that the picture is part of an anchor access unit.
8 BF_8	MVC inter_view_flag. Specifies that the picture is used for inter-view prediction.
7-4 BF_4	1st ROI area delta QP. qp = Qp - Roi1DeltaQp [0..15]
3-0 BF_0	2nd ROI area delta QP. qp = Qp - Roi2DeltaQp [0..15]

15.3.2.1.49 VPU H1 Register 63 (SWREG63)

15.3.2.1.49.1 Offset

Register	Offset
SWREG63	FCh

15.3.2.1.49.2 Diagram



15.3.2.1.49.3 Fields

Field	Function
31 BF_31	HW Chrominance search area internal buffer. 0=not supported.

Table continues on the next page...

Field	Function
	1=supported.
30 BF_30	Down-scaling supported by HW. 0=not supported. 1=supported.
29 BF_29	HW search area height. 0=5 MB rows. 1=3 MB rows.
28 BF_28	RGB to YUV conversion supported by HW. 0=not supported. 1=supported.
27 BF_27	H.264 encoding supported by HW. 0=not supported. 1=supported.
26 BF_26	VP8 encoding supported by HW. 0=not supported. 1=supported.
25 BF_25	JPEG encoding supported by HW. 0=not supported. 1=supported.
24 BF_24	Stabilization supported by HW. 0=not supported. 1=supported.
23-20 BF_20	Bus connection of HW. 1=AHB. 2=OCP. 3=AXI. 4=PCI. 5=AXIAHB. 6=AXIAPB.
19-16 BF_16	Synthesis language. 1=vhdl. 2=verilog.
15-12 BF_12	Bus width of HW. 0=32b. 1=64b. 2=128b.
11-0 BF_0	Maximum video width supported by HW (pixels)

15.3.2.1.50 VPU H1 Register 96 (SWREG96)

15.3.2.1.50.1 Offset

Register	Offset
SWREG96	180h

15.3.2.1.50.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.50.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values 0-3

15.3.2.1.51 VPU H1 Register 97 (SWREG97)

15.3.2.1.51.1 Offset

Register	Offset
SWREG97	184h

15.3.2.1.51.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.51.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values 4-7

15.3.2.1.52 VPU H1 Register 98 (SWREG98)

15.3.2.1.52.1 Offset

Register	Offset
SWREG98	188h

15.3.2.1.52.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.52.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.53 VPU H1 Register 99 (SWREG99)

15.3.2.1.53.1 Offset

Register	Offset
SWREG99	18Ch

15.3.2.1.53.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.53.3 Fields

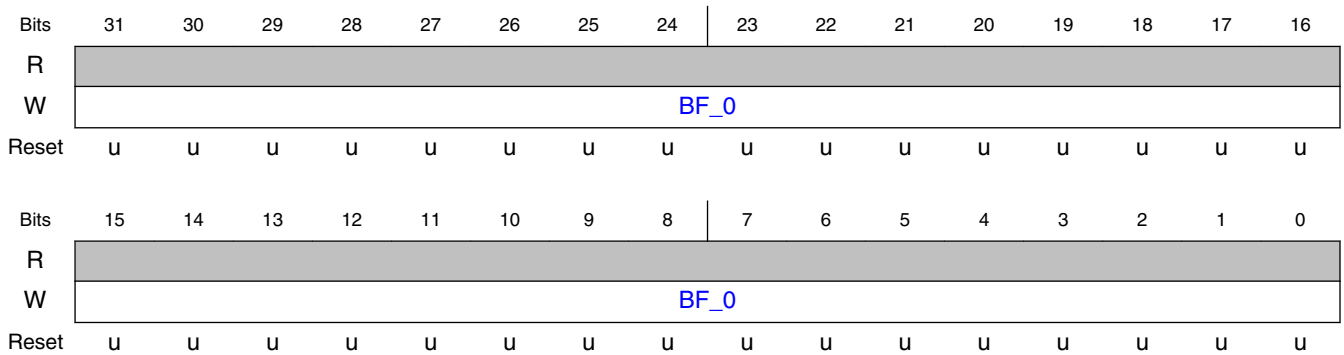
Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.54 VPU H1 Register 100 (SWREG100)

15.3.2.1.54.1 Offset

Register	Offset
SWREG100	190h

15.3.2.1.54.2 Diagram



15.3.2.1.54.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.55 VPU H1 Register 101 (SWREG101)

15.3.2.1.55.1 Offset

Register	Offset
SWREG101	194h

15.3.2.1.55.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.55.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.56 VPU H1 Register 102 (SWREG102)

15.3.2.1.56.1 Offset

Register	Offset
SWREG102	198h

15.3.2.1.56.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.56.3 Fields

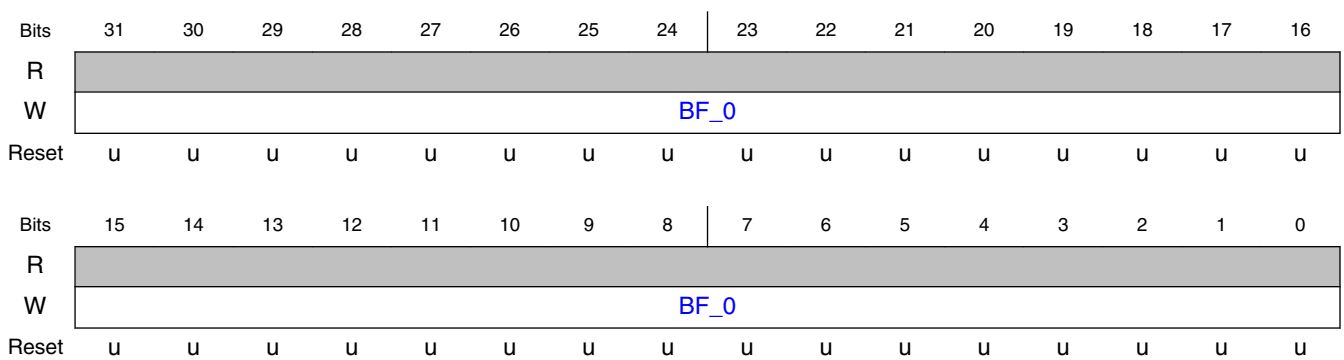
Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.57 VPU H1 Register 103 (SWREG103)

15.3.2.1.57.1 Offset

Register	Offset
SWREG103	19Ch

15.3.2.1.57.2 Diagram



15.3.2.1.57.3 Fields

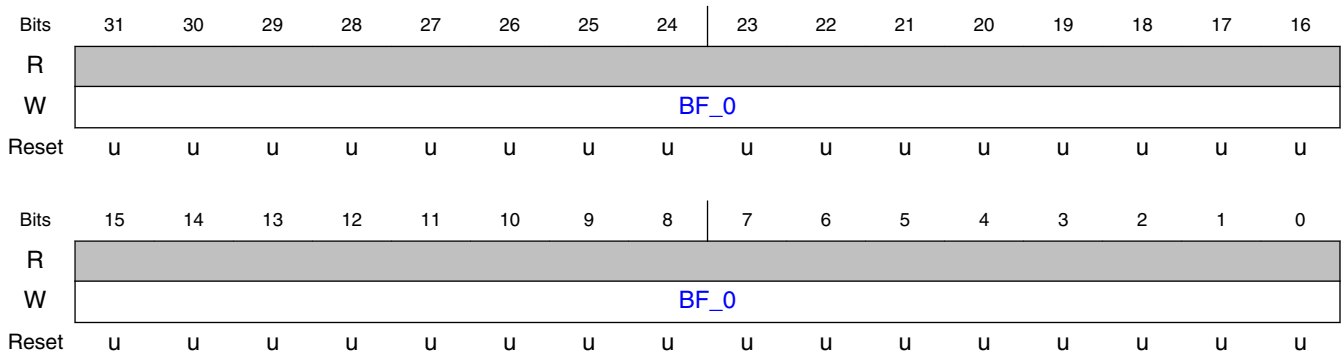
Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.58 VPU H1 Register 104 (SWREG104)

15.3.2.1.58.1 Offset

Register	Offset
SWREG104	1A0h

15.3.2.1.58.2 Diagram



15.3.2.1.58.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.59 VPU H1 Register 105 (SWREG105)

15.3.2.1.59.1 Offset

Register	Offset
SWREG105	1A4h

15.3.2.1.59.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.59.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.60 VPU H1 Register 106 (SWREG106)

15.3.2.1.60.1 Offset

Register	Offset
SWREG106	1A8h

15.3.2.1.60.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.60.3 Fields

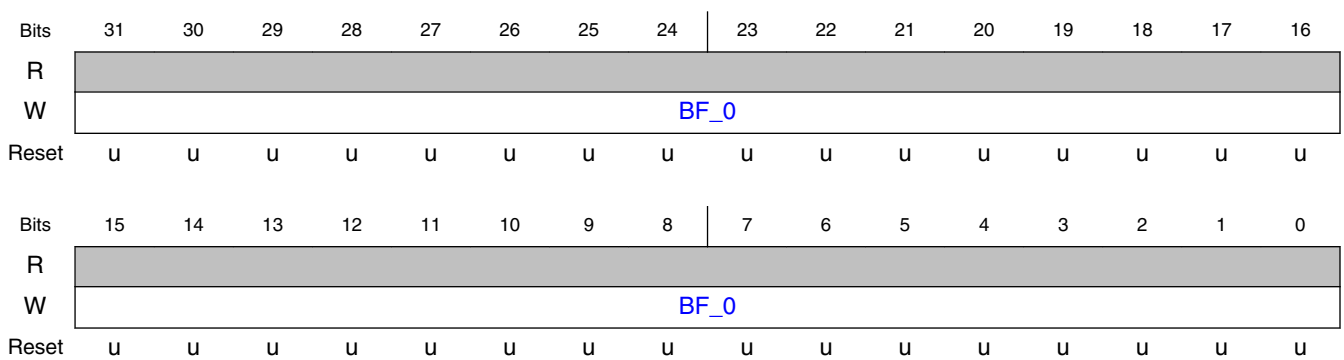
Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.61 VPU H1 Register 107 (SWREG107)

15.3.2.1.61.1 Offset

Register	Offset
SWREG107	1ACh

15.3.2.1.61.2 Diagram



15.3.2.1.61.3 Fields

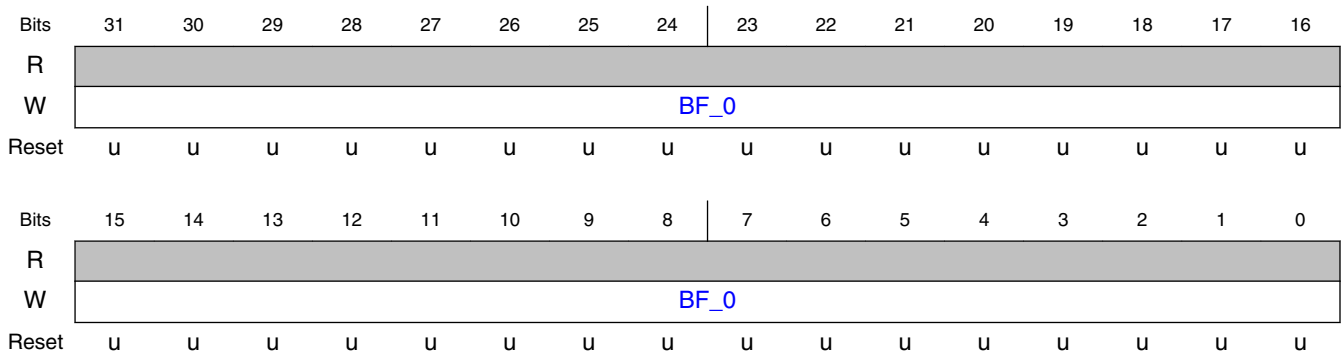
Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.62 VPU H1 Register 108 (SWREG108)

15.3.2.1.62.1 Offset

Register	Offset
SWREG108	1B0h

15.3.2.1.62.2 Diagram



15.3.2.1.62.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.63 VPU H1 Register 109 (SWREG109)

15.3.2.1.63.1 Offset

Register	Offset
SWREG109	1B4h

15.3.2.1.63.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.63.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.64 VPU H1 Register 110 (SWREG110)

15.3.2.1.64.1 Offset

Register	Offset
SWREG110	1B8h

15.3.2.1.64.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.64.3 Fields

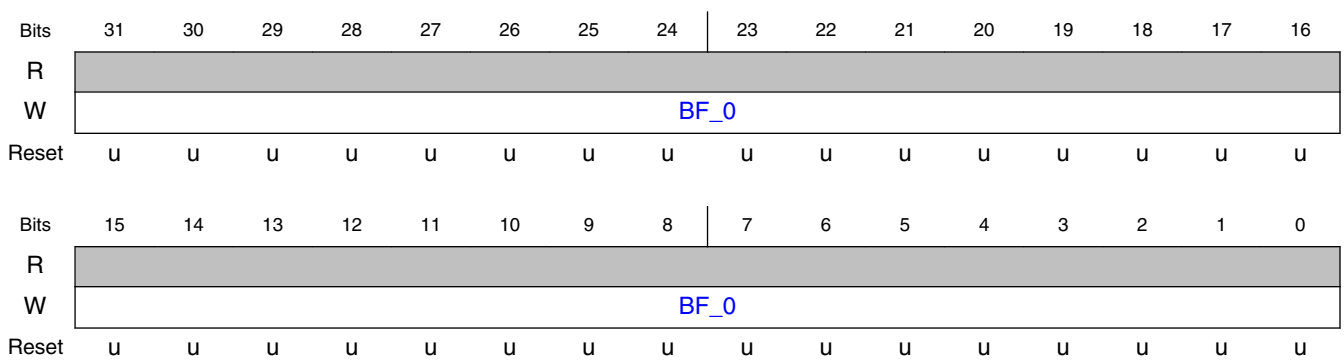
Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.65 VPU H1 Register 111 (SWREG111)

15.3.2.1.65.1 Offset

Register	Offset
SWREG111	1BCh

15.3.2.1.65.2 Diagram



15.3.2.1.65.3 Fields

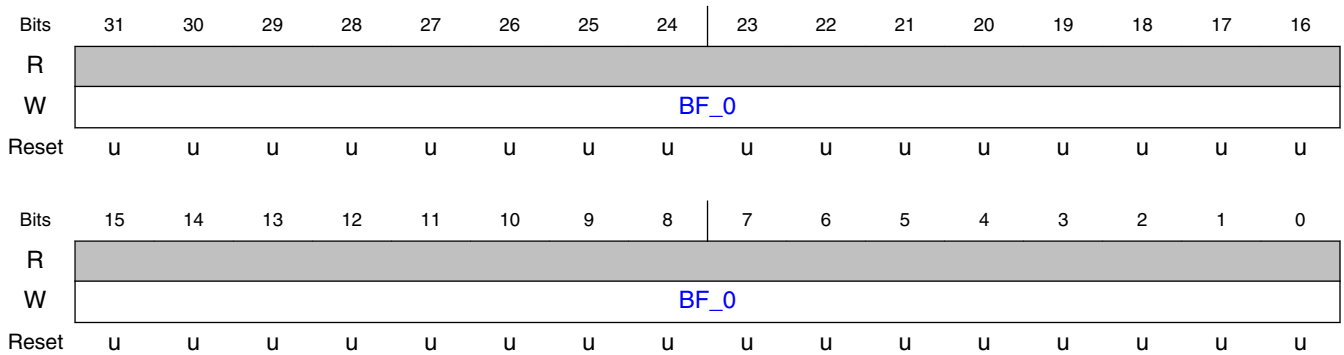
Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.66 VPU H1 Register 112 (SWREG112)

15.3.2.1.66.1 Offset

Register	Offset
SWREG112	1C0h

15.3.2.1.66.2 Diagram



15.3.2.1.66.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.67 VPU H1 Register 113 (SWREG113)

15.3.2.1.67.1 Offset

Register	Offset
SWREG113	1C4h

15.3.2.1.67.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.67.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.68 VPU H1 Register 114 (SWREG114)

15.3.2.1.68.1 Offset

Register	Offset
SWREG114	1C8h

15.3.2.1.68.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.68.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.69 VPU H1 Register 115 (SWREG115)

15.3.2.1.69.1 Offset

Register	Offset
SWREG115	1CCh

15.3.2.1.69.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.69.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.70 VPU H1 Register 116 (SWREG116)

15.3.2.1.70.1 Offset

Register	Offset
SWREG116	1D0h

15.3.2.1.70.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.70.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.71 VPU H1 Register 117 (SWREG117)

15.3.2.1.71.1 Offset

Register	Offset
SWREG117	1D4h

15.3.2.1.71.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.71.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.72 VPU H1 Register 118 (SWREG118)

15.3.2.1.72.1 Offset

Register	Offset
SWREG118	1D8h

15.3.2.1.72.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.72.3 Fields

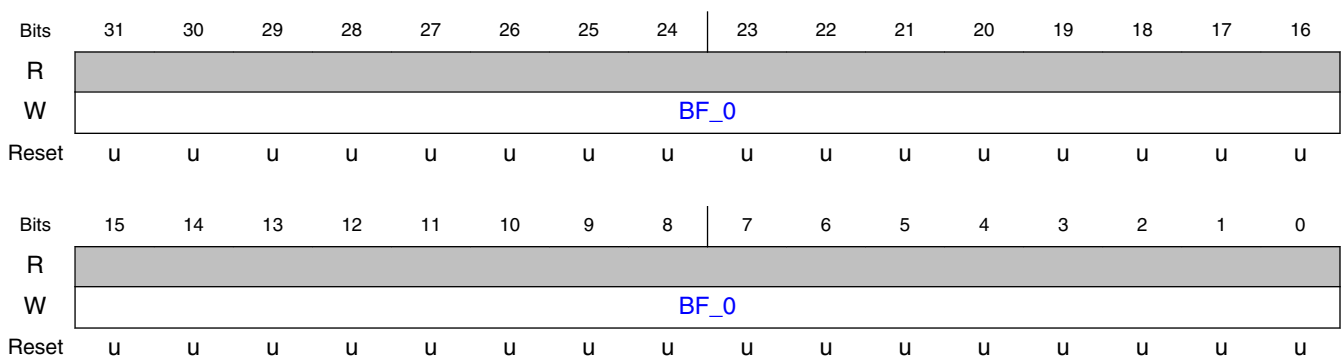
Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.73 VPU H1 Register 119 (SWREG119)

15.3.2.1.73.1 Offset

Register	Offset
SWREG119	1DCh

15.3.2.1.73.2 Diagram



15.3.2.1.73.3 Fields

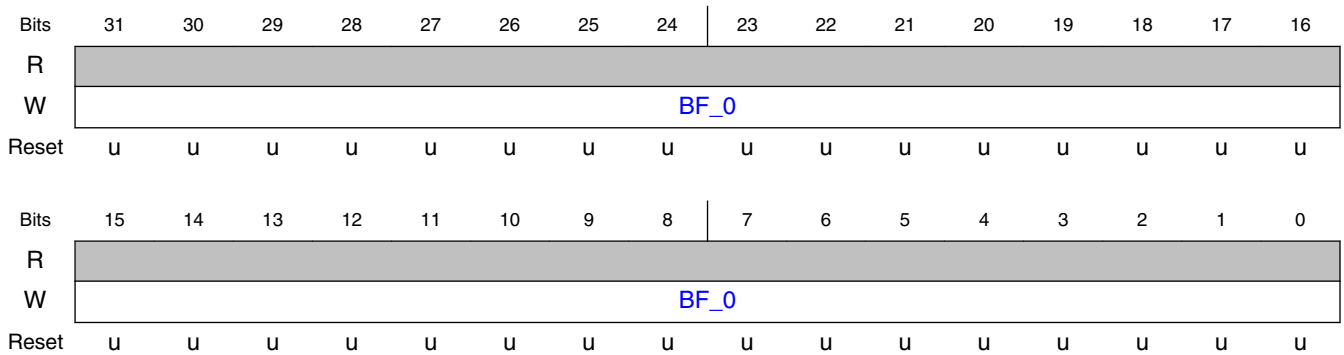
Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.74 VPU H1 Register 120 (SWREG120)

15.3.2.1.74.1 Offset

Register	Offset
SWREG120	1E0h

15.3.2.1.74.2 Diagram



15.3.2.1.74.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.75 VPU H1 Register 121 (SWREG121)

15.3.2.1.75.1 Offset

Register	Offset
SWREG121	1E4h

15.3.2.1.75.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.75.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.76 VPU H1 Register 122 (SWREG122)

15.3.2.1.76.1 Offset

Register	Offset
SWREG122	1E8h

15.3.2.1.76.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.76.3 Fields

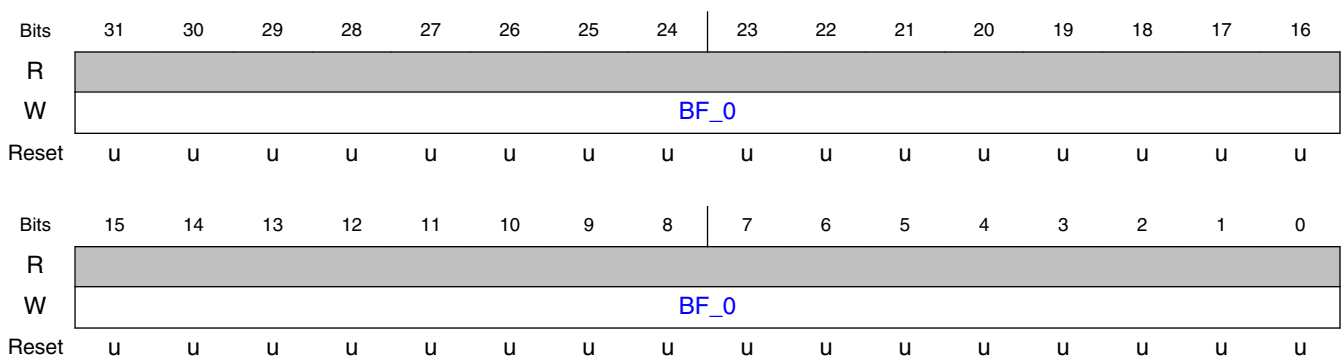
Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.77 VPU H1 Register 123 (SWREG123)

15.3.2.1.77.1 Offset

Register	Offset
SWREG123	1ECh

15.3.2.1.77.2 Diagram



15.3.2.1.77.3 Fields

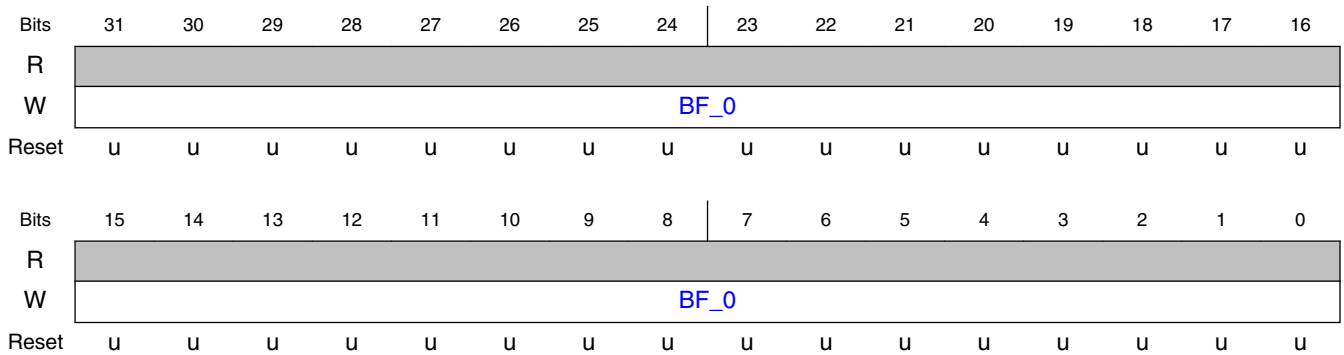
Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.78 VPU H1 Register 124 (SWREG124)

15.3.2.1.78.1 Offset

Register	Offset
SWREG124	1F0h

15.3.2.1.78.2 Diagram



15.3.2.1.78.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.79 VPU H1 Register 125 (SWREG125)

15.3.2.1.79.1 Offset

Register	Offset
SWREG125	1F4h

15.3.2.1.79.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.79.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.80 VPU H1 Register 126 (SWREG126)

15.3.2.1.80.1 Offset

Register	Offset
SWREG126	1F8h

15.3.2.1.80.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.80.3 Fields

Field	Function
31-0 BF_0	DMV 4p/1p penalty values

15.3.2.1.81 VPU H1 Register 127 (SWREG127)

15.3.2.1.81.1 Offset

Register	Offset
SWREG127	1FCh

15.3.2.1.81.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.81.3 Fields

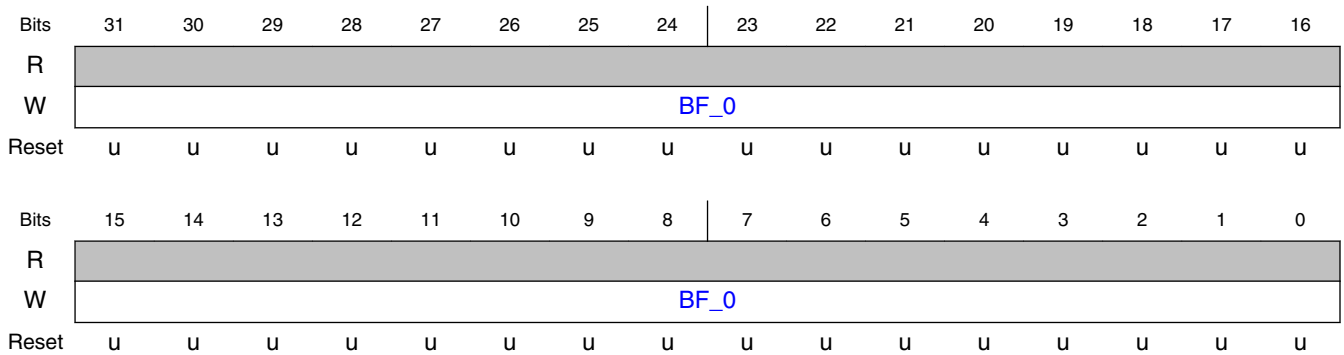
Field	Function
31-0 BF_0	DMV 4p/1p penalty values 124-127

15.3.2.1.82 VPU H1 Register 128 (SWREG128)

15.3.2.1.82.1 Offset

Register	Offset
SWREG128	200h

15.3.2.1.82.2 Diagram



15.3.2.1.82.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values 0-3

15.3.2.1.83 VPU H1 Register 129 (SWREG129)

15.3.2.1.83.1 Offset

Register	Offset
SWREG129	204h

15.3.2.1.83.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.83.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values 4-7

15.3.2.1.84 VPU H1 Register 130 (SWREG130)

15.3.2.1.84.1 Offset

Register	Offset
SWREG130	208h

15.3.2.1.84.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.84.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.85 VPU H1 Register 131 (SWREG131)

15.3.2.1.85.1 Offset

Register	Offset
SWREG131	20Ch

15.3.2.1.85.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.85.3 Fields

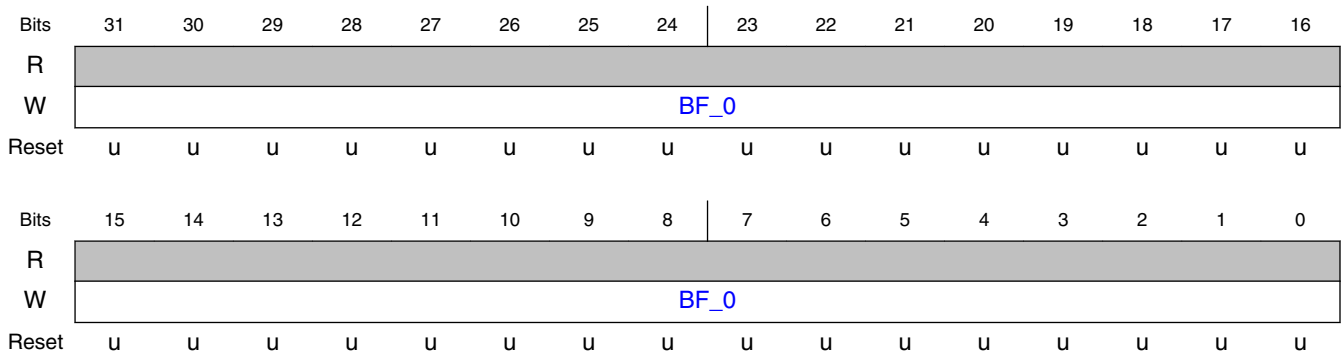
Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.86 VPU H1 Register 132 (SWREG132)

15.3.2.1.86.1 Offset

Register	Offset
SWREG132	210h

15.3.2.1.86.2 Diagram



15.3.2.1.86.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.87 VPU H1 Register 133 (SWREG133)

15.3.2.1.87.1 Offset

Register	Offset
SWREG133	214h

15.3.2.1.87.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.87.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.88 VPU H1 Register 134 (SWREG134)

15.3.2.1.88.1 Offset

Register	Offset
SWREG134	218h

15.3.2.1.88.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.88.3 Fields

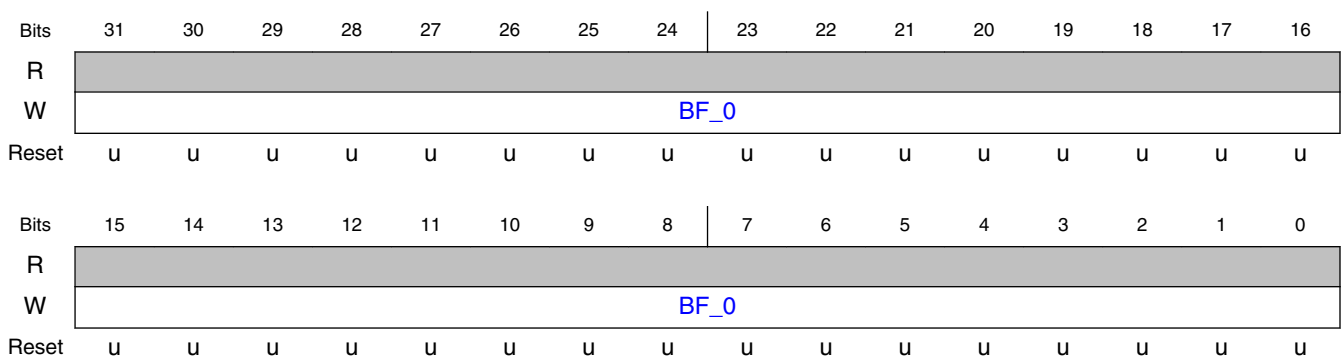
Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.89 VPU H1 Register 135 (SWREG135)

15.3.2.1.89.1 Offset

Register	Offset
SWREG135	21Ch

15.3.2.1.89.2 Diagram



15.3.2.1.89.3 Fields

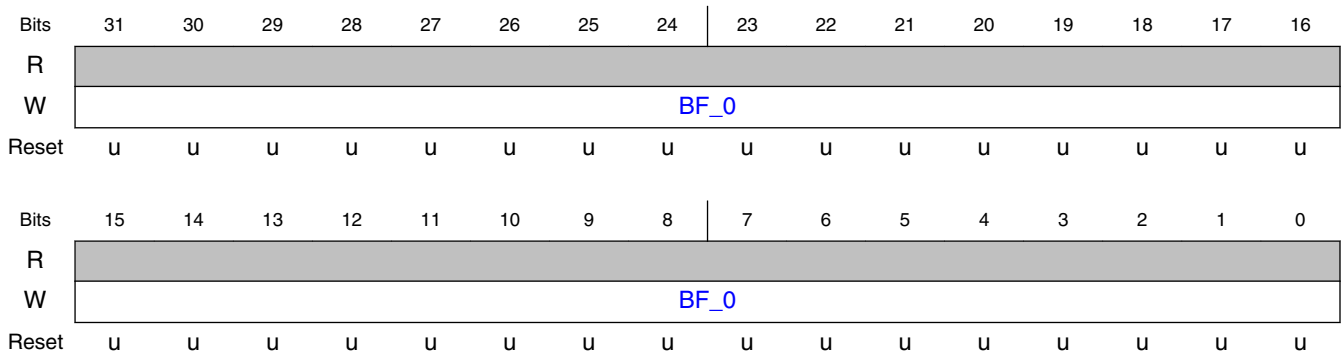
Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.90 VPU H1 Register 136 (SWREG136)

15.3.2.1.90.1 Offset

Register	Offset
SWREG136	220h

15.3.2.1.90.2 Diagram



15.3.2.1.90.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.91 VPU H1 Register 137 (SWREG137)

15.3.2.1.91.1 Offset

Register	Offset
SWREG137	224h

15.3.2.1.91.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.91.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.92 VPU H1 Register 138 (SWREG138)

15.3.2.1.92.1 Offset

Register	Offset
SWREG138	228h

15.3.2.1.92.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.92.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.93 VPU H1 Register 139 (SWREG139)

15.3.2.1.93.1 Offset

Register	Offset
SWREG139	22Ch

15.3.2.1.93.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.93.3 Fields

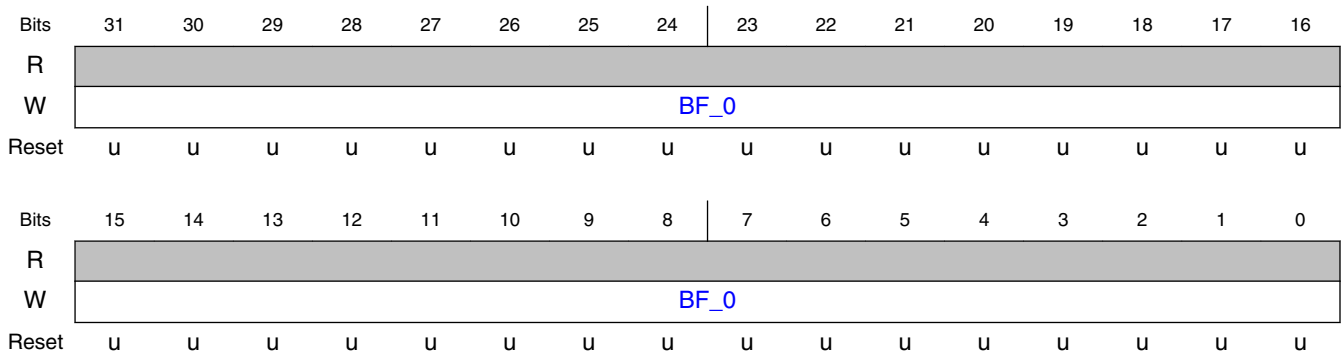
Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.94 VPU H1 Register 140 (SWREG140)

15.3.2.1.94.1 Offset

Register	Offset
SWREG140	230h

15.3.2.1.94.2 Diagram



15.3.2.1.94.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.95 VPU H1 Register 141 (SWREG141)

15.3.2.1.95.1 Offset

Register	Offset
SWREG141	234h

15.3.2.1.95.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.95.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.96 VPU H1 Register 142 (SWREG142)

15.3.2.1.96.1 Offset

Register	Offset
SWREG142	238h

15.3.2.1.96.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.96.3 Fields

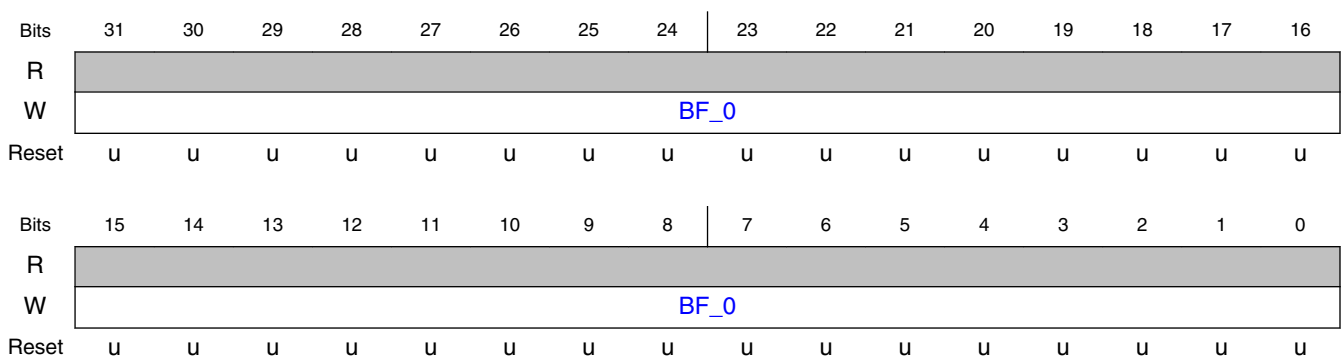
Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.97 VPU H1 Register 143 (SWREG143)

15.3.2.1.97.1 Offset

Register	Offset
SWREG143	23Ch

15.3.2.1.97.2 Diagram



15.3.2.1.97.3 Fields

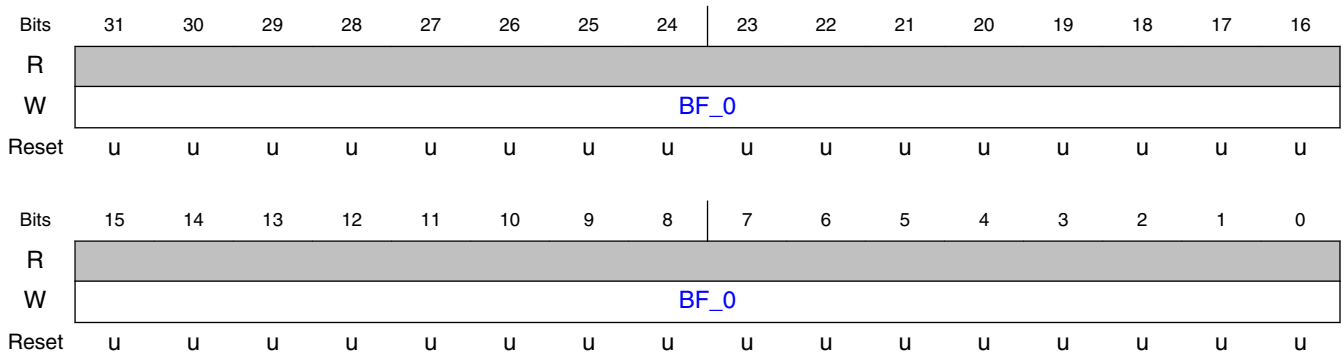
Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.98 VPU H1 Register 144 (SWREG144)

15.3.2.1.98.1 Offset

Register	Offset
SWREG144	240h

15.3.2.1.98.2 Diagram



15.3.2.1.98.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.99 VPU H1 Register 145 (SWREG145)

15.3.2.1.99.1 Offset

Register	Offset
SWREG145	244h

15.3.2.1.99.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.99.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.100 VPU H1 Register 146 (SWREG146)

15.3.2.1.100.1 Offset

Register	Offset
SWREG146	248h

15.3.2.1.100.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.100.3 Fields

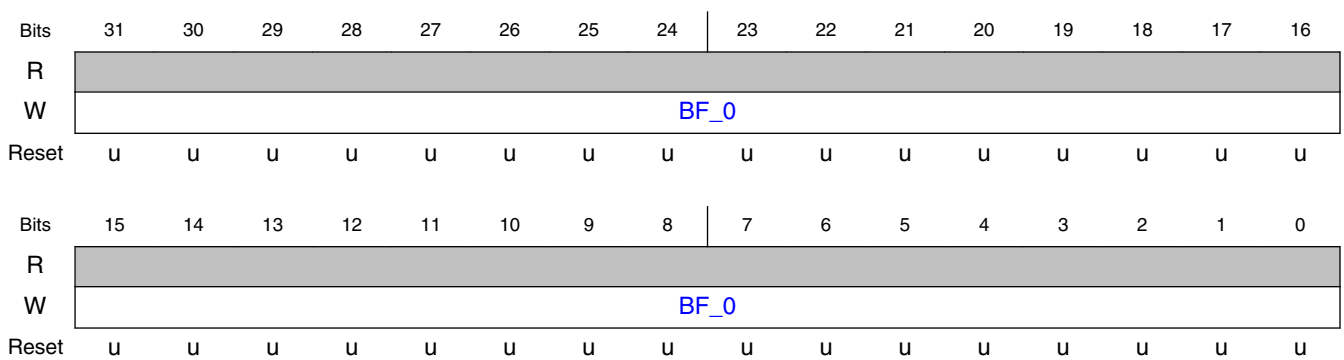
Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.101 VPU H1 Register 147 (SWREG147)

15.3.2.1.101.1 Offset

Register	Offset
SWREG147	24Ch

15.3.2.1.101.2 Diagram



15.3.2.1.101.3 Fields

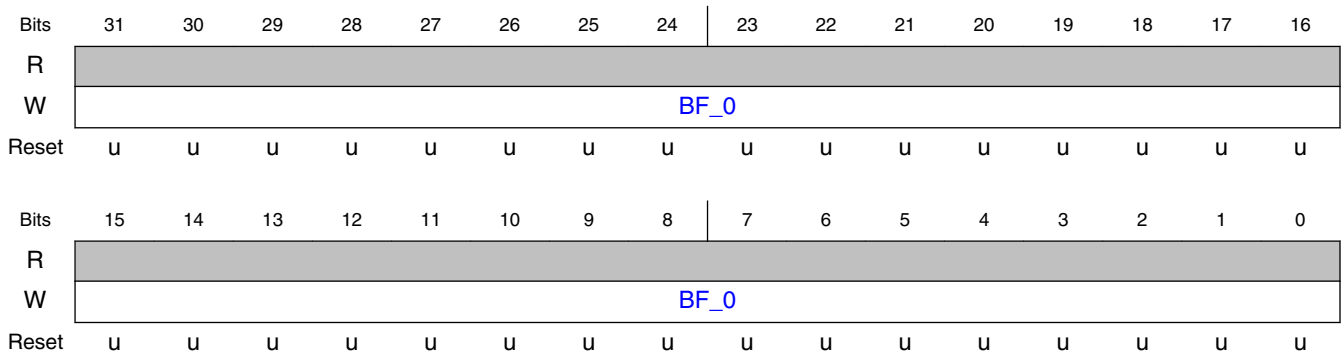
Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.102 VPU H1 Register 148 (SWREG148)

15.3.2.1.102.1 Offset

Register	Offset
SWREG148	250h

15.3.2.1.102.2 Diagram



15.3.2.1.102.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.103 VPU H1 Register 149 (SWREG149)

15.3.2.1.103.1 Offset

Register	Offset
SWREG149	254h

15.3.2.1.103.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.103.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.104 VPU H1 Register 150 (SWREG150)

15.3.2.1.104.1 Offset

Register	Offset
SWREG150	258h

15.3.2.1.104.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.104.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.105 VPU H1 Register 151 (SWREG151)

15.3.2.1.105.1 Offset

Register	Offset
SWREG151	25Ch

15.3.2.1.105.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.105.3 Fields

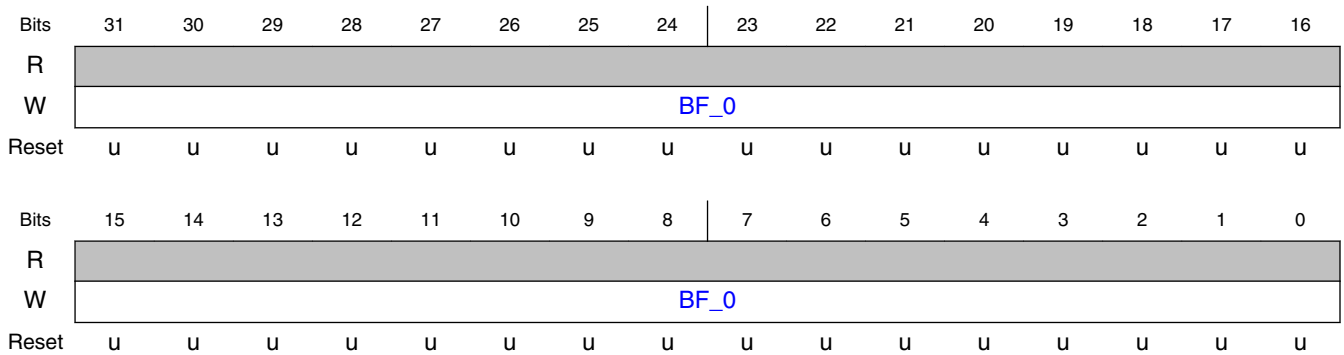
Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.106 VPU H1 Register 152 (SWREG152)

15.3.2.1.106.1 Offset

Register	Offset
SWREG152	260h

15.3.2.1.106.2 Diagram



15.3.2.1.106.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.107 VPU H1 Register 153 (SWREG153)

15.3.2.1.107.1 Offset

Register	Offset
SWREG153	264h

15.3.2.1.107.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.107.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.108 VPU H1 Register 154 (SWREG154)

15.3.2.1.108.1 Offset

Register	Offset
SWREG154	268h

15.3.2.1.108.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.108.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.109 VPU H1 Register 155 (SWREG155)

15.3.2.1.109.1 Offset

Register	Offset
SWREG155	26Ch

15.3.2.1.109.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.109.3 Fields

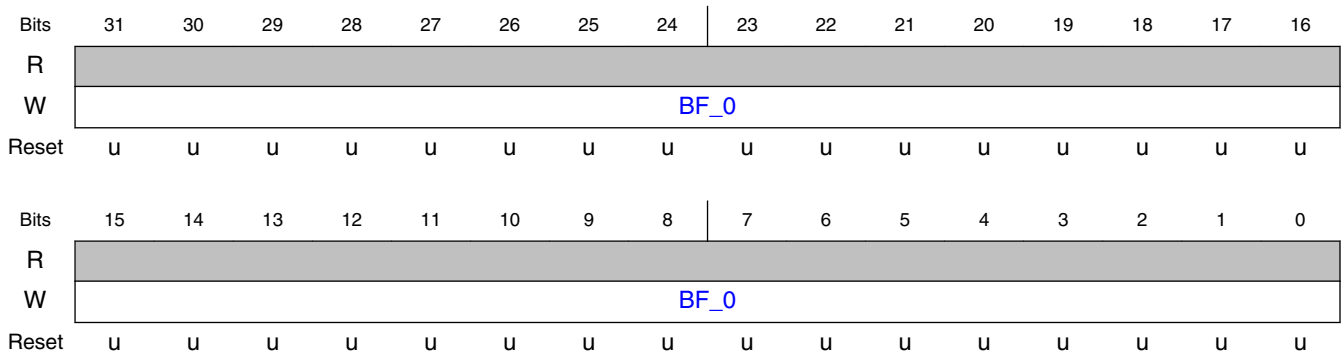
Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.110 VPU H1 Register 156 (SWREG156)

15.3.2.1.110.1 Offset

Register	Offset
SWREG156	270h

15.3.2.1.110.2 Diagram



15.3.2.1.110.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.111 VPU H1 Register 157 (SWREG157)

15.3.2.1.111.1 Offset

Register	Offset
SWREG157	274h

15.3.2.1.111.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.111.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.112 VPU H1 Register 158 (SWREG158)

15.3.2.1.112.1 Offset

Register	Offset
SWREG158	278h

15.3.2.1.112.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.112.3 Fields

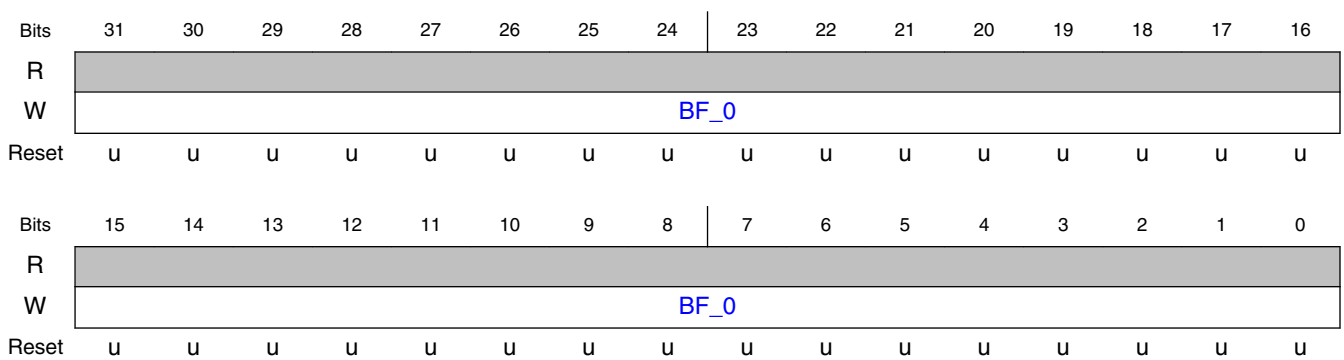
Field	Function
31-0 BF_0	DMV qpel penalty values

15.3.2.1.113 VPU H1 Register 159 (SWREG159)

15.3.2.1.113.1 Offset

Register	Offset
SWREG159	27Ch

15.3.2.1.113.2 Diagram



15.3.2.1.113.3 Fields

Field	Function
31-0 BF_0	DMV qpel penalty values 124-127

15.3.2.1.114 VPU H1 Register 160 (SWREG160)

15.3.2.1.114.1 Offset

Register	Offset
SWREG160	280h

15.3.2.1.114.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_12							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_12				BF_0											
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.114.3 Fields

Field	Function
31-24 BF_24	Penalty for using zero-MV in 16x8/8x16/8x8 split
23-12 BF_12	VP8 coeff for dmv penalty for intra/inter selection
11-0 BF_0	VP8 bit cost of inter type

15.3.2.1.115 VPU H1 Register 161 (SWREG161)

15.3.2.1.115.1 Offset

Register	Offset
SWREG161	284h

15.3.2.1.115.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				BF_0											
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.115.3 Fields

Field	Function
31-12 —	Reserved.
11-0 BF_0	VP8 bit cost of golden ref frame (not used)

15.3.2.1.116 VPU H1 Register 162 (SWREG162)

15.3.2.1.116.1 Offset

Register	Offset
SWREG162	288h

15.3.2.1.116.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				BF_21							BF_14				
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_14		BF_7						BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.116.3 Fields

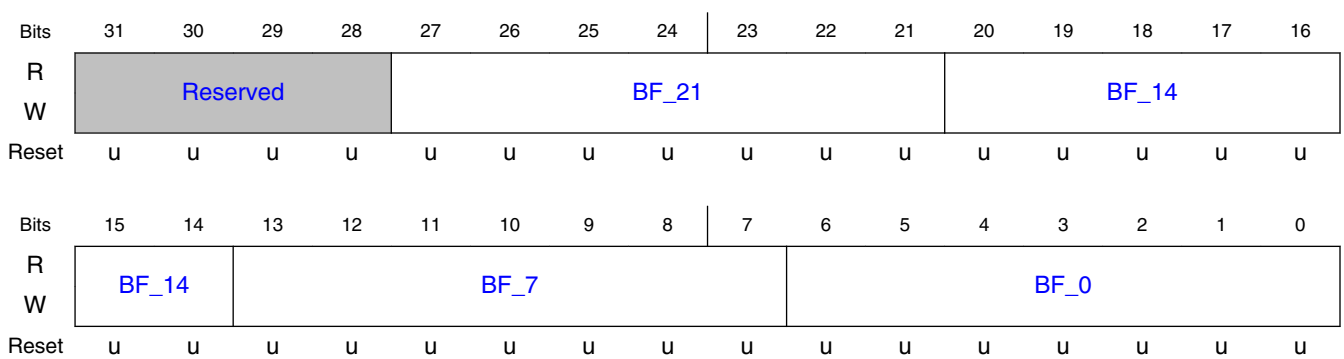
Field	Function
31-28 —	Reserved.
27-21 BF_21	VP8 loop filter delta for alt ref
20-14 BF_14	VP8 loop filter delta for golden ref
13-7 BF_7	VP8 loop filter delta for last ref
6-0 BF_0	VP8 loop filter delta for intra mb

15.3.2.1.117 VPU H1 Register 163 (SWREG163)

15.3.2.1.117.1 Offset

Register	Offset
SWREG163	28Ch

15.3.2.1.117.2 Diagram



15.3.2.1.117.3 Fields

Field	Function
31-28	Reserved.

Table continues on the next page...

Field	Function
—	
27-21 BF_21	VP8 loop filter delta for SPLITMV
20-14 BF_14	VP8 loop filter delta for NEWMV
13-7 BF_7	VP8 loop filter delta for ZEROMV
6-0 BF_0	VP8 loop filter delta for BPRED

15.3.2.1.118 VPU H1 Register 164 (SWREG164)

15.3.2.1.118.1 Offset

Register	Offset
SWREG164	290h

15.3.2.1.118.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.118.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.119 VPU H1 Register 165 (SWREG165)

15.3.2.1.119.1 Offset

Register	Offset
SWREG165	294h

15.3.2.1.119.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.119.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.120 VPU H1 Register 166 (SWREG166)

15.3.2.1.120.1 Offset

Register	Offset
SWREG166	298h

15.3.2.1.120.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.120.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.121 VPU H1 Register 167 (SWREG167)

15.3.2.1.121.1 Offset

Register	Offset
SWREG167	29Ch

15.3.2.1.121.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.121.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.122 VPU H1 Register 168 (SWREG168)

15.3.2.1.122.1 Offset

Register	Offset
SWREG168	2A0h

15.3.2.1.122.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.122.3 Fields

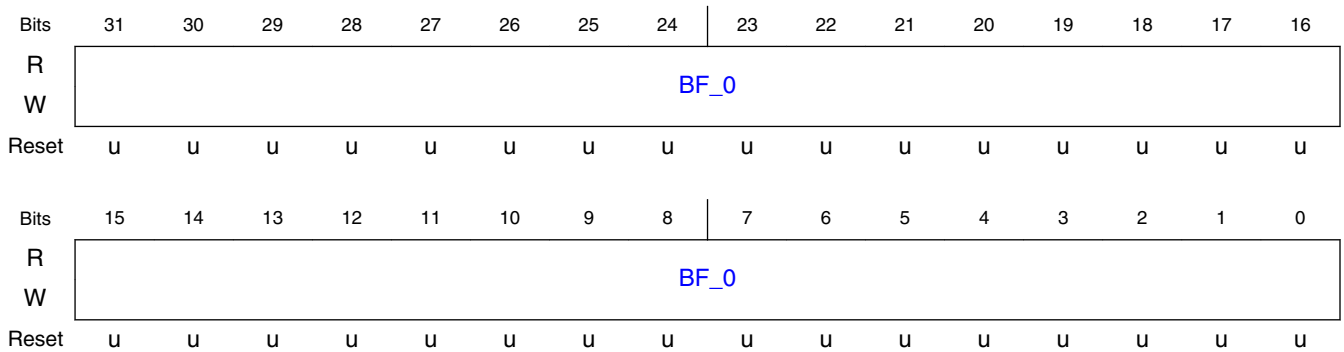
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.123 VPU H1 Register 169 (SWREG169)

15.3.2.1.123.1 Offset

Register	Offset
SWREG169	2A4h

15.3.2.1.123.2 Diagram



15.3.2.1.123.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.124 VPU H1 Register 170 (SWREG170)

15.3.2.1.124.1 Offset

Register	Offset
SWREG170	2A8h

15.3.2.1.124.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.124.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.125 VPU H1 Register 171 (SWREG171)

15.3.2.1.125.1 Offset

Register	Offset
SWREG171	2ACh

15.3.2.1.125.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.125.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.126 VPU H1 Register 172 (SWREG172)

15.3.2.1.126.1 Offset

Register	Offset
SWREG172	2B0h

15.3.2.1.126.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.126.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.127 VPU H1 Register 173 (SWREG173)

15.3.2.1.127.1 Offset

Register	Offset
SWREG173	2B4h

15.3.2.1.127.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.127.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.128 VPU H1 Register 174 (SWREG174)

15.3.2.1.128.1 Offset

Register	Offset
SWREG174	2B8h

15.3.2.1.128.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.128.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.129 VPU H1 Register 175 (SWREG175)

15.3.2.1.129.1 Offset

Register	Offset
SWREG175	2BCh

15.3.2.1.129.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.129.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.130 VPU H1 Register 176 (SWREG176)

15.3.2.1.130.1 Offset

Register	Offset
SWREG176	2C0h

15.3.2.1.130.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.130.3 Fields

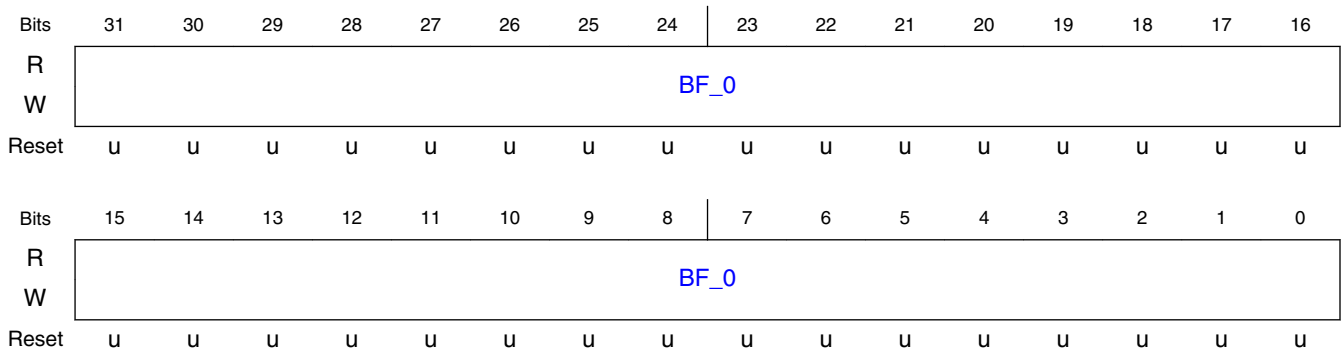
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.131 VPU H1 Register 177 (SWREG177)

15.3.2.1.131.1 Offset

Register	Offset
SWREG177	2C4h

15.3.2.1.131.2 Diagram



15.3.2.1.131.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.132 VPU H1 Register 178 (SWREG178)

15.3.2.1.132.1 Offset

Register	Offset
SWREG178	2C8h

15.3.2.1.132.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.132.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.133 VPU H1 Register 179 (SWREG179)

15.3.2.1.133.1 Offset

Register	Offset
SWREG179	2CCh

15.3.2.1.133.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.133.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.134 VPU H1 Register 180 (SWREG180)

15.3.2.1.134.1 Offset

Register	Offset
SWREG180	2D0h

15.3.2.1.134.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.134.3 Fields

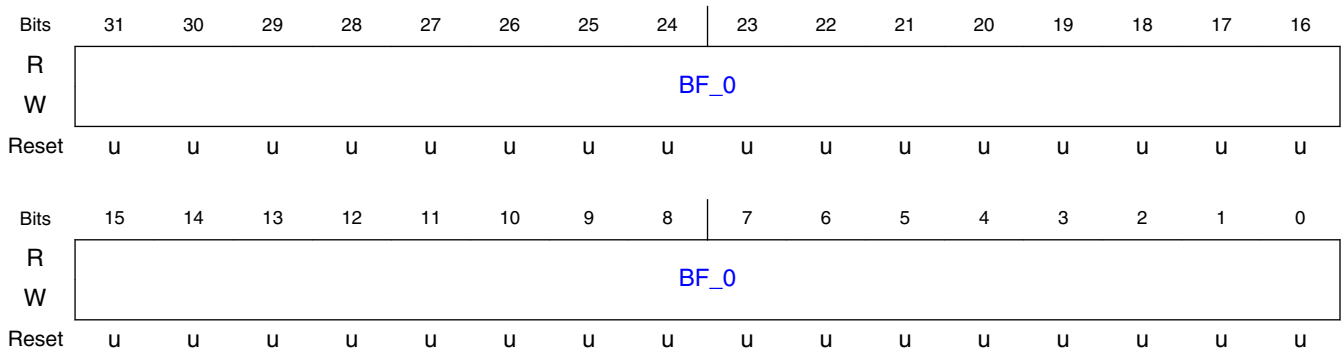
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.135 VPU H1 Register 181 (SWREG181)

15.3.2.1.135.1 Offset

Register	Offset
SWREG181	2D4h

15.3.2.1.135.2 Diagram



15.3.2.1.135.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.136 VPU H1 Register 182 (SWREG182)

15.3.2.1.136.1 Offset

Register	Offset
SWREG182	2D8h

15.3.2.1.136.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.136.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.137 VPU H1 Register 183 (SWREG183)

15.3.2.1.137.1 Offset

Register	Offset
SWREG183	2DCh

15.3.2.1.137.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.137.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.138 VPU H1 Register 184 (SWREG184)

15.3.2.1.138.1 Offset

Register	Offset
SWREG184	2E0h

15.3.2.1.138.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.138.3 Fields

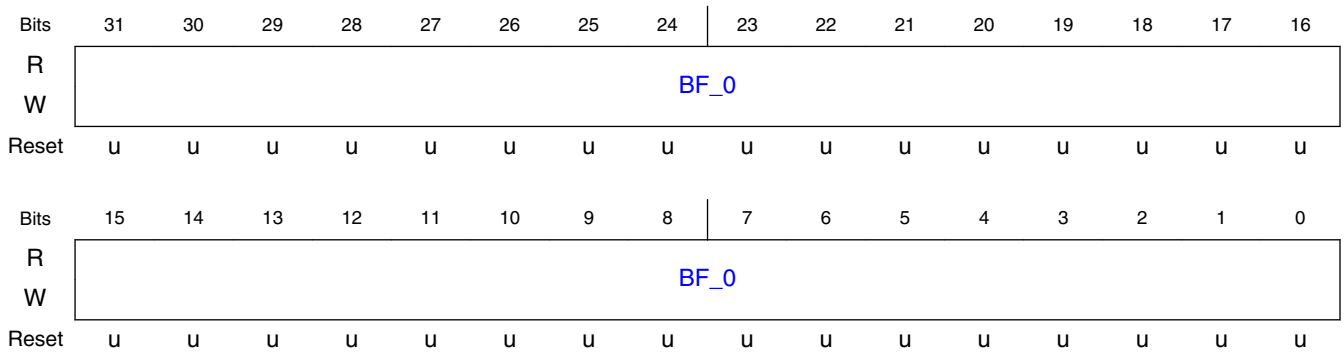
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.139 VPU H1 Register 185 (SWREG185)

15.3.2.1.139.1 Offset

Register	Offset
SWREG185	2E4h

15.3.2.1.139.2 Diagram



15.3.2.1.139.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.140 VPU H1 Register 186 (SWREG186)

15.3.2.1.140.1 Offset

Register	Offset
SWREG186	2E8h

15.3.2.1.140.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.140.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.141 VPU H1 Register 187 (SWREG187)

15.3.2.1.141.1 Offset

Register	Offset
SWREG187	2ECh

15.3.2.1.141.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.141.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.142 VPU H1 Register 188 (SWREG188)

15.3.2.1.142.1 Offset

Register	Offset
SWREG188	2F0h

15.3.2.1.142.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.142.3 Fields

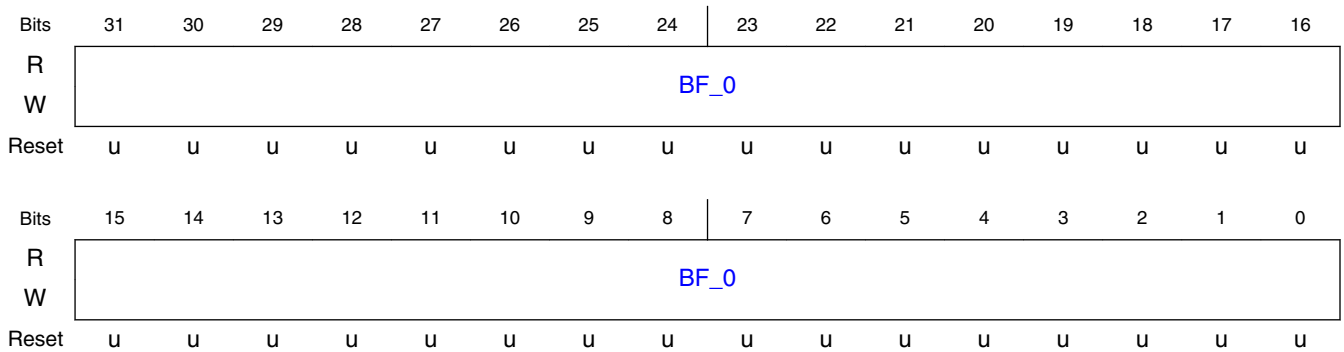
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.143 VPU H1 Register 189 (SWREG189)

15.3.2.1.143.1 Offset

Register	Offset
SWREG189	2F4h

15.3.2.1.143.2 Diagram



15.3.2.1.143.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.144 VPU H1 Register 190 (SWREG190)

15.3.2.1.144.1 Offset

Register	Offset
SWREG190	2F8h

15.3.2.1.144.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.144.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.145 VPU H1 Register 191 (SWREG191)

15.3.2.1.145.1 Offset

Register	Offset
SWREG191	2FCh

15.3.2.1.145.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.145.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.146 VPU H1 Register 192 (SWREG192)

15.3.2.1.146.1 Offset

Register	Offset
SWREG192	300h

15.3.2.1.146.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.146.3 Fields

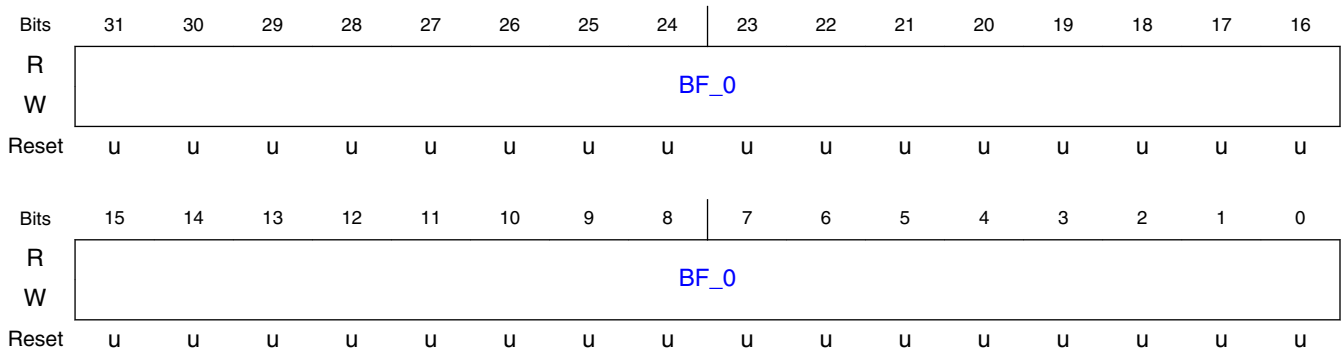
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.147 VPU H1 Register 193 (SWREG193)

15.3.2.1.147.1 Offset

Register	Offset
SWREG193	304h

15.3.2.1.147.2 Diagram



15.3.2.1.147.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.148 VPU H1 Register 194 (SWREG194)

15.3.2.1.148.1 Offset

Register	Offset
SWREG194	308h

15.3.2.1.148.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.148.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.149 VPU H1 Register 195 (SWREG195)

15.3.2.1.149.1 Offset

Register	Offset
SWREG195	30Ch

15.3.2.1.149.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.149.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.150 VPU H1 Register 196 (SWREG196)

15.3.2.1.150.1 Offset

Register	Offset
SWREG196	310h

15.3.2.1.150.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.150.3 Fields

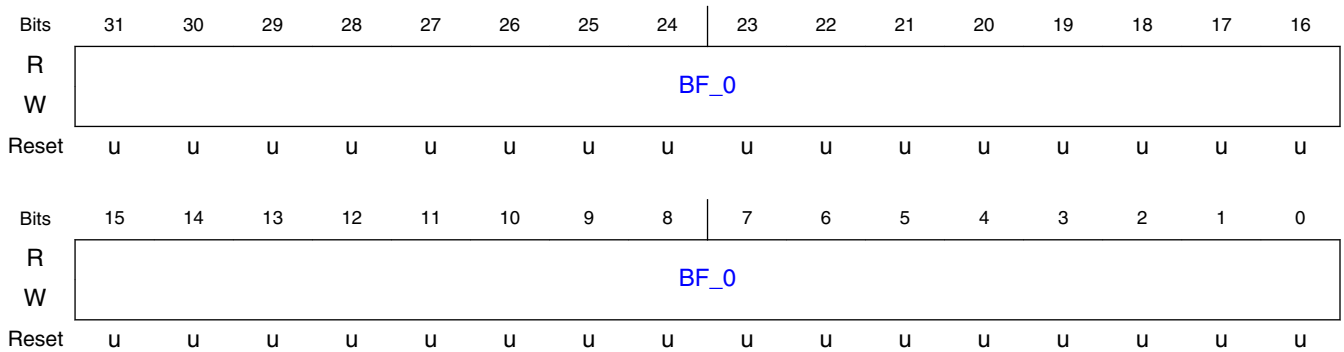
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.151 VPU H1 Register 197 (SWREG197)

15.3.2.1.151.1 Offset

Register	Offset
SWREG197	314h

15.3.2.1.151.2 Diagram



15.3.2.1.151.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.152 VPU H1 Register 198 (SWREG198)

15.3.2.1.152.1 Offset

Register	Offset
SWREG198	318h

15.3.2.1.152.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.152.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.153 VPU H1 Register 199 (SWREG199)

15.3.2.1.153.1 Offset

Register	Offset
SWREG199	31Ch

15.3.2.1.153.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.153.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.154 VPU H1 Register 200 (SWREG200)

15.3.2.1.154.1 Offset

Register	Offset
SWREG200	320h

15.3.2.1.154.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.154.3 Fields

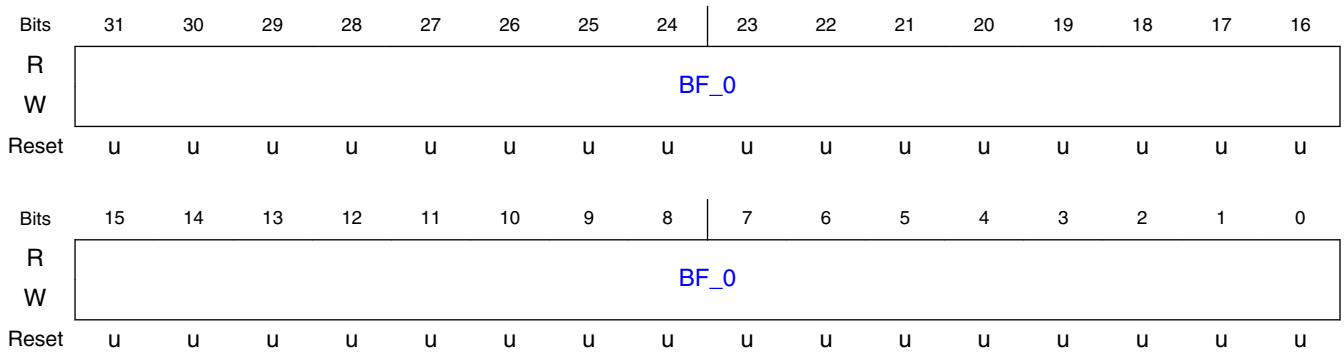
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.155 VPU H1 Register 201 (SWREG201)

15.3.2.1.155.1 Offset

Register	Offset
SWREG201	324h

15.3.2.1.155.2 Diagram



15.3.2.1.155.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.156 VPU H1 Register 202 (SWREG202)

15.3.2.1.156.1 Offset

Register	Offset
SWREG202	328h

15.3.2.1.156.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.156.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.157 VPU H1 Register 203 (SWREG203)

15.3.2.1.157.1 Offset

Register	Offset
SWREG203	32Ch

15.3.2.1.157.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.157.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.158 VPU H1 Register 204 (SWREG204)

15.3.2.1.158.1 Offset

Register	Offset
SWREG204	330h

15.3.2.1.158.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.158.3 Fields

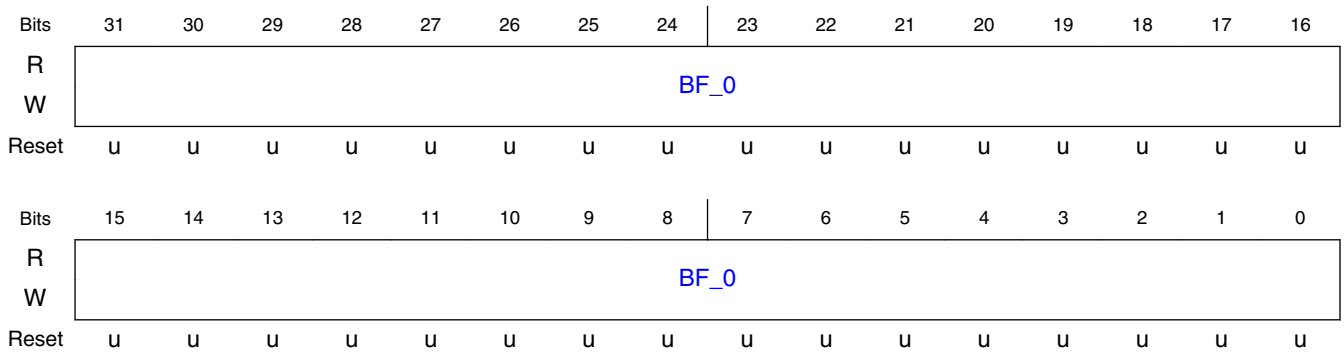
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.159 VPU H1 Register 205 (SWREG205)

15.3.2.1.159.1 Offset

Register	Offset
SWREG205	334h

15.3.2.1.159.2 Diagram



15.3.2.1.159.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.160 VPU H1 Register 206 (SWREG206)

15.3.2.1.160.1 Offset

Register	Offset
SWREG206	338h

15.3.2.1.160.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.160.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.161 VPU H1 Register 207 (SWREG207)

15.3.2.1.161.1 Offset

Register	Offset
SWREG207	33Ch

15.3.2.1.161.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.161.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.162 VPU H1 Register 208 (SWREG208)

15.3.2.1.162.1 Offset

Register	Offset
SWREG208	340h

15.3.2.1.162.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.162.3 Fields

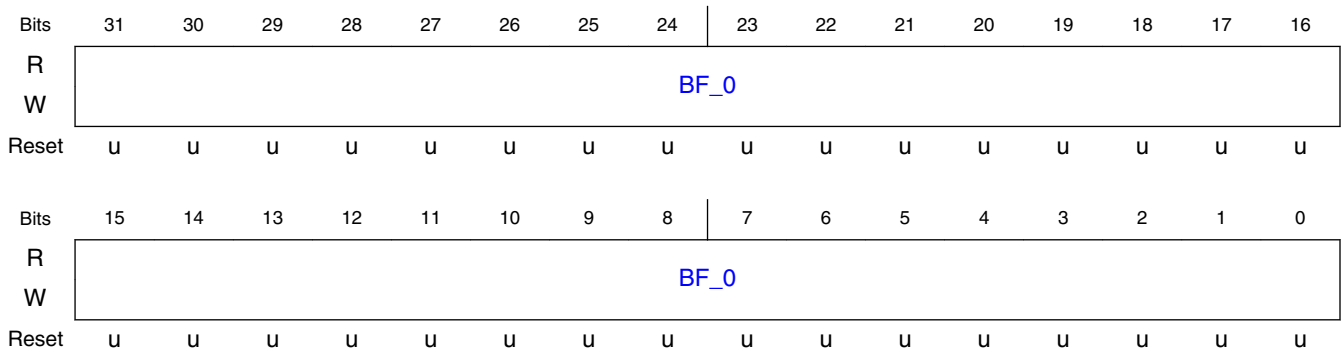
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.163 VPU H1 Register 209 (SWREG209)

15.3.2.1.163.1 Offset

Register	Offset
SWREG209	344h

15.3.2.1.163.2 Diagram



15.3.2.1.163.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.164 VPU H1 Register 210 (SWREG210)

15.3.2.1.164.1 Offset

Register	Offset
SWREG210	348h

15.3.2.1.164.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.164.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.165 VPU H1 Register 211 (SWREG211)

15.3.2.1.165.1 Offset

Register	Offset
SWREG211	34Ch

15.3.2.1.165.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.165.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.166 VPU H1 Register 212 (SWREG212)

15.3.2.1.166.1 Offset

Register	Offset
SWREG212	350h

15.3.2.1.166.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.166.3 Fields

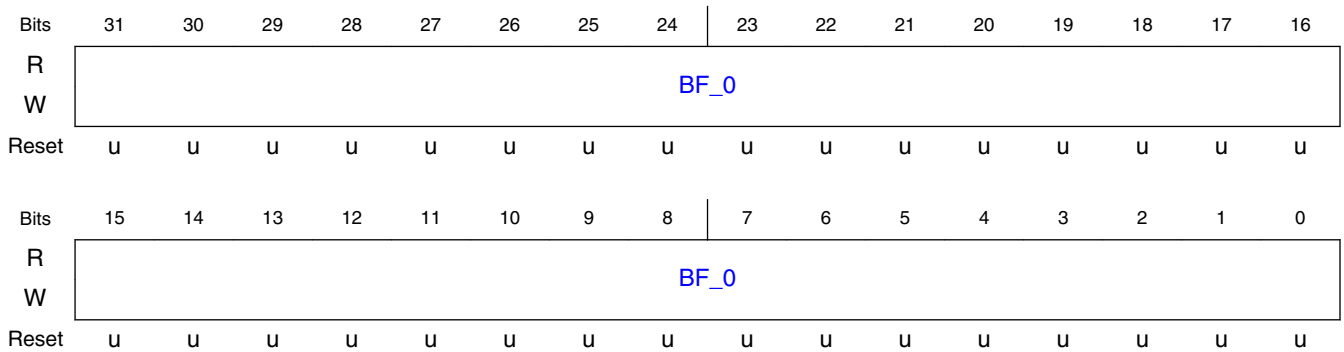
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.167 VPU H1 Register 213 (SWREG213)

15.3.2.1.167.1 Offset

Register	Offset
SWREG213	354h

15.3.2.1.167.2 Diagram



15.3.2.1.167.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.168 VPU H1 Register 214 (SWREG214)

15.3.2.1.168.1 Offset

Register	Offset
SWREG214	358h

15.3.2.1.168.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.168.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.169 VPU H1 Register 215 (SWREG215)

15.3.2.1.169.1 Offset

Register	Offset
SWREG215	35Ch

15.3.2.1.169.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.169.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.170 VPU H1 Register 216 (SWREG216)

15.3.2.1.170.1 Offset

Register	Offset
SWREG216	360h

15.3.2.1.170.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.170.3 Fields

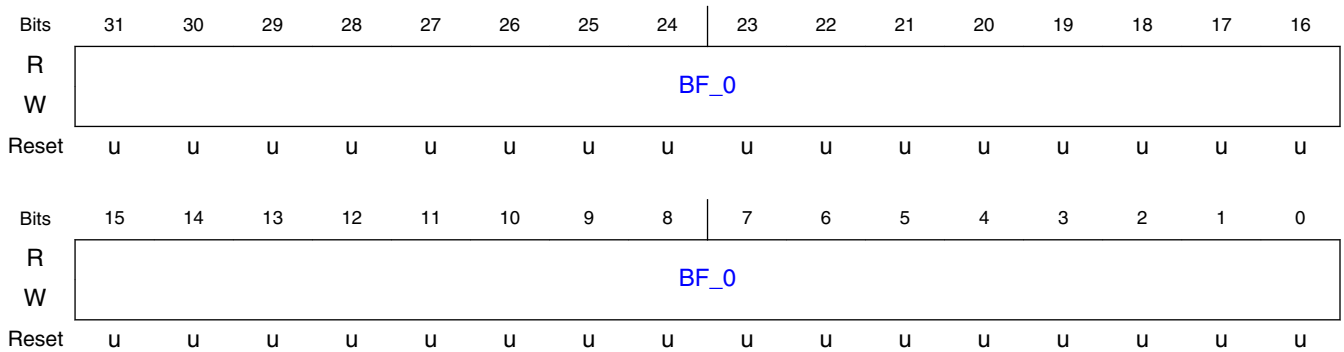
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.171 VPU H1 Register 217 (SWREG217)

15.3.2.1.171.1 Offset

Register	Offset
SWREG217	364h

15.3.2.1.171.2 Diagram



15.3.2.1.171.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.172 VPU H1 Register 218 (SWREG218)

15.3.2.1.172.1 Offset

Register	Offset
SWREG218	368h

15.3.2.1.172.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.172.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.173 VPU H1 Register 219 (SWREG219)

15.3.2.1.173.1 Offset

Register	Offset
SWREG219	36Ch

15.3.2.1.173.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.173.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.174 VPU H1 Register 220 (SWREG220)

15.3.2.1.174.1 Offset

Register	Offset
SWREG220	370h

15.3.2.1.174.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.174.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.175 VPU H1 Register 221 (SWREG221)

15.3.2.1.175.1 Offset

Register	Offset
SWREG221	374h

15.3.2.1.175.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.175.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.176 VPU H1 Register 222 (SWREG222)

15.3.2.1.176.1 Offset

Register	Offset
SWREG222	378h

15.3.2.1.176.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.176.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.177 VPU H1 Register 223 (SWREG223)

15.3.2.1.177.1 Offset

Register	Offset
SWREG223	37Ch

15.3.2.1.177.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.177.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.178 VPU H1 Register 224 (SWREG224)

15.3.2.1.178.1 Offset

Register	Offset
SWREG224	380h

15.3.2.1.178.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.178.3 Fields

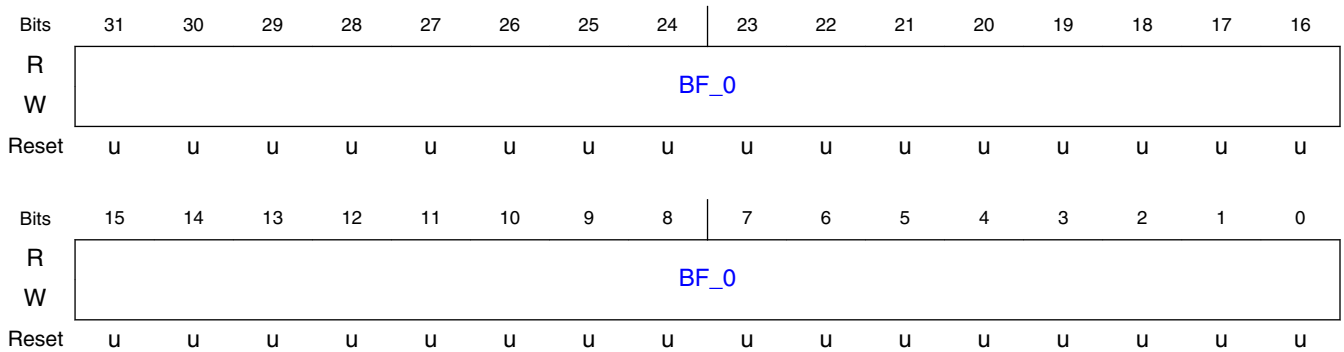
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.179 VPU H1 Register 225 (SWREG225)

15.3.2.1.179.1 Offset

Register	Offset
SWREG225	384h

15.3.2.1.179.2 Diagram



15.3.2.1.179.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.180 VPU H1 Register 226 (SWREG226)

15.3.2.1.180.1 Offset

Register	Offset
SWREG226	388h

15.3.2.1.180.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.180.3 Fields

Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.181 VPU H1 Register 227 (SWREG227)

15.3.2.1.181.1 Offset

Register	Offset
SWREG227	38Ch

15.3.2.1.181.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.181.3 Fields

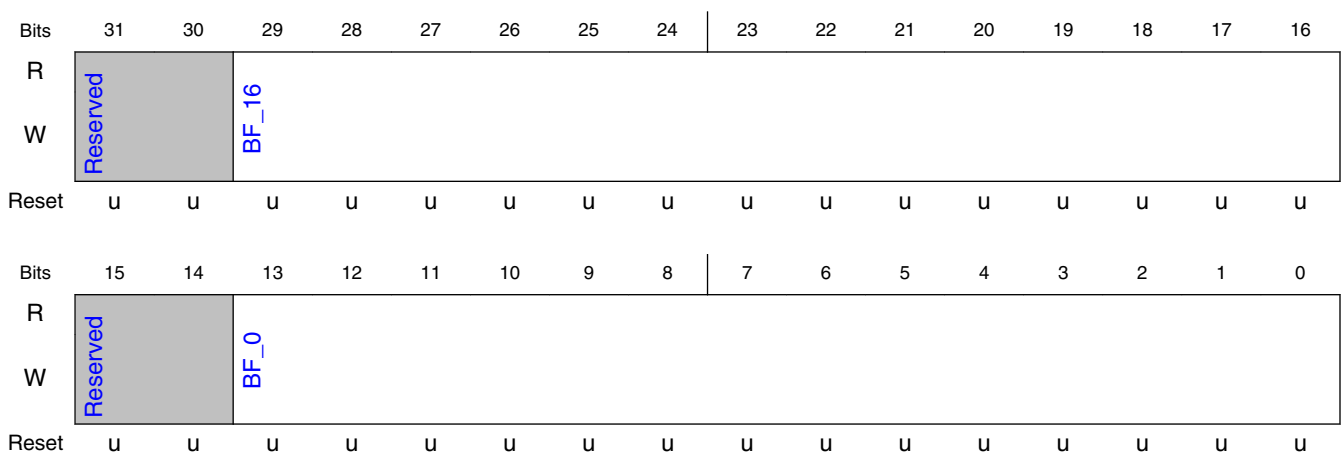
Field	Function
31-0 BF_0	VP8 deadzone lookup table

15.3.2.1.182 VPU H1 Register 228 (SWREG228)

15.3.2.1.182.1 Offset

Register	Offset
SWREG228	390h

15.3.2.1.182.2 Diagram



15.3.2.1.182.3 Fields

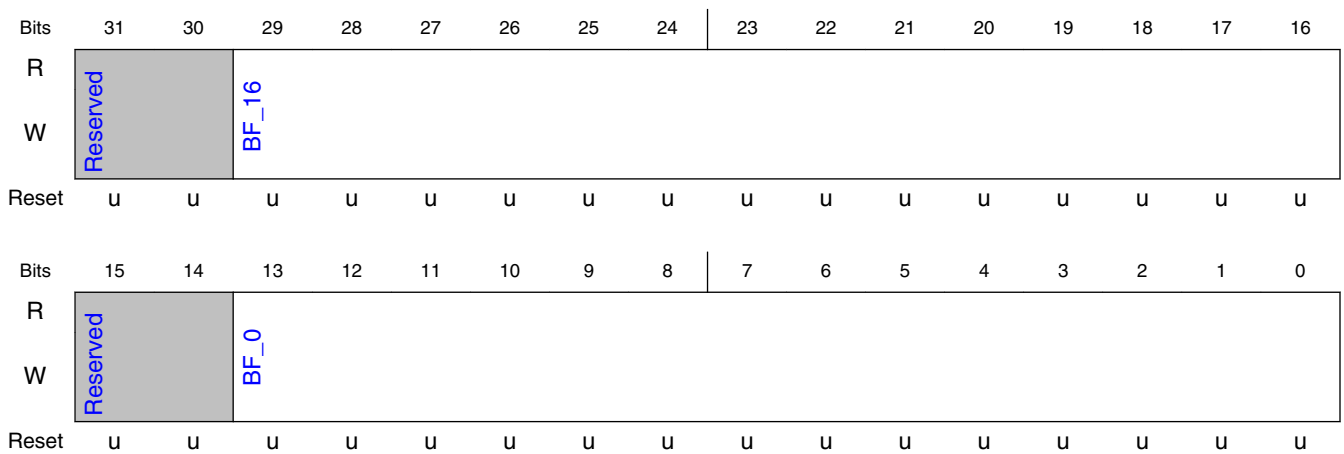
Field	Function
31-30 —	Reserved.
29-16 BF_16	VP8 deadzone rate multiplier for plane 0
15-14 —	Reserved.
13-0 BF_0	VP8 deadzone rate multiplier for plane 1

15.3.2.1.183 VPU H1 Register 229 (SWREG229)

15.3.2.1.183.1 Offset

Register	Offset
SWREG229	394h

15.3.2.1.183.2 Diagram



15.3.2.1.183.3 Fields

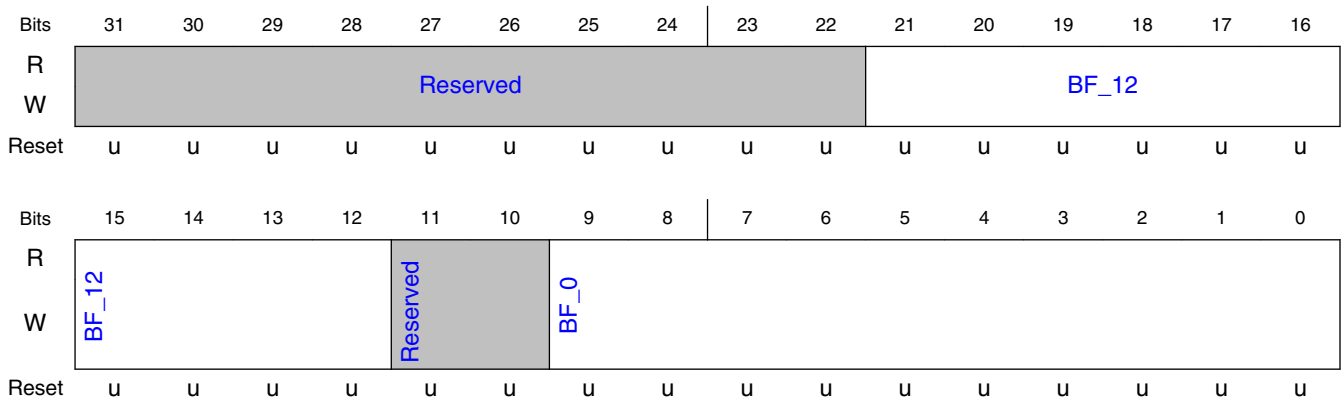
Field	Function
31-30 —	Reserved.
29-16 BF_16	VP8 deadzone rate multiplier for plane 2
15-14 —	Reserved.
13-0 BF_0	VP8 deadzone rate multiplier for plane 3

15.3.2.1.184 VPU H1 Register 230 (SWREG230)

15.3.2.1.184.1 Offset

Register	Offset
SWREG230	398h

15.3.2.1.184.2 Diagram



15.3.2.1.184.3 Fields

Field	Function
31-22 —	Reserved.
21-12 BF_12	VP8 deadzone rate for macroblock skip token 0
11-10 —	Reserved.
9-0 BF_0	VP8 deadzone rate for macroblock skip token 1

15.3.2.1.185 VPU H1 Register 231 (SWREG231)

15.3.2.1.185.1 Offset

Register	Offset
SWREG231	39Ch

15.3.2.1.185.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.185.3 Fields

Field	Function
31-0 BF_0	Base address for output of down-scaled encoder image in YUYV 4:2:2 format

15.3.2.1.186 VPU H1 Register 232 (SWREG232)

15.3.2.1.186.1 Offset

Register	Offset
SWREG232	3A0h

15.3.2.1.186.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_30		Reserved	BF_17												BF_0
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.186.3 Fields

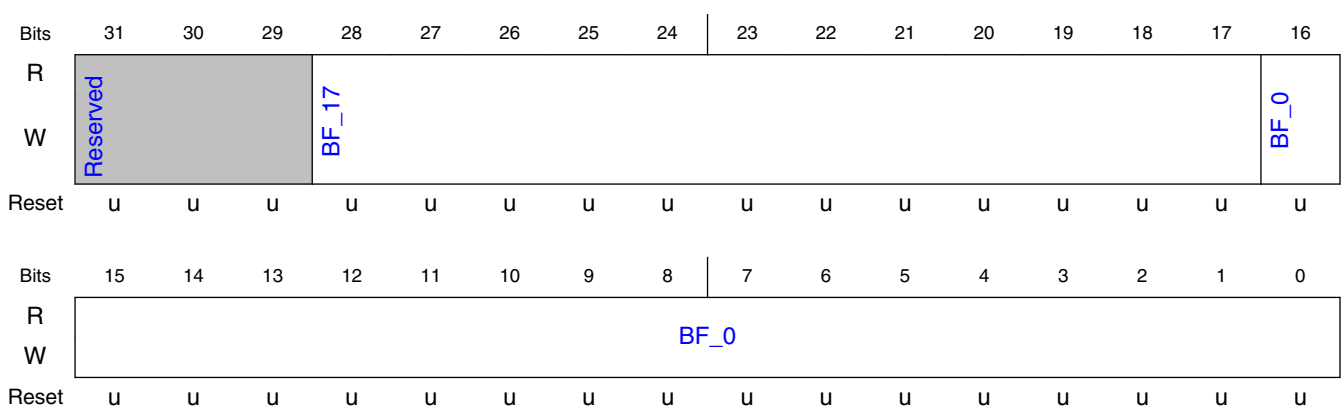
Field	Function
31-30 BF_30	Scaling mode. ScalingMode. 0=disabled. 1=scaling only. 2=scale+encode.
29 —	Reserved.
28-17 BF_17	Scaling width of down-scaled image. ScaledWidth. [96..4076]
16-0 BF_0	Scaling ratio for width of down-scaled image. Fixed point integer 1.16.

15.3.2.1.187 VPU H1 Register 233 (SWREG233)

15.3.2.1.187.1 Offset

Register	Offset
SWREG233	3A4h

15.3.2.1.187.2 Diagram



15.3.2.1.187.3 Fields

Field	Function
31-29	Reserved.

Table continues on the next page...

Field	Function
—	
28-17 BF_17	Scaling width of down-scaled image. ScaledHeight. [2..4078]
16-0 BF_0	Scaling ratio for height of down-scaled image. Fixed point integer 1.16.

15.3.2.1.188 VPU H1 Register 236 (SWREG236)

15.3.2.1.188.1 Offset

Register	Offset
SWREG236	3B0h

15.3.2.1.188.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.188.3 Fields

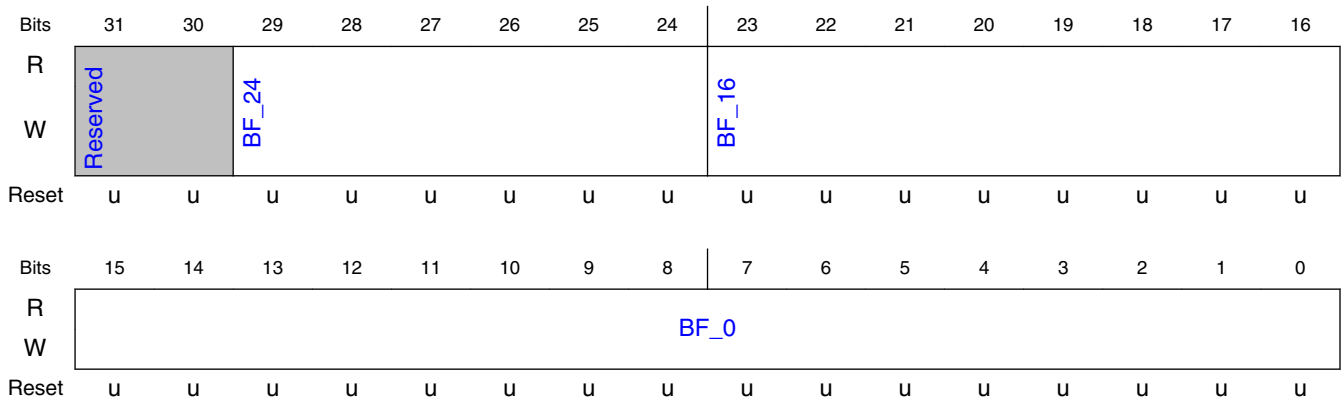
Field	Function
31-0 BF_0	Squared error output calculated for 13x13 pixels per macroblock. 32-bit max

15.3.2.1.189 VPU H1 Register 237 (SWREG237)

15.3.2.1.189.1 Offset

Register	Offset
SWREG237	3B4h

15.3.2.1.189.2 Diagram



15.3.2.1.189.3 Fields

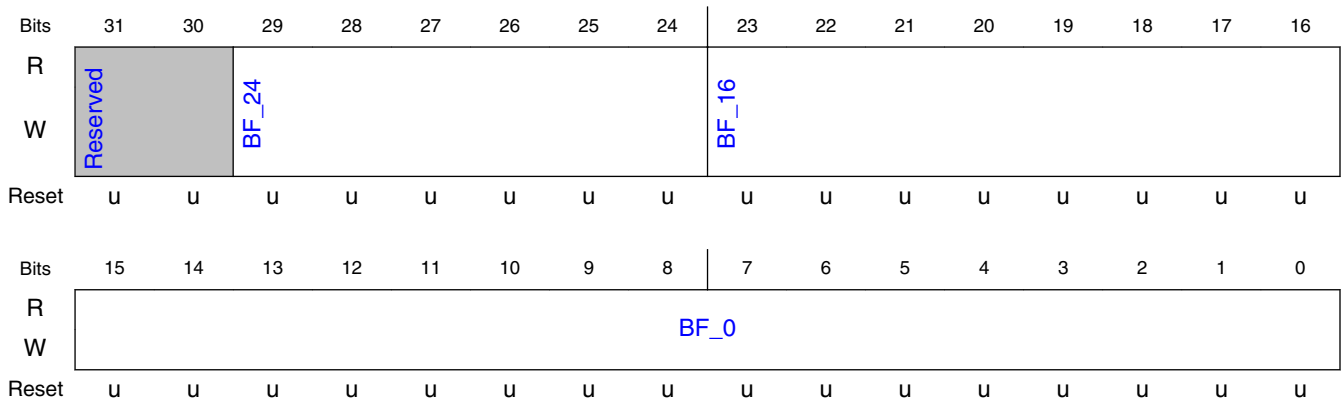
Field	Function
31-30 —	Reserved.
29-24 BF_24	MAD threshold div256. MadThreshold2 < MadThreshold
23-16 BF_16	MAD based QP adjustment. madQpChange [-127..127]
15-0 BF_0	Macroblock count with MAD value under threshold output

15.3.2.1.190 VPU H1 Register 238 (SWREG238)

15.3.2.1.190.1 Offset

Register	Offset
SWREG238	3B8h

15.3.2.1.190.2 Diagram



15.3.2.1.190.3 Fields

Field	Function
31-30 —	Reserved.
29-24 BF_24	MAD threshold div256. MadThreshold3 < MadThreshold2
23-16 BF_16	MAD based QP adjustment. madQpChange [-127..127]
15-0 BF_0	Macroblock count with MAD value under threshold output

15.3.2.1.191 VPU H1 Register 239 (SWREG239)

15.3.2.1.191.1 Offset

Register	Offset
SWREG239	3BCh

15.3.2.1.191.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_29			BF_24					BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_3					BF_0		
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.191.3 Fields

Field	Function
31-29 BF_29	VP8 predictor interpolation coefficient for full pixel position tap=1
28-24 BF_24	VP8 predictor interpolation coefficient for full pixel position tap=2
23-16 BF_16	VP8 predictor interpolation coefficient for full pixel position tap=3
15-8 BF_8	VP8 predictor interpolation coefficient for full pixel position tap=4
7-3 BF_3	VP8 predictor interpolation coefficient for full pixel position tap=5
2-0 BF_0	VP8 predictor interpolation coefficient for full pixel position tap=6

15.3.2.1.192 VPU H1 Register 240 (SWREG240)

15.3.2.1.192.1 Offset

Register	Offset
SWREG240	3C0h

15.3.2.1.192.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_29			BF_24					BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_3					BF_0		
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.192.3 Fields

Field	Function
31-29 BF_29	VP8 predictor interpolation coefficient for 1/8 pixel position tap=1
28-24 BF_24	VP8 predictor interpolation coefficient for 1/8 pixel position tap=2
23-16 BF_16	VP8 predictor interpolation coefficient for 1/8 pixel position tap=3
15-8 BF_8	VP8 predictor interpolation coefficient for 1/8 pixel position tap=4
7-3 BF_3	VP8 predictor interpolation coefficient for 1/8 pixel position tap=5
2-0 BF_0	VP8 predictor interpolation coefficient for 1/8 pixel position tap=6

15.3.2.1.193 VPU H1 Register 241 (SWREG241)

15.3.2.1.193.1 Offset

Register	Offset
SWREG241	3C4h

15.3.2.1.193.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_29			BF_24					BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_3					BF_0		
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.193.3 Fields

Field	Function
31-29 BF_29	VP8 predictor interpolation coefficient for 2/8 pixel position tap=1
28-24 BF_24	VP8 predictor interpolation coefficient for 2/8 pixel position tap=2
23-16 BF_16	VP8 predictor interpolation coefficient for 2/8 pixel position tap=3
15-8 BF_8	VP8 predictor interpolation coefficient for 2/8 pixel position tap=4
7-3 BF_3	VP8 predictor interpolation coefficient for 2/8 pixel position tap=5
2-0 BF_0	VP8 predictor interpolation coefficient for 2/8 pixel position tap=6

15.3.2.1.194 VPU H1 Register 242 (SWREG242)

15.3.2.1.194.1 Offset

Register	Offset
SWREG242	3C8h

15.3.2.1.194.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_29			BF_24					BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_3					BF_0		
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.194.3 Fields

Field	Function
31-29 BF_29	VP8 predictor interpolation coefficient for 3/8 pixel position tap=1
28-24 BF_24	VP8 predictor interpolation coefficient for 3/8 pixel position tap=2
23-16 BF_16	VP8 predictor interpolation coefficient for 3/8 pixel position tap=3
15-8 BF_8	VP8 predictor interpolation coefficient for 3/8 pixel position tap=4
7-3 BF_3	VP8 predictor interpolation coefficient for 3/8 pixel position tap=5
2-0 BF_0	VP8 predictor interpolation coefficient for 3/8 pixel position tap=6

15.3.2.1.195 VPU H1 Register 243 (SWREG243)

15.3.2.1.195.1 Offset

Register	Offset
SWREG243	3CCh

15.3.2.1.195.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_29			BF_24					BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_3					BF_0		
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.195.3 Fields

Field	Function
31-29 BF_29	VP8 predictor interpolation coefficient for 4/8 pixel position tap=1
28-24 BF_24	VP8 predictor interpolation coefficient for 4/8 pixel position tap=2
23-16 BF_16	VP8 predictor interpolation coefficient for 4/8 pixel position tap=3
15-8 BF_8	VP8 predictor interpolation coefficient for 4/8 pixel position tap=4
7-3 BF_3	VP8 predictor interpolation coefficient for 4/8 pixel position tap=5
2-0 BF_0	VP8 predictor interpolation coefficient for 4/8 pixel position tap=6

15.3.2.1.196 VPU H1 Register 244 (SWREG244)

15.3.2.1.196.1 Offset

Register	Offset
SWREG244	3D0h

15.3.2.1.196.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.196.3 Fields

Field	Function
31-0 BF_0	Base address for VP8 3rd DCT partition

15.3.2.1.197 VPU H1 Register 245 (SWREG245)

15.3.2.1.197.1 Offset

Register	Offset
SWREG245	3D4h

15.3.2.1.197.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.197.3 Fields

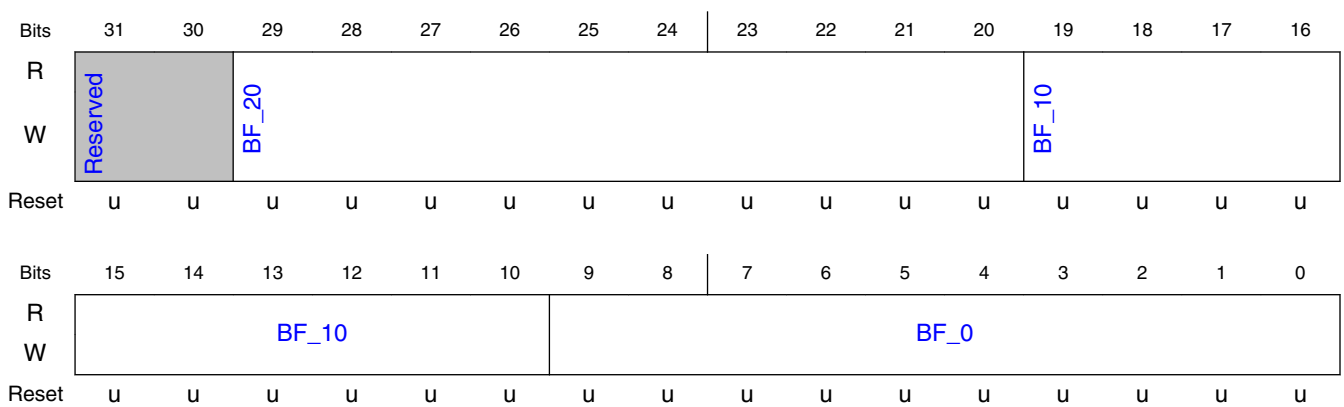
Field	Function
31-0 BF_0	Base address for VP8 4th DCT partition

15.3.2.1.198 VPU H1 Register 256 (SWREG256)

15.3.2.1.198.1 Offset

Register	Offset
SWREG256	400h

15.3.2.1.198.2 Diagram



15.3.2.1.198.3 Fields

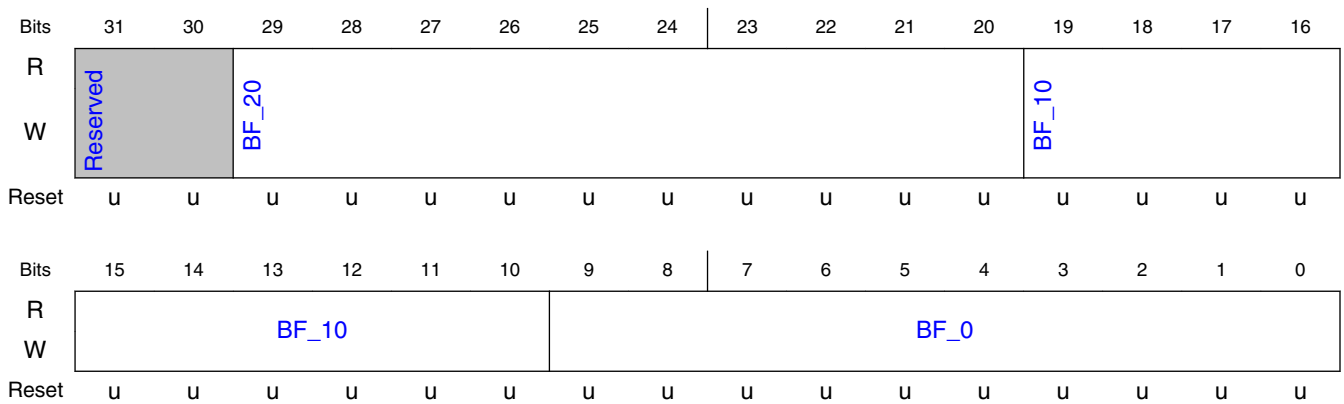
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 1: Intra 16x16 mode 2 penalty
19-10 BF_10	Segment 1: Intra 16x16 mode 1 penalty
9-0 BF_0	Segment 1: Intra 16x16 mode 0 penalty

15.3.2.1.199 VPU H1 Register 257 (SWREG257)

15.3.2.1.199.1 Offset

Register	Offset
SWREG257	404h

15.3.2.1.199.2 Diagram



15.3.2.1.199.3 Fields

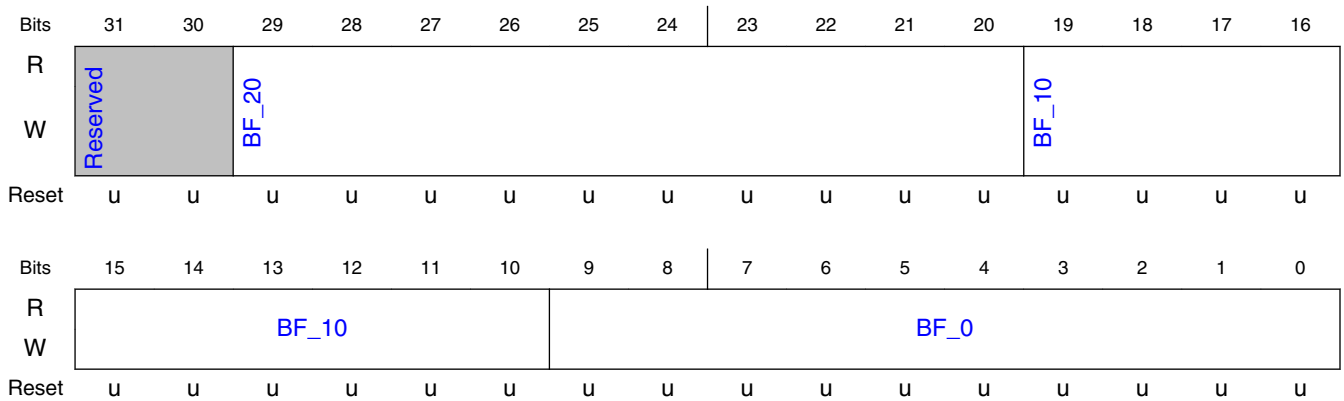
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 1: Intra 4x4 mode 1 penalty
19-10 BF_10	Segment 1: Intra 4x4 mode 0 penalty
9-0 BF_0	Segment 1: Intra 16x16 mode 3 penalty

15.3.2.1.200 VPU H1 Register 258 (SWREG258)

15.3.2.1.200.1 Offset

Register	Offset
SWREG258	408h

15.3.2.1.200.2 Diagram



15.3.2.1.200.3 Fields

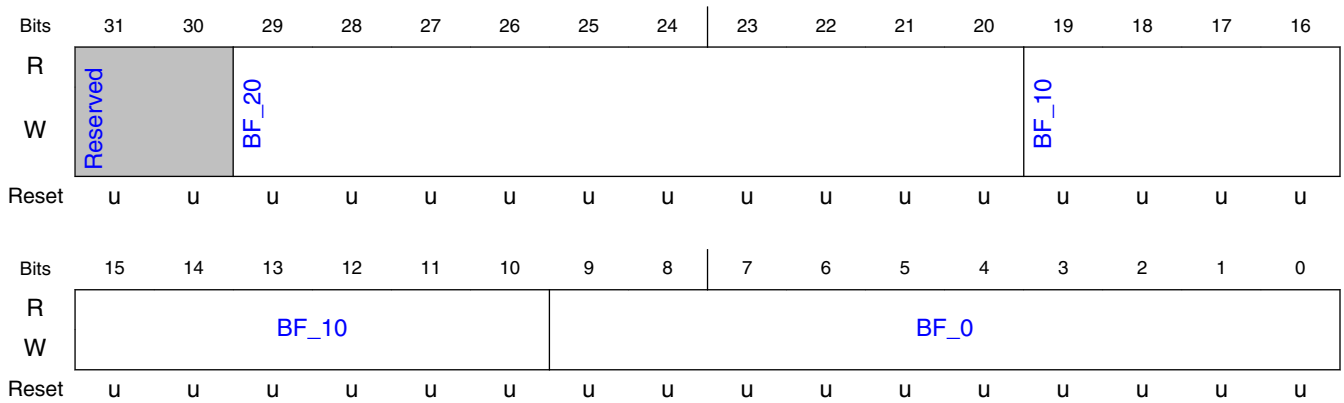
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 1: Intra 4x4 mode 4 penalty
19-10 BF_10	Segment 1: Intra 4x4 mode 3 penalty
9-0 BF_0	Segment 1: Intra 4x4 mode 2 penalty

15.3.2.1.201 VPU H1 Register 259 (SWREG259)

15.3.2.1.201.1 Offset

Register	Offset
SWREG259	40Ch

15.3.2.1.201.2 Diagram



15.3.2.1.201.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 1: Intra 4x4 mode 7 penalty
19-10 BF_10	Segment 1: Intra 4x4 mode 6 penalty
9-0 BF_0	Segment 1: Intra 4x4 mode 5 penalty

15.3.2.1.202 VPU H1 Register 260 (SWREG260)

15.3.2.1.202.1 Offset

Register	Offset
SWREG260	410h

15.3.2.1.202.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				BF_20								BF_10			
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_10						BF_0									
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.202.3 Fields

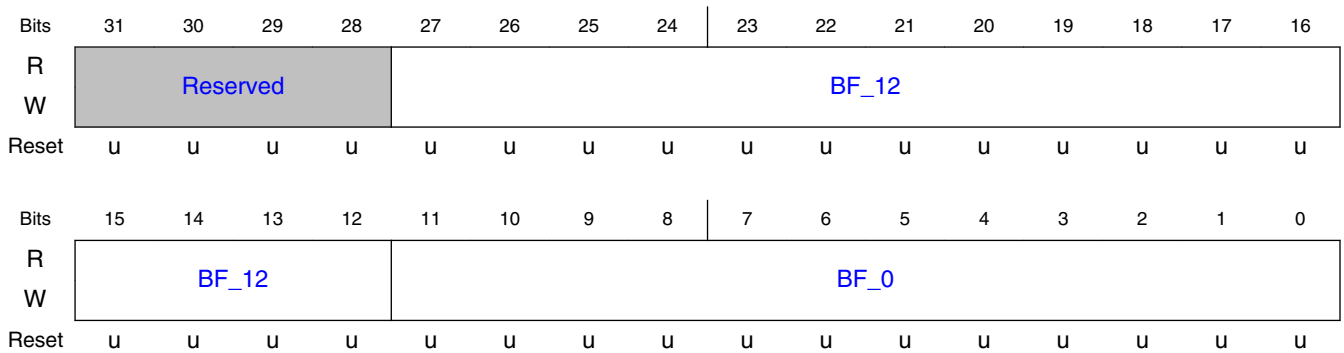
Field	Function
31-28 —	Reserved.
27-20 BF_20	Segment 1: Intra 4x4 previous mode favor for H.264
19-10 BF_10	Segment 1: Intra 4x4 mode 9 penalty
9-0 BF_0	Segment 1: Intra 4x4 mode 8 penalty

15.3.2.1.203 VPU H1 Register 261 (SWREG261)

15.3.2.1.203.1 Offset

Register	Offset
SWREG261	414h

15.3.2.1.203.2 Diagram



15.3.2.1.203.3 Fields

Field	Function
31-28 —	Reserved.
27-12 BF_12	Segment 1: Intra 16x16 mode favor in intra 16x16/4x4 selection
11-0 BF_0	Segment 1: Bit cost of inter type

15.3.2.1.204 VPU H1 Register 262 (SWREG262)

15.3.2.1.204.1 Offset

Register	Offset
SWREG262	418h

15.3.2.1.204.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.204.3 Fields

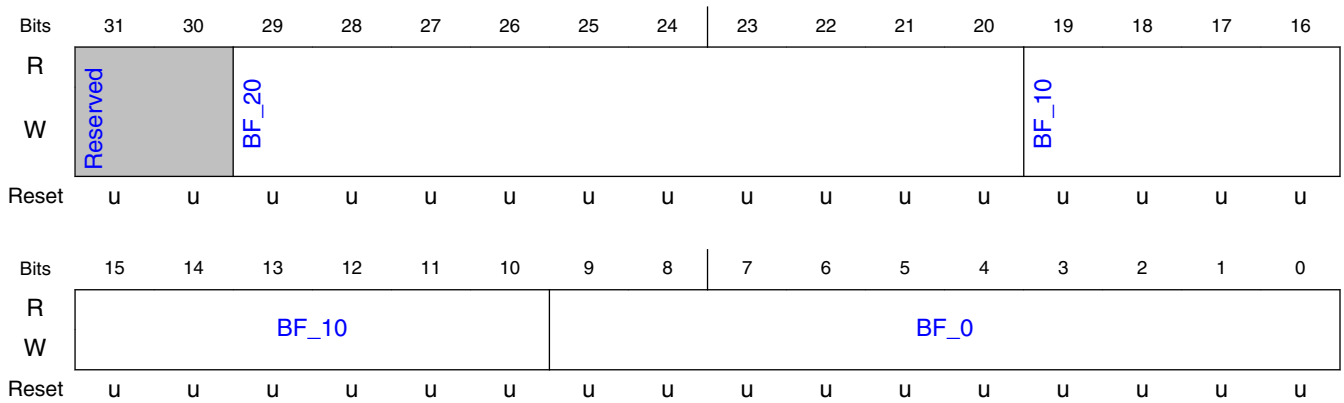
Field	Function
31-24 BF_24	Segment 1: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]
23-16 BF_16	Segment 1: Skip mode (zero/nearest/near) penalty
15-0 BF_0	Segment 1: Inter MB mode favor in intra/inter selection

15.3.2.1.205 VPU H1 Register 263 (SWREG263)

15.3.2.1.205.1 Offset

Register	Offset
SWREG263	41Ch

15.3.2.1.205.2 Diagram



15.3.2.1.205.3 Fields

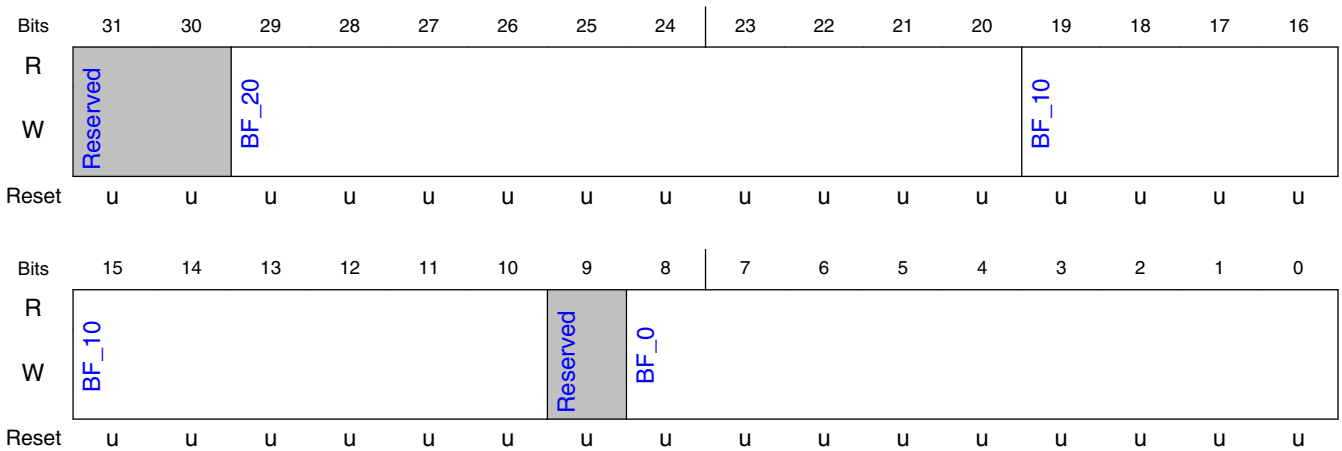
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 1: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 1: Penalty for using 8x8 MV.
9-0 BF_0	Segment 1: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.206 VPU H1 Register 264 (SWREG264)

15.3.2.1.206.1 Offset

Register	Offset
SWREG264	420h

15.3.2.1.206.2 Diagram



15.3.2.1.206.3 Fields

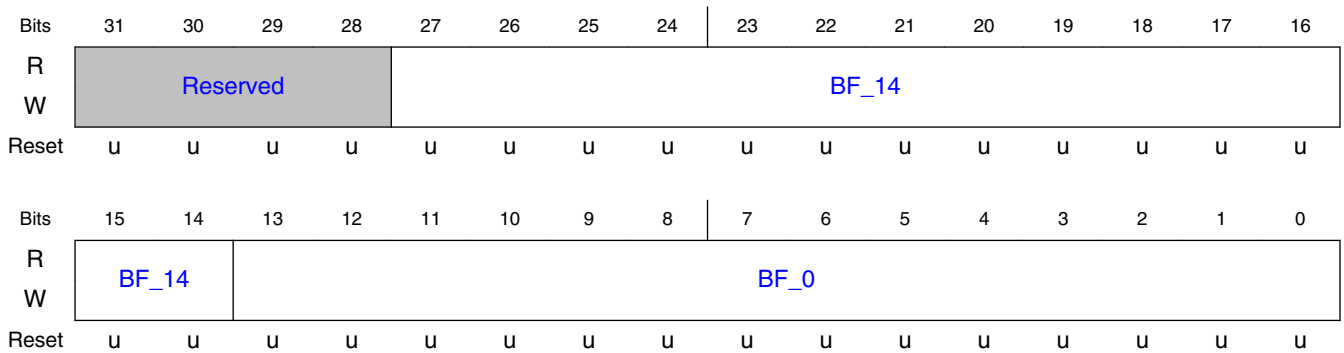
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 1: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-10 BF_10	Segment 1: Penalty for using zero-MV in 16x8/8x16/8x8 split
9 —	Reserved.
8-0 BF_0	Segment 1: Penalty for using 4x4 MV.

15.3.2.1.207 VPU H1 Register 265 (SWREG265)

15.3.2.1.207.1 Offset

Register	Offset
SWREG265	424h

15.3.2.1.207.2 Diagram



15.3.2.1.207.3 Fields

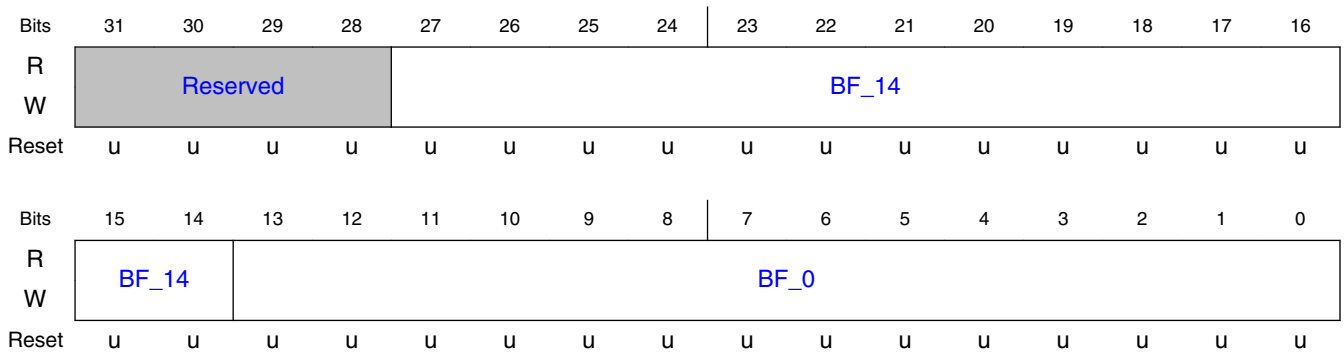
Field	Function
31-28 —	Reserved.
27-14 BF_14	Segment 1: Deadzone rate multiplier for plane 1
13-0 BF_0	Segment 1: Deadzone rate multiplier for plane 0

15.3.2.1.208 VPU H1 Register 266 (SWREG266)

15.3.2.1.208.1 Offset

Register	Offset
SWREG266	428h

15.3.2.1.208.2 Diagram



15.3.2.1.208.3 Fields

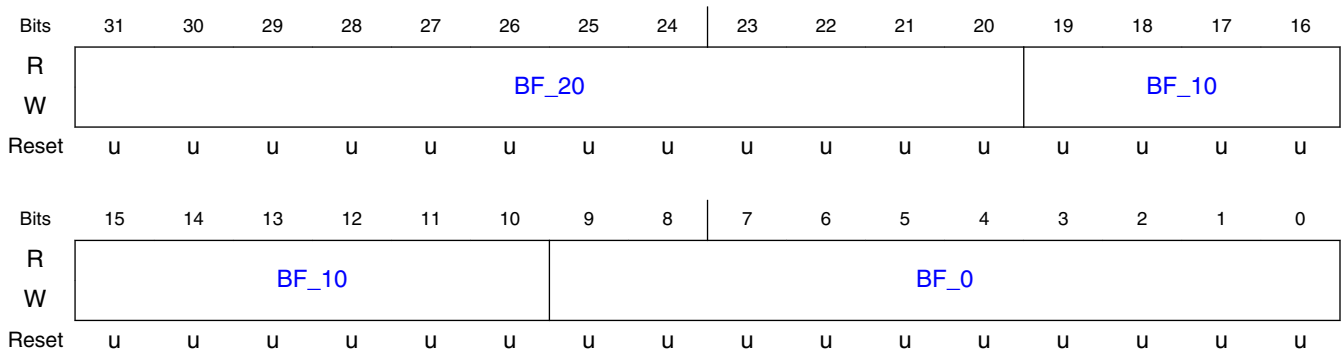
Field	Function
31-28 —	Reserved.
27-14 BF_14	Segment 1: Deadzone rate multiplier for plane 3
13-0 BF_0	Segment 1: Deadzone rate multiplier for plane 2

15.3.2.1.209 VPU H1 Register 267 (SWREG267)

15.3.2.1.209.1 Offset

Register	Offset
SWREG267	42Ch

15.3.2.1.209.2 Diagram



15.3.2.1.209.3 Fields

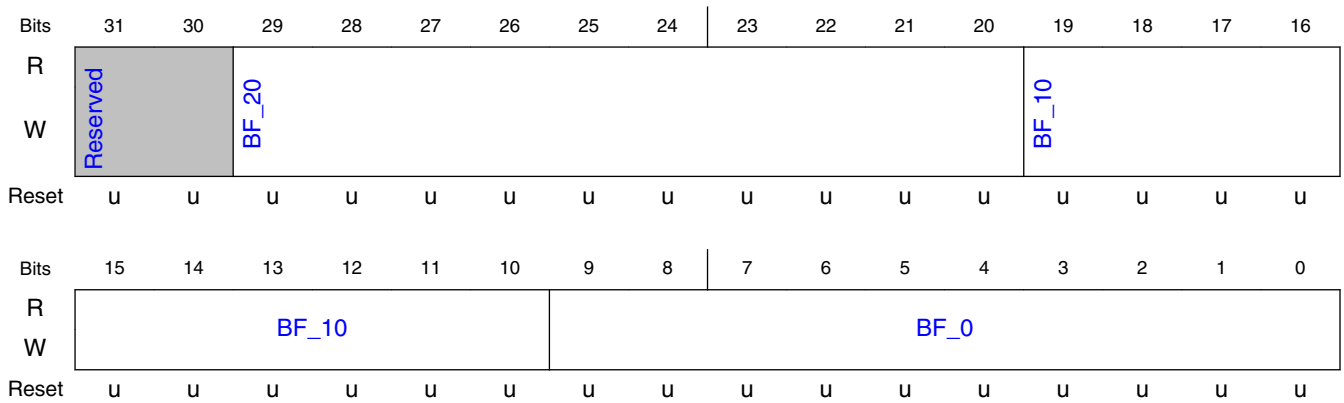
Field	Function
31-20 BF_20	Segment 1: Coeff for dmv penalty for intra/inter selection
19-10 BF_10	Segment 1: Deadzone rate for macroblock skip token 1
9-0 BF_0	Segment 1: Deadzone rate for macroblock skip token 0

15.3.2.1.210 VPU H1 Register 268 (SWREG268)

15.3.2.1.210.1 Offset

Register	Offset
SWREG268	430h

15.3.2.1.210.2 Diagram



15.3.2.1.210.3 Fields

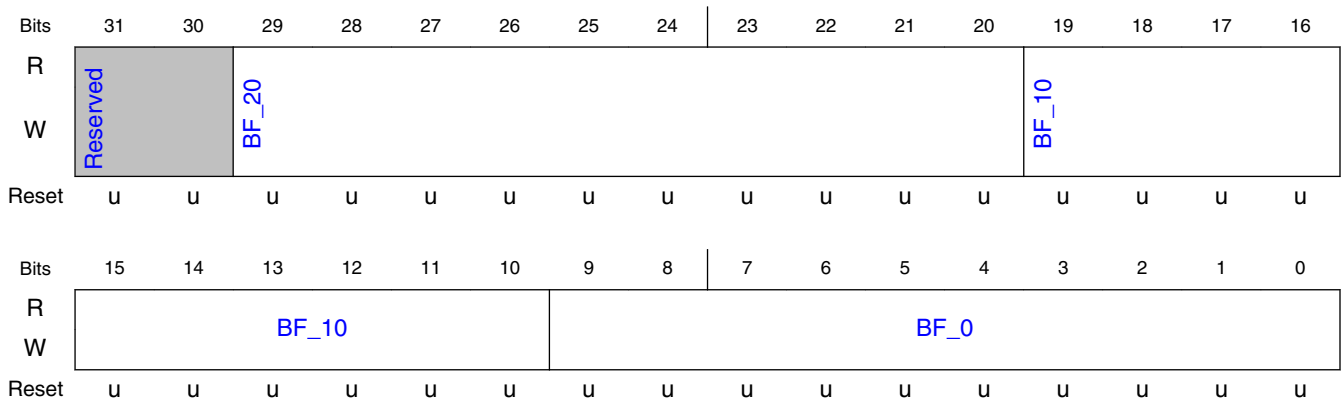
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 2: Intra 16x16 mode 2 penalty
19-10 BF_10	Segment 2: Intra 16x16 mode 1 penalty
9-0 BF_0	Segment 2: Intra 16x16 mode 0 penalty

15.3.2.1.211 VPU H1 Register 269 (SWREG269)

15.3.2.1.211.1 Offset

Register	Offset
SWREG269	434h

15.3.2.1.211.2 Diagram



15.3.2.1.211.3 Fields

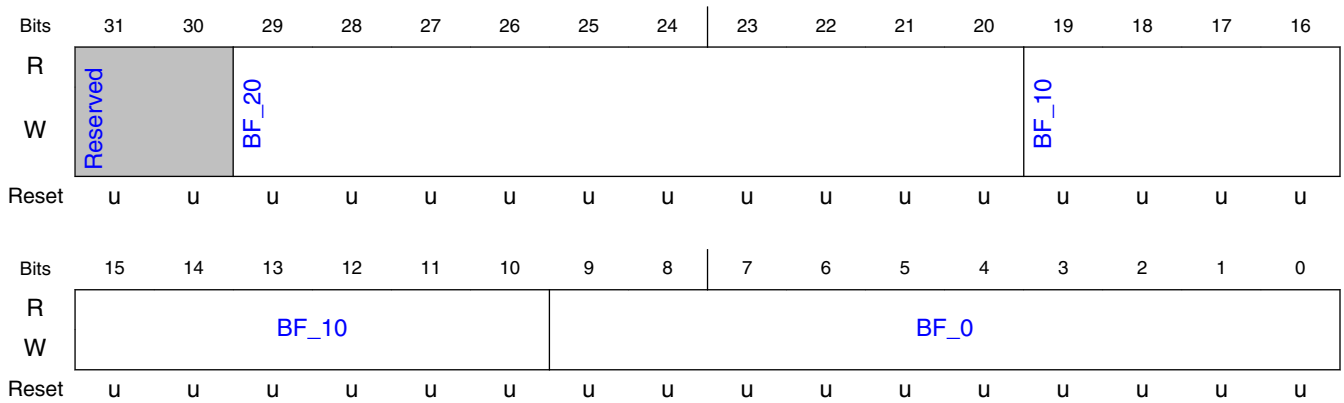
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 2: Intra 4x4 mode 1 penalty
19-10 BF_10	Segment 2: Intra 4x4 mode 0 penalty
9-0 BF_0	Segment 2: Intra 16x16 mode 3 penalty

15.3.2.1.212 VPU H1 Register 270 (SWREG270)

15.3.2.1.212.1 Offset

Register	Offset
SWREG270	438h

15.3.2.1.212.2 Diagram



15.3.2.1.212.3 Fields

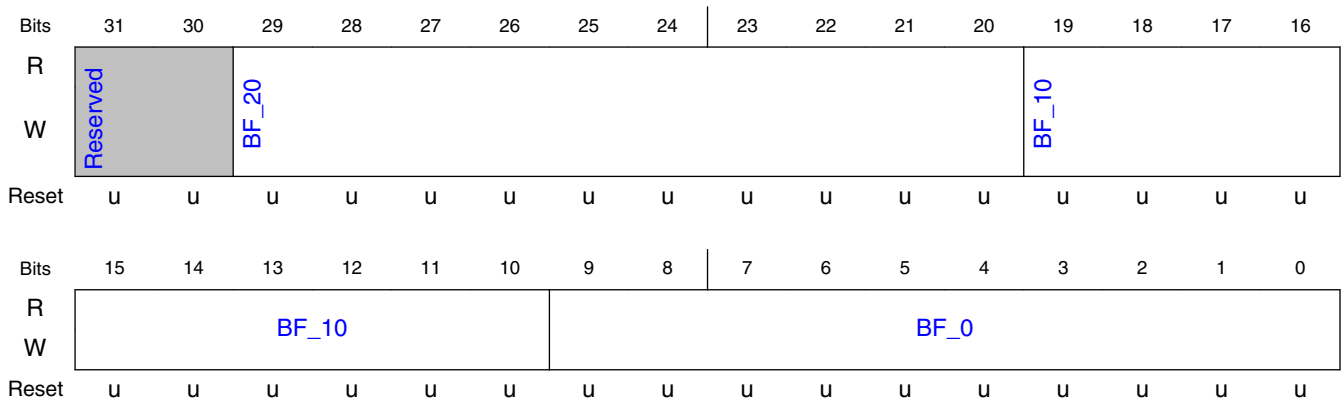
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 2: Intra 4x4 mode 4 penalty
19-10 BF_10	Segment 2: Intra 4x4 mode 3 penalty
9-0 BF_0	Segment 2: Intra 4x4 mode 2 penalty

15.3.2.1.213 VPU H1 Register 271 (SWREG271)

15.3.2.1.213.1 Offset

Register	Offset
SWREG271	43Ch

15.3.2.1.213.2 Diagram



15.3.2.1.213.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 2: Intra 4x4 mode 7 penalty
19-10 BF_10	Segment 2: Intra 4x4 mode 6 penalty
9-0 BF_0	Segment 2: Intra 4x4 mode 5 penalty

15.3.2.1.214 VPU H1 Register 272 (SWREG272)

15.3.2.1.214.1 Offset

Register	Offset
SWREG272	440h

15.3.2.1.214.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				BF_20								BF_10			
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_10						BF_0									
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.214.3 Fields

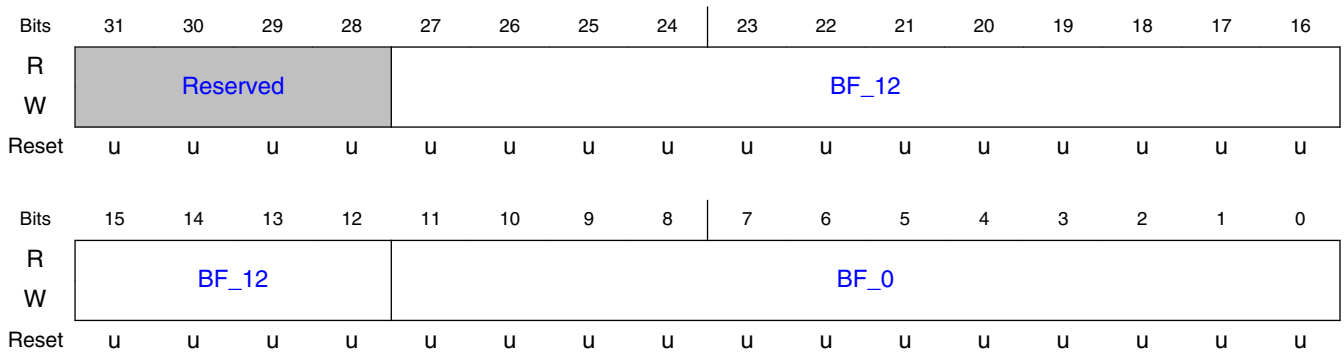
Field	Function
31-28 —	Reserved.
27-20 BF_20	Segment 2: Intra 4x4 previous mode favor for H.264
19-10 BF_10	Segment 2: Intra 4x4 mode 9 penalty
9-0 BF_0	Segment 2: Intra 4x4 mode 8 penalty

15.3.2.1.215 VPU H1 Register 273 (SWREG273)

15.3.2.1.215.1 Offset

Register	Offset
SWREG273	444h

15.3.2.1.215.2 Diagram



15.3.2.1.215.3 Fields

Field	Function
31-28 —	Reserved.
27-12 BF_12	Segment 2: Intra 16x16 mode favor in intra 16x16/4x4 selection
11-0 BF_0	Segment 2: Bit cost of inter type

15.3.2.1.216 VPU H1 Register 274 (SWREG274)

15.3.2.1.216.1 Offset

Register	Offset
SWREG274	448h

15.3.2.1.216.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.216.3 Fields

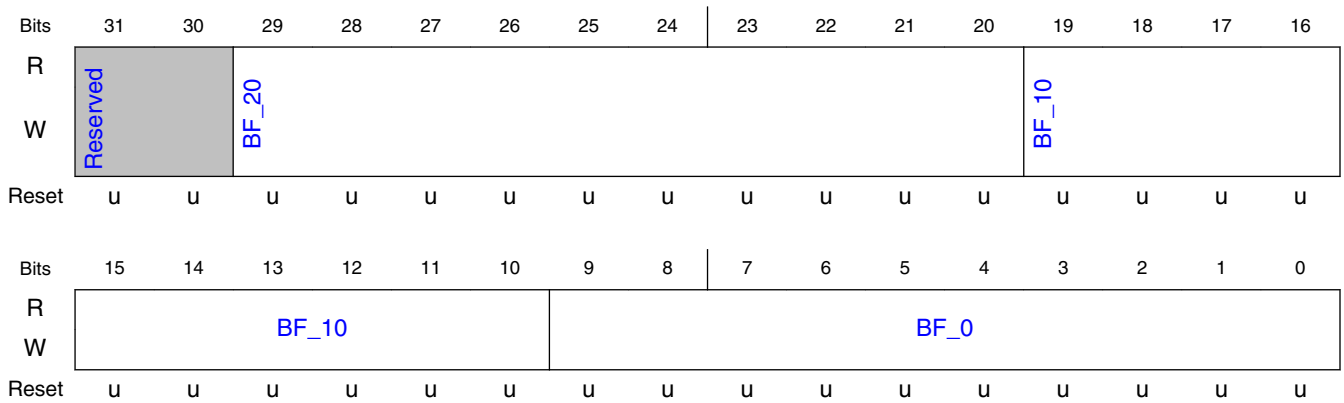
Field	Function
31-24 BF_24	Segment 2: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]
23-16 BF_16	Segment 2: Skip mode (zero/nearest/near) penalty
15-0 BF_0	Segment 2: Inter MB mode favor in intra/inter selection

15.3.2.1.217 VPU H1 Register 275 (SWREG275)

15.3.2.1.217.1 Offset

Register	Offset
SWREG275	44Ch

15.3.2.1.217.2 Diagram



15.3.2.1.217.3 Fields

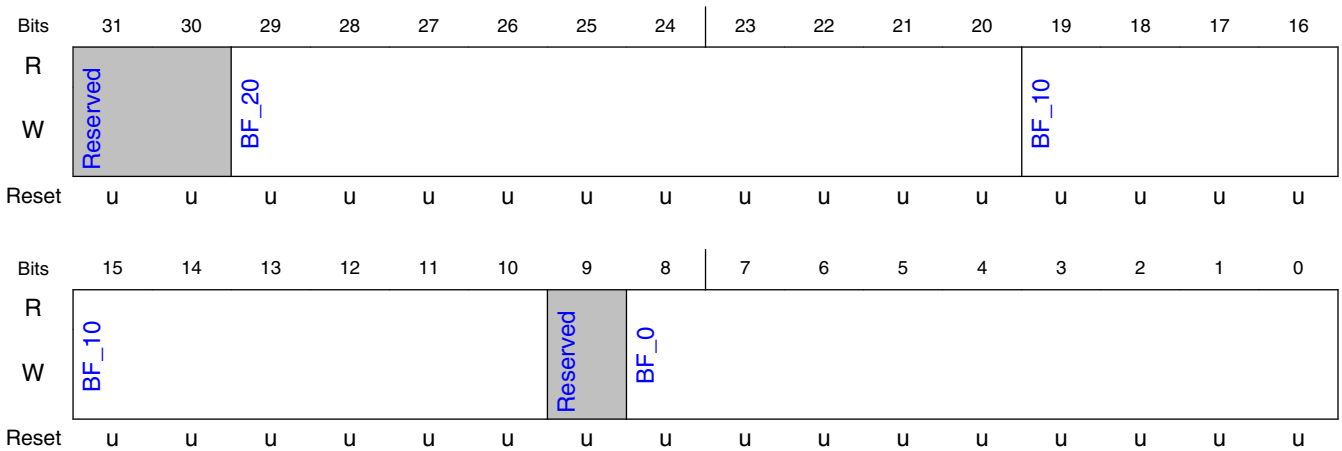
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 2: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 2: Penalty for using 8x8 MV.
9-0 BF_0	Segment 2: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.218 VPU H1 Register 276 (SWREG276)

15.3.2.1.218.1 Offset

Register	Offset
SWREG276	450h

15.3.2.1.218.2 Diagram



15.3.2.1.218.3 Fields

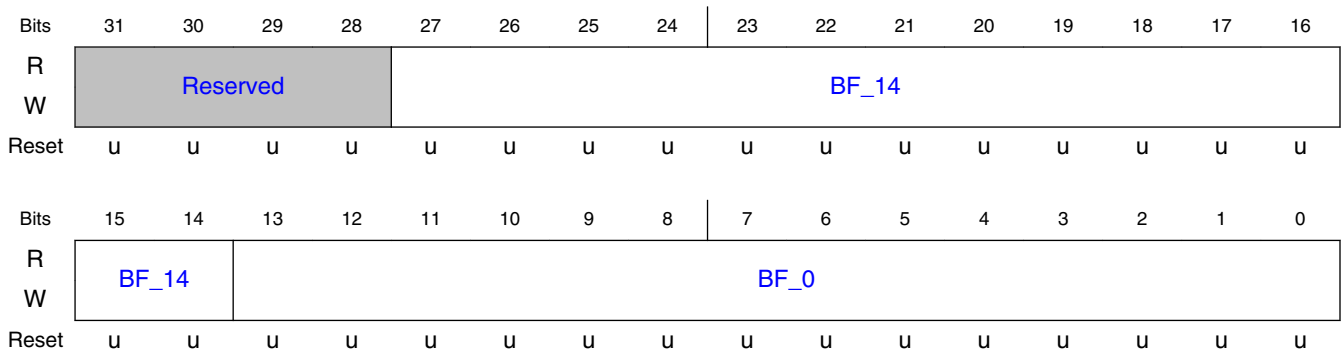
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 2: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-10 BF_10	Segment 2: Penalty for using zero-MV in 16x8/8x16/8x8 split
9 —	Reserved.
8-0 BF_0	Segment 2: Penalty for using 4x4 MV.

15.3.2.1.219 VPU H1 Register 277 (SWREG277)

15.3.2.1.219.1 Offset

Register	Offset
SWREG277	454h

15.3.2.1.219.2 Diagram



15.3.2.1.219.3 Fields

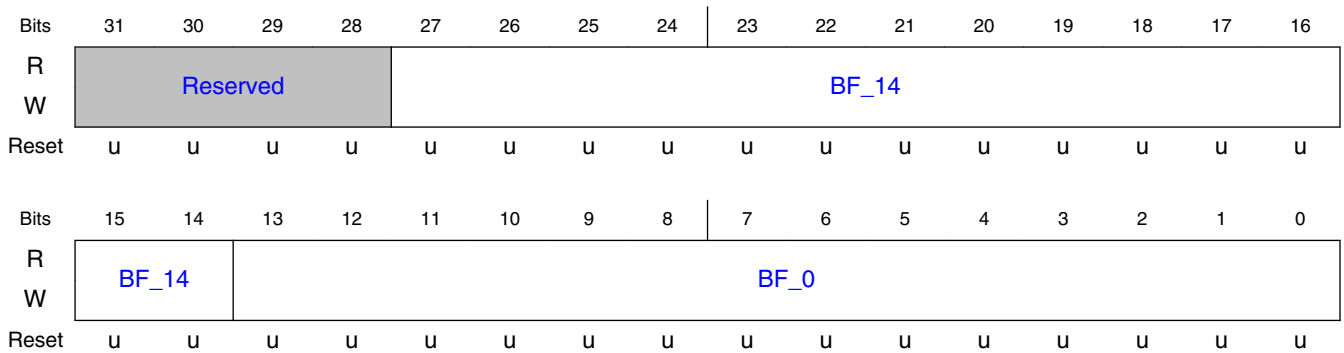
Field	Function
31-28 —	Reserved.
27-14 BF_14	Segment 2: Deadzone rate multiplier for plane 1
13-0 BF_0	Segment 2: Deadzone rate multiplier for plane 0

15.3.2.1.220 VPU H1 Register 278 (SWREG278)

15.3.2.1.220.1 Offset

Register	Offset
SWREG278	458h

15.3.2.1.220.2 Diagram



15.3.2.1.220.3 Fields

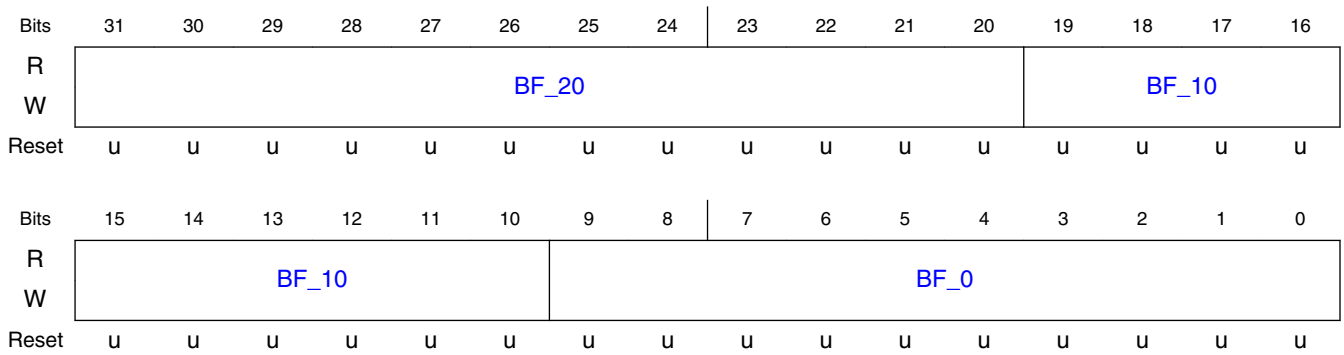
Field	Function
31-28 —	Reserved.
27-14 BF_14	Segment 2: Deadzone rate multiplier for plane 3
13-0 BF_0	Segment 2: Deadzone rate multiplier for plane 2

15.3.2.1.221 VPU H1 Register 279 (SWREG279)

15.3.2.1.221.1 Offset

Register	Offset
SWREG279	45Ch

15.3.2.1.221.2 Diagram



15.3.2.1.221.3 Fields

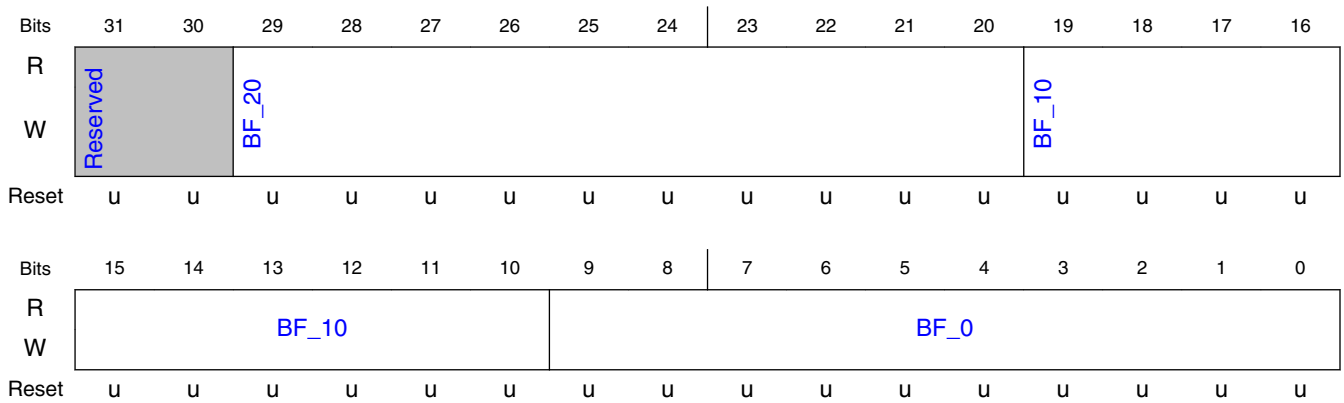
Field	Function
31-20 BF_20	Segment 2: Coeff for dmv penalty for intra/inter selection
19-10 BF_10	Segment 2: Deadzone rate for macroblock skip token 1
9-0 BF_0	Segment 2: Deadzone rate for macroblock skip token 0

15.3.2.1.222 VPU H1 Register 280 (SWREG280)

15.3.2.1.222.1 Offset

Register	Offset
SWREG280	460h

15.3.2.1.222.2 Diagram



15.3.2.1.222.3 Fields

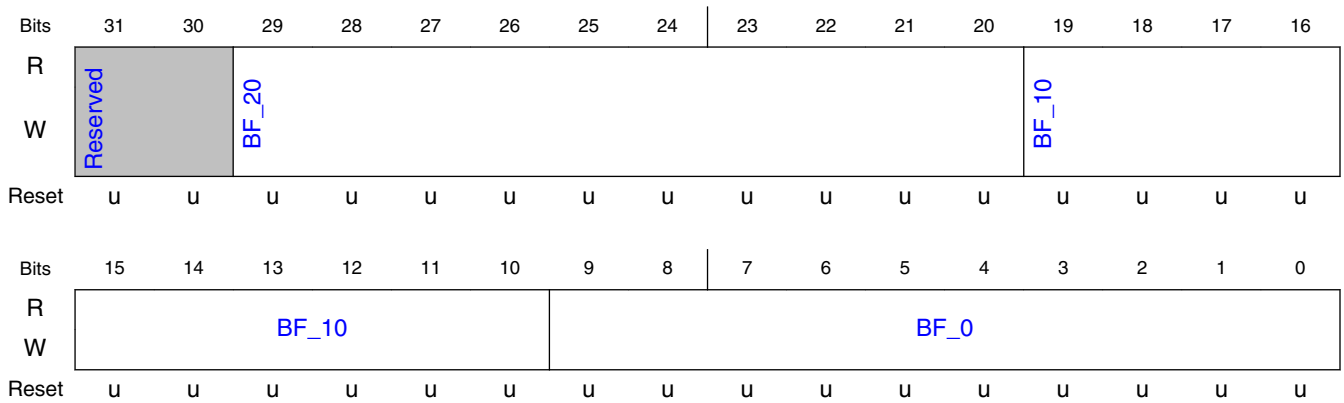
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 3: Intra 16x16 mode 2 penalty
19-10 BF_10	Segment 3: Intra 16x16 mode 1 penalty
9-0 BF_0	Segment 3: Intra 16x16 mode 0 penalty

15.3.2.1.223 VPU H1 Register 281 (SWREG281)

15.3.2.1.223.1 Offset

Register	Offset
SWREG281	464h

15.3.2.1.223.2 Diagram



15.3.2.1.223.3 Fields

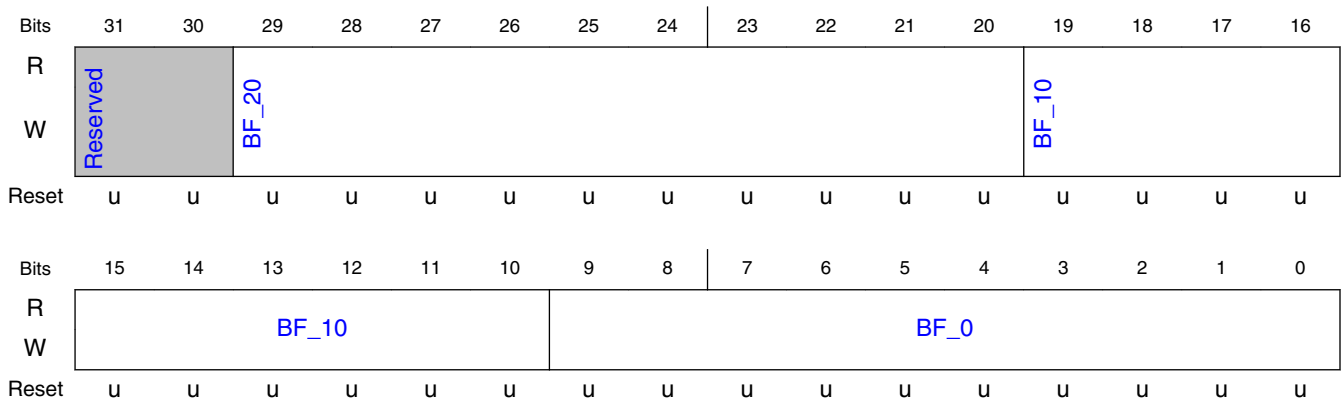
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 3: Intra 4x4 mode 1 penalty
19-10 BF_10	Segment 3: Intra 4x4 mode 0 penalty
9-0 BF_0	Segment 3: Intra 16x16 mode 3 penalty

15.3.2.1.224 VPU H1 Register 282 (SWREG282)

15.3.2.1.224.1 Offset

Register	Offset
SWREG282	468h

15.3.2.1.224.2 Diagram



15.3.2.1.224.3 Fields

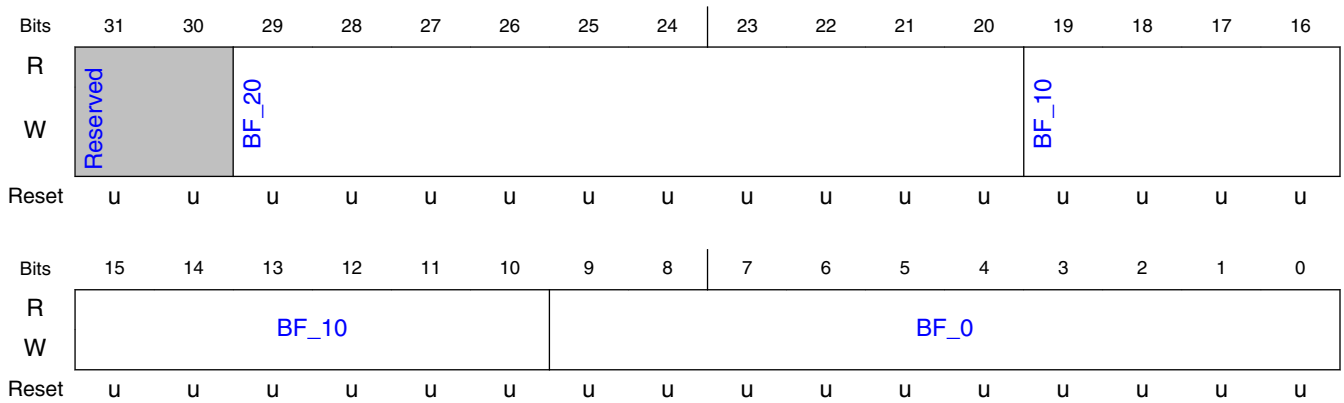
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 3: Intra 4x4 mode 4 penalty
19-10 BF_10	Segment 3: Intra 4x4 mode 3 penalty
9-0 BF_0	Segment 3: Intra 4x4 mode 2 penalty

15.3.2.1.225 VPU H1 Register 283 (SWREG283)

15.3.2.1.225.1 Offset

Register	Offset
SWREG283	46Ch

15.3.2.1.225.2 Diagram



15.3.2.1.225.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 3: Intra 4x4 mode 7 penalty
19-10 BF_10	Segment 3: Intra 4x4 mode 6 penalty
9-0 BF_0	Segment 3: Intra 4x4 mode 5 penalty

15.3.2.1.226 VPU H1 Register 284 (SWREG284)

15.3.2.1.226.1 Offset

Register	Offset
SWREG284	470h

15.3.2.1.226.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				BF_20								BF_10			
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_10						BF_0									
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.226.3 Fields

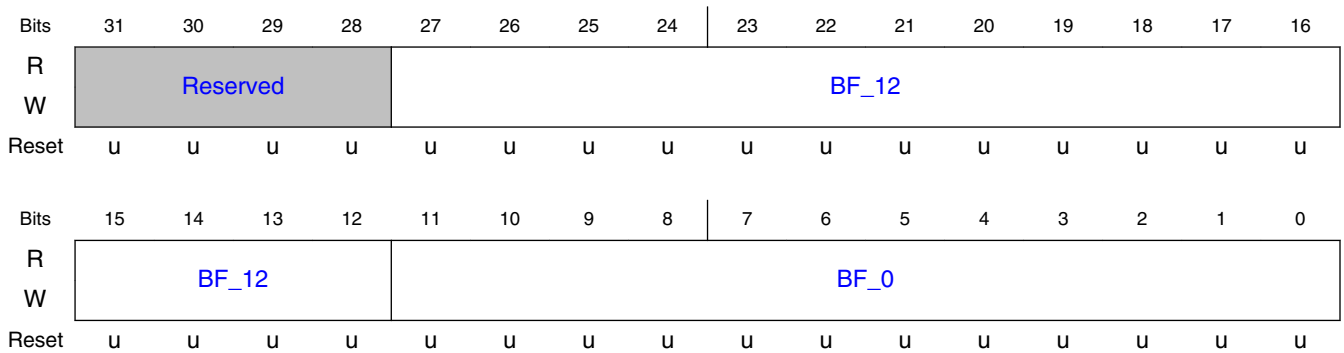
Field	Function
31-28 —	Reserved.
27-20 BF_20	Segment 3: Intra 4x4 previous mode favor for H.264
19-10 BF_10	Segment 3: Intra 4x4 mode 9 penalty
9-0 BF_0	Segment 3: Intra 4x4 mode 8 penalty

15.3.2.1.227 VPU H1 Register 285 (SWREG285)

15.3.2.1.227.1 Offset

Register	Offset
SWREG285	474h

15.3.2.1.227.2 Diagram



15.3.2.1.227.3 Fields

Field	Function
31-28 —	Reserved.
27-12 BF_12	Segment 3: Intra 16x16 mode favor in intra 16x16/4x4 selection
11-0 BF_0	Segment 3: Bit cost of inter type

15.3.2.1.228 VPU H1 Register 286 (SWREG286)

15.3.2.1.228.1 Offset

Register	Offset
SWREG286	478h

15.3.2.1.228.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.228.3 Fields

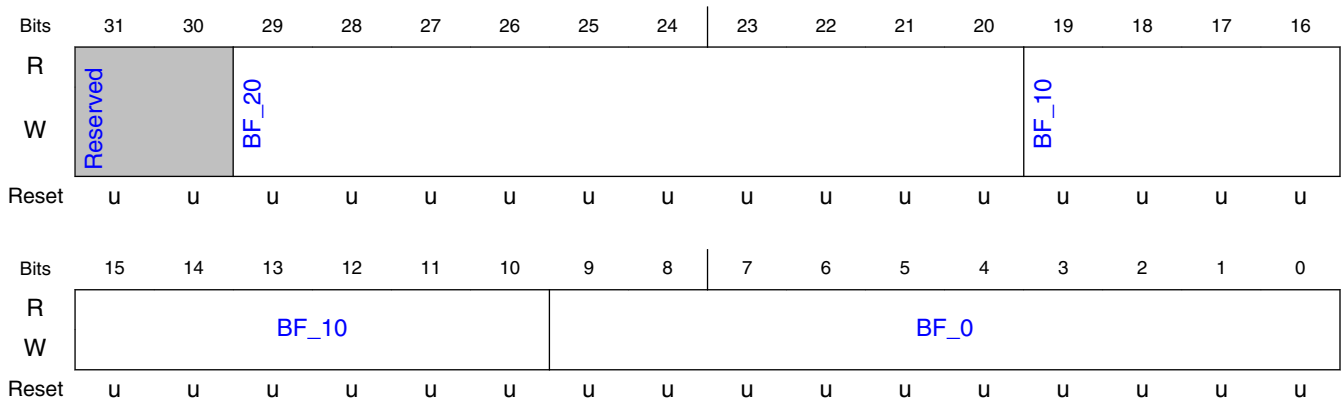
Field	Function
31-24 BF_24	Segment 3: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]
23-16 BF_16	Segment 3: Skip mode (zero/nearest/near) penalty
15-0 BF_0	Segment 3: Inter MB mode favor in intra/inter selection

15.3.2.1.229 VPU H1 Register 287 (SWREG287)

15.3.2.1.229.1 Offset

Register	Offset
SWREG287	47Ch

15.3.2.1.229.2 Diagram



15.3.2.1.229.3 Fields

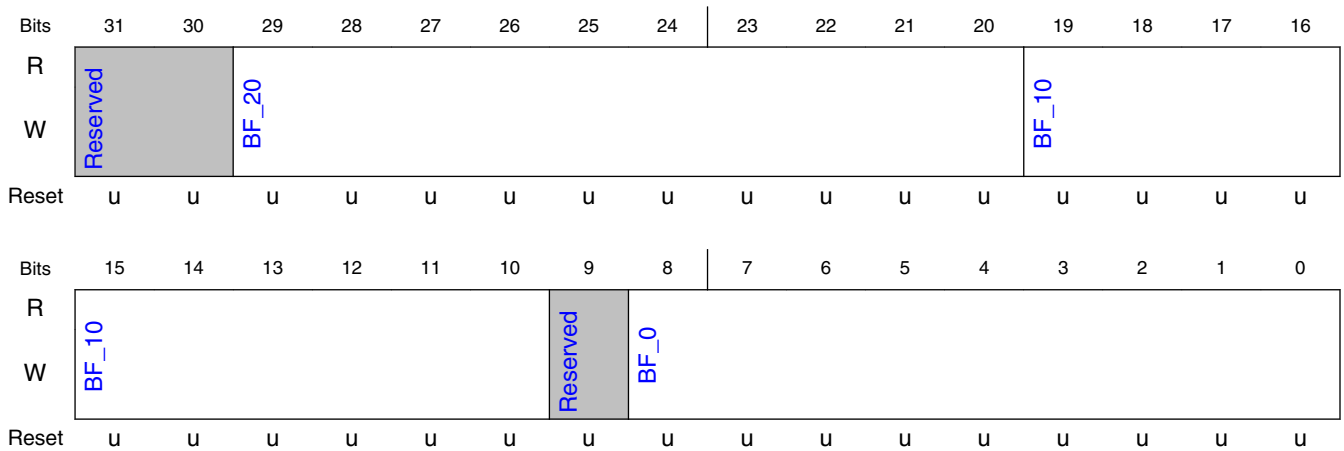
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 3: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 3: Penalty for using 8x8 MV.
9-0 BF_0	Segment 3: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.230 VPU H1 Register 288 (SWREG288)

15.3.2.1.230.1 Offset

Register	Offset
SWREG288	480h

15.3.2.1.230.2 Diagram



15.3.2.1.230.3 Fields

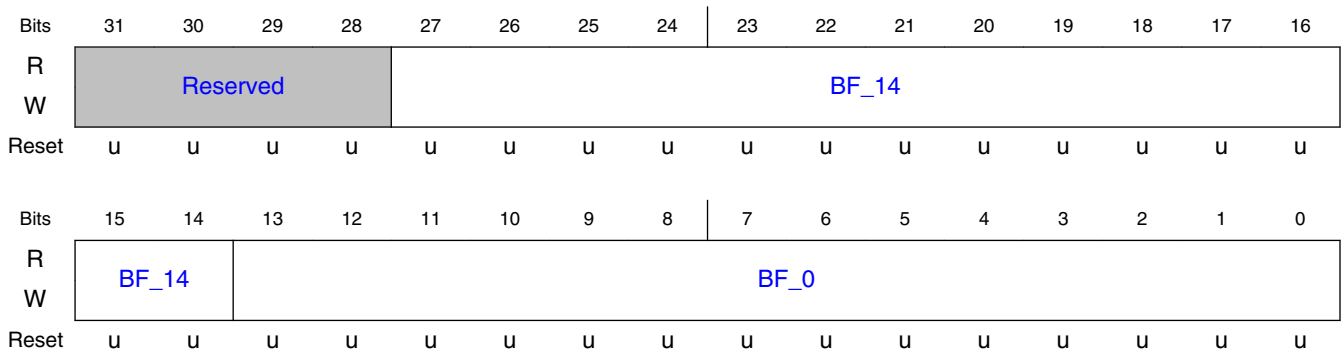
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 3: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-10 BF_10	Segment 3: Penalty for using zero-MV in 16x8/8x16/8x8 split
9 —	Reserved.
8-0 BF_0	Segment 3: Penalty for using 4x4 MV.

15.3.2.1.231 VPU H1 Register 289 (SWREG289)

15.3.2.1.231.1 Offset

Register	Offset
SWREG289	484h

15.3.2.1.231.2 Diagram



15.3.2.1.231.3 Fields

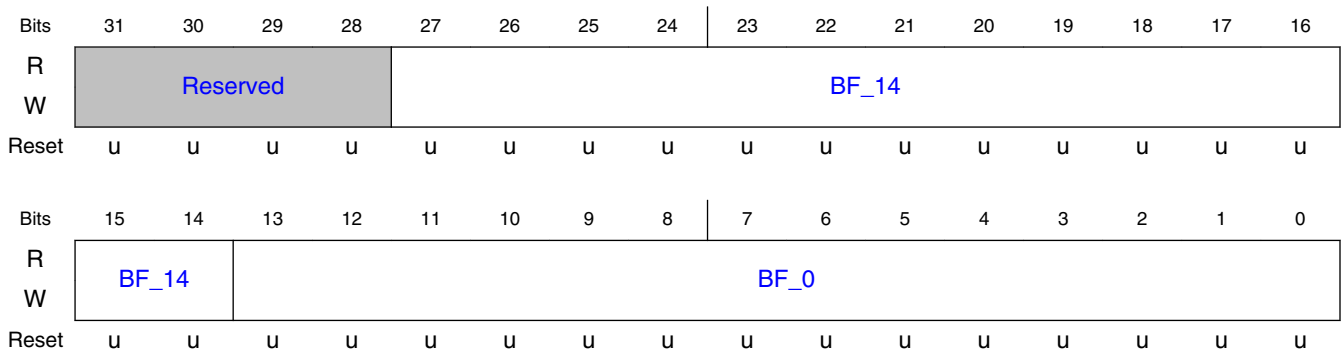
Field	Function
31-28 —	Reserved.
27-14 BF_14	Segment 3: Deadzone rate multiplier for plane 1
13-0 BF_0	Segment 3: Deadzone rate multiplier for plane 0

15.3.2.1.232 VPU H1 Register 290 (SWREG290)

15.3.2.1.232.1 Offset

Register	Offset
SWREG290	488h

15.3.2.1.232.2 Diagram



15.3.2.1.232.3 Fields

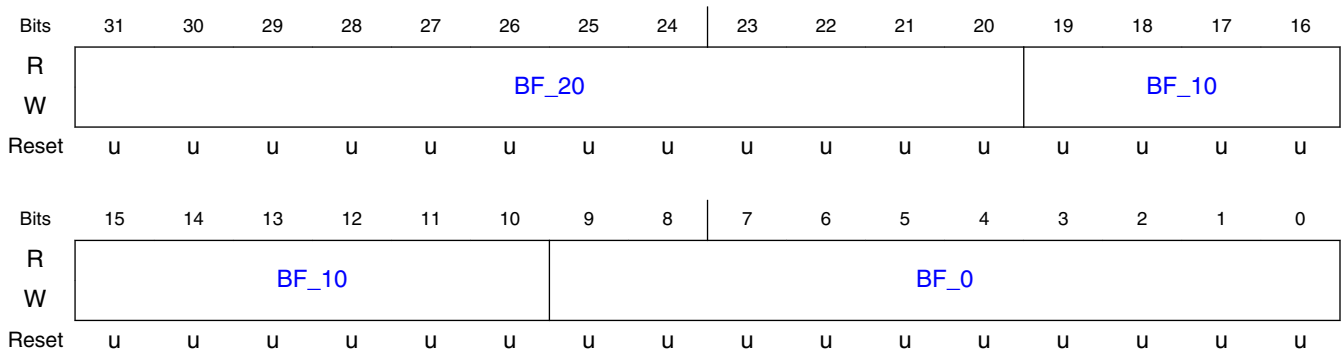
Field	Function
31-28 —	Reserved.
27-14 BF_14	Segment 3: Deadzone rate multiplier for plane 3
13-0 BF_0	Segment 3: Deadzone rate multiplier for plane 2

15.3.2.1.233 VPU H1 Register 291 (SWREG291)

15.3.2.1.233.1 Offset

Register	Offset
SWREG291	48Ch

15.3.2.1.233.2 Diagram



15.3.2.1.233.3 Fields

Field	Function
31-20 BF_20	Segment 3: Coeff for dmv penalty for intra/inter selection
19-10 BF_10	Segment 3: Deadzone rate for macroblock skip token 1
9-0 BF_0	Segment 3: Deadzone rate for macroblock skip token 0

15.3.2.1.234 VPU H1 Register 292 (SWREG292)

15.3.2.1.234.1 Offset

Register	Offset
SWREG292	490h

15.3.2.1.234.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.234.3 Fields

Field	Function
31-0 BF_0	VP8 average variance in prev frame

15.3.2.1.235 VPU H1 Register 293 (SWREG293)

15.3.2.1.235.1 Offset

Register	Offset
SWREG293	494h

15.3.2.1.235.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved												BF_18		BF_17	BF_16
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.235.3 Fields

Field	Function
31-20 —	Reserved.
19-18 BF_18	H.264 Field parity conditions for chroma MV. 0=same parity. 1=top/bottom. 2=bottom/top.
17 BF_17	H.264 BottomFieldFlag. 0=top. 1=bottom field.
16 BF_16	H.264 FieldPicFlag. 0=frame. 1=field.
15-0 BF_0	VP8 16384/avgVar in prev frame

15.3.2.1.236 VPU H1 Register 294 (SWREG294)

15.3.2.1.236.1 Offset

Register	Offset
SWREG294	498h

15.3.2.1.236.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_19													BF_6		
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_6										BF_0					
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.236.3 Fields

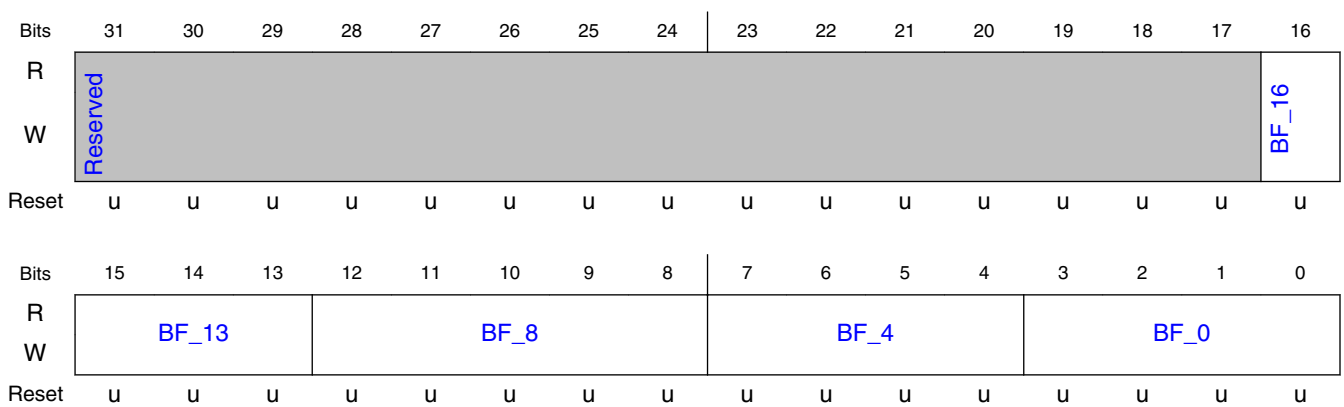
Field	Function
31-19 BF_19	Mb boost variance2
18-6 BF_6	Mb boost variance1
5-0 BF_0	Mb boost qp

15.3.2.1.237 VPU H1 Register 295 (SWREG295)

15.3.2.1.237.1 Offset

Register	Offset
SWREG295	49Ch

15.3.2.1.237.2 Diagram



15.3.2.1.237.3 Fields

Field	Function
31-17 —	Reserved.
16 BF_16	Pskip coding mode

Table continues on the next page...

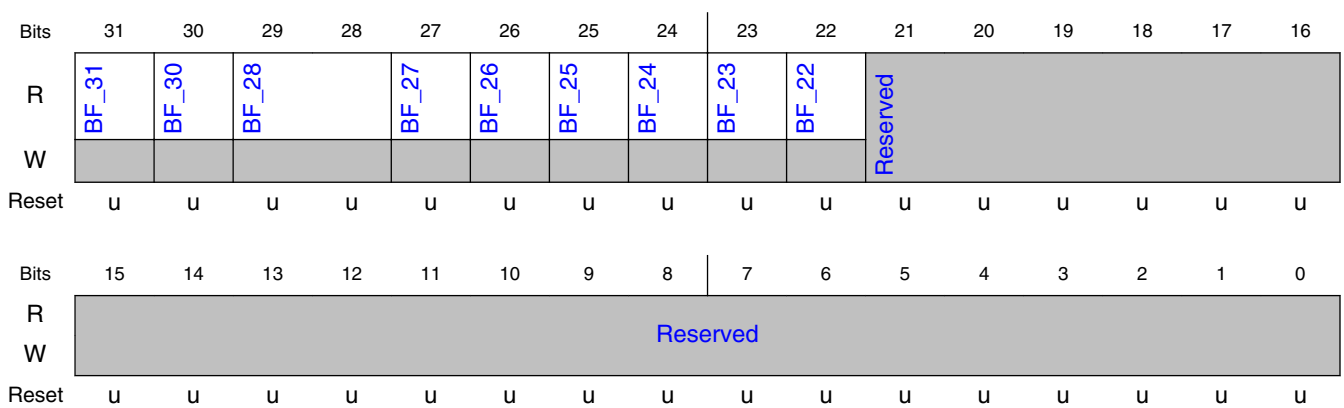
Field	Function
15-13 BF_13	Variance add
12-8 BF_8	Variance multiplier
7-4 BF_4	Variance interFavor
3-0 BF_0	Variance limit

15.3.2.1.238 VPU H1 Register 296 (SWREG296)

15.3.2.1.238.1 Offset

Register	Offset
SWREG296	4A0h

15.3.2.1.238.2 Diagram



15.3.2.1.238.3 Fields

Field	Function
31 BF_31	HW Address bit width. 0=32 bits. 1=64 bits.
30	HW Denoise Support.

Table continues on the next page...

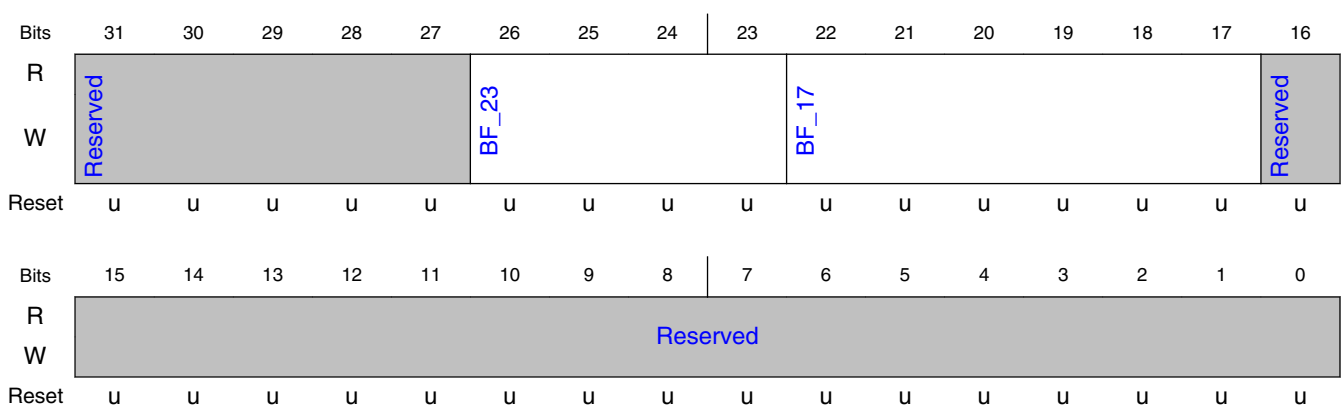
Field	Function
BF_30	
29-28 BF_28	HW Reference Compression Support for Luma and Chroma.
27 BF_27	HW Enhancement for Quality.
26 BF_26	HW Low Latency Control.
25 BF_25	HW SVCT Support.
24 BF_24	HW AXI Read ID for Input Support.
23 BF_23	HW IRQ is clear in write-one way.
22 BF_22	HW Input Loopback Support. not support now
21-0 —	Reserved.

15.3.2.1.239 VPU H1 Register 297 (SWREG297)

15.3.2.1.239.1 Offset

Register	Offset
SWREG297	4A4h

15.3.2.1.239.2 Diagram



15.3.2.1.239.3 Fields

Field	Function
31-27 —	Reserved.
26-23 BF_23	Segment 4: Intra 16x16 mode 2 penalty
22-17 BF_17	Segment 4: Intra 16x16 mode 1 penalty
16-0 —	Reserved.

15.3.2.1.240 VPU H1 Register 298 (SWREG298)

15.3.2.1.240.1 Offset

Register	Offset
SWREG298	4A8h

15.3.2.1.240.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.240.3 Fields

Field	Function
31-24 BF_24	Segment 4: Intra 4x4 previous mode favor for H.264

Table continues on the next page...

Field	Function
23-8 BF_8	Segment 4: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 4: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.241 VPU H1 Register 299 (SWREG299)

15.3.2.1.241.1 Offset

Register	Offset
SWREG299	4ACh

15.3.2.1.241.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.241.3 Fields

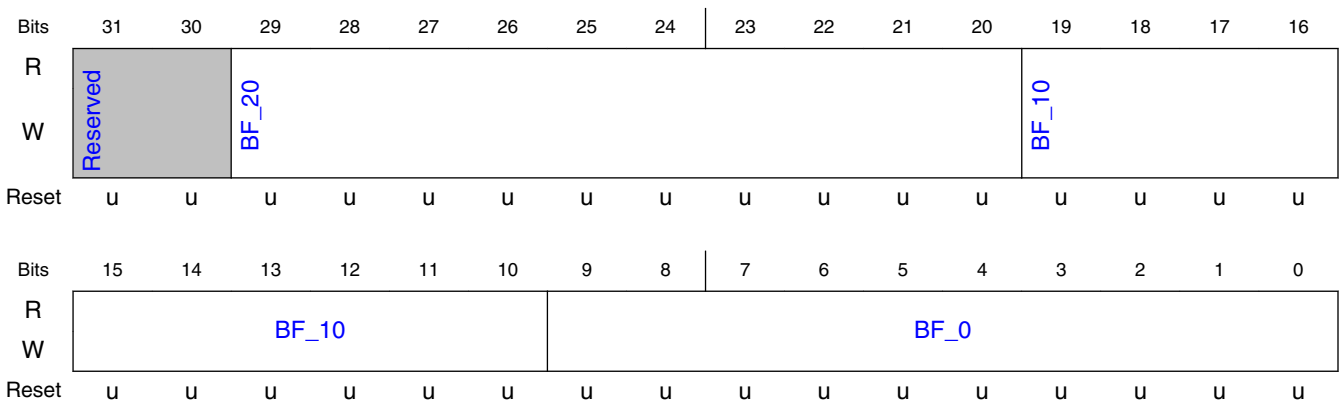
Field	Function
31-24 BF_24	Segment 4: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 4: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.242 VPU H1 Register 300 (SWREG300)

15.3.2.1.242.1 Offset

Register	Offset
SWREG300	4B0h

15.3.2.1.242.2 Diagram



15.3.2.1.242.3 Fields

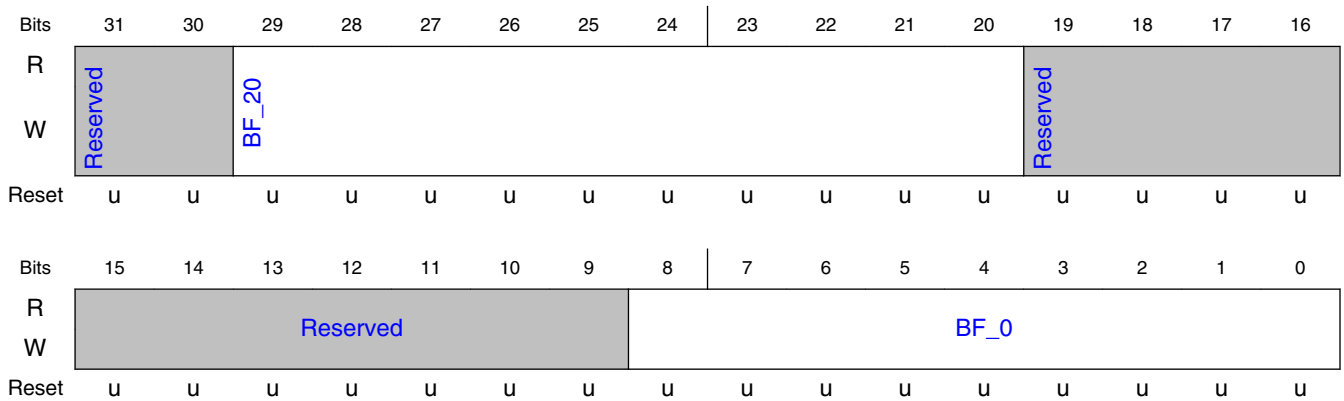
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 4: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 4: Penalty for using 8x8 MV.
9-0 BF_0	Segment 4: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.243 VPU H1 Register 301 (SWREG301)

15.3.2.1.243.1 Offset

Register	Offset
SWREG301	4B4h

15.3.2.1.243.2 Diagram



15.3.2.1.243.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 4: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 4: Penalty for using 4x4 MV.

15.3.2.1.244 VPU H1 Register 302 (SWREG302)

15.3.2.1.244.1 Offset

Register	Offset
SWREG302	4B8h

15.3.2.1.244.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.244.3 Fields

Field	Function
31-24 BF_24	Segment 5: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 5: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 5: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.245 VPU H1 Register 303 (SWREG303)

15.3.2.1.245.1 Offset

Register	Offset
SWREG303	4BCh

15.3.2.1.245.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.245.3 Fields

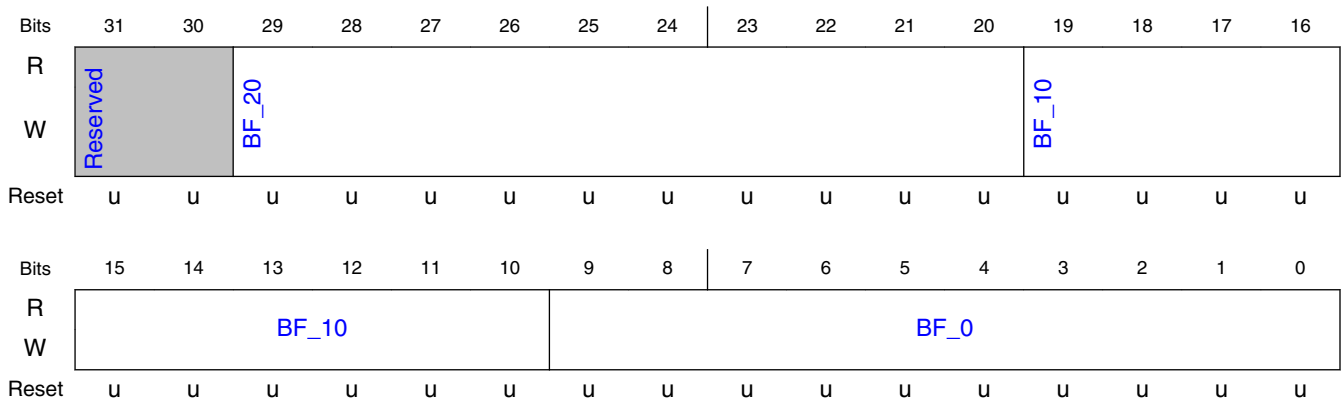
Field	Function
31-24 BF_24	Segment 5: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 5: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.246 VPU H1 Register 304 (SWREG304)

15.3.2.1.246.1 Offset

Register	Offset
SWREG304	4C0h

15.3.2.1.246.2 Diagram



15.3.2.1.246.3 Fields

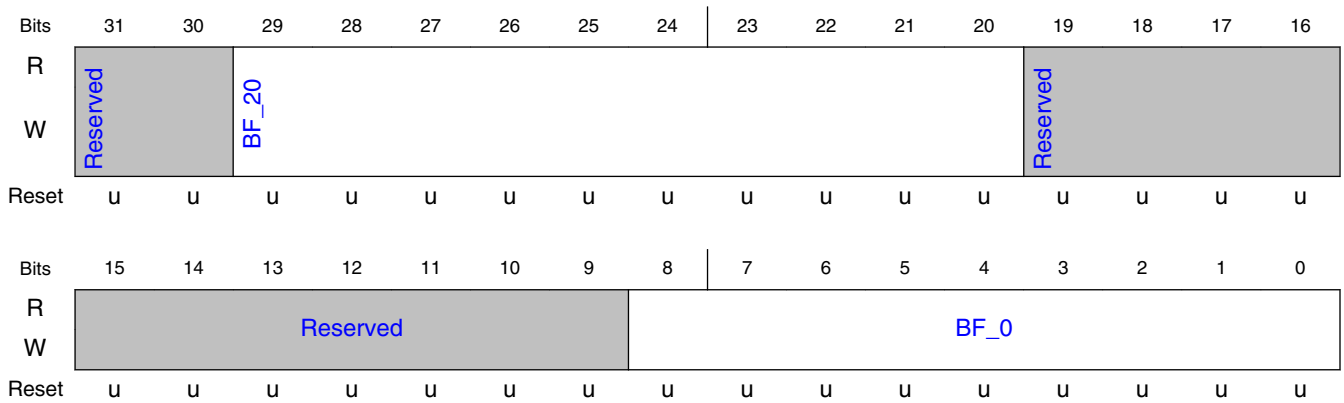
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 5: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 5: Penalty for using 8x8 MV.
9-0 BF_0	Segment 5: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.247 VPU H1 Register 305 (SWREG305)

15.3.2.1.247.1 Offset

Register	Offset
SWREG305	4C4h

15.3.2.1.247.2 Diagram



15.3.2.1.247.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 5: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 5: Penalty for using 4x4 MV.

15.3.2.1.248 VPU H1 Register 306 (SWREG306)

15.3.2.1.248.1 Offset

Register	Offset
SWREG306	4C8h

15.3.2.1.248.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.248.3 Fields

Field	Function
31-24 BF_24	Segment 6: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 6: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 6: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.249 VPU H1 Register 307 (SWREG307)

15.3.2.1.249.1 Offset

Register	Offset
SWREG307	4CCh

15.3.2.1.249.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.249.3 Fields

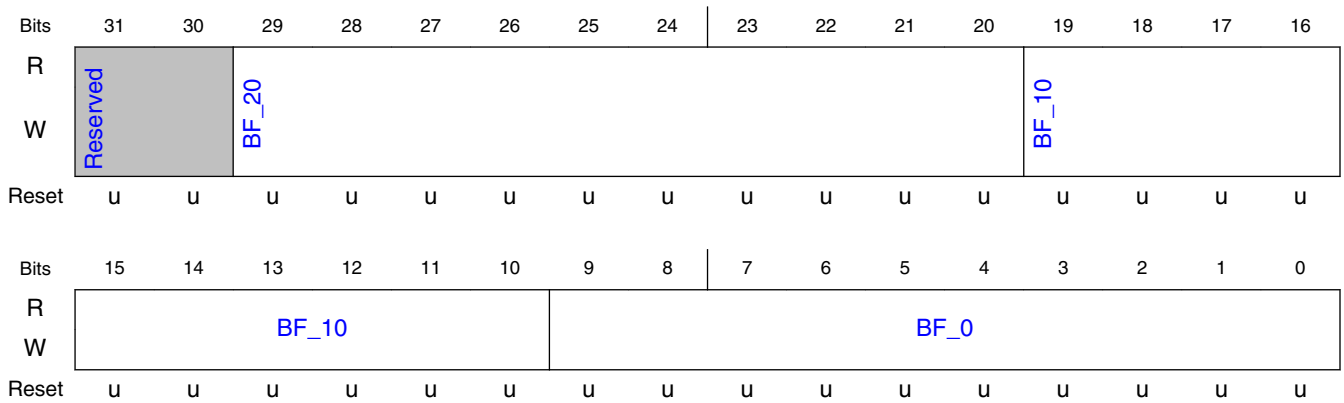
Field	Function
31-24 BF_24	Segment 6: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 6: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.250 VPU H1 Register 308 (SWREG308)

15.3.2.1.250.1 Offset

Register	Offset
SWREG308	4D0h

15.3.2.1.250.2 Diagram



15.3.2.1.250.3 Fields

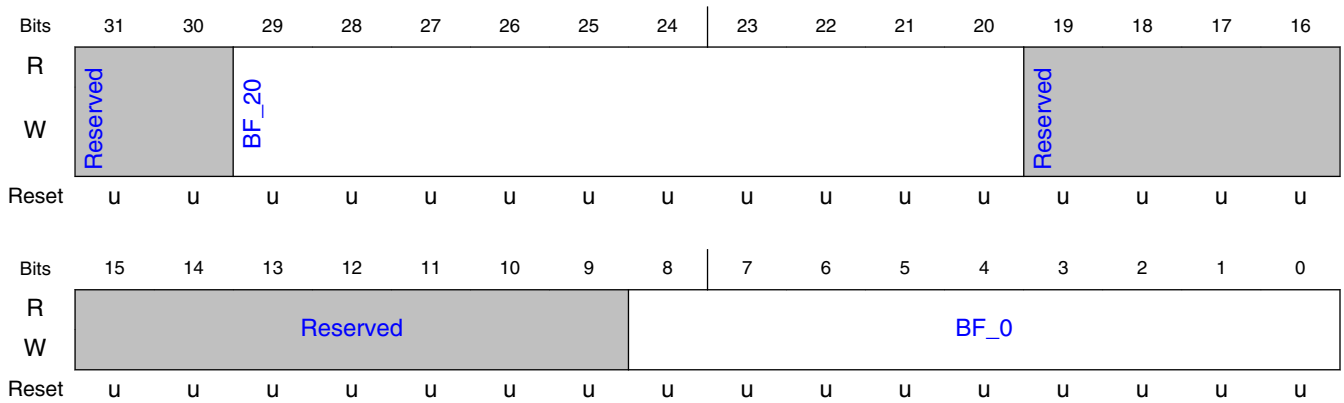
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 6: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 6: Penalty for using 8x8 MV.
9-0 BF_0	Segment 6: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.251 VPU H1 Register 309 (SWREG309)

15.3.2.1.251.1 Offset

Register	Offset
SWREG309	4D4h

15.3.2.1.251.2 Diagram



15.3.2.1.251.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 6: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 6: Penalty for using 4x4 MV.

15.3.2.1.252 VPU H1 Register 310 (SWREG310)

15.3.2.1.252.1 Offset

Register	Offset
SWREG310	4D8h

15.3.2.1.252.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.252.3 Fields

Field	Function
31-24 BF_24	Segment 7: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 7: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 7: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.253 VPU H1 Register 311 (SWREG311)

15.3.2.1.253.1 Offset

Register	Offset
SWREG311	4DCh

15.3.2.1.253.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.253.3 Fields

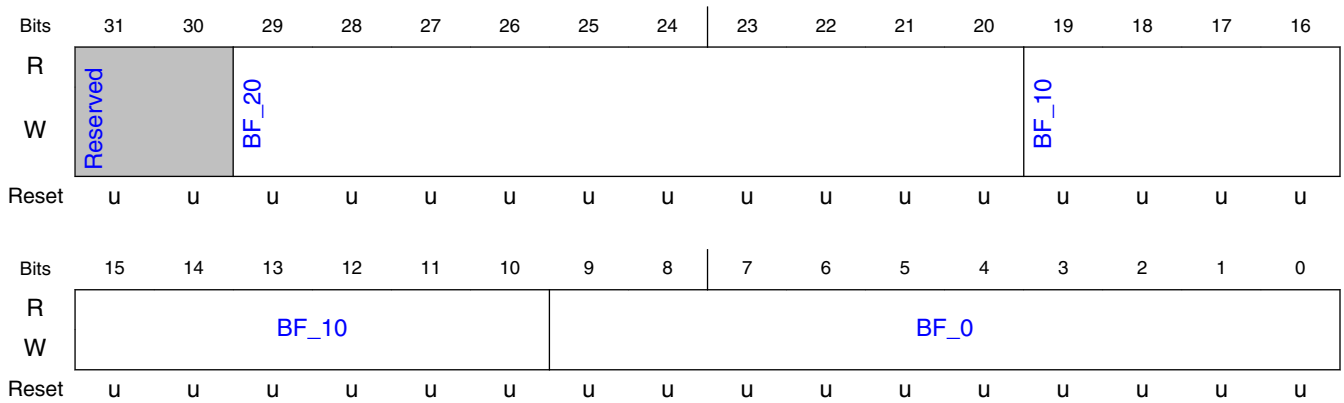
Field	Function
31-24 BF_24	Segment 7: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 7: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.254 VPU H1 Register 312 (SWREG312)

15.3.2.1.254.1 Offset

Register	Offset
SWREG312	4E0h

15.3.2.1.254.2 Diagram



15.3.2.1.254.3 Fields

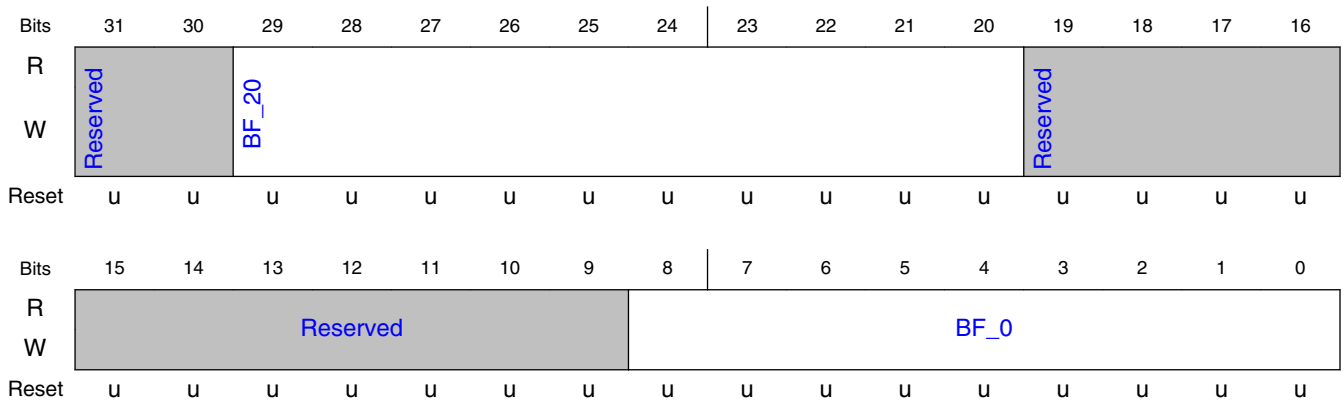
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 7: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 7: Penalty for using 8x8 MV.
9-0 BF_0	Segment 7: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.255 VPU H1 Register 313 (SWREG313)

15.3.2.1.255.1 Offset

Register	Offset
SWREG313	4E4h

15.3.2.1.255.2 Diagram



15.3.2.1.255.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 7: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 7: Penalty for using 4x4 MV.

15.3.2.1.256 VPU H1 Register 314 (SWREG314)

15.3.2.1.256.1 Offset

Register	Offset
SWREG314	4E8h

15.3.2.1.256.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.256.3 Fields

Field	Function
31-24 BF_24	Segment 8: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 8: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 8: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.257 VPU H1 Register 315 (SWREG315)

15.3.2.1.257.1 Offset

Register	Offset
SWREG315	4ECh

15.3.2.1.257.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.257.3 Fields

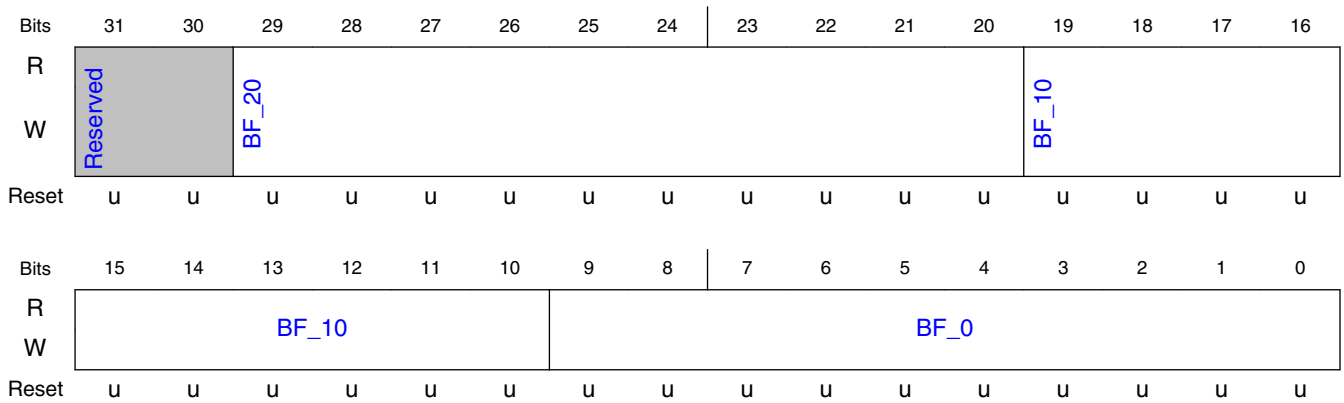
Field	Function
31-24 BF_24	Segment 8: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 8: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.258 VPU H1 Register 316 (SWREG316)

15.3.2.1.258.1 Offset

Register	Offset
SWREG316	4F0h

15.3.2.1.258.2 Diagram



15.3.2.1.258.3 Fields

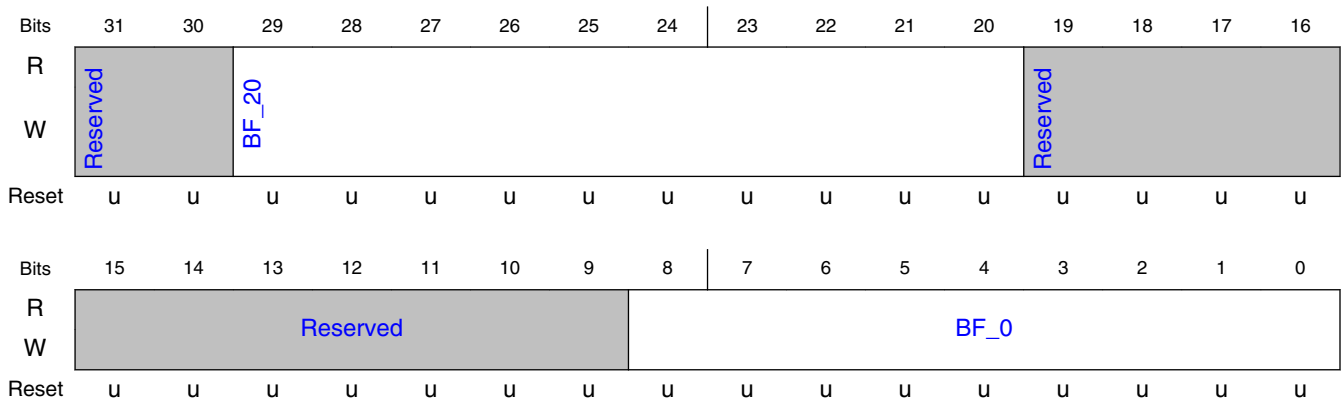
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 8: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 8: Penalty for using 8x8 MV.
9-0 BF_0	Segment 8: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.259 VPU H1 Register 317 (SWREG317)

15.3.2.1.259.1 Offset

Register	Offset
SWREG317	4F4h

15.3.2.1.259.2 Diagram



15.3.2.1.259.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 8: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 8: Penalty for using 4x4 MV.

15.3.2.1.260 VPU H1 Register 318 (SWREG318)

15.3.2.1.260.1 Offset

Register	Offset
SWREG318	4F8h

15.3.2.1.260.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.260.3 Fields

Field	Function
31-24 BF_24	Segment 9: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 9: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 9: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.261 VPU H1 Register 319 (SWREG319)

15.3.2.1.261.1 Offset

Register	Offset
SWREG319	4FCh

15.3.2.1.261.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.261.3 Fields

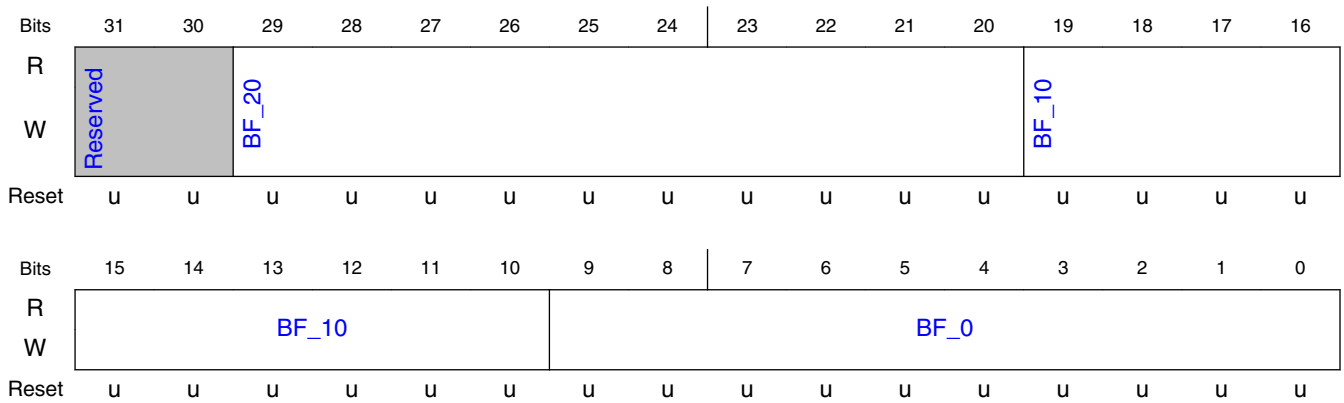
Field	Function
31-24 BF_24	Segment 9: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 9: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.262 VPU H1 Register 320 (SWREG320)

15.3.2.1.262.1 Offset

Register	Offset
SWREG320	500h

15.3.2.1.262.2 Diagram



15.3.2.1.262.3 Fields

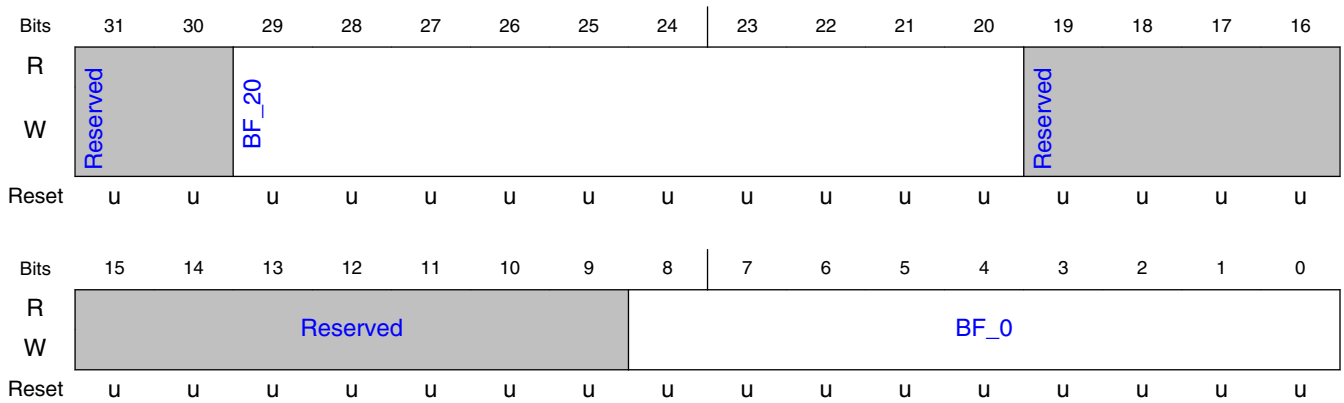
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 9: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 9: Penalty for using 8x8 MV.
9-0 BF_0	Segment 9: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.263 VPU H1 Register 321 (SWREG321)

15.3.2.1.263.1 Offset

Register	Offset
SWREG321	504h

15.3.2.1.263.2 Diagram



15.3.2.1.263.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 9: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 9: Penalty for using 4x4 MV.

15.3.2.1.264 VPU H1 Register 322 (SWREG322)

15.3.2.1.264.1 Offset

Register	Offset
SWREG322	508h

15.3.2.1.264.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.264.3 Fields

Field	Function
31-24 BF_24	Segment 10: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 10: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 10: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.265 VPU H1 Register 323 (SWREG323)

15.3.2.1.265.1 Offset

Register	Offset
SWREG323	50Ch

15.3.2.1.265.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.265.3 Fields

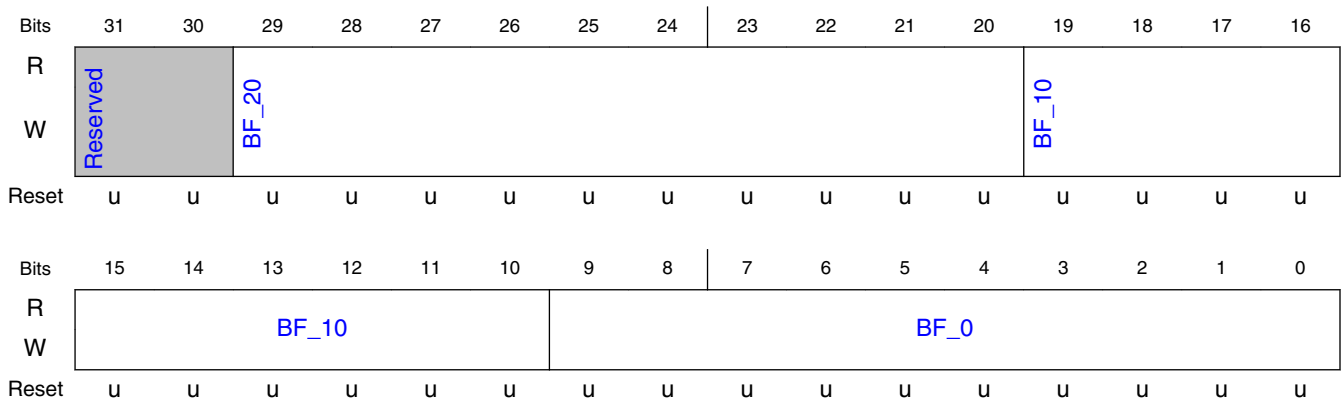
Field	Function
31-24 BF_24	Segment 10: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 10: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.266 VPU H1 Register 324 (SWREG324)

15.3.2.1.266.1 Offset

Register	Offset
SWREG324	510h

15.3.2.1.266.2 Diagram



15.3.2.1.266.3 Fields

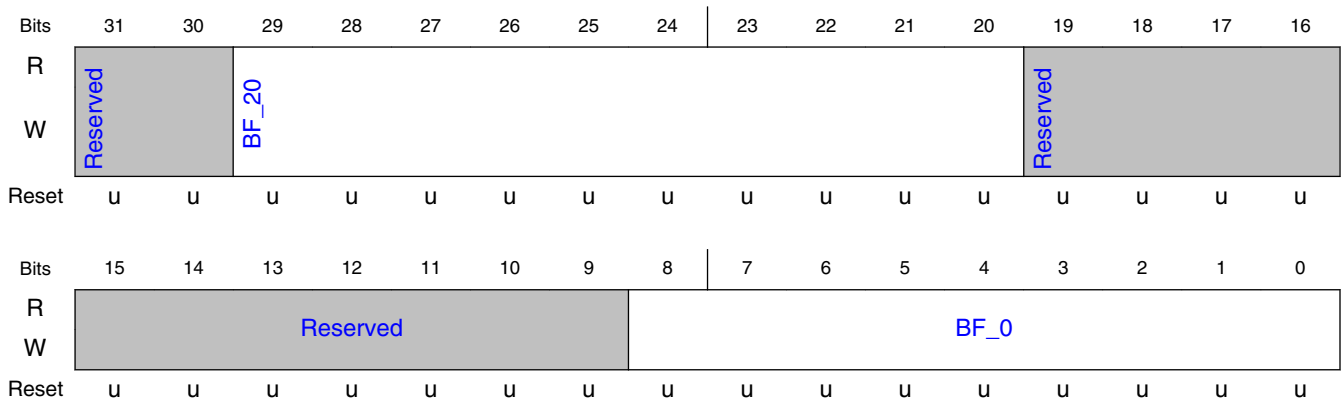
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 10: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 10: Penalty for using 8x8 MV.
9-0 BF_0	Segment 10: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.267 VPU H1 Register 325 (SWREG325)

15.3.2.1.267.1 Offset

Register	Offset
SWREG325	514h

15.3.2.1.267.2 Diagram



15.3.2.1.267.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 10: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 10: Penalty for using 4x4 MV.

15.3.2.1.268 VPU H1 Register 326 (SWREG326)

15.3.2.1.268.1 Offset

Register	Offset
SWREG326	518h

15.3.2.1.268.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.268.3 Fields

Field	Function
31-24 BF_24	Segment 11: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 11: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 11: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.269 VPU H1 Register 327 (SWREG327)

15.3.2.1.269.1 Offset

Register	Offset
SWREG327	51Ch

15.3.2.1.269.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.269.3 Fields

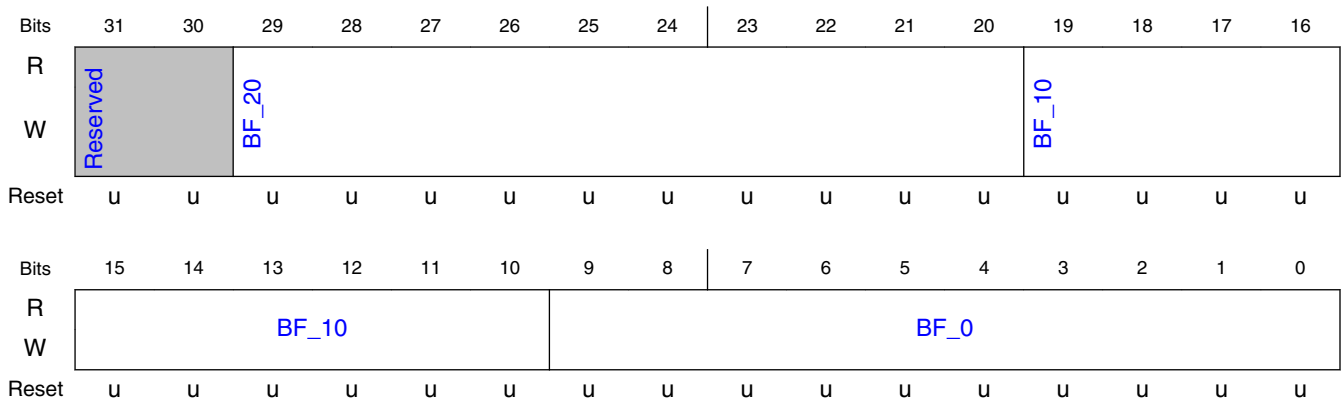
Field	Function
31-24 BF_24	Segment 11: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 11: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.270 VPU H1 Register 328 (SWREG328)

15.3.2.1.270.1 Offset

Register	Offset
SWREG328	520h

15.3.2.1.270.2 Diagram



15.3.2.1.270.3 Fields

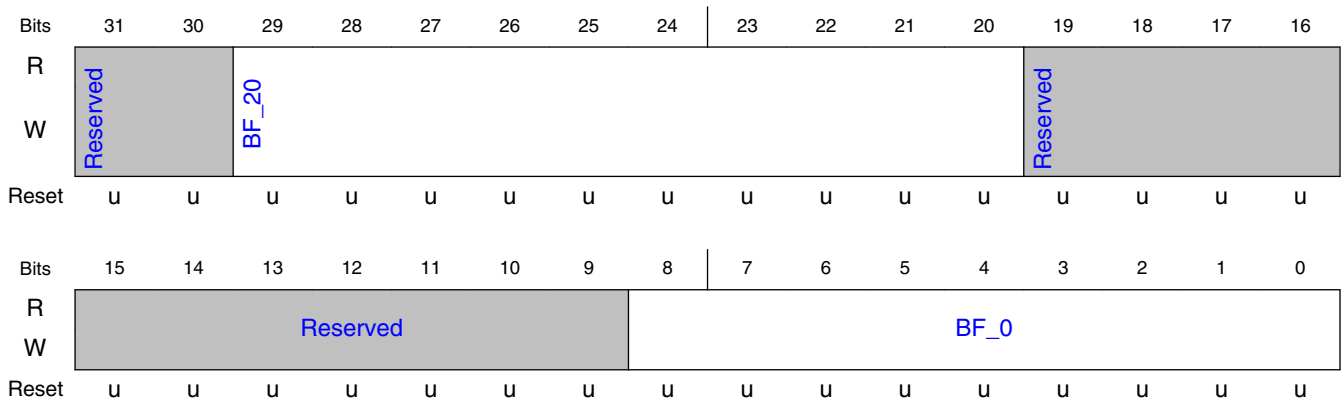
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 11: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 11: Penalty for using 8x8 MV.
9-0 BF_0	Segment 11: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.271 VPU H1 Register 329 (SWREG329)

15.3.2.1.271.1 Offset

Register	Offset
SWREG329	524h

15.3.2.1.271.2 Diagram



15.3.2.1.271.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 11: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 11: Penalty for using 4x4 MV.

15.3.2.1.272 VPU H1 Register 330 (SWREG330)

15.3.2.1.272.1 Offset

Register	Offset
SWREG330	528h

15.3.2.1.272.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.272.3 Fields

Field	Function
31-24 BF_24	Segment 12: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 12: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 12: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.273 VPU H1 Register 331 (SWREG331)

15.3.2.1.273.1 Offset

Register	Offset
SWREG331	52Ch

15.3.2.1.273.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.273.3 Fields

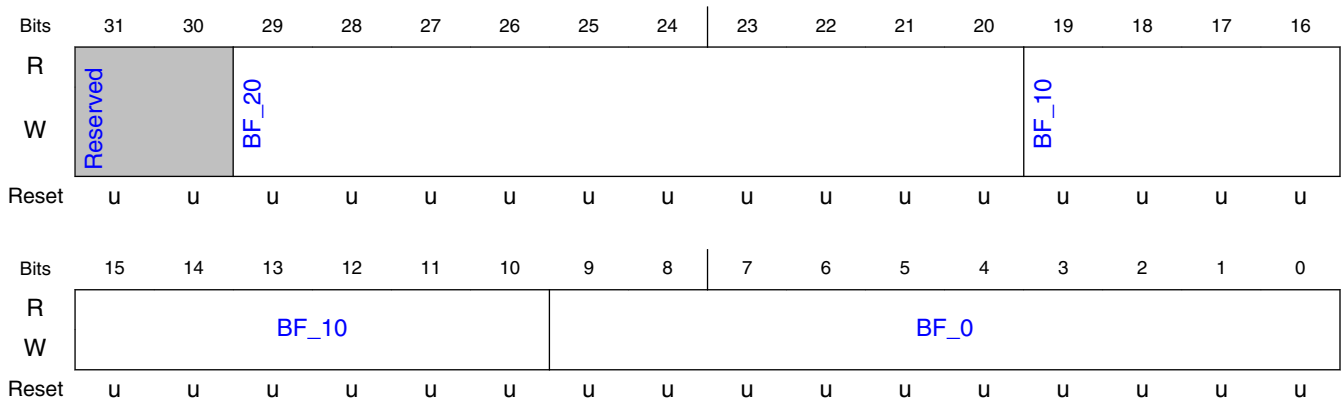
Field	Function
31-24 BF_24	Segment 12: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 12: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.274 VPU H1 Register 332 (SWREG332)

15.3.2.1.274.1 Offset

Register	Offset
SWREG332	530h

15.3.2.1.274.2 Diagram



15.3.2.1.274.3 Fields

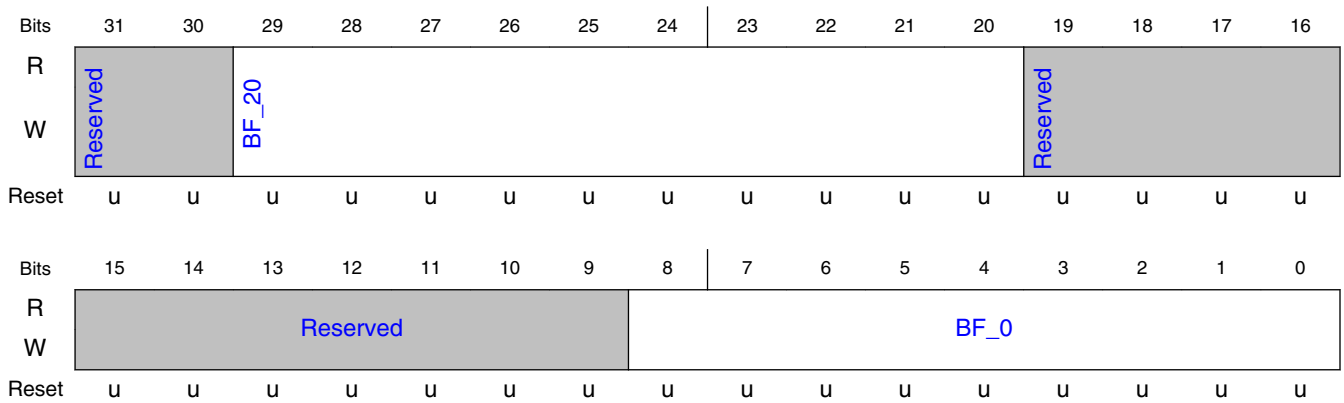
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 12: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 12: Penalty for using 8x8 MV.
9-0 BF_0	Segment 12: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.275 VPU H1 Register 333 (SWREG333)

15.3.2.1.275.1 Offset

Register	Offset
SWREG333	534h

15.3.2.1.275.2 Diagram



15.3.2.1.275.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 12: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 12: Penalty for using 4x4 MV.

15.3.2.1.276 VPU H1 Register 334 (SWREG334)

15.3.2.1.276.1 Offset

Register	Offset
SWREG334	538h

15.3.2.1.276.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.276.3 Fields

Field	Function
31-24 BF_24	Segment 13: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 13: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 13: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.277 VPU H1 Register 335 (SWREG335)

15.3.2.1.277.1 Offset

Register	Offset
SWREG335	53Ch

15.3.2.1.277.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.277.3 Fields

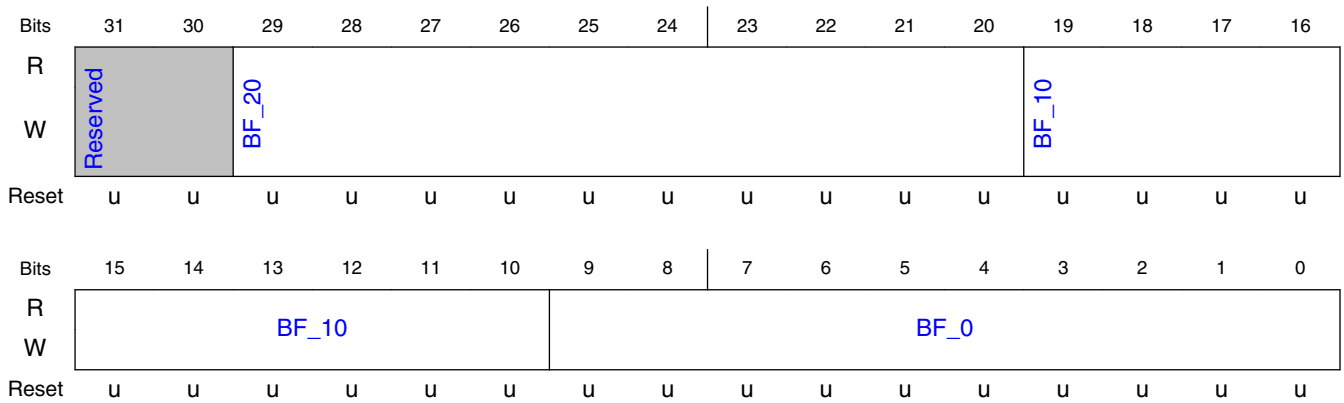
Field	Function
31-24 BF_24	Segment 13: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 13: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.278 VPU H1 Register 336 (SWREG336)

15.3.2.1.278.1 Offset

Register	Offset
SWREG336	540h

15.3.2.1.278.2 Diagram



15.3.2.1.278.3 Fields

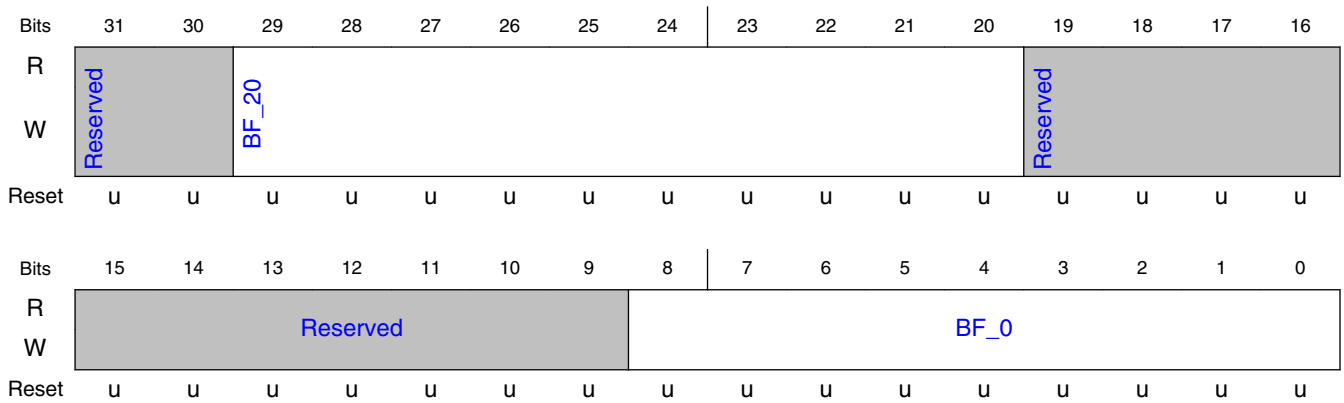
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 13: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 13: Penalty for using 8x8 MV.
9-0 BF_0	Segment 13: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.279 VPU H1 Register 337 (SWREG337)

15.3.2.1.279.1 Offset

Register	Offset
SWREG337	544h

15.3.2.1.279.2 Diagram



15.3.2.1.279.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 13: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 13: Penalty for using 4x4 MV.

15.3.2.1.280 VPU H1 Register 338 (SWREG338)

15.3.2.1.280.1 Offset

Register	Offset
SWREG338	548h

15.3.2.1.280.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.280.3 Fields

Field	Function
31-24 BF_24	Segment 14: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 14: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 14: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.281 VPU H1 Register 339 (SWREG339)

15.3.2.1.281.1 Offset

Register	Offset
SWREG339	54Ch

15.3.2.1.281.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.281.3 Fields

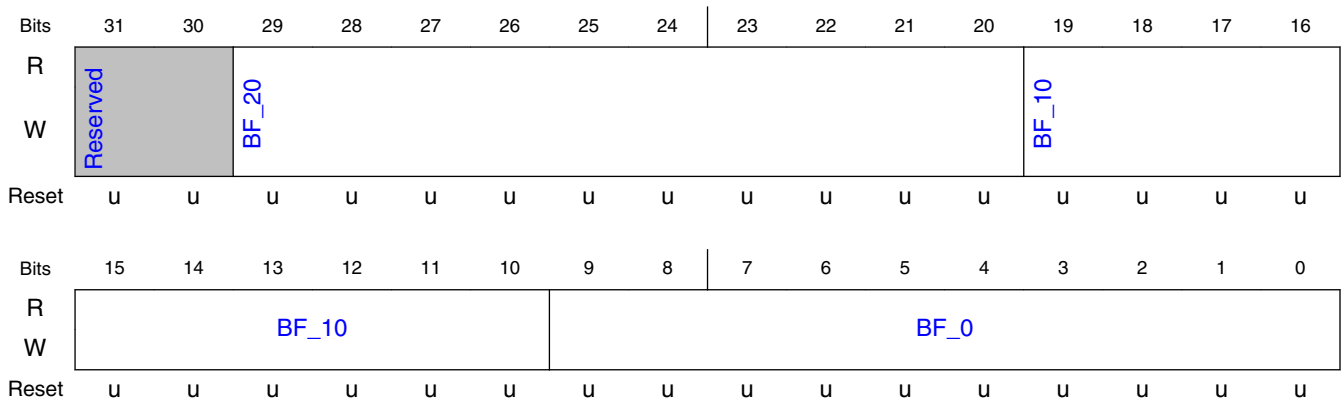
Field	Function
31-24 BF_24	Segment 14: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 14: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.282 VPU H1 Register 340 (SWREG340)

15.3.2.1.282.1 Offset

Register	Offset
SWREG340	550h

15.3.2.1.282.2 Diagram



15.3.2.1.282.3 Fields

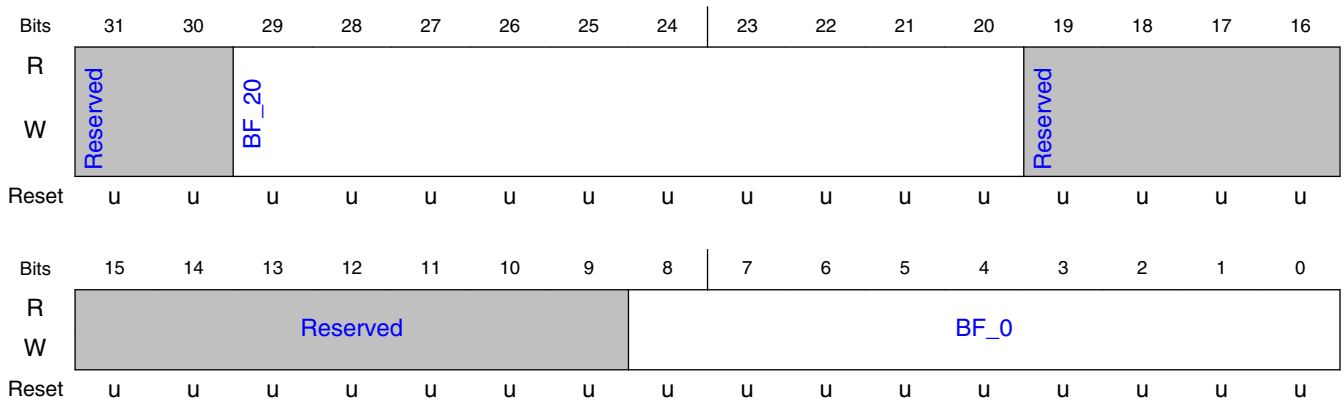
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 14: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 14: Penalty for using 8x8 MV.
9-0 BF_0	Segment 14: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.283 VPU H1 Register 341 (SWREG341)

15.3.2.1.283.1 Offset

Register	Offset
SWREG341	554h

15.3.2.1.283.2 Diagram



15.3.2.1.283.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 14: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 14: Penalty for using 4x4 MV.

15.3.2.1.284 VPU H1 Register 342 (SWREG342)

15.3.2.1.284.1 Offset

Register	Offset
SWREG342	558h

15.3.2.1.284.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.284.3 Fields

Field	Function
31-24 BF_24	Segment 15: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 15: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 15: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.285 VPU H1 Register 343 (SWREG343)

15.3.2.1.285.1 Offset

Register	Offset
SWREG343	55Ch

15.3.2.1.285.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.285.3 Fields

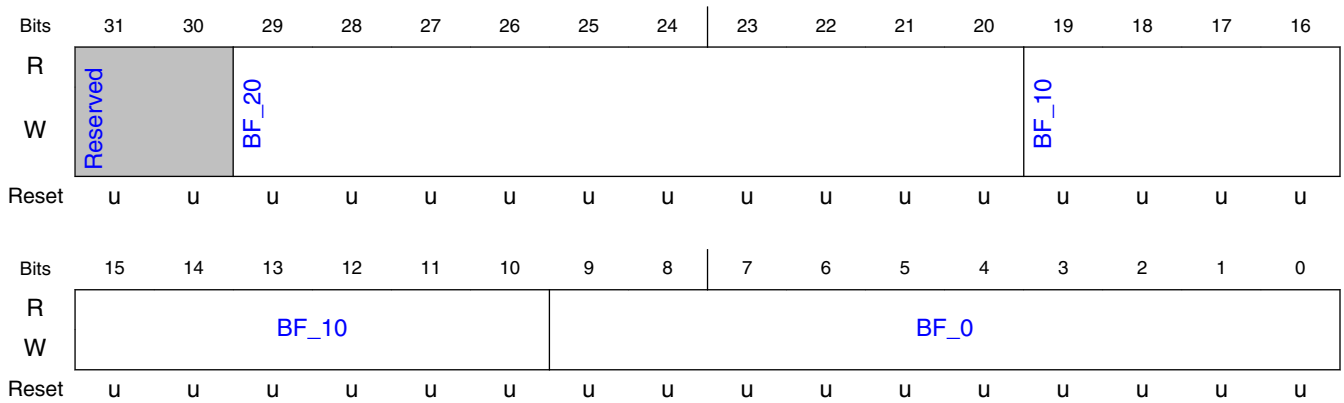
Field	Function
31-24 BF_24	Segment 15: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 15: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.286 VPU H1 Register 344 (SWREG344)

15.3.2.1.286.1 Offset

Register	Offset
SWREG344	560h

15.3.2.1.286.2 Diagram



15.3.2.1.286.3 Fields

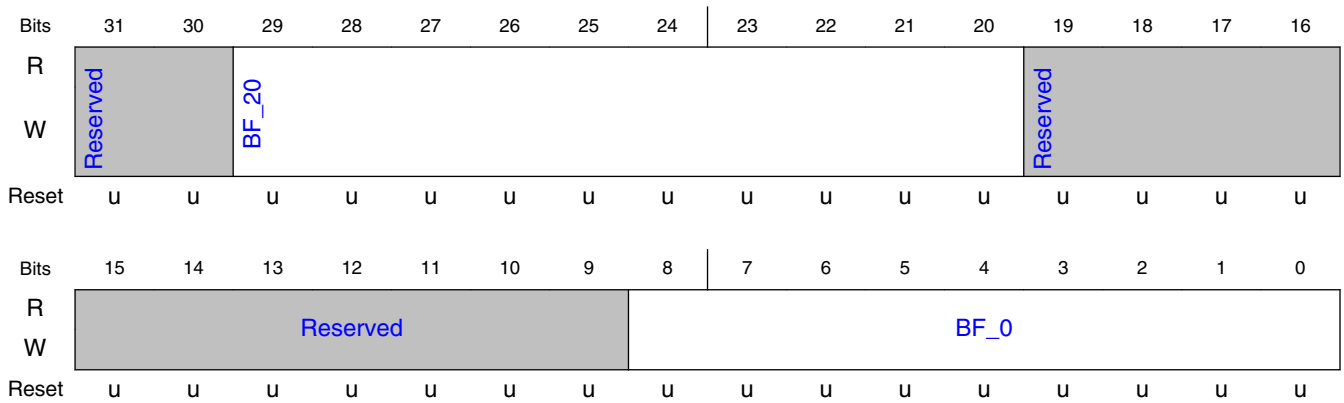
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 15: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 15: Penalty for using 8x8 MV.
9-0 BF_0	Segment 15: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.287 VPU H1 Register 345 (SWREG345)

15.3.2.1.287.1 Offset

Register	Offset
SWREG345	564h

15.3.2.1.287.2 Diagram



15.3.2.1.287.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 15: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 15: Penalty for using 4x4 MV.

15.3.2.1.288 VPU H1 Register 346 (SWREG346)

15.3.2.1.288.1 Offset

Register	Offset
SWREG346	568h

15.3.2.1.288.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.288.3 Fields

Field	Function
31-24 BF_24	Segment 16: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 16: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 16: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.289 VPU H1 Register 347 (SWREG347)

15.3.2.1.289.1 Offset

Register	Offset
SWREG347	56Ch

15.3.2.1.289.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.289.3 Fields

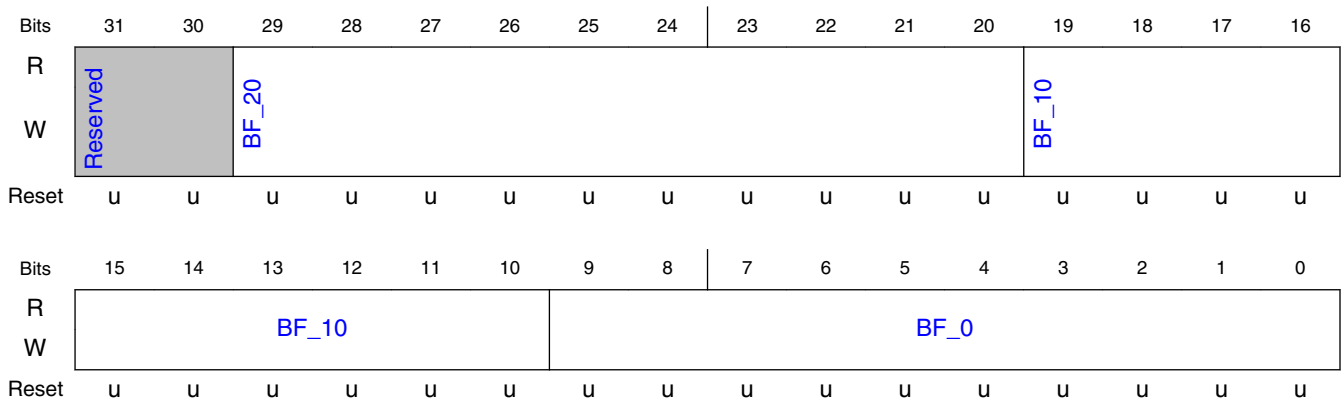
Field	Function
31-24 BF_24	Segment 16: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 16: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.290 VPU H1 Register 348 (SWREG348)

15.3.2.1.290.1 Offset

Register	Offset
SWREG348	570h

15.3.2.1.290.2 Diagram



15.3.2.1.290.3 Fields

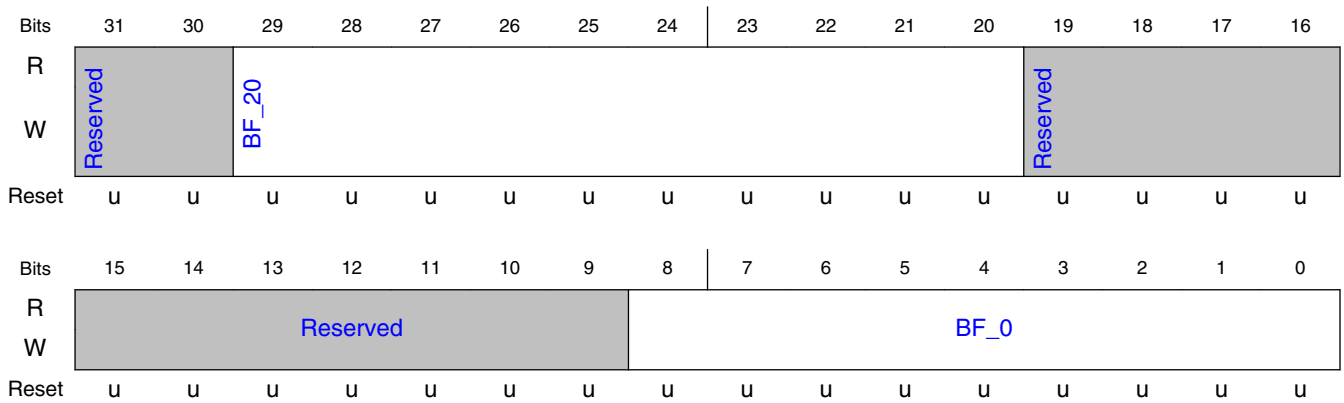
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 16: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 16: Penalty for using 8x8 MV.
9-0 BF_0	Segment 16: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.291 VPU H1 Register 349 (SWREG349)

15.3.2.1.291.1 Offset

Register	Offset
SWREG349	574h

15.3.2.1.291.2 Diagram



15.3.2.1.291.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 16: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 16: Penalty for using 4x4 MV.

15.3.2.1.292 VPU H1 Register 350 (SWREG350)

15.3.2.1.292.1 Offset

Register	Offset
SWREG350	578h

15.3.2.1.292.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.292.3 Fields

Field	Function
31-24 BF_24	Segment 17: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 17: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 17: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.293 VPU H1 Register 351 (SWREG351)

15.3.2.1.293.1 Offset

Register	Offset
SWREG351	57Ch

15.3.2.1.293.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.293.3 Fields

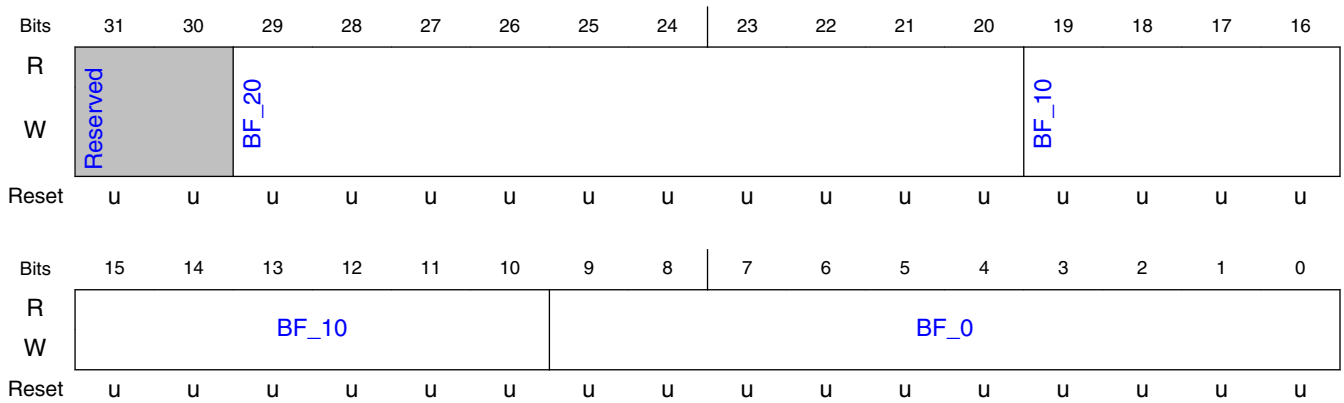
Field	Function
31-24 BF_24	Segment 17: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 17: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.294 VPU H1 Register 352 (SWREG352)

15.3.2.1.294.1 Offset

Register	Offset
SWREG352	580h

15.3.2.1.294.2 Diagram



15.3.2.1.294.3 Fields

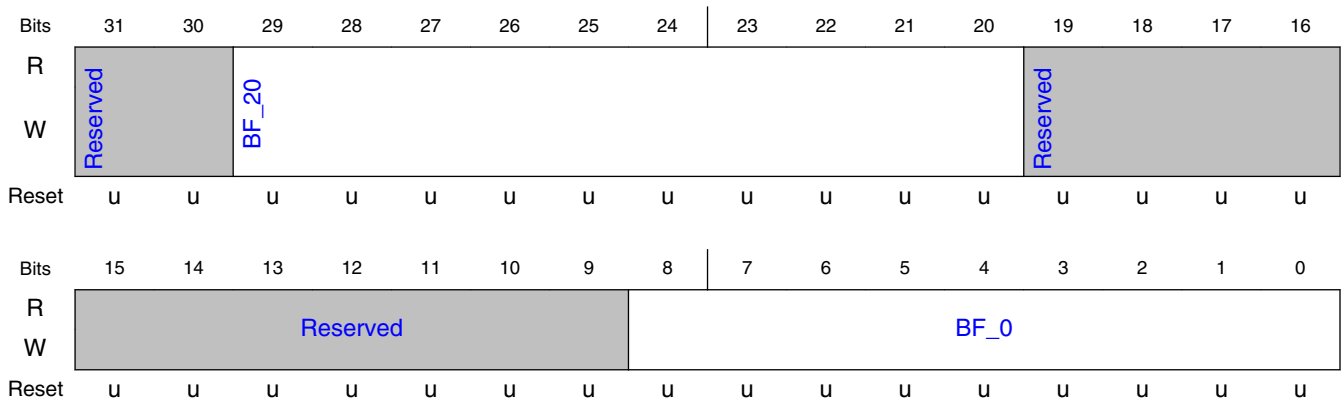
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 17: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 17: Penalty for using 8x8 MV.
9-0 BF_0	Segment 17: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.295 VPU H1 Register 353 (SWREG353)

15.3.2.1.295.1 Offset

Register	Offset
SWREG353	584h

15.3.2.1.295.2 Diagram



15.3.2.1.295.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 17: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 17: Penalty for using 4x4 MV.

15.3.2.1.296 VPU H1 Register 354 (SWREG354)

15.3.2.1.296.1 Offset

Register	Offset
SWREG354	588h

15.3.2.1.296.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.296.3 Fields

Field	Function
31-24 BF_24	Segment 18: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 18: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 18: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.297 VPU H1 Register 355 (SWREG355)

15.3.2.1.297.1 Offset

Register	Offset
SWREG355	58Ch

15.3.2.1.297.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.297.3 Fields

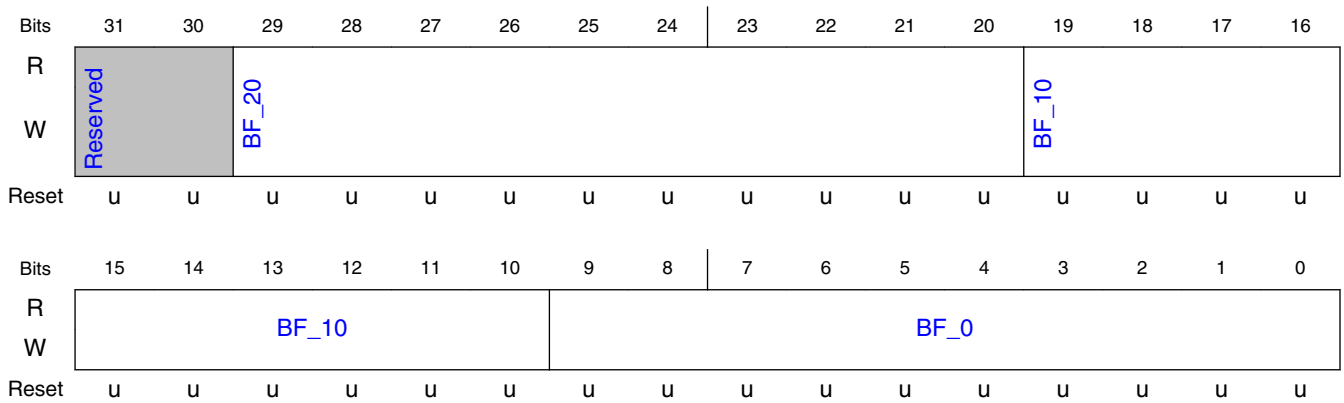
Field	Function
31-24 BF_24	Segment 18: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 18: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.298 VPU H1 Register 356 (SWREG356)

15.3.2.1.298.1 Offset

Register	Offset
SWREG356	590h

15.3.2.1.298.2 Diagram



15.3.2.1.298.3 Fields

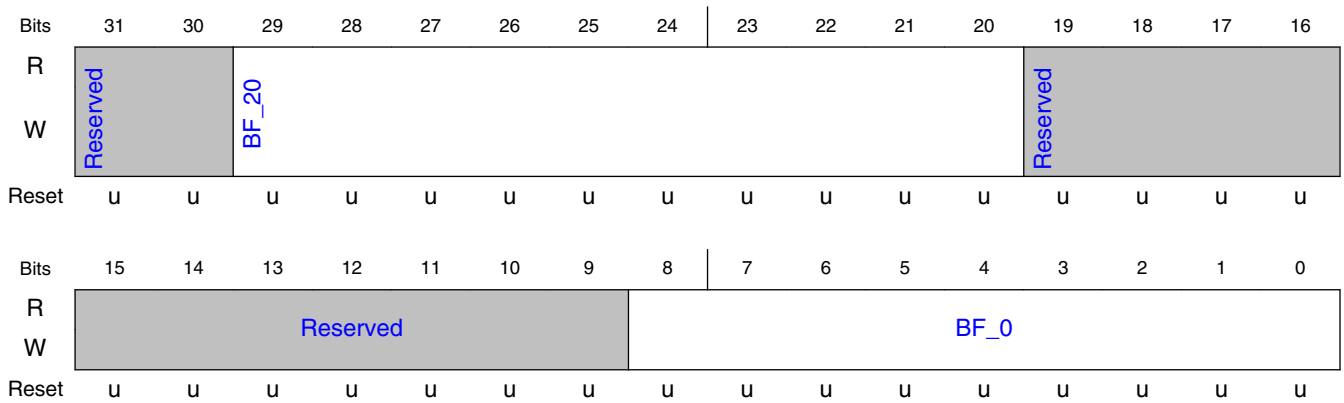
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 18: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 18: Penalty for using 8x8 MV.
9-0 BF_0	Segment 18: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.299 VPU H1 Register 357 (SWREG357)

15.3.2.1.299.1 Offset

Register	Offset
SWREG357	594h

15.3.2.1.299.2 Diagram



15.3.2.1.299.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 18: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 18: Penalty for using 4x4 MV.

15.3.2.1.300 VPU H1 Register 358 (SWREG358)

15.3.2.1.300.1 Offset

Register	Offset
SWREG358	598h

15.3.2.1.300.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.300.3 Fields

Field	Function
31-24 BF_24	Segment 19: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 19: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 19: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.301 VPU H1 Register 359 (SWREG359)

15.3.2.1.301.1 Offset

Register	Offset
SWREG359	59Ch

15.3.2.1.301.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.301.3 Fields

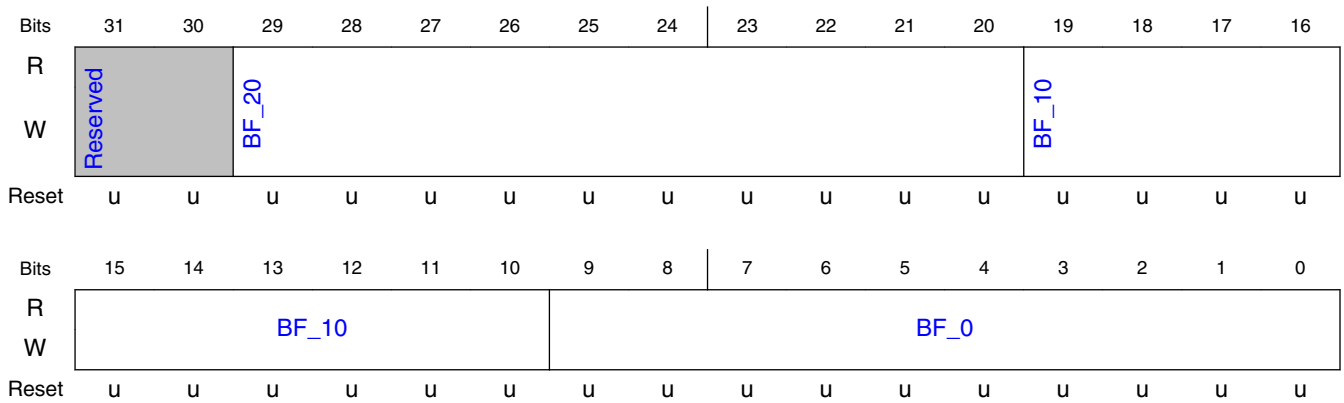
Field	Function
31-24 BF_24	Segment 19: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 19: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.302 VPU H1 Register 360 (SWREG360)

15.3.2.1.302.1 Offset

Register	Offset
SWREG360	5A0h

15.3.2.1.302.2 Diagram



15.3.2.1.302.3 Fields

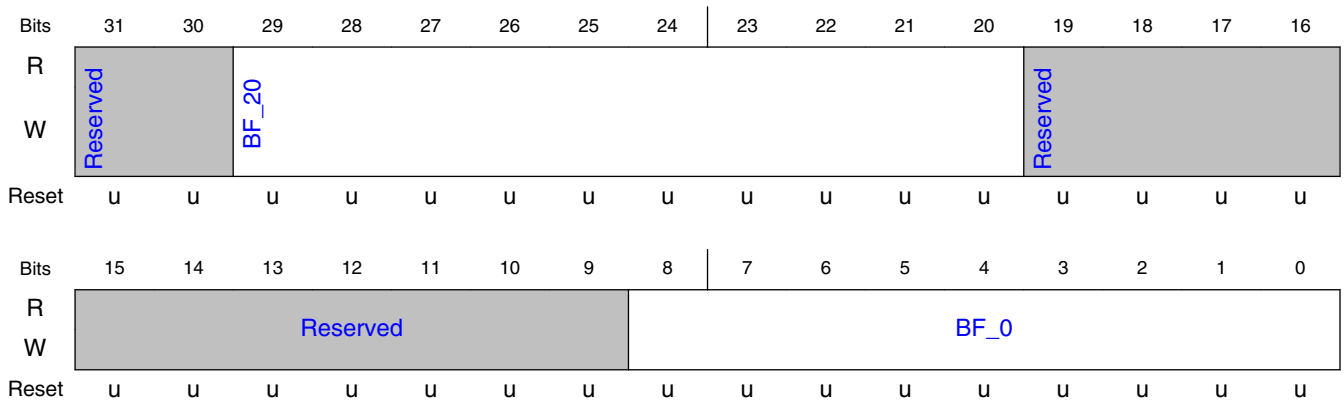
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 19: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 19: Penalty for using 8x8 MV.
9-0 BF_0	Segment 19: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.303 VPU H1 Register 361 (SWREG361)

15.3.2.1.303.1 Offset

Register	Offset
SWREG361	5A4h

15.3.2.1.303.2 Diagram



15.3.2.1.303.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 19: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 19: Penalty for using 4x4 MV.

15.3.2.1.304 VPU H1 Register 362 (SWREG362)

15.3.2.1.304.1 Offset

Register	Offset
SWREG362	5A8h

15.3.2.1.304.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.304.3 Fields

Field	Function
31-24 BF_24	Segment 20: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 20: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 20: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.305 VPU H1 Register 363 (SWREG363)

15.3.2.1.305.1 Offset

Register	Offset
SWREG363	5ACh

15.3.2.1.305.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.305.3 Fields

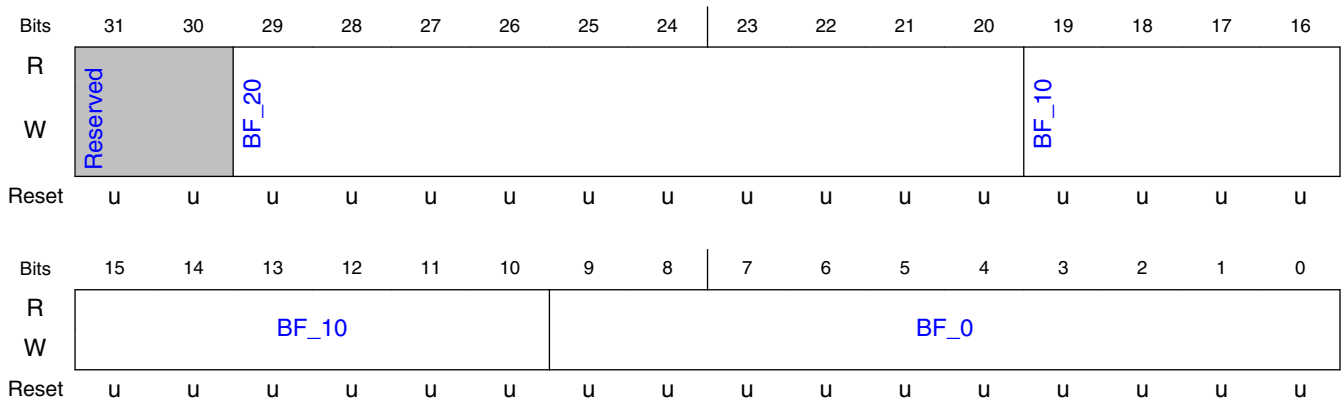
Field	Function
31-24 BF_24	Segment 20: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 20: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.306 VPU H1 Register 364 (SWREG364)

15.3.2.1.306.1 Offset

Register	Offset
SWREG364	5B0h

15.3.2.1.306.2 Diagram



15.3.2.1.306.3 Fields

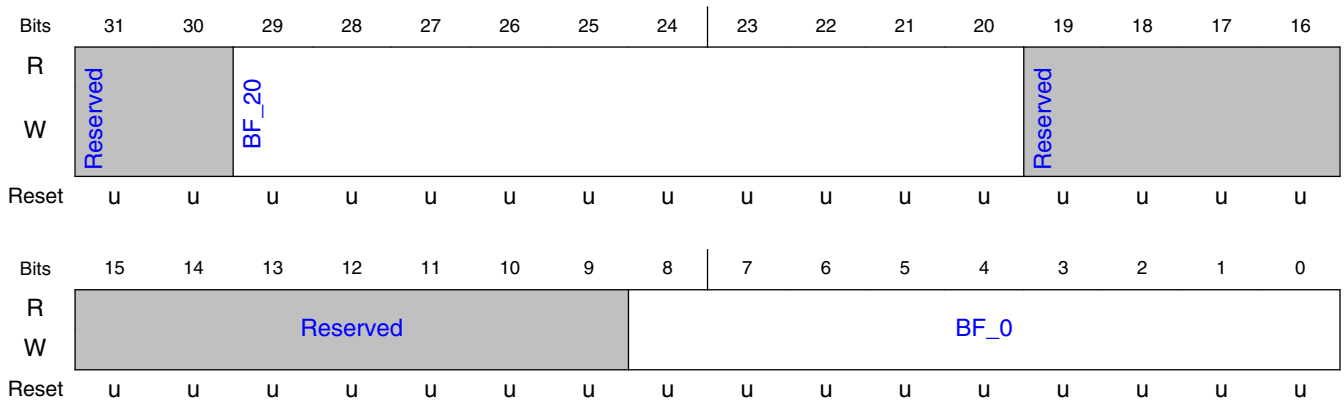
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 20: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 20: Penalty for using 8x8 MV.
9-0 BF_0	Segment 20: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.307 VPU H1 Register 365 (SWREG365)

15.3.2.1.307.1 Offset

Register	Offset
SWREG365	5B4h

15.3.2.1.307.2 Diagram



15.3.2.1.307.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 20: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 20: Penalty for using 4x4 MV.

15.3.2.1.308 VPU H1 Register 366 (SWREG366)

15.3.2.1.308.1 Offset

Register	Offset
SWREG366	5B8h

15.3.2.1.308.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.308.3 Fields

Field	Function
31-24 BF_24	Segment 21: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 21: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 21: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.309 VPU H1 Register 367 (SWREG367)

15.3.2.1.309.1 Offset

Register	Offset
SWREG367	5BCh

15.3.2.1.309.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.309.3 Fields

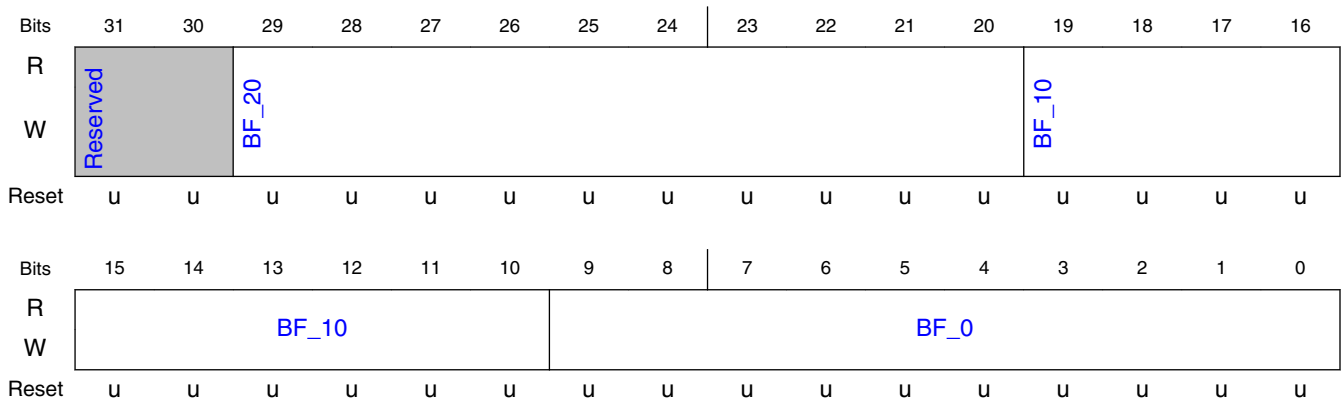
Field	Function
31-24 BF_24	Segment 21: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 21: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.310 VPU H1 Register 368 (SWREG368)

15.3.2.1.310.1 Offset

Register	Offset
SWREG368	5C0h

15.3.2.1.310.2 Diagram



15.3.2.1.310.3 Fields

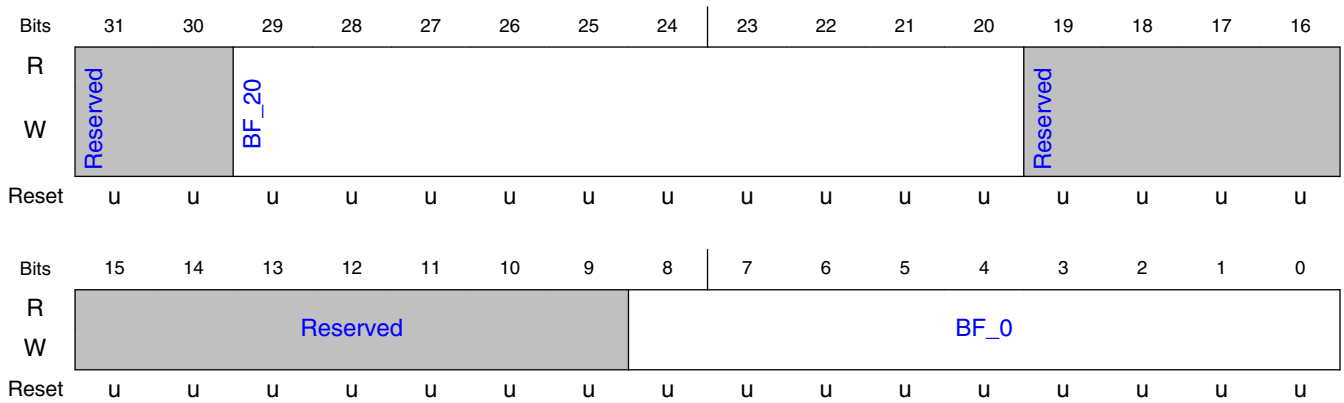
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 21: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 21: Penalty for using 8x8 MV.
9-0 BF_0	Segment 21: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.311 VPU H1 Register 369 (SWREG369)

15.3.2.1.311.1 Offset

Register	Offset
SWREG369	5C4h

15.3.2.1.311.2 Diagram



15.3.2.1.311.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 21: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 21: Penalty for using 4x4 MV.

15.3.2.1.312 VPU H1 Register 370 (SWREG370)

15.3.2.1.312.1 Offset

Register	Offset
SWREG370	5C8h

15.3.2.1.312.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.312.3 Fields

Field	Function
31-24 BF_24	Segment 22: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 22: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 22: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.313 VPU H1 Register 371 (SWREG371)

15.3.2.1.313.1 Offset

Register	Offset
SWREG371	5CCh

15.3.2.1.313.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.313.3 Fields

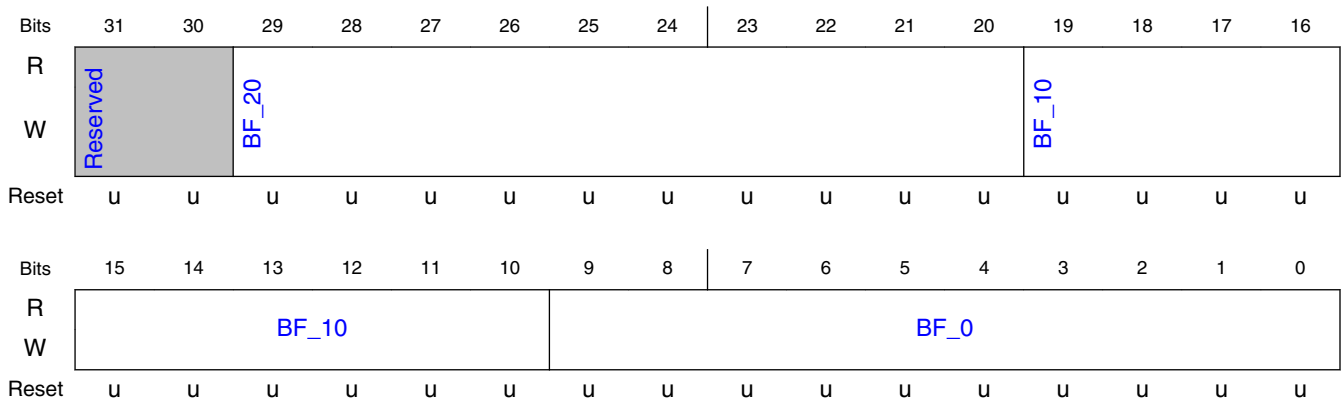
Field	Function
31-24 BF_24	Segment 22: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 22: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.314 VPU H1 Register 372 (SWREG372)

15.3.2.1.314.1 Offset

Register	Offset
SWREG372	5D0h

15.3.2.1.314.2 Diagram



15.3.2.1.314.3 Fields

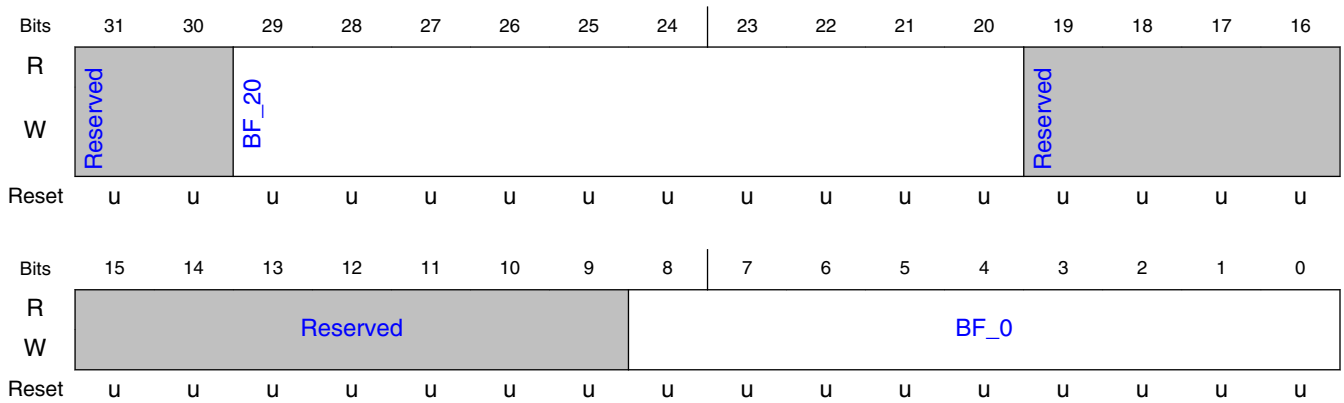
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 22: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 22: Penalty for using 8x8 MV.
9-0 BF_0	Segment 22: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.315 VPU H1 Register 373 (SWREG373)

15.3.2.1.315.1 Offset

Register	Offset
SWREG373	5D4h

15.3.2.1.315.2 Diagram



15.3.2.1.315.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 22: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 22: Penalty for using 4x4 MV.

15.3.2.1.316 VPU H1 Register 374 (SWREG374)

15.3.2.1.316.1 Offset

Register	Offset
SWREG374	5D8h

15.3.2.1.316.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.316.3 Fields

Field	Function
31-24 BF_24	Segment 23: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 23: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 23: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.317 VPU H1 Register 375 (SWREG375)

15.3.2.1.317.1 Offset

Register	Offset
SWREG375	5DCh

15.3.2.1.317.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.317.3 Fields

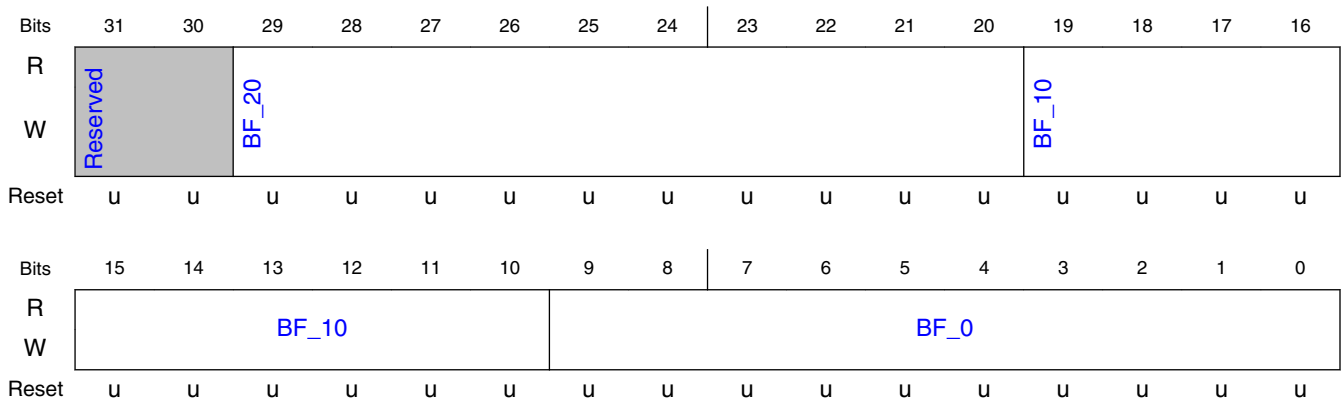
Field	Function
31-24 BF_24	Segment 23: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 23: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.318 VPU H1 Register 376 (SWREG376)

15.3.2.1.318.1 Offset

Register	Offset
SWREG376	5E0h

15.3.2.1.318.2 Diagram



15.3.2.1.318.3 Fields

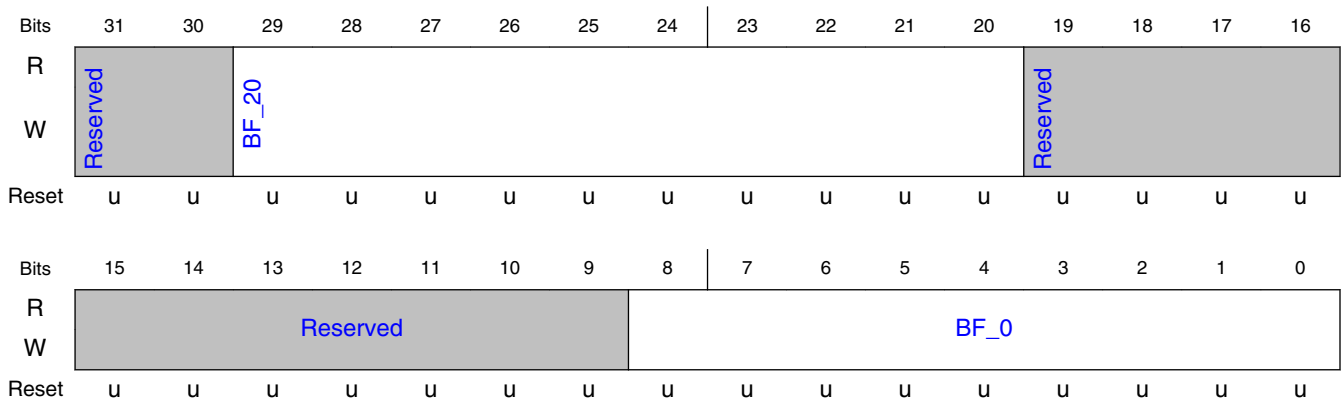
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 23: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 23: Penalty for using 8x8 MV.
9-0 BF_0	Segment 23: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.319 VPU H1 Register 377 (SWREG377)

15.3.2.1.319.1 Offset

Register	Offset
SWREG377	5E4h

15.3.2.1.319.2 Diagram



15.3.2.1.319.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 23: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 23: Penalty for using 4x4 MV.

15.3.2.1.320 VPU H1 Register 378 (SWREG378)

15.3.2.1.320.1 Offset

Register	Offset
SWREG378	5E8h

15.3.2.1.320.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.320.3 Fields

Field	Function
31-24 BF_24	Segment 24: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 24: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 24: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.321 VPU H1 Register 379 (SWREG379)

15.3.2.1.321.1 Offset

Register	Offset
SWREG379	5ECh

15.3.2.1.321.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.321.3 Fields

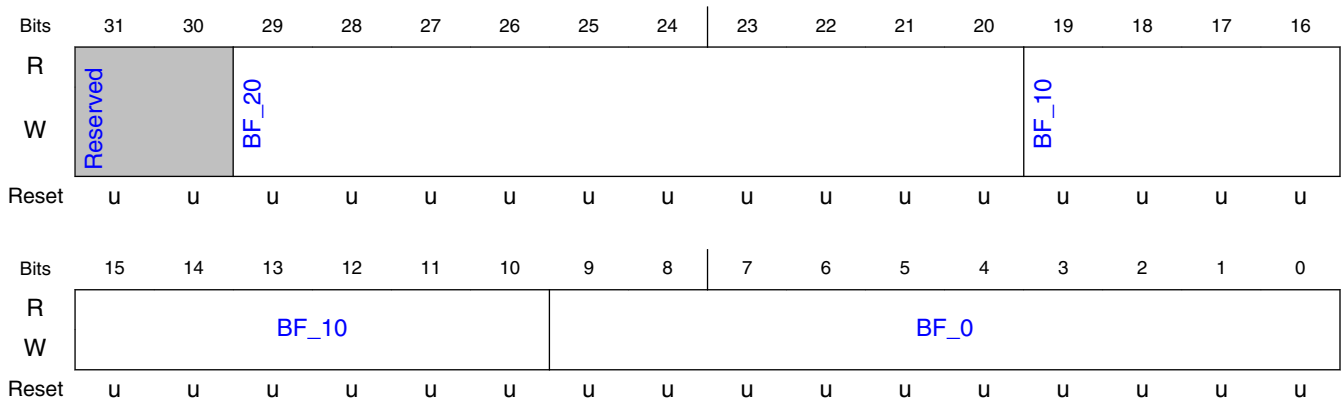
Field	Function
31-24 BF_24	Segment 24: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 24: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.322 VPU H1 Register 380 (SWREG380)

15.3.2.1.322.1 Offset

Register	Offset
SWREG380	5F0h

15.3.2.1.322.2 Diagram



15.3.2.1.322.3 Fields

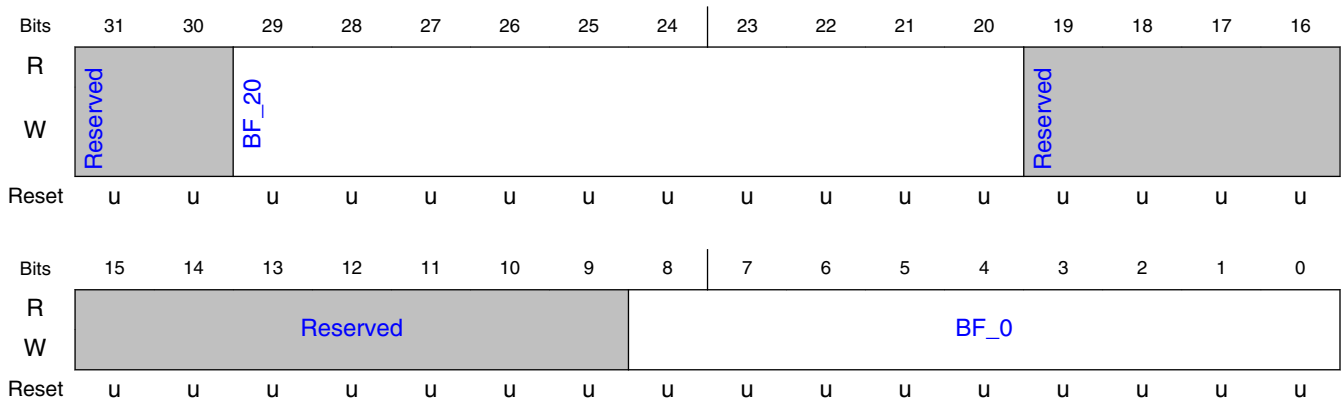
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 24: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 24: Penalty for using 8x8 MV.
9-0 BF_0	Segment 24: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.323 VPU H1 Register 381 (SWREG381)

15.3.2.1.323.1 Offset

Register	Offset
SWREG381	5F4h

15.3.2.1.323.2 Diagram



15.3.2.1.323.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 24: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 24: Penalty for using 4x4 MV.

15.3.2.1.324 VPU H1 Register 382 (SWREG382)

15.3.2.1.324.1 Offset

Register	Offset
SWREG382	5F8h

15.3.2.1.324.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.324.3 Fields

Field	Function
31-24 BF_24	Segment 25: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 25: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 25: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.325 VPU H1 Register 383 (SWREG383)

15.3.2.1.325.1 Offset

Register	Offset
SWREG383	5FCh

15.3.2.1.325.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.325.3 Fields

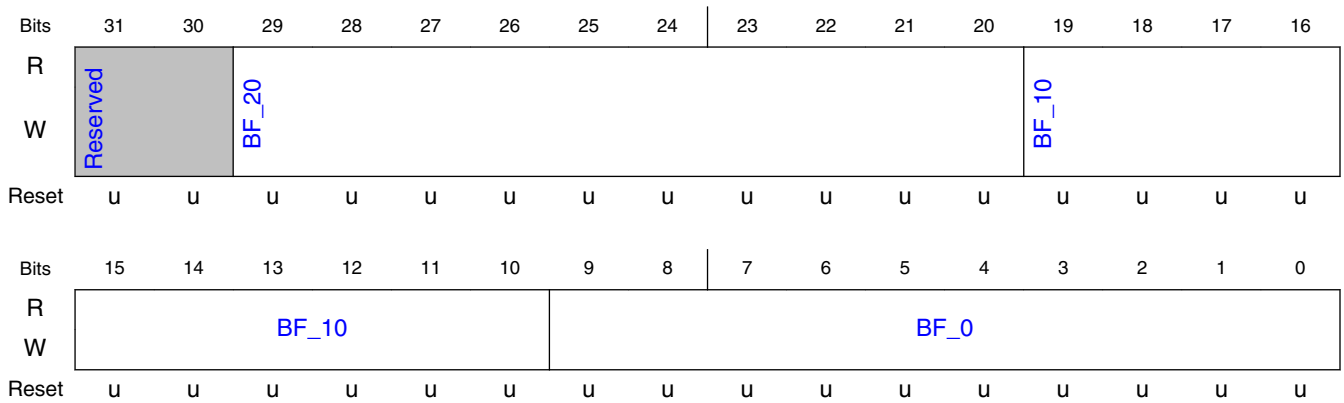
Field	Function
31-24 BF_24	Segment 25: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 25: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.326 VPU H1 Register 384 (SWREG384)

15.3.2.1.326.1 Offset

Register	Offset
SWREG384	600h

15.3.2.1.326.2 Diagram



15.3.2.1.326.3 Fields

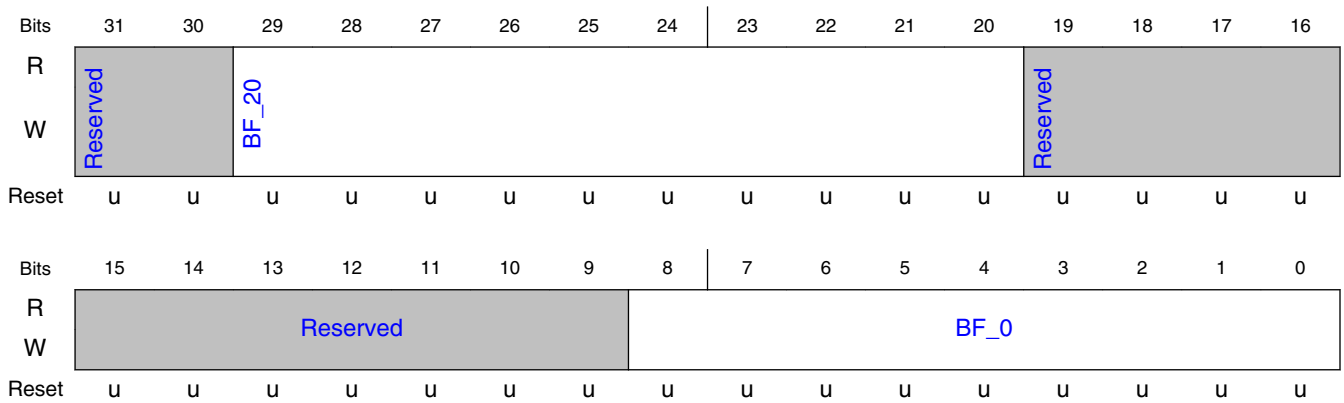
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 25: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 25: Penalty for using 8x8 MV.
9-0 BF_0	Segment 25: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.327 VPU H1 Register 385 (SWREG385)

15.3.2.1.327.1 Offset

Register	Offset
SWREG385	604h

15.3.2.1.327.2 Diagram



15.3.2.1.327.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 25: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 25: Penalty for using 4x4 MV.

15.3.2.1.328 VPU H1 Register 386 (SWREG386)

15.3.2.1.328.1 Offset

Register	Offset
SWREG386	608h

15.3.2.1.328.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.328.3 Fields

Field	Function
31-24 BF_24	Segment 26: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 26: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 26: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.329 VPU H1 Register 387 (SWREG387)

15.3.2.1.329.1 Offset

Register	Offset
SWREG387	60Ch

15.3.2.1.329.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.329.3 Fields

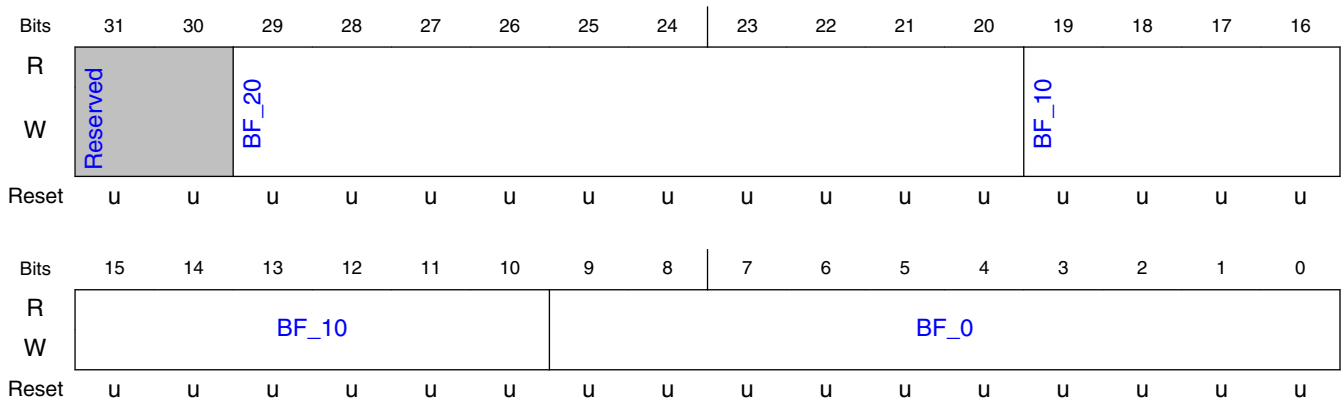
Field	Function
31-24 BF_24	Segment 26: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 26: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.330 VPU H1 Register 388 (SWREG388)

15.3.2.1.330.1 Offset

Register	Offset
SWREG388	610h

15.3.2.1.330.2 Diagram



15.3.2.1.330.3 Fields

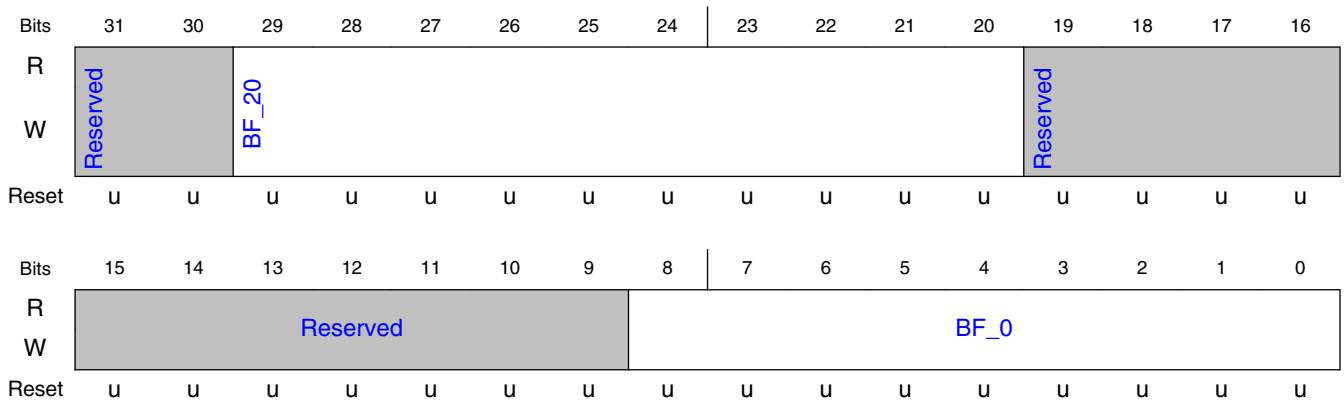
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 26: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 26: Penalty for using 8x8 MV.
9-0 BF_0	Segment 26: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.331 VPU H1 Register 389 (SWREG389)

15.3.2.1.331.1 Offset

Register	Offset
SWREG389	614h

15.3.2.1.331.2 Diagram



15.3.2.1.331.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 26: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 26: Penalty for using 4x4 MV.

15.3.2.1.332 VPU H1 Register 390 (SWREG390)

15.3.2.1.332.1 Offset

Register	Offset
SWREG390	618h

15.3.2.1.332.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.332.3 Fields

Field	Function
31-24 BF_24	Segment 27: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 27: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segm

15.3.2.1.333 VPU H1 Register 391 (SWREG391)

15.3.2.1.333.1 Offset

Register	Offset
SWREG391	61Ch

15.3.2.1.333.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.333.3 Fields

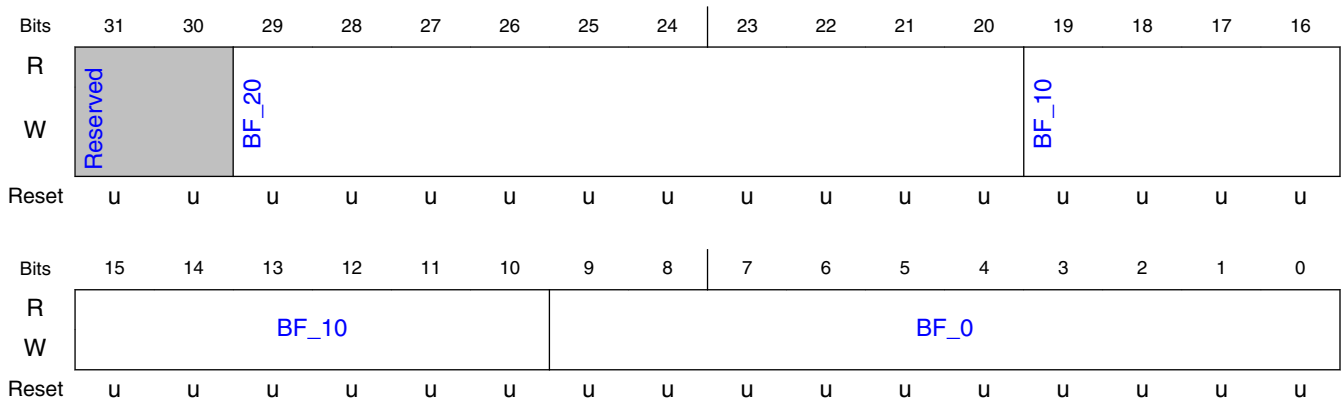
Field	Function
31-24 BF_24	Segment 27: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 27: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.334 VPU H1 Register 392 (SWREG392)

15.3.2.1.334.1 Offset

Register	Offset
SWREG392	620h

15.3.2.1.334.2 Diagram



15.3.2.1.334.3 Fields

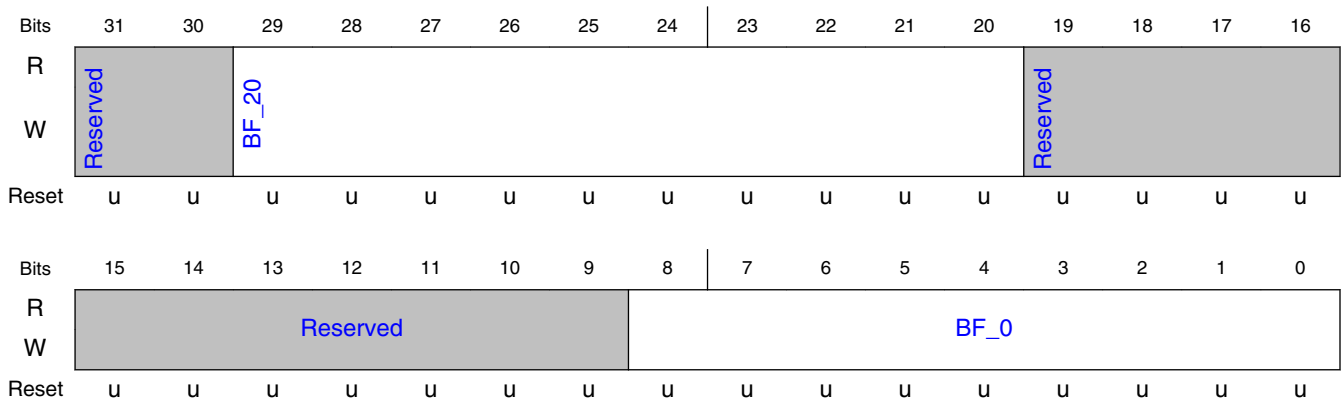
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 27: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 27: Penalty for using 8x8 MV.
9-0 BF_0	Segment 27: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.335 VPU H1 Register 393 (SWREG393)

15.3.2.1.335.1 Offset

Register	Offset
SWREG393	624h

15.3.2.1.335.2 Diagram



15.3.2.1.335.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 27: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 27: Penalty for using 4x4 MV.

15.3.2.1.336 VPU H1 Register 394 (SWREG394)

15.3.2.1.336.1 Offset

Register	Offset
SWREG394	628h

15.3.2.1.336.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.336.3 Fields

Field	Function
31-24 BF_24	Segment 28: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 28: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 28: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.337 VPU H1 Register 395 (SWREG395)

15.3.2.1.337.1 Offset

Register	Offset
SWREG395	62Ch

15.3.2.1.337.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.337.3 Fields

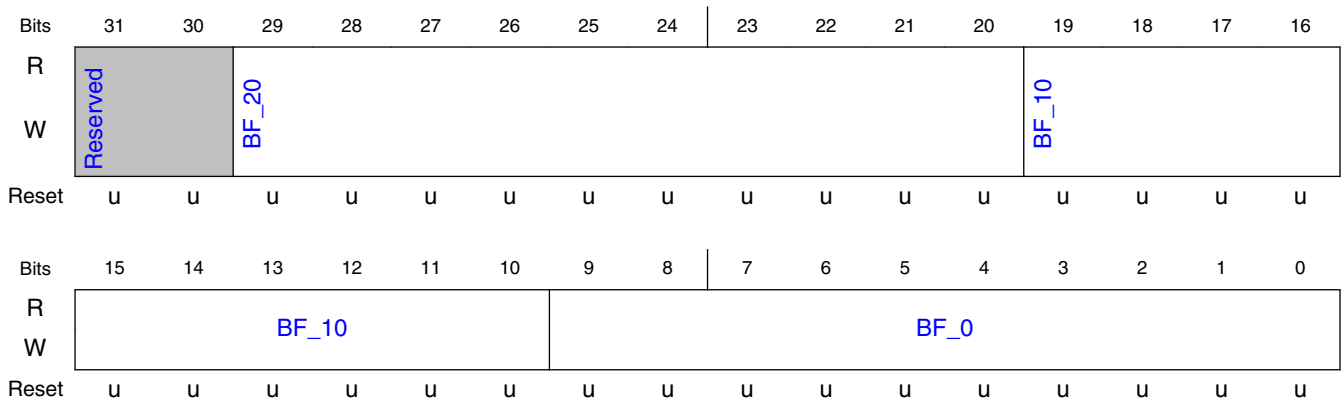
Field	Function
31-24 BF_24	Segment 28: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 28: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.338 VPU H1 Register 396 (SWREG396)

15.3.2.1.338.1 Offset

Register	Offset
SWREG396	630h

15.3.2.1.338.2 Diagram



15.3.2.1.338.3 Fields

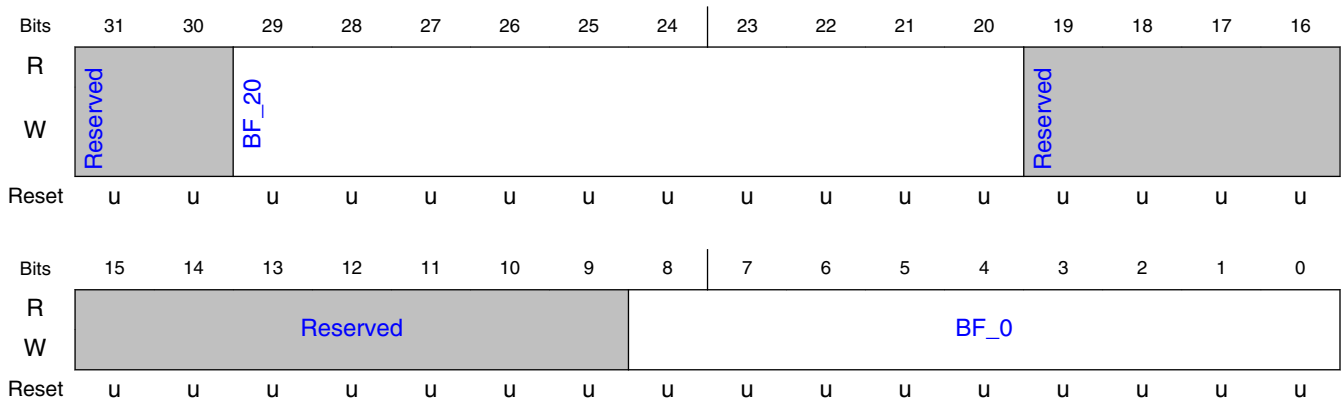
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 28: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 28: Penalty for using 8x8 MV.
9-0 BF_0	Segment 28: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.339 VPU H1 Register 397 (SWREG397)

15.3.2.1.339.1 Offset

Register	Offset
SWREG397	634h

15.3.2.1.339.2 Diagram



15.3.2.1.339.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 28: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 28: Penalty for using 4x4 MV.

15.3.2.1.340 VPU H1 Register 398 (SWREG398)

15.3.2.1.340.1 Offset

Register	Offset
SWREG398	638h

15.3.2.1.340.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.340.3 Fields

Field	Function
31-24 BF_24	Segment 29: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 29: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 29: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.341 VPU H1 Register 399 (SWREG399)

15.3.2.1.341.1 Offset

Register	Offset
SWREG399	63Ch

15.3.2.1.341.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.341.3 Fields

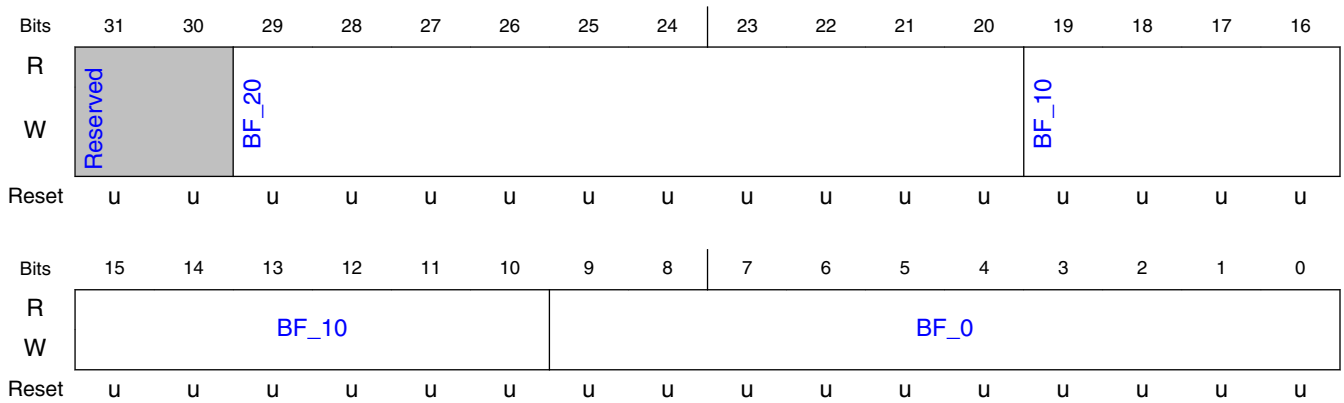
Field	Function
31-24 BF_24	Segment 29: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 29: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.342 VPU H1 Register 400 (SWREG400)

15.3.2.1.342.1 Offset

Register	Offset
SWREG400	640h

15.3.2.1.342.2 Diagram



15.3.2.1.342.3 Fields

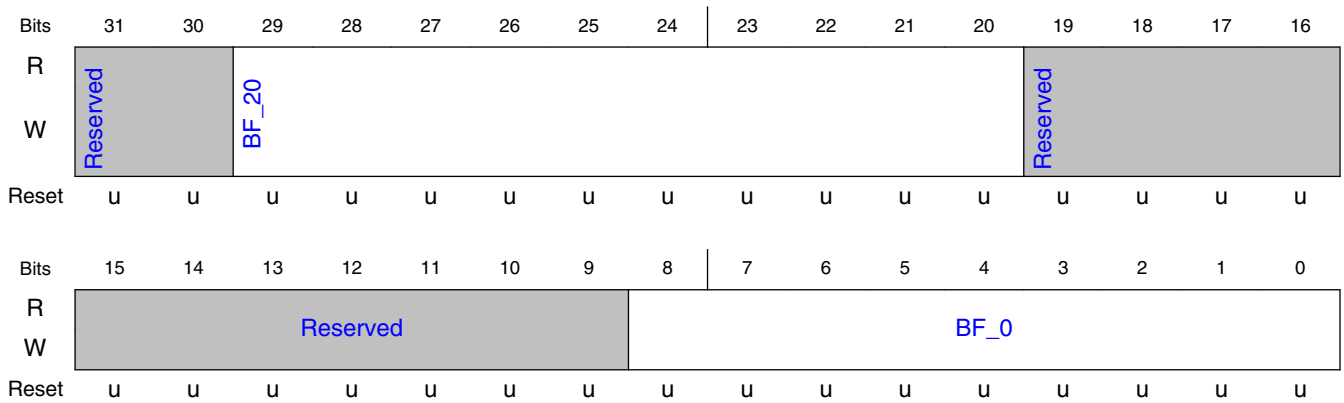
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 29: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 29: Penalty for using 8x8 MV.
9-0 BF_0	Segment 29: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.343 VPU H1 Register 401 (SWREG401)

15.3.2.1.343.1 Offset

Register	Offset
SWREG401	644h

15.3.2.1.343.2 Diagram



15.3.2.1.343.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 29: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 29: Penalty for using 4x4 MV.

15.3.2.1.344 VPU H1 Register 402 (SWREG402)

15.3.2.1.344.1 Offset

Register	Offset
SWREG402	648h

15.3.2.1.344.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.344.3 Fields

Field	Function
31-24 BF_24	Segment 30: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 30: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 30: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.345 VPU H1 Register 403 (SWREG403)

15.3.2.1.345.1 Offset

Register	Offset
SWREG403	64Ch

15.3.2.1.345.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.345.3 Fields

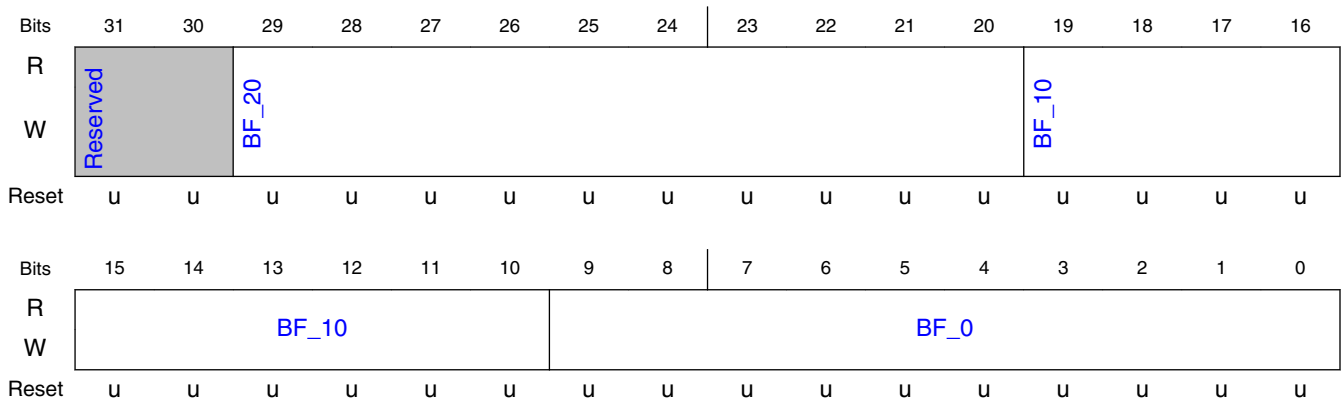
Field	Function
31-24 BF_24	Segment 30: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 30: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.346 VPU H1 Register 404 (SWREG404)

15.3.2.1.346.1 Offset

Register	Offset
SWREG404	650h

15.3.2.1.346.2 Diagram



15.3.2.1.346.3 Fields

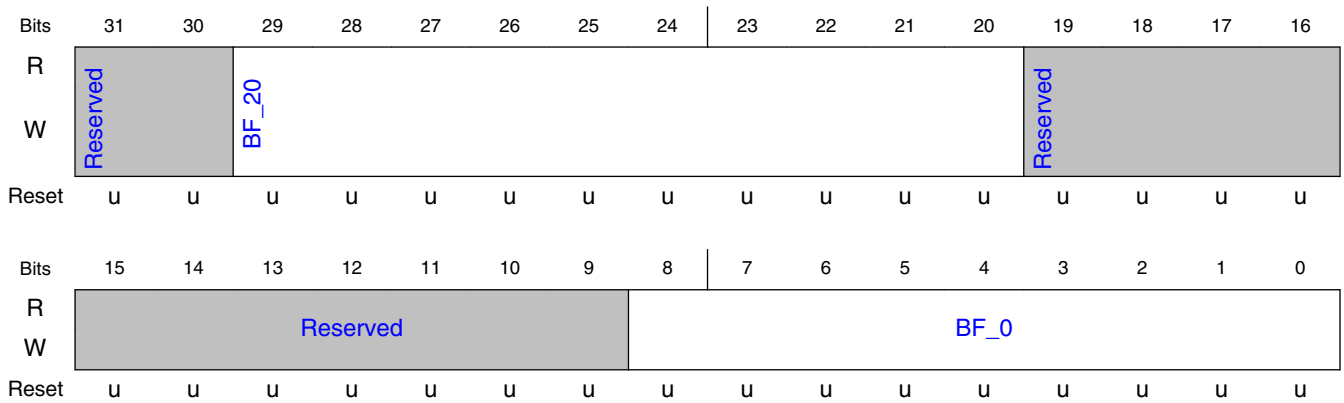
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 30: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 30: Penalty for using 8x8 MV.
9-0 BF_0	Segment 30: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.347 VPU H1 Register 405 (SWREG405)

15.3.2.1.347.1 Offset

Register	Offset
SWREG405	654h

15.3.2.1.347.2 Diagram



15.3.2.1.347.3 Fields

Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 30: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 30: Penalty for using 4x4 MV.

15.3.2.1.348 VPU H1 Register 406 (SWREG406)

15.3.2.1.348.1 Offset

Register	Offset
SWREG406	658h

15.3.2.1.348.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.348.3 Fields

Field	Function
31-24 BF_24	Segment 31: Intra 4x4 previous mode favor for H.264
23-8 BF_8	Segment 31: Intra 16x16 mode favor in intra 16x16/4x4 selection
7-0 BF_0	Segment 31: Penalty value for second reference frame zero-mv. VP8 golden / H.264 LTR. [0..255]

15.3.2.1.349 VPU H1 Register 407 (SWREG407)

15.3.2.1.349.1 Offset

Register	Offset
SWREG407	65Ch

15.3.2.1.349.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_24								BF_8							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_8								Reserved							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.349.3 Fields

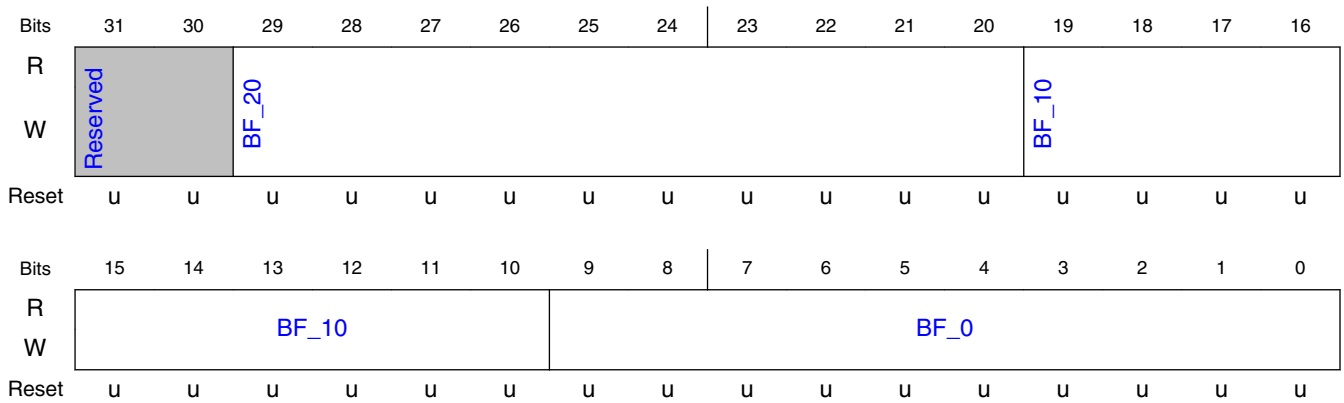
Field	Function
31-24 BF_24	Segment 31: Skip mode (zero/nearest/near) penalty
23-8 BF_8	Segment 31: Inter MB mode favor in intra/inter selection
7-0 —	Reserved.

15.3.2.1.350 VPU H1 Register 408 (SWREG408)

15.3.2.1.350.1 Offset

Register	Offset
SWREG408	660h

15.3.2.1.350.2 Diagram



15.3.2.1.350.3 Fields

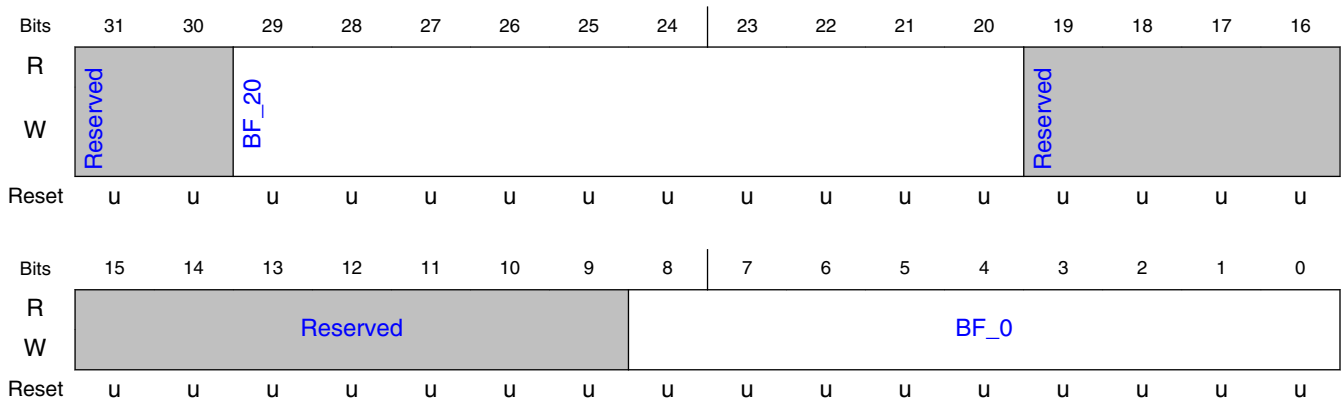
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 31: Penalty for using 16x8 or 8x16 MV.
19-10 BF_10	Segment 31: Penalty for using 8x8 MV.
9-0 BF_0	Segment 31: Penalty for using 8x4 or 4x8 MV.

15.3.2.1.351 VPU H1 Register 409 (SWREG409)

15.3.2.1.351.1 Offset

Register	Offset
SWREG409	664h

15.3.2.1.351.2 Diagram



15.3.2.1.351.3 Fields

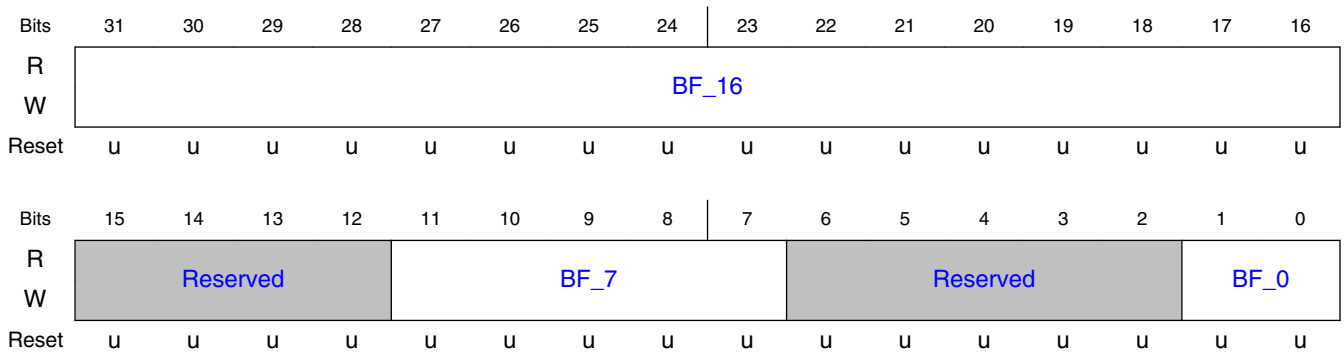
Field	Function
31-30 —	Reserved.
29-20 BF_20	Segment 31: Differential MV penalty for 1/4p ME. DMVPenaltyQp
19-9 —	Reserved.
8-0 BF_0	Segment 31: Penalty for using 4x4 MV.

15.3.2.1.352 VPU H1 Register 410 (SWREG410)

15.3.2.1.352.1 Offset

Register	Offset
SWREG410	668h

15.3.2.1.352.2 Diagram



15.3.2.1.352.3 Fields

Field	Function
31-16 BF_16	fractional part of frame qp. .16 format
15-12 —	Reserved.
11-7 BF_7	offset of MB complexity
6-2 —	Reserved.
1-0 BF_0	MB RC Enable. 0 = disable. 1 = enable.

15.3.2.1.353 VPU H1 Register 411 (SWREG411)

15.3.2.1.353.1 Offset

Register	Offset
SWREG411	66Ch

15.3.2.1.353.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.353.3 Fields

Field	Function
31-16 —	Reserved.
15-0 BF_0	gain of MB qp delta. 8.8 format

15.3.2.1.354 VPU H1 Register 412 (SWREG412)

15.3.2.1.354.1 Offset

Register	Offset
SWREG412	670h

15.3.2.1.354.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.354.3 Fields

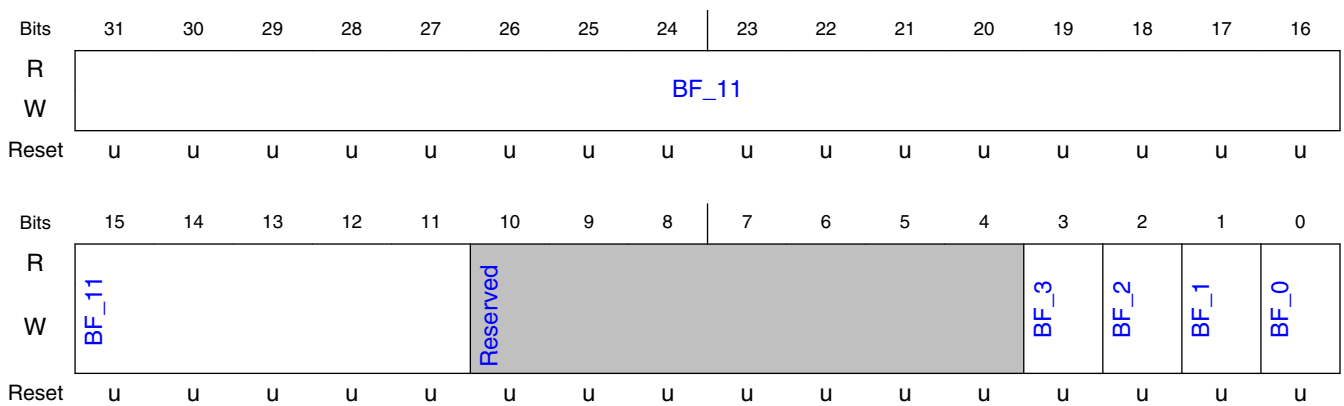
Field	Function
31-0 BF_0	average of MB complexity

15.3.2.1.355 VPU H1 Register 413 (SWREG413)

15.3.2.1.355.1 Offset

Register	Offset
SWREG413	674h

15.3.2.1.355.2 Diagram



15.3.2.1.355.3 Fields

Field	Function
31-11 BF_11	Limit of luma RFC buffer in unit of 64-bit. 0 indicating no limit. Send an IRQ if overflow.
10-4 —	Reserved.
3 BF_3	RFC overflow IRQ Enable. 0=Disable. 1=Enable.

Table continues on the next page...

Field	Function
2 BF_2	Reference Compression in discrete storage mode. 0=OFF. 1=ON.
1 BF_1	Reference Frame Chroma Compression. 0=OFF. 1=ON.
0 BF_0	Reference Frame Luma Compression. 0=OFF. 1=ON.

15.3.2.1.356 VPU H1 Register 414 (SWREG414)

15.3.2.1.356.1 Offset

Register	Offset
SWREG414	678h

15.3.2.1.356.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.356.3 Fields

Field	Function
31-0 BF_0	Base address for reference luma compression table

15.3.2.1.357 VPU H1 Register 415 (SWREG415)

15.3.2.1.357.1 Offset

Register	Offset
SWREG415	67Ch

15.3.2.1.357.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.357.3 Fields

Field	Function
31-0 BF_0	Base address for reference chroma compression table

15.3.2.1.358 VPU H1 Register 416 (SWREG416)

15.3.2.1.358.1 Offset

Register	Offset
SWREG416	680h

15.3.2.1.358.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.358.3 Fields

Field	Function
31-0 BF_0	Base address for reconstructed luma compression table

15.3.2.1.359 VPU H1 Register 417 (SWREG417)

15.3.2.1.359.1 Offset

Register	Offset
SWREG417	684h

15.3.2.1.359.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.359.3 Fields

Field	Function
31-0 BF_0	Base address for reconstructed chroma compression table

15.3.2.1.360 VPU H1 Register 418 (SWREG418)

15.3.2.1.360.1 Offset

Register	Offset
SWREG418	688h

15.3.2.1.360.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.360.3 Fields

Field	Function
31-0 BF_0	Base address for second reference luma compression table

15.3.2.1.361 VPU H1 Register 419 (SWREG419)

15.3.2.1.361.1 Offset

Register	Offset
SWREG419	68Ch

15.3.2.1.361.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.361.3 Fields

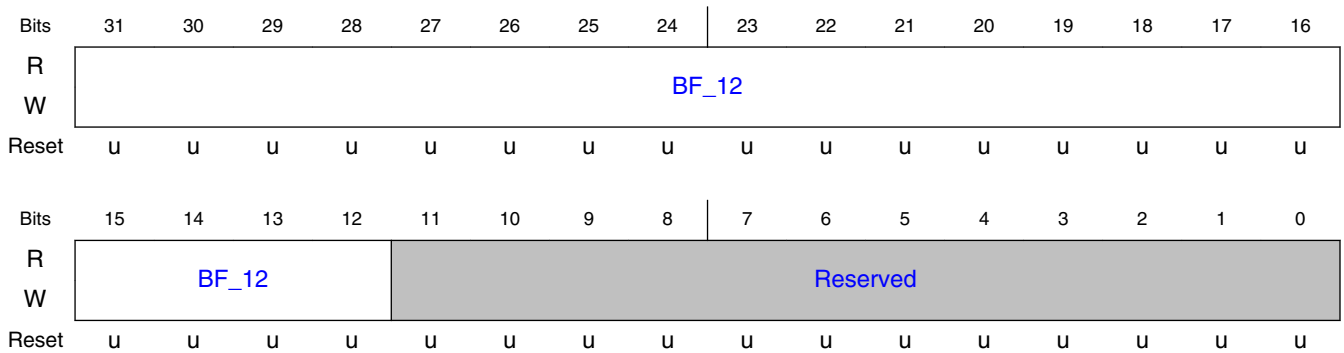
Field	Function
31-0 BF_0	Base address for second reference chroma compression table

15.3.2.1.362 VPU H1 Register 420 (SWREG420)

15.3.2.1.362.1 Offset

Register	Offset
SWREG420	690h

15.3.2.1.362.2 Diagram



15.3.2.1.362.3 Fields

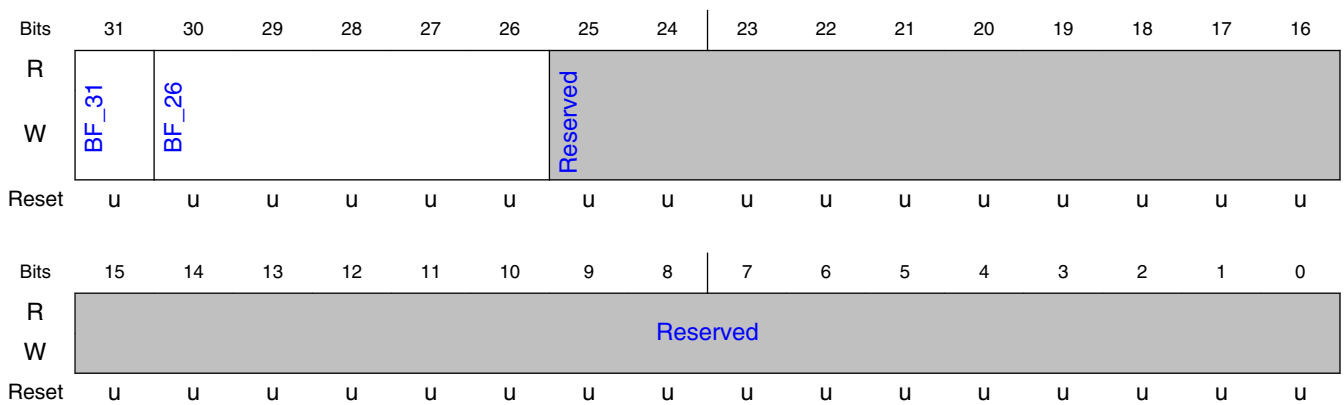
Field	Function
31-12 BF_12	Limit of chroma RFC buffer in unit of 64-bit. 0 indicating no limit. Send an IRQ if overflow.
11-0 —	Reserved.

15.3.2.1.363 VPU H1 Register 421 (SWREG421)

15.3.2.1.363.1 Offset

Register	Offset
SWREG421	694h

15.3.2.1.363.2 Diagram



15.3.2.1.363.3 Fields

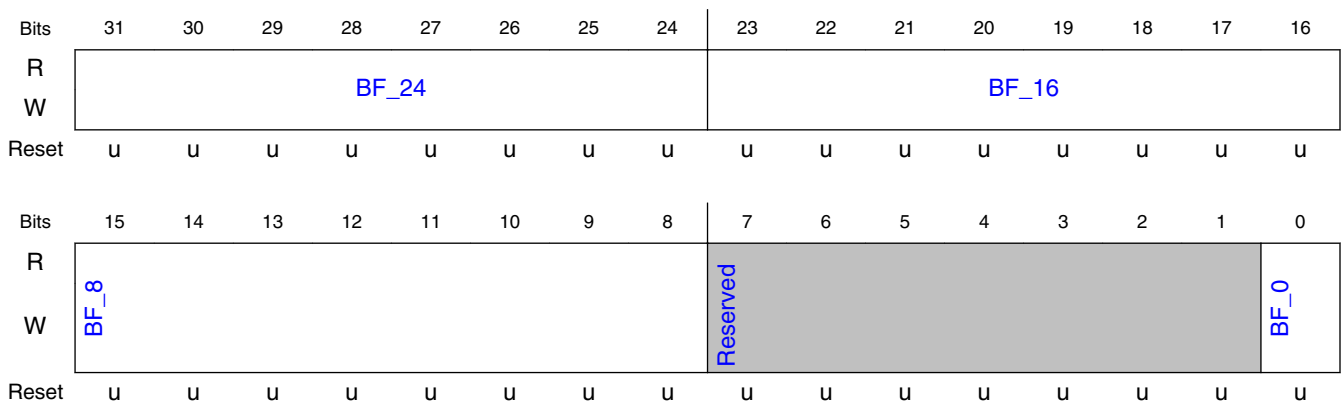
Field	Function
31 BF_31	Modification Flag of Pic Nums.
30-26 BF_26	Modification Absolute Diff Pic Nums.
25-0 —	Reserved.

15.3.2.1.364 VPU H1 Register 422 (SWREG422)

15.3.2.1.364.1 Offset

Register	Offset
SWREG422	698h

15.3.2.1.364.2 Diagram



15.3.2.1.364.3 Fields

Field	Function
31-24 BF_24	AXI Read ID for Input Channel 0.

Table continues on the next page...

Field	Function
23-16 BF_16	AXI Read ID for Input Channel 1.
15-8 BF_8	AXI Read ID for Input Channel 2.
7-1 —	Reserved.
0 BF_0	Enable AXI Read ID and different burst.

15.3.2.1.365 VPU H1 Register 423 (SWREG423)

15.3.2.1.365.1 Offset

Register	Offset
SWREG423	69Ch

15.3.2.1.365.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.365.3 Fields

Field	Function
31-0 BF_0	Base address MSB for reference luma compression table.

15.3.2.1.366 VPU H1 Register 424 (SWREG424)

15.3.2.1.366.1 Offset

Register	Offset
SWREG424	6A0h

15.3.2.1.366.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.366.3 Fields

Field	Function
31-0 BF_0	Base address MSB for reference chroma compression table

15.3.2.1.367 VPU H1 Register 425 (SWREG425)

15.3.2.1.367.1 Offset

Register	Offset
SWREG425	6A4h

15.3.2.1.367.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.367.3 Fields

Field	Function
31-0 BF_0	Base address MSB for reconstructed luma compression table

15.3.2.1.368 VPU H1 Register 426 (SWREG426)

15.3.2.1.368.1 Offset

Register	Offset
SWREG426	6A8h

15.3.2.1.368.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.368.3 Fields

Field	Function
31-0 BF_0	Base address for reconstructed chroma compression table

15.3.2.1.369 VPU H1 Register 427 (SWREG427)

15.3.2.1.369.1 Offset

Register	Offset
SWREG427	6ACh

15.3.2.1.369.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.369.3 Fields

Field	Function
31-0 BF_0	Base address MSB for second reference luma compression table

15.3.2.1.370 VPU H1 Register 428 (SWREG428)

15.3.2.1.370.1 Offset

Register	Offset
SWREG428	6B0h

15.3.2.1.370.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.370.3 Fields

Field	Function
31-0 BF_0	Base address MSB for second reference chroma compression table

15.3.2.1.371 VPU H1 Register 429 (SWREG429)

15.3.2.1.371.1 Offset

Register	Offset
SWREG429	6B4h

15.3.2.1.371.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.371.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for output stream data

15.3.2.1.372 VPU H1 Register 430 (SWREG430)

15.3.2.1.372.1 Offset

Register	Offset
SWREG430	6B8h

15.3.2.1.372.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.372.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for output control data

15.3.2.1.373 VPU H1 Register 431 (SWREG431)

15.3.2.1.373.1 Offset

Register	Offset
SWREG431	6BCh

15.3.2.1.373.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.373.3 Fields

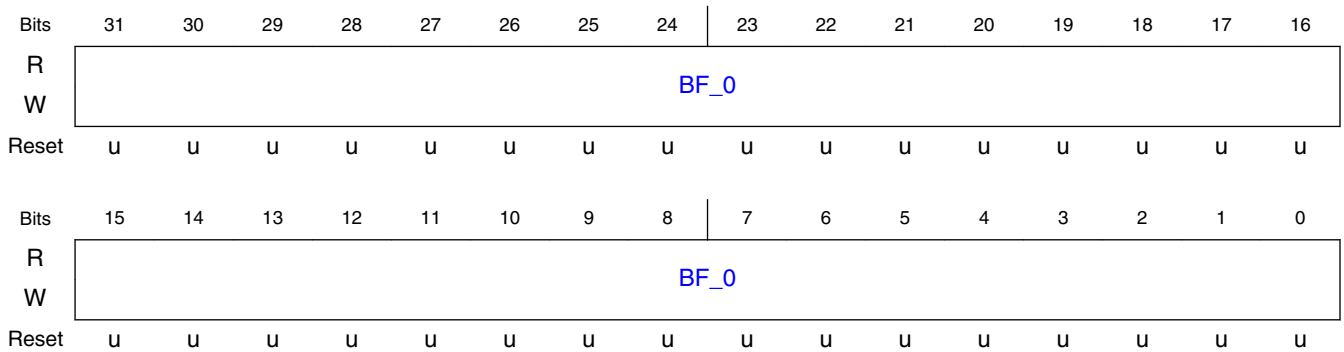
Field	Function
31-0 BF_0	high 32 bits of Base address for reference luma

15.3.2.1.374 VPU H1 Register 432 (SWREG432)

15.3.2.1.374.1 Offset

Register	Offset
SWREG432	6C0h

15.3.2.1.374.2 Diagram



15.3.2.1.374.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for reference chroma

15.3.2.1.375 VPU H1 Register 433 (SWREG433)

15.3.2.1.375.1 Offset

Register	Offset
SWREG433	6C4h

15.3.2.1.375.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.375.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for reconstructed luma

15.3.2.1.376 VPU H1 Register 434 (SWREG434)

15.3.2.1.376.1 Offset

Register	Offset
SWREG434	6C8h

15.3.2.1.376.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.376.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for reconstructed chroma

15.3.2.1.377 VPU H1 Register 435 (SWREG435)

15.3.2.1.377.1 Offset

Register	Offset
SWREG435	6CCh

15.3.2.1.377.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.377.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for input picture luma

15.3.2.1.378 VPU H1 Register 436 (SWREG436)

15.3.2.1.378.1 Offset

Register	Offset
SWREG436	6D0h

15.3.2.1.378.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.378.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for input picture cb

15.3.2.1.379 VPU H1 Register 437 (SWREG437)

15.3.2.1.379.1 Offset

Register	Offset
SWREG437	6D4h

15.3.2.1.379.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.379.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for input picture cr

15.3.2.1.380 VPU H1 Register 438 (SWREG438)

15.3.2.1.380.1 Offset

Register	Offset
SWREG438	6D8h

15.3.2.1.380.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.380.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for second reference luma

15.3.2.1.381 VPU H1 Register 439 (SWREG439)

15.3.2.1.381.1 Offset

Register	Offset
SWREG439	6DCh

15.3.2.1.381.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.381.3 Fields

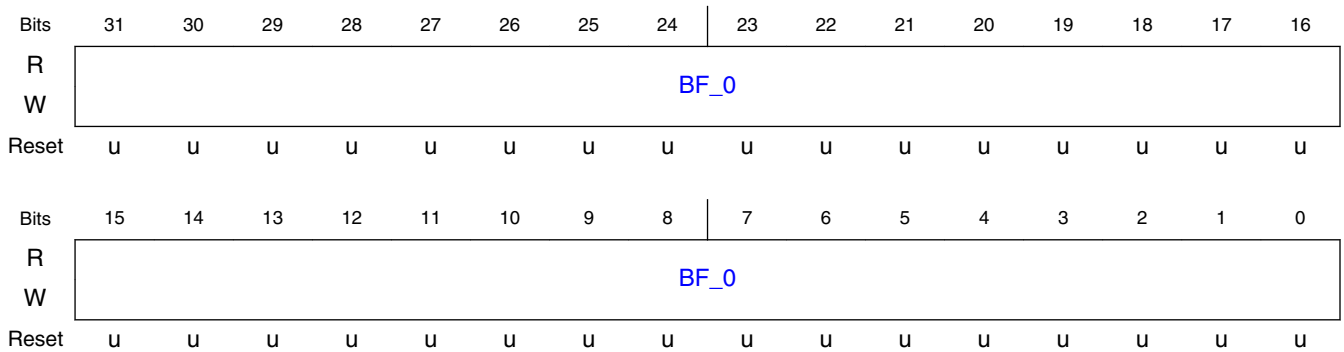
Field	Function
31-0 BF_0	high 32 bits of Base address for second reference chroma

15.3.2.1.382 VPU H1 Register 440 (SWREG440)

15.3.2.1.382.1 Offset

Register	Offset
SWREG440	6E0h

15.3.2.1.382.2 Diagram



15.3.2.1.382.3 Fields

Field	Function
31-0 BF_0	high 32 bits of H264 secondary ref pic base

15.3.2.1.383 VPU H1 Register 441 (SWREG441)

15.3.2.1.383.1 Offset

Register	Offset
SWREG441	6E4h

15.3.2.1.383.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.383.3 Fields

Field	Function
31-0 BF_0	high 32 bits of H264 secondary ref pic base

15.3.2.1.384 VPU H1 Register 442 (SWREG442)

15.3.2.1.384.1 Offset

Register	Offset
SWREG442	6E8h

15.3.2.1.384.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.384.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for next pic luminance

15.3.2.1.385 VPU H1 Register 443 (SWREG443)

15.3.2.1.385.1 Offset

Register	Offset
SWREG443	6ECh

15.3.2.1.385.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.385.3 Fields

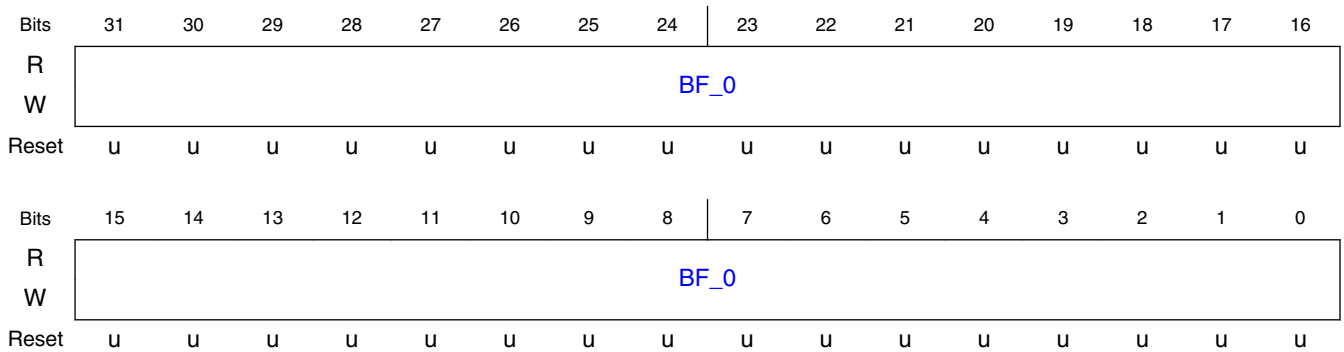
Field	Function
31-0 BF_0	high 32 bits of Base address for cabac context tables (H264) or probability tables (VP8)

15.3.2.1.386 VPU H1 Register 444 (SWREG444)

15.3.2.1.386.1 Offset

Register	Offset
SWREG444	6F0h

15.3.2.1.386.2 Diagram



15.3.2.1.386.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for MV output writing

15.3.2.1.387 VPU H1 Register 445 (SWREG445)

15.3.2.1.387.1 Offset

Register	Offset
SWREG445	6F4h

15.3.2.1.387.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.387.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for VP8 1st DCT partition

15.3.2.1.388 VPU H1 Register 446 (SWREG446)

15.3.2.1.388.1 Offset

Register	Offset
SWREG446	6F8h

15.3.2.1.388.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.388.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for VP8 2nd DCT partition

15.3.2.1.389 VPU H1 Register 447 (SWREG447)

15.3.2.1.389.1 Offset

Register	Offset
SWREG447	6FCh

15.3.2.1.389.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.389.3 Fields

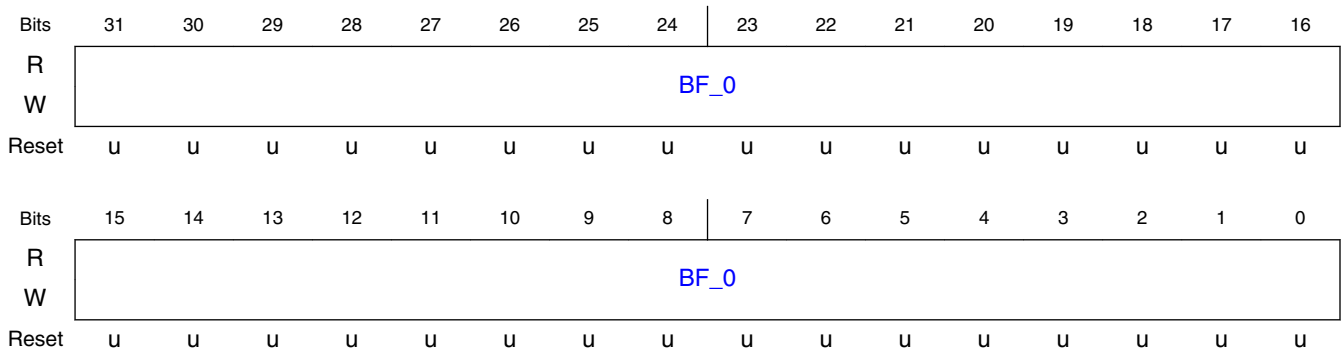
Field	Function
31-0 BF_0	high 32 bits of Base address for VP8 counters for probability updates

15.3.2.1.390 VPU H1 Register 448 (SWREG448)

15.3.2.1.390.1 Offset

Register	Offset
SWREG448	700h

15.3.2.1.390.2 Diagram



15.3.2.1.390.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for VP8 segmentation map. segmentId 2-bits/macroblock

15.3.2.1.391 VPU H1 Register 449 (SWREG449)

15.3.2.1.391.1 Offset

Register	Offset
SWREG449	704h

15.3.2.1.391.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.391.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for output of down-scaled encoder image in YUYV 4:2:2 format

15.3.2.1.392 VPU H1 Register 450 (SWREG450)

15.3.2.1.392.1 Offset

Register	Offset
SWREG450	708h

15.3.2.1.392.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.392.3 Fields

Field	Function
31-0 BF_0	high 32 bits of Base address for VP8 3rd DCT partition

15.3.2.1.393 VPU H1 Register 451 (SWREG451)

15.3.2.1.393.1 Offset

Register	Offset
SWREG451	70Ch

15.3.2.1.393.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.393.3 Fields

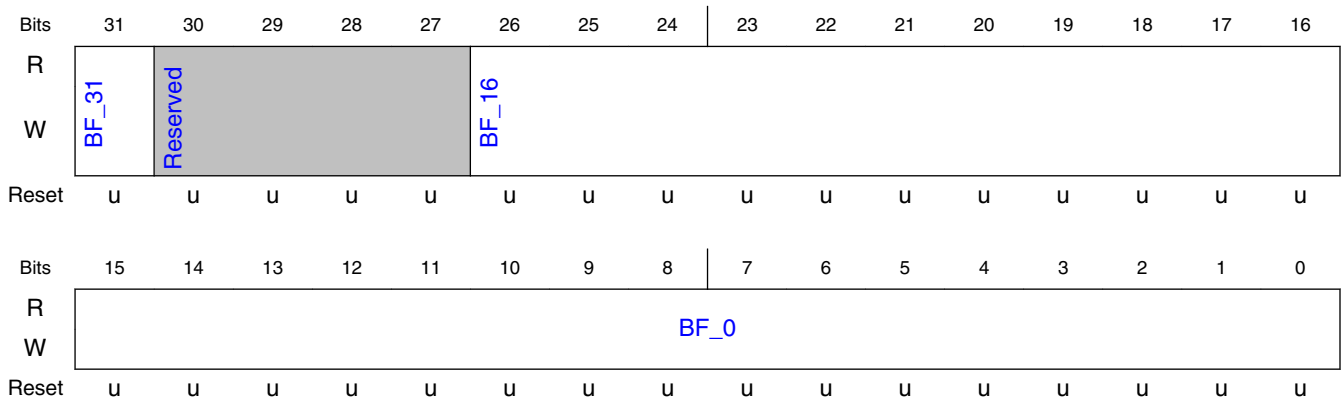
Field	Function
31-0 BF_0	high 32 bits of Base address for VP8 4th DCT partition

15.3.2.1.394 VPU H1 Register 452 (SWREG452)

15.3.2.1.394.1 Offset

Register	Offset
SWREG452	710h

15.3.2.1.394.2 Diagram



15.3.2.1.394.3 Fields

Field	Function
31 BF_31	Enable/disable the de-noise function
30-27 —	Reserved.
26-16 BF_16	Denoise filter strength
15-0 BF_0	Number of Noisy MB

15.3.2.1.395 VPU H1 Register 453 (SWREG453)

15.3.2.1.395.1 Offset

Register	Offset
SWREG453	714h

15.3.2.1.395.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.395.3 Fields

Field	Function
31-16 —	Reserved.
15-0 BF_0	Max noise level

15.3.2.1.396 VPU H1 Register 454 (SWREG454)

15.3.2.1.396.1 Offset

Register	Offset
SWREG454	718h

15.3.2.1.396.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				BF_16											
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				BF_0											
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.396.3 Fields

Field	Function
31-28 —	Reserved.
27-16 BF_16	Noise level inverted for chroma
15-12 —	Reserved.
11-0 BF_0	Noise level inverted for luma

15.3.2.1.397 VPU H1 Register 455 (SWREG455)

15.3.2.1.397.1 Offset

Register	Offset
SWREG455	71Ch

15.3.2.1.397.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.397.3 Fields

Field	Function
31-0 BF_0	calculated sigma of coding frame

15.3.2.1.398 VPU H1 Register 456 (SWREG456)

15.3.2.1.398.1 Offset

Register	Offset
SWREG456	720h

15.3.2.1.398.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.398.3 Fields

Field	Function
31-0 BF_0	noise sigma for current frame

15.3.2.1.399 VPU H1 Register 457 (SWREG457)

15.3.2.1.399.1 Offset

Register	Offset
SWREG457	724h

15.3.2.1.399.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.399.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 1, No.3.
23-16 BF_16	Denoise Filter Parameters Set 1, No.2.
15-8 BF_8	Denoise Filter Parameters Set 1, No.1.
7-0 BF_0	Denoise Filter Parameters Set 1, No.0.

15.3.2.1.400 VPU H1 Register 458 (SWREG458)

15.3.2.1.400.1 Offset

Register	Offset
SWREG458	728h

15.3.2.1.400.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.400.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 1, No.7.
23-16 BF_16	Denoise Filter Parameters Set 1, No.6.
15-8 BF_8	Denoise Filter Parameters Set 1, No.5.
7-0 BF_0	Denoise Filter Parameters Set 1, No.4.

15.3.2.1.401 VPU H1 Register 459 (SWREG459)

15.3.2.1.401.1 Offset

Register	Offset
SWREG459	72Ch

15.3.2.1.401.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.401.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 1, No.11.
23-16 BF_16	Denoise Filter Parameters Set 1, No.10.
15-8 BF_8	Denoise Filter Parameters Set 1, No.9.
7-0 BF_0	Denoise Filter Parameters Set 1, No.8.

15.3.2.1.402 VPU H1 Register 460 (SWREG460)

15.3.2.1.402.1 Offset

Register	Offset
SWREG460	730h

15.3.2.1.402.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.402.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 1, No.15.
23-16 BF_16	Denoise Filter Parameters Set 1, No.14.
15-8 BF_8	Denoise Filter Parameters Set 1, No.13.
7-0 BF_0	Denoise Filter Parameters Set 1, No.12.

15.3.2.1.403 VPU H1 Register 461 (SWREG461)

15.3.2.1.403.1 Offset

Register	Offset
SWREG461	734h

15.3.2.1.403.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.403.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 2, No.3.
23-16 BF_16	Denoise Filter Parameters Set 2, No.2.
15-8 BF_8	Denoise Filter Parameters Set 2, No.1.
7-0 BF_0	Denoise Filter Parameters Set 2, No.0.

15.3.2.1.404 VPU H1 Register 462 (SWREG462)

15.3.2.1.404.1 Offset

Register	Offset
SWREG462	738h

15.3.2.1.404.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.404.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 2, No.7.
23-16 BF_16	Denoise Filter Parameters Set 2, No.6.
15-8 BF_8	Denoise Filter Parameters Set 2, No.5.
7-0 BF_0	Denoise Filter Parameters Set 2, No.4.

15.3.2.1.405 VPU H1 Register 463 (SWREG463)

15.3.2.1.405.1 Offset

Register	Offset
SWREG463	73Ch

15.3.2.1.405.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.405.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 2, No.11.
23-16 BF_16	Denoise Filter Parameters Set 2, No.10.
15-8 BF_8	Denoise Filter Parameters Set 2, No.9.
7-0 BF_0	Denoise Filter Parameters Set 2, No.8.

15.3.2.1.406 VPU H1 Register 464 (SWREG464)

15.3.2.1.406.1 Offset

Register	Offset
SWREG464	740h

15.3.2.1.406.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.406.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 2, No.15.
23-16 BF_16	Denoise Filter Parameters Set 2, No.14.
15-8 BF_8	Denoise Filter Parameters Set 2, No.13.
7-0 BF_0	Denoise Filter Parameters Set 2, No.12.

15.3.2.1.407 VPU H1 Register 465 (SWREG465)

15.3.2.1.407.1 Offset

Register	Offset
SWREG465	744h

15.3.2.1.407.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.407.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.3.
23-16 BF_16	Denoise Filter Parameters Set 3, No.2.
15-8 BF_8	Denoise Filter Parameters Set 3, No.1.
7-0 BF_0	Denoise Filter Parameters Set 3, No.0.

15.3.2.1.408 VPU H1 Register 466 (SWREG466)

15.3.2.1.408.1 Offset

Register	Offset
SWREG466	748h

15.3.2.1.408.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.408.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.7.
23-16 BF_16	Denoise Filter Parameters Set 3, No.6.
15-8 BF_8	Denoise Filter Parameters Set 3, No.5.
7-0 BF_0	Denoise Filter Parameters Set 3, No.4.

15.3.2.1.409 VPU H1 Register 467 (SWREG467)

15.3.2.1.409.1 Offset

Register	Offset
SWREG467	74Ch

15.3.2.1.409.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.409.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.11.
23-16 BF_16	Denoise Filter Parameters Set 3, No.10.
15-8 BF_8	Denoise Filter Parameters Set 3, No.9.
7-0 BF_0	Denoise Filter Parameters Set 3, No.8.

15.3.2.1.410 VPU H1 Register 468 (SWREG468)

15.3.2.1.410.1 Offset

Register	Offset
SWREG468	750h

15.3.2.1.410.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.410.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.15.
23-16 BF_16	Denoise Filter Parameters Set 3, No.14.
15-8 BF_8	Denoise Filter Parameters Set 3, No.13.
7-0 BF_0	Denoise Filter Parameters Set 3, No.12.

15.3.2.1.411 VPU H1 Register 469 (SWREG469)

15.3.2.1.411.1 Offset

Register	Offset
SWREG469	754h

15.3.2.1.411.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.411.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.19.
23-16 BF_16	Denoise Filter Parameters Set 3, No.18.
15-8 BF_8	Denoise Filter Parameters Set 3, No.17.
7-0 BF_0	Denoise Filter Parameters Set 3, No.16.

15.3.2.1.412 VPU H1 Register 470 (SWREG470)

15.3.2.1.412.1 Offset

Register	Offset
SWREG470	758h

15.3.2.1.412.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.412.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.23.
23-16 BF_16	Denoise Filter Parameters Set 3, No.22.
15-8 BF_8	Denoise Filter Parameters Set 3, No.21.
7-0 BF_0	Denoise Filter Parameters Set 3, No.20.

15.3.2.1.413 VPU H1 Register 471 (SWREG471)

15.3.2.1.413.1 Offset

Register	Offset
SWREG471	75Ch

15.3.2.1.413.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.413.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.27.
23-16 BF_16	Denoise Filter Parameters Set 3, No.26.
15-8 BF_8	Denoise Filter Parameters Set 3, No.25.
7-0 BF_0	Denoise Filter Parameters Set 3, No.24.

15.3.2.1.414 VPU H1 Register 472 (SWREG472)

15.3.2.1.414.1 Offset

Register	Offset
SWREG472	760h

15.3.2.1.414.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.414.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.31.
23-16 BF_16	Denoise Filter Parameters Set 3, No.30.
15-8 BF_8	Denoise Filter Parameters Set 3, No.29.
7-0 BF_0	Denoise Filter Parameters Set 3, No.28.

15.3.2.1.415 VPU H1 Register 473 (SWREG473)

15.3.2.1.415.1 Offset

Register	Offset
SWREG473	764h

15.3.2.1.415.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.415.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.35.
23-16 BF_16	Denoise Filter Parameters Set 3, No.34.
15-8 BF_8	Denoise Filter Parameters Set 3, No.33.
7-0 BF_0	Denoise Filter Parameters Set 3, No.32.

15.3.2.1.416 VPU H1 Register 474 (SWREG474)

15.3.2.1.416.1 Offset

Register	Offset
SWREG474	768h

15.3.2.1.416.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.416.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.39.
23-16 BF_16	Denoise Filter Parameters Set 3, No.38.
15-8 BF_8	Denoise Filter Parameters Set 3, No.37.
7-0 BF_0	Denoise Filter Parameters Set 3, No.36.

15.3.2.1.417 VPU H1 Register 475 (SWREG475)

15.3.2.1.417.1 Offset

Register	Offset
SWREG475	76Ch

15.3.2.1.417.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.417.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.43.
23-16 BF_16	Denoise Filter Parameters Set 3, No.42.
15-8 BF_8	Denoise Filter Parameters Set 3, No.41.
7-0 BF_0	Denoise Filter Parameters Set 3, No.40.

15.3.2.1.418 VPU H1 Register 476 (SWREG476)

15.3.2.1.418.1 Offset

Register	Offset
SWREG476	770h

15.3.2.1.418.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.418.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.47.
23-16 BF_16	Denoise Filter Parameters Set 3, No.46.
15-8 BF_8	Denoise Filter Parameters Set 3, No.45.
7-0 BF_0	Denoise Filter Parameters Set 3, No.44.

15.3.2.1.419 VPU H1 Register 477 (SWREG477)

15.3.2.1.419.1 Offset

Register	Offset
SWREG477	774h

15.3.2.1.419.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.419.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.51.
23-16 BF_16	Denoise Filter Parameters Set 3, No.50.
15-8 BF_8	Denoise Filter Parameters Set 3, No.49.
7-0 BF_0	Denoise Filter Parameters Set 3, No.48.

15.3.2.1.420 VPU H1 Register 478 (SWREG478)

15.3.2.1.420.1 Offset

Register	Offset
SWREG478	778h

15.3.2.1.420.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.420.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.55.
23-16 BF_16	Denoise Filter Parameters Set 3, No.54.
15-8 BF_8	Denoise Filter Parameters Set 3, No.53.
7-0 BF_0	Denoise Filter Parameters Set 3, No.52.

15.3.2.1.421 VPU H1 Register 479 (SWREG479)

15.3.2.1.421.1 Offset

Register	Offset
SWREG479	77Ch

15.3.2.1.421.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.421.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.59.
23-16 BF_16	Denoise Filter Parameters Set 3, No.58.
15-8 BF_8	Denoise Filter Parameters Set 3, No.57.
7-0 BF_0	Denoise Filter Parameters Set 3, No.56.

15.3.2.1.422 VPU H1 Register 480 (SWREG480)

15.3.2.1.422.1 Offset

Register	Offset
SWREG480	780h

15.3.2.1.422.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.422.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 3, No.63.
23-16 BF_16	Denoise Filter Parameters Set 3, No.62.
15-8 BF_8	Denoise Filter Parameters Set 3, No.61.
7-0 BF_0	Denoise Filter Parameters Set 3, No.60.

15.3.2.1.423 VPU H1 Register 481 (SWREG481)

15.3.2.1.423.1 Offset

Register	Offset
SWREG481	784h

15.3.2.1.423.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.423.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.3.
23-16 BF_16	Denoise Filter Parameters Set 4, No.2.
15-8 BF_8	Denoise Filter Parameters Set 4, No.1.
7-0 BF_0	Denoise Filter Parameters Set 4, No.0.

15.3.2.1.424 VPU H1 Register 482 (SWREG482)

15.3.2.1.424.1 Offset

Register	Offset
SWREG482	788h

15.3.2.1.424.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.424.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.7.
23-16 BF_16	Denoise Filter Parameters Set 4, No.6.
15-8 BF_8	Denoise Filter Parameters Set 4, No.5.
7-0 BF_0	Denoise Filter Parameters Set 4, No.4.

15.3.2.1.425 VPU H1 Register 483 (SWREG483)

15.3.2.1.425.1 Offset

Register	Offset
SWREG483	78Ch

15.3.2.1.425.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.425.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.11.
23-16 BF_16	Denoise Filter Parameters Set 4, No.10.
15-8 BF_8	Denoise Filter Parameters Set 4, No.9.
7-0 BF_0	Denoise Filter Parameters Set 4, No.8.

15.3.2.1.426 VPU H1 Register 484 (SWREG484)

15.3.2.1.426.1 Offset

Register	Offset
SWREG484	790h

15.3.2.1.426.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.426.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.15.
23-16 BF_16	Denoise Filter Parameters Set 4, No.14.
15-8 BF_8	Denoise Filter Parameters Set 4, No.13.
7-0 BF_0	Denoise Filter Parameters Set 4, No.12.

15.3.2.1.427 VPU H1 Register 485 (SWREG485)

15.3.2.1.427.1 Offset

Register	Offset
SWREG485	794h

15.3.2.1.427.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.427.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.19.
23-16 BF_16	Denoise Filter Parameters Set 4, No.18.
15-8 BF_8	Denoise Filter Parameters Set 4, No.17.
7-0 BF_0	Denoise Filter Parameters Set 4, No.16.

15.3.2.1.428 VPU H1 Register 486 (SWREG486)

15.3.2.1.428.1 Offset

Register	Offset
SWREG486	798h

15.3.2.1.428.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.428.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.23.
23-16 BF_16	Denoise Filter Parameters Set 4, No.22.
15-8 BF_8	Denoise Filter Parameters Set 4, No.21.
7-0 BF_0	Denoise Filter Parameters Set 4, No.20.

15.3.2.1.429 VPU H1 Register 487 (SWREG487)

15.3.2.1.429.1 Offset

Register	Offset
SWREG487	79Ch

15.3.2.1.429.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.429.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.27.
23-16 BF_16	Denoise Filter Parameters Set 4, No.26.
15-8 BF_8	Denoise Filter Parameters Set 4, No.25.
7-0 BF_0	Denoise Filter Parameters Set 4, No.24.

15.3.2.1.430 VPU H1 Register 488 (SWREG488)

15.3.2.1.430.1 Offset

Register	Offset
SWREG488	7A0h

15.3.2.1.430.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.430.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.31.
23-16 BF_16	Denoise Filter Parameters Set 4, No.30.
15-8 BF_8	Denoise Filter Parameters Set 4, No.29.
7-0 BF_0	Denoise Filter Parameters Set 4, No.28.

15.3.2.1.431 VPU H1 Register 489 (SWREG489)

15.3.2.1.431.1 Offset

Register	Offset
SWREG489	7A4h

15.3.2.1.431.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.431.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.35.
23-16 BF_16	Denoise Filter Parameters Set 4, No.34.
15-8 BF_8	Denoise Filter Parameters Set 4, No.33.
7-0 BF_0	Denoise Filter Parameters Set 4, No.32.

15.3.2.1.432 VPU H1 Register 490 (SWREG490)

15.3.2.1.432.1 Offset

Register	Offset
SWREG490	7A8h

15.3.2.1.432.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.432.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.39.
23-16 BF_16	Denoise Filter Parameters Set 4, No.38.
15-8 BF_8	Denoise Filter Parameters Set 4, No.37.
7-0 BF_0	Denoise Filter Parameters Set 4, No.36.

15.3.2.1.433 VPU H1 Register 491 (SWREG491)

15.3.2.1.433.1 Offset

Register	Offset
SWREG491	7ACh

15.3.2.1.433.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.433.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.43.
23-16 BF_16	Denoise Filter Parameters Set 4, No.42.
15-8 BF_8	Denoise Filter Parameters Set 4, No.41.
7-0 BF_0	Denoise Filter Parameters Set 4, No.40.

15.3.2.1.434 VPU H1 Register 492 (SWREG492)

15.3.2.1.434.1 Offset

Register	Offset
SWREG492	7B0h

15.3.2.1.434.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.434.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.47.
23-16 BF_16	Denoise Filter Parameters Set 4, No.46.
15-8 BF_8	Denoise Filter Parameters Set 4, No.45.
7-0 BF_0	Denoise Filter Parameters Set 4, No.44.

15.3.2.1.435 VPU H1 Register 493 (SWREG493)

15.3.2.1.435.1 Offset

Register	Offset
SWREG493	7B4h

15.3.2.1.435.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.435.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.51.
23-16 BF_16	Denoise Filter Parameters Set 4, No.50.
15-8 BF_8	Denoise Filter Parameters Set 4, No.49.
7-0 BF_0	Denoise Filter Parameters Set 4, No.48.

15.3.2.1.436 VPU H1 Register 494 (SWREG494)

15.3.2.1.436.1 Offset

Register	Offset
SWREG494	7B8h

15.3.2.1.436.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.436.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.55.
23-16 BF_16	Denoise Filter Parameters Set 4, No.54.
15-8 BF_8	Denoise Filter Parameters Set 4, No.53.
7-0 BF_0	Denoise Filter Parameters Set 4, No.52.

15.3.2.1.437 VPU H1 Register 495 (SWREG495)

15.3.2.1.437.1 Offset

Register	Offset
SWREG495	7BCh

15.3.2.1.437.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.437.3 Fields

Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.59.
23-16 BF_16	Denoise Filter Parameters Set 4, No.58.
15-8 BF_8	Denoise Filter Parameters Set 4, No.57.
7-0 BF_0	Denoise Filter Parameters Set 4, No.56.

15.3.2.1.438 VPU H1 Register 496 (SWREG496)

15.3.2.1.438.1 Offset

Register	Offset
SWREG496	7C0h

15.3.2.1.438.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	BF_24								BF_16							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	BF_8								BF_0							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.1.438.3 Fields

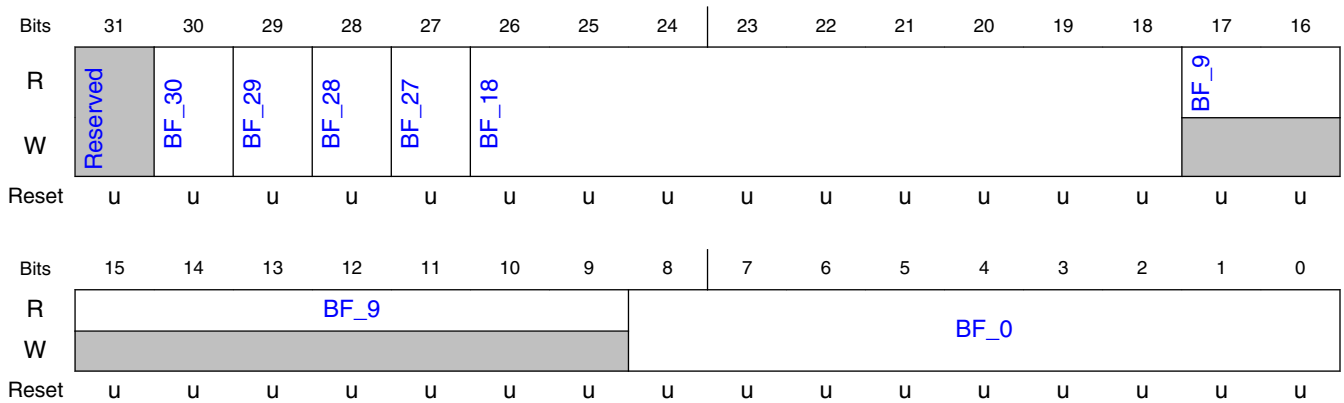
Field	Function
31-24 BF_24	Denoise Filter Parameters Set 4, No.63.
23-16 BF_16	Denoise Filter Parameters Set 4, No.62.
15-8 BF_8	Denoise Filter Parameters Set 4, No.61.
7-0 BF_0	Denoise Filter Parameters Set 4, No.60.

15.3.2.1.439 VPU H1 Register 497 (SWREG497)

15.3.2.1.439.1 Offset

Register	Offset
SWREG497	7C4h

15.3.2.1.439.2 Diagram



15.3.2.1.439.3 Fields

Field	Function
31 —	Reserved.
30 BF_30	Enable input Buffer underflow interrupt.
29 BF_29	Low Latency Hardware Interface Enable. Use hardware handshaking.
28 BF_28	Input Buffer Loop Back Enable. valid only when sw_num_mb_rows_per_sync is smaller than frame size.
27 BF_27	Line Buffer Enable
26-18 BF_18	The number of MB rows featched by encoder which will introduce HW shakehand happens.
17-9 BF_9	The number of MB rows that is already read out from the input buffer
8-0 BF_0	The number of MB rows that is already filled in the input buffer

15.3.2.2 VPU_H1 H264 encoder registers descriptions

15.3.2.2.1 VPU H1 H264 encoder registers memory map

VPU_H1_H264 base address: 3832_0000h

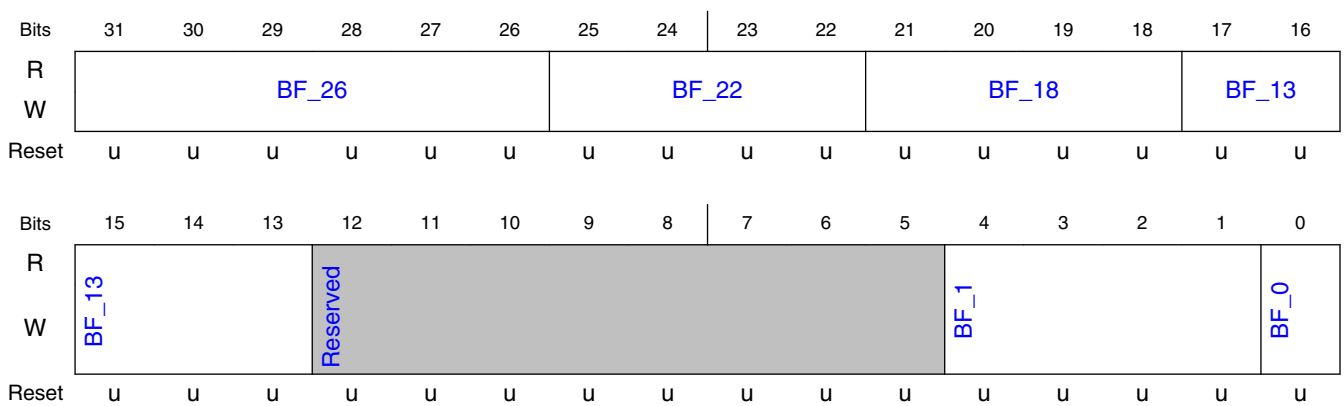
Offset	Register	Width (In bits)	Access	Reset value
40h	VPU H1 Register 16 - for H.264 (SWREG16_H264)	32	RW	Table 15-9
44h	VPU H1 Register 17 - for H.264 (SWREG17_H264)	32	RW	Table 15-9
68h	VPU H1 Register 26 - for H.264 (SWREG26_H264)	32	RW	Table 15-9
6Ch	VPU H1 Register 27 - for H.264 (SWREG27_H264)	32	RW	Table 15-9
70h	VPU H1 Register 28 - for H.264 (SWREG28_H264)	32	RW	Table 15-9
74h	VPU H1 Register 29 - for H.264 (SWREG29_H264)	32	RW	Table 15-9
78h	VPU H1 Register 30 - for H.264 (SWREG30_H264)	32	RW	Table 15-9
7Ch	VPU H1 Register 31 - for H.264 (SWREG31_H264)	32	RW	Table 15-9
80h	VPU H1 Register 32 - for H.264 (SWREG32_H264)	32	RW	Table 15-9
84h	VPU H1 Register 33 - for H.264 (SWREG33_H264)	32	RW	Table 15-9
88h	VPU H1 Register 34 - for H.264 (SWREG34_H264)	32	RW	Table 15-9
8Ch	VPU H1 Register 35 - for H.264 (SWREG35_H264)	32	RW	Table 15-9
90h	VPU H1 Register 36 - for H.264 (SWREG36_H264)	32	RW	Table 15-9
E8h	VPU H1 Register 58 - for H.264 (SWREG58_H264)	32	RW	Table 15-9
ECh	VPU H1 Register 59 - for H.264 (SWREG59_H264)	32	RW	Table 15-9

15.3.2.2.2 VPU H1 Register 16 - for H.264 (SWREG16_H264)

15.3.2.2.2.1 Offset

Register	Offset
SWREG16_H264	40h

15.3.2.2.2.2 Diagram



15.3.2.2.2.3 Fields

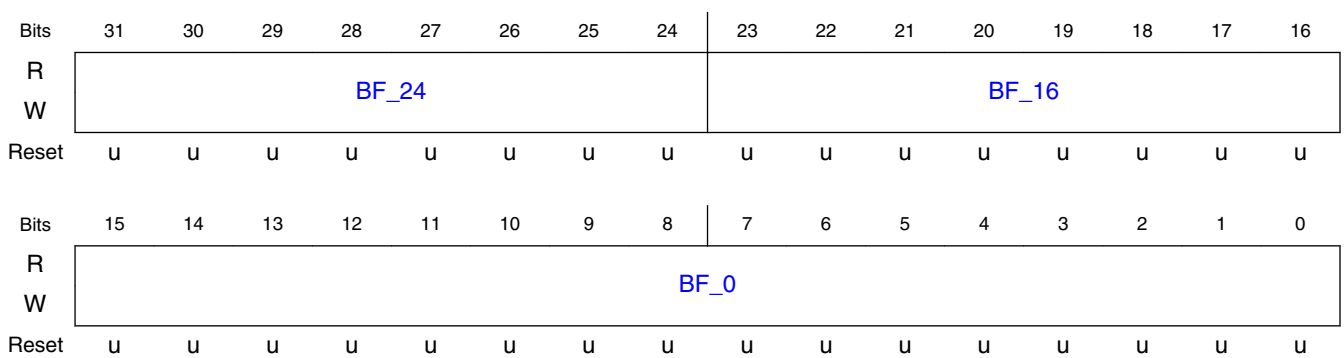
Field	Function
31-26 BF_26	H.264 Pic init qp in PPS [0..51]
25-22 BF_22	H.264 Slice filter alpha c0 offset div2 [-6..6]
21-18 BF_18	H.264 Slice filter beta offset div2 [-6..6]
17-13 BF_13	H.264 Chroma qp index offset [-12..12]
12-5 —	Reserved.
4-1 BF_1	H.264 IDR picture ID
0 BF_0	H.264 Constrained intra prediction enable. constIntraPred

15.3.2.2.3 VPU H1 Register 17 - for H.264 (SWREG17_H264)

15.3.2.2.3.1 Offset

Register	Offset
SWREG17_H264	44h

15.3.2.2.3.2 Diagram



15.3.2.2.3.3 Fields

Field	Function
31-24 BF_24	H.264 pic_parameter_set_id
23-16 BF_16	H.264 Intra prediction previous 4x4 mode favor
15-0 BF_0	H.264 Frame num

15.3.2.2.4 VPU H1 Register 26 - for H.264 (SWREG26_H264)

15.3.2.2.4.1 Offset

Register	Offset
SWREG26_H264	68h

15.3.2.2.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.2.4.3 Fields

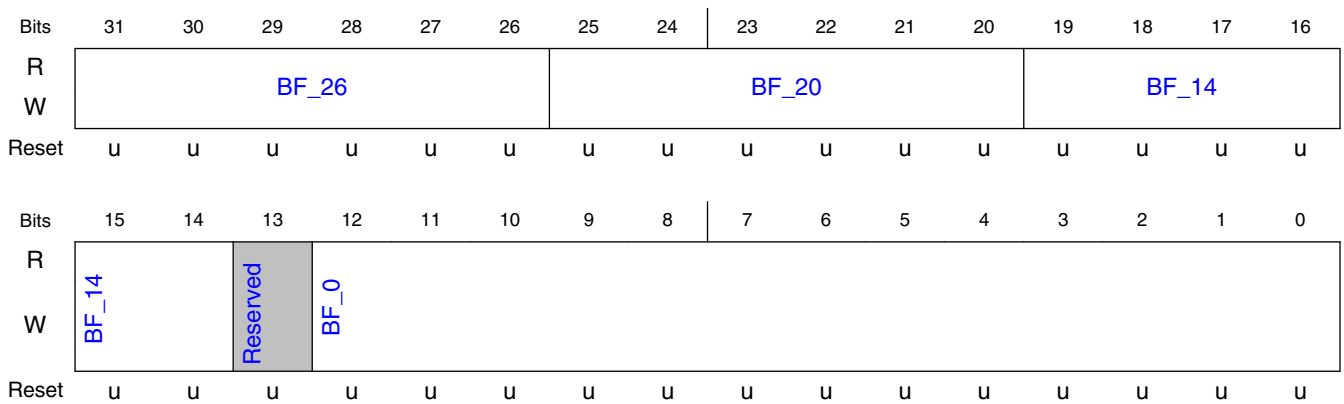
Field	Function
31-0 BF_0	Intra slice bitmap for slices 64..95. LSB=slice64. MSB=slice95. 1=intra.

15.3.2.2.5 VPU H1 Register 27 - for H.264 (SWREG27_H264)

15.3.2.2.5.1 Offset

Register	Offset
SWREG27_H264	6Ch

15.3.2.2.5.2 Diagram



15.3.2.2.5.3 Fields

Field	Function
31-26 BF_26	H.264 Initial QP. qpLum [0..51]
25-20 BF_20	H.264 Maximum QP. qpMax [0..51]
19-14 BF_14	H.264 Minimum QP. qpMin [0..51]
13 —	Reserved.
12-0 BF_0	H.264 Checkpoint distance (mb). 0=disabled [0..8191]

15.3.2.2.6 VPU H1 Register 28 - for H.264 (SWREG28_H264)

15.3.2.2.6.1 Offset

Register	Offset
SWREG28_H264	70h

15.3.2.2.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_16															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.2.6.3 Fields

Field	Function
31-16 BF_16	H.264 Checkpoint 1 word target/usage div32 [0..65535]
15-0 BF_0	H.264 Checkpoint 2 word target/usage div32 [0..65535]

15.3.2.2.7 VPU H1 Register 29 - for H.264 (SWREG29_H264)

15.3.2.2.7.1 Offset

Register	Offset
SWREG29_H264	74h

15.3.2.2.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_16															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.2.7.3 Fields

Field	Function
31-16 BF_16	H.264 Checkpoint 3 word target/usage div32 [0..65535]
15-0 BF_0	H.264 Checkpoint 4 word target/usage div32 [0..65535]

15.3.2.2.8 VPU H1 Register 30 - for H.264 (SWREG30_H264)

15.3.2.2.8.1 Offset

Register	Offset
SWREG30_H264	78h

15.3.2.2.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_16															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.2.8.3 Fields

Field	Function
31-16 BF_16	H.264 Checkpoint 5 word target/usage div32 [0..65535]
15-0 BF_0	H.264 Checkpoint 6 word target/usage div32 [0..65535]

15.3.2.2.9 VPU H1 Register 31 - for H.264 (SWREG31_H264)

15.3.2.2.9.1 Offset

Register	Offset
SWREG31_H264	7Ch

15.3.2.2.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_16															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.2.9.3 Fields

Field	Function
31-16 BF_16	H.264 Checkpoint 7 word target/usage div32 [0..65535]
15-0 BF_0	H.264 Checkpoint 8 word target/usage div32 [0..65535]

15.3.2.2.10 VPU H1 Register 32 - for H.264 (SWREG32_H264)

15.3.2.2.10.1 Offset

Register	Offset
SWREG32_H264	80h

15.3.2.2.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_16															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.2.10.3 Fields

Field	Function
31-16 BF_16	H.264 Checkpoint 9 word target/usage div32 [0..65535]
15-0 BF_0	H.264 Checkpoint 10 word target/usage div32 [0..65535]

15.3.2.2.11 VPU H1 Register 33 - for H.264 (SWREG33_H264)

15.3.2.2.11.1 Offset

Register	Offset
SWREG33_H264	84h

15.3.2.2.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_16															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.2.11.3 Fields

Field	Function
31-16 BF_16	H.264 Checkpoint word error 1 div4 [-32768..32767]
15-0 BF_0	H.264 Checkpoint word error 2 div4 [-32768..32767]

15.3.2.2.12 VPU H1 Register 34 - for H.264 (SWREG34_H264)

15.3.2.2.12.1 Offset

Register	Offset
SWREG34_H264	88h

15.3.2.2.12.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_16															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.2.12.3 Fields

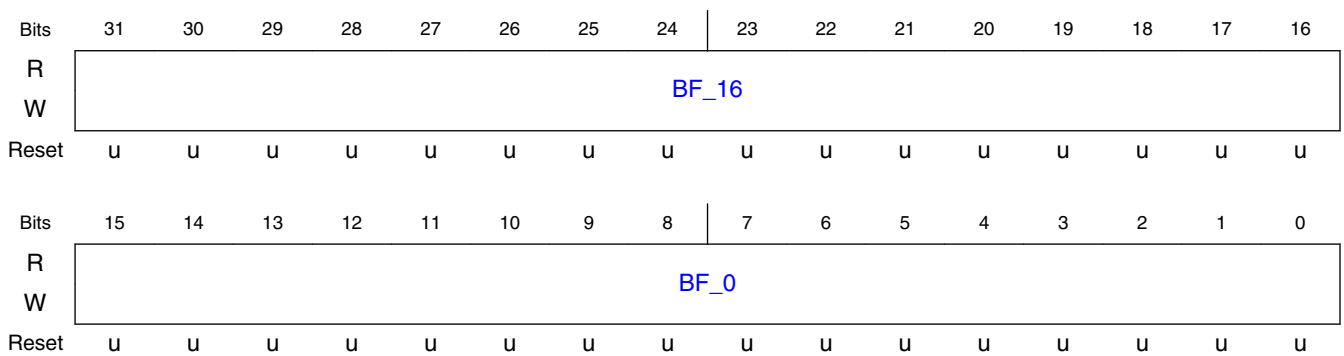
Field	Function
31-16 BF_16	H.264 Checkpoint word error 3 div4 [-32768..32767]
15-0 BF_0	H.264 Checkpoint word error 4 div4 [-32768..32767]

15.3.2.2.13 VPU H1 Register 35 - for H.264 (SWREG35_H264)

15.3.2.2.13.1 Offset

Register	Offset
SWREG35_H264	8Ch

15.3.2.2.13.2 Diagram



15.3.2.2.13.3 Fields

Field	Function
31-16 BF_16	H.264 Checkpoint word error 5 div4 [-32768..32767]
15-0 BF_0	H.264 Checkpoint word error 6 div4 [-32768..32767]

15.3.2.2.14 VPU H1 Register 36 - for H.264 (SWREG36_H264)

15.3.2.2.14.1 Offset

Register	Offset
SWREG36_H264	90h

15.3.2.2.14.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				BF_24				BF_20				BF_16			
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_12				BF_8				BF_4				BF_0			
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.2.14.3 Fields

Field	Function
31-28 —	Reserved.
27-24 BF_24	H.264 Checkpoint delta QP 1 [-8..7]
23-20 BF_20	H.264 Checkpoint delta QP 2 [-8..7]
19-16 BF_16	H.264 Checkpoint delta QP 3 [-8..7]
15-12 BF_12	H.264 Checkpoint delta QP 4 [-8..7]
11-8 BF_8	H.264 Checkpoint delta QP 5 [-8..7]
7-4 BF_4	H.264 Checkpoint delta QP 6 [-8..7]
3-0 BF_0	H.264 Checkpoint delta QP 7 [-8..7]

15.3.2.2.15 VPU H1 Register 58 - for H.264 (SWREG58_H264)

15.3.2.2.15.1 Offset

Register	Offset
SWREG58_H264	E8h

15.3.2.2.15.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.2.15.3 Fields

Field	Function
31-0 BF_0	Intra slice bitmap for slices 0..31. LSB=slice0. MSB=slice31. 1=intra.

15.3.2.2.16 VPU H1 Register 59 - for H.264 (SWREG59_H264)

15.3.2.2.16.1 Offset

Register	Offset
SWREG59_H264	ECh

15.3.2.2.16.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.2.16.3 Fields

Field	Function
31-0 BF_0	Intra slice bitmap for slices 32..63. LSB=slice32. MSB=slice63. 1=intra.

15.3.2.3 VPU_H1 VP8 encoder registers descriptions

15.3.2.3.1 VPU H1 VP8 encoder registers memory map

VPU_H1_VP8 base address: 3832_0000h

Offset	Register	Width (In bits)	Access	Reset value
68h	VPU H1 Register 26 - for VP8 (SWREG26_VP8)	32	RW	Table 15-9
6Ch	VPU H1 Register 27 - for VP8 (SWREG27_VP8)	32	RW	Table 15-9
70h	VPU H1 Register 28 - for VP8 (SWREG28_VP8)	32	RW	Table 15-9
74h	VPU H1 Register 29 - for VP8 (SWREG29_VP8)	32	RW	Table 15-9
78h	VPU H1 Register 30 - for VP8 (SWREG30_VP8)	32	RW	Table 15-9
7Ch	VPU H1 Register 31 - for VP8 (SWREG31_VP8)	32	RW	Table 15-9
80h	VPU H1 Register 32 - for VP8 (SWREG32_VP8)	32	RW	Table 15-9
84h	VPU H1 Register 33 - for VP8 (SWREG33_VP8)	32	RW	Table 15-9
88h	VPU H1 Register 34 - for VP8 (SWREG34_VP8)	32	RW	Table 15-9
8Ch	VPU H1 Register 35 - for VP8 (SWREG35_VP8)	32	RW	Table 15-9
90h	VPU H1 Register 36 - for VP8 (SWREG36_VP8)	32	RW	Table 15-9
E8h	VPU H1 Register 58 - for VP8 (SWREG58_VP8)	32	RW	Table 15-9

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
ECh	VPU H1 Register 59 - for VP8 (SWREG59_VP8)	32	RW	Table 15-9
100h	VPU H1 Register 64 - for VP8 (SWREG64_VP8)	32	RW	Table 15-9
104h	VPU H1 Register 65 - for VP8 (SWREG65_VP8)	32	RW	Table 15-9
108h	VPU H1 Register 66 - for VP8 (SWREG66_VP8)	32	RW	Table 15-9
10Ch	VPU H1 Register 67 - for VP8 (SWREG67_VP8)	32	RW	Table 15-9
110h	VPU H1 Register 68 - for VP8 (SWREG68_VP8)	32	RW	Table 15-9
114h	VPU H1 Register 69 - for VP8 (SWREG69_VP8)	32	RW	Table 15-9
118h	VPU H1 Register 70 - for VP8 (SWREG70_VP8)	32	RW	Table 15-9
11Ch	VPU H1 Register 71 - for VP8 (SWREG71_VP8)	32	RW	Table 15-9
120h	VPU H1 Register 72 - for VP8 (SWREG72_VP8)	32	RW	Table 15-9
124h	VPU H1 Register 73 - for VP8 (SWREG73_VP8)	32	RW	Table 15-9
128h	VPU H1 Register 74 - for VP8 (SWREG74_VP8)	32	RW	Table 15-9
12Ch	VPU H1 Register 75 - for VP8 (SWREG75_VP8)	32	RW	Table 15-9
130h	VPU H1 Register 76 - for VP8 (SWREG76_VP8)	32	RW	Table 15-9
134h	VPU H1 Register 77 - for VP8 (SWREG77_VP8)	32	RW	Table 15-9
138h	VPU H1 Register 78 - for VP8 (SWREG78_VP8)	32	RW	Table 15-9
13Ch	VPU H1 Register 79 - for VP8 (SWREG79_VP8)	32	RW	Table 15-9
140h	VPU H1 Register 80 - for VP8 (SWREG80_VP8)	32	RW	Table 15-9
144h	VPU H1 Register 81 - for VP8 (SWREG81_VP8)	32	RW	Table 15-9
148h	VPU H1 Register 82 - for VP8 (SWREG82_VP8)	32	RW	Table 15-9
14Ch	VPU H1 Register 83 - for VP8 (SWREG83_VP8)	32	RW	Table 15-9
150h	VPU H1 Register 84 - for VP8 (SWREG84_VP8)	32	RW	Table 15-9
154h	VPU H1 Register 85 - for VP8 (SWREG85_VP8)	32	RW	Table 15-9
158h	VPU H1 Register 86 - for VP8 (SWREG86_VP8)	32	RW	Table 15-9
15Ch	VPU H1 Register 87 - for VP8 (SWREG87_VP8)	32	RW	Table 15-9
160h	VPU H1 Register 88 - for VP8 (SWREG88_VP8)	32	RW	Table 15-9
164h	VPU H1 Register 89 - for VP8 (SWREG89_VP8)	32	RW	Table 15-9
168h	VPU H1 Register 90 - for VP8 (SWREG90_VP8)	32	RW	Table 15-9
16Ch	VPU H1 Register 91 - for VP8 (SWREG91_VP8)	32	RW	Table 15-9
170h	VPU H1 Register 92 - for VP8 (SWREG92_VP8)	32	RW	Table 15-9
174h	VPU H1 Register 93 - for VP8 (SWREG93_VP8)	32	RW	Table 15-9
178h	VPU H1 Register 94 - for VP8 (SWREG94_VP8)	32	RW	Table 15-9
17Ch	VPU H1 Register 95 - for VP8 (SWREG95_VP8)	32	RW	Table 15-9

15.3.2.3.2 VPU H1 Register 26 - for VP8 (SWREG26_VP8)

15.3.2.3.2.1 Offset

Register	Offset
SWREG26_VP8	68h

15.3.2.3.2.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.3.2.3 Fields

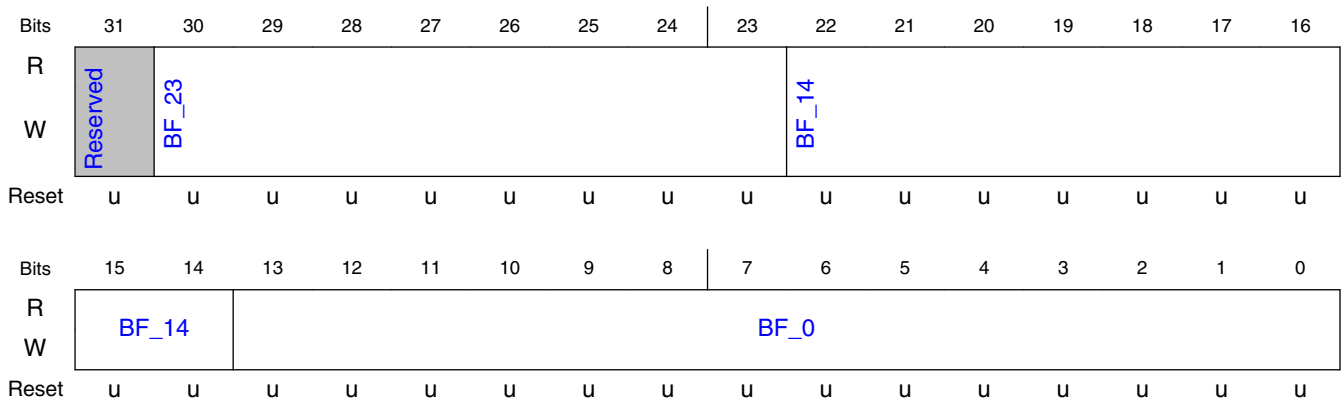
Field	Function
31-0 BF_0	Base address for VP8 counters for probability updates

15.3.2.3.3 VPU H1 Register 27 - for VP8 (SWREG27_VP8)

15.3.2.3.3.1 Offset

Register	Offset
SWREG27_VP8	6Ch

15.3.2.3.3.2 Diagram



15.3.2.3.3.3 Fields

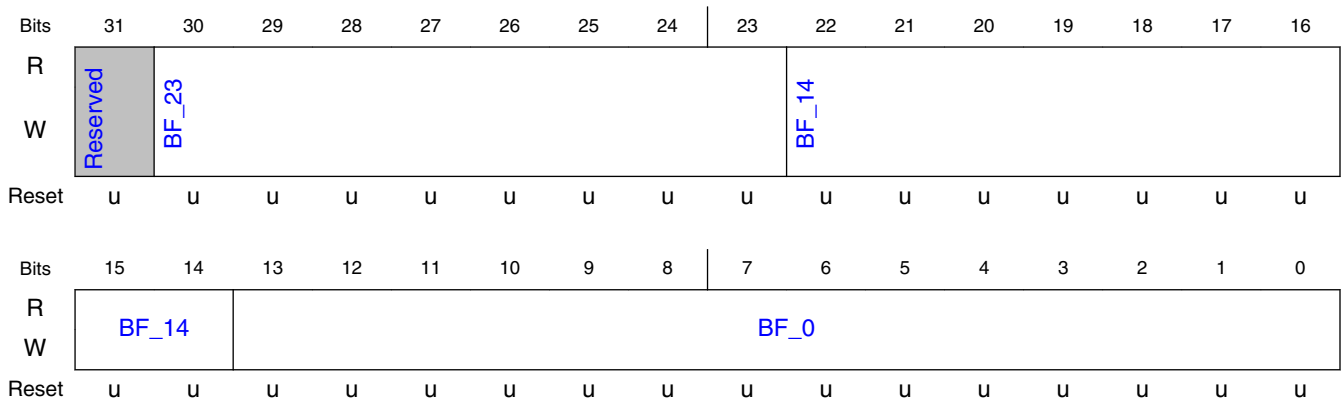
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 qpY1RoundDc 8b
22-14 BF_14	VP8 qpY1ZbinDc 9b
13-0 BF_0	VP8 qpY1QuantDc 14b

15.3.2.3.4 VPU H1 Register 28 - for VP8 (SWREG28_VP8)

15.3.2.3.4.1 Offset

Register	Offset
SWREG28_VP8	70h

15.3.2.3.4.2 Diagram



15.3.2.3.4.3 Fields

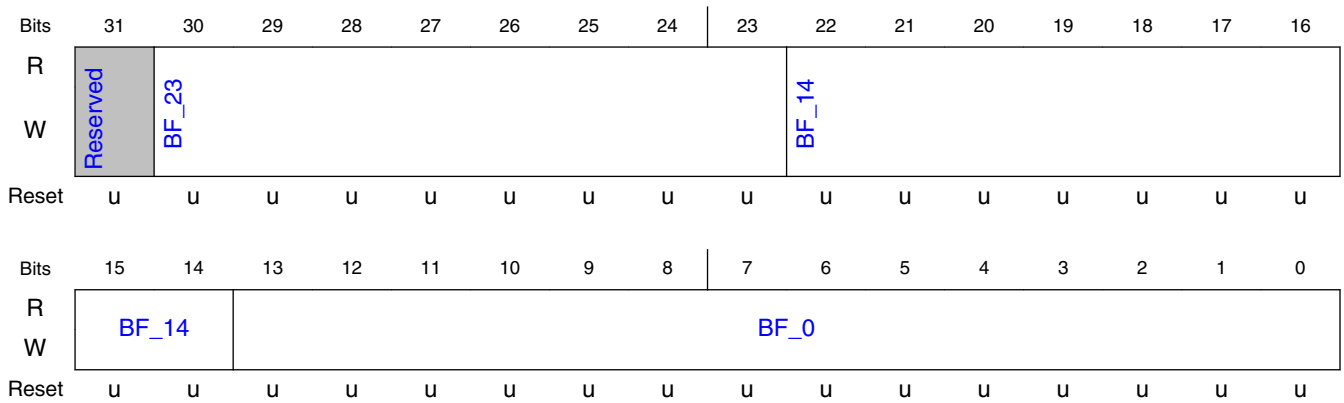
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 qpY1RoundAc 8b
22-14 BF_14	VP8 qpY1ZbinAc 9b
13-0 BF_0	VP8 qpY1QuantAc 14b

15.3.2.3.5 VPU H1 Register 29 - for VP8 (SWREG29_VP8)

15.3.2.3.5.1 Offset

Register	Offset
SWREG29_VP8	74h

15.3.2.3.5.2 Diagram



15.3.2.3.5.3 Fields

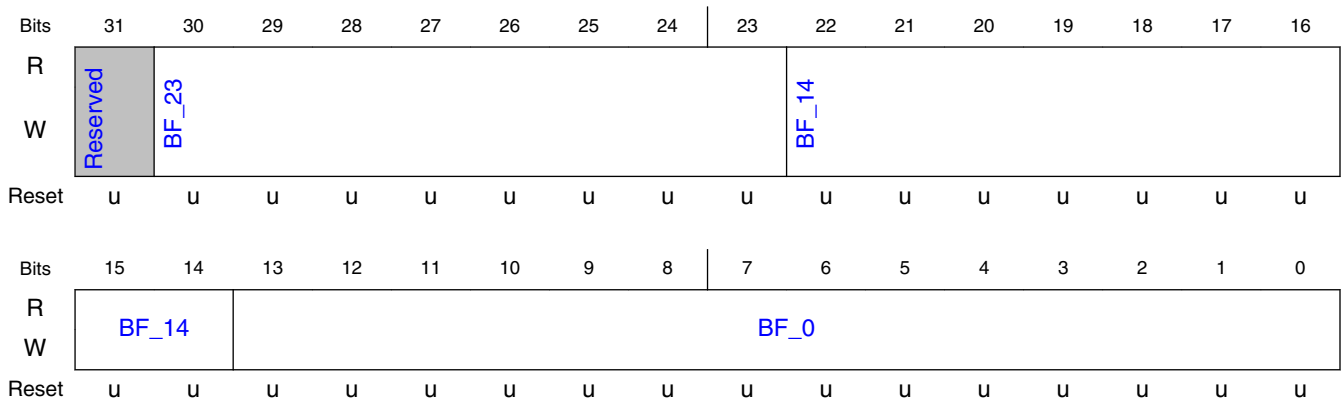
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 qpY2RoundDc 8b
22-14 BF_14	VP8 qpY2ZbinDc 9b
13-0 BF_0	VP8 qpY2QuantDc 14b

15.3.2.3.6 VPU H1 Register 30 - for VP8 (SWREG30_VP8)

15.3.2.3.6.1 Offset

Register	Offset
SWREG30_VP8	78h

15.3.2.3.6.2 Diagram



15.3.2.3.6.3 Fields

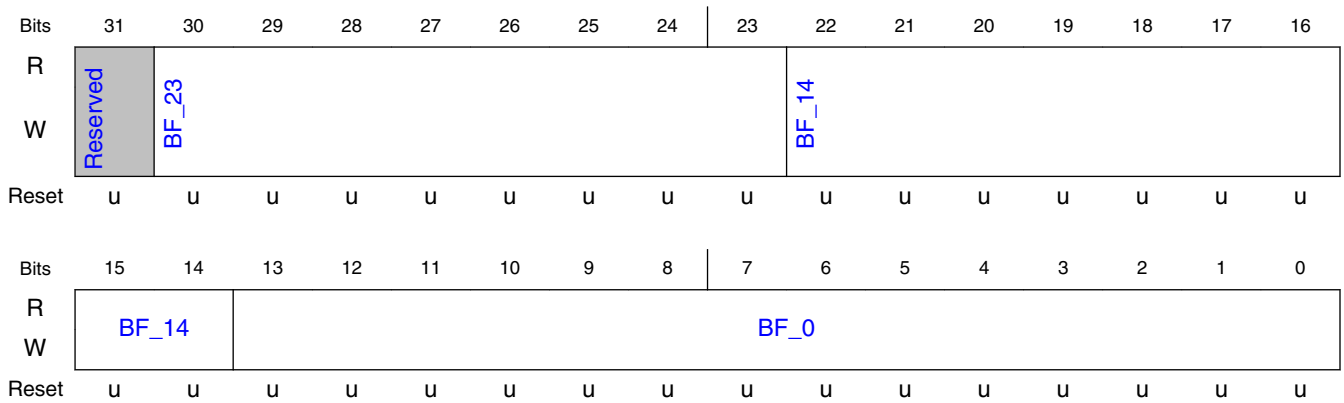
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 qpY2RoundAc 8b
22-14 BF_14	VP8 qpY2ZbinAc 9b
13-0 BF_0	VP8 qpY2QuantAc 14b

15.3.2.3.7 VPU H1 Register 31 - for VP8 (SWREG31_VP8)

15.3.2.3.7.1 Offset

Register	Offset
SWREG31_VP8	7Ch

15.3.2.3.7.2 Diagram



15.3.2.3.7.3 Fields

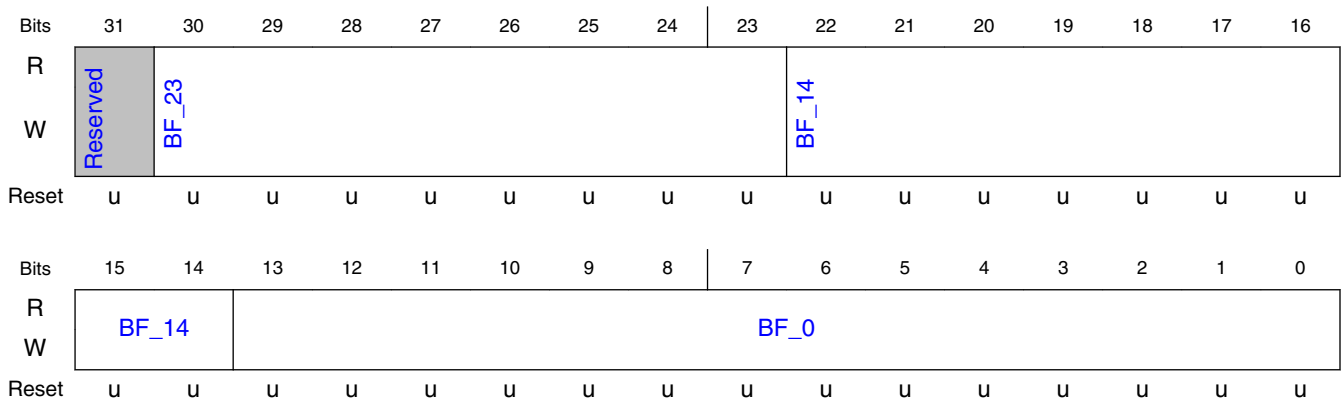
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 qpChRoundDc 8b
22-14 BF_14	VP8 qpChZbinDc 9b
13-0 BF_0	VP8 qpChQuantDc 14b

15.3.2.3.8 VPU H1 Register 32 - for VP8 (SWREG32_VP8)

15.3.2.3.8.1 Offset

Register	Offset
SWREG32_VP8	80h

15.3.2.3.8.2 Diagram



15.3.2.3.8.3 Fields

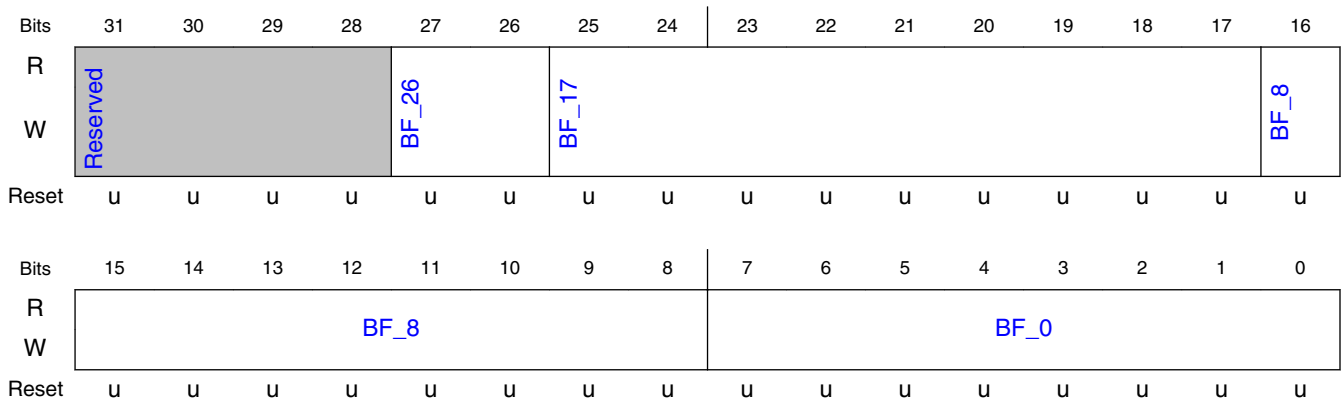
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 qpChRoundAc 8b
22-14 BF_14	VP8 qpChZbinAc 9b
13-0 BF_0	VP8 qpChQuantAc 14b

15.3.2.3.9 VPU H1 Register 33 - for VP8 (SWREG33_VP8)

15.3.2.3.9.1 Offset

Register	Offset
SWREG33_VP8	84h

15.3.2.3.9.2 Diagram



15.3.2.3.9.3 Fields

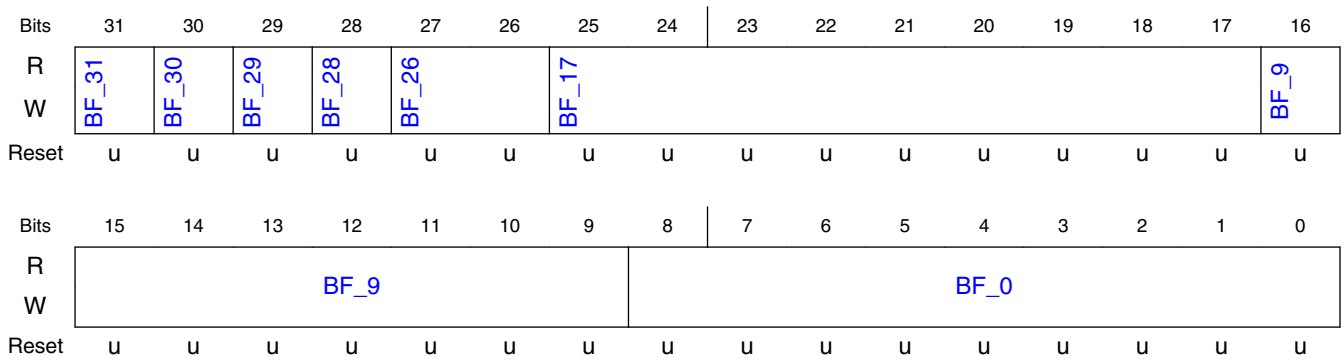
Field	Function
31-28 —	Reserved.
27-26 BF_26	VP8 mvRefIdx for first reference frame. 0=ipf. 1=grf. 2=arf.
25-17 BF_17	VP8 qpY2DequantDc 9b
16-8 BF_8	VP8 qpY1DequantAc 9b
7-0 BF_0	VP8 qpY1DequantDc 8b

15.3.2.3.10 VPU H1 Register 34 - for VP8 (SWREG34_VP8)

15.3.2.3.10.1 Offset

Register	Offset
SWREG34_VP8	88h

15.3.2.3.10.2 Diagram



15.3.2.3.10.3 Fields

Field	Function
31 BF_31	VP8 enable for deadzone.
30 BF_30	VP8 enable for segmentation map update. Map is different from previous frame and is written in stream.
29 BF_29	VP8 enable for segmentation. Segmentation map is stored in BaseSegmentMap.
28 BF_28	VP8 enable for second reference frame.
27-26 BF_26	VP8 mvRefIdx for second reference frame. 0=ipf. 1=grf. 2=arf.
25-17 BF_17	VP8 qpChDequantAc 9b
16-9 BF_9	VP8 qpChDequantDc 8b
8-0 BF_0	VP8 qpY2DequantAc 9b

15.3.2.3.11 VPU H1 Register 35 - for VP8 (SWREG35_VP8)

15.3.2.3.11.1 Offset

Register	Offset
SWREG35_VP8	8Ch

15.3.2.3.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.3.11.3 Fields

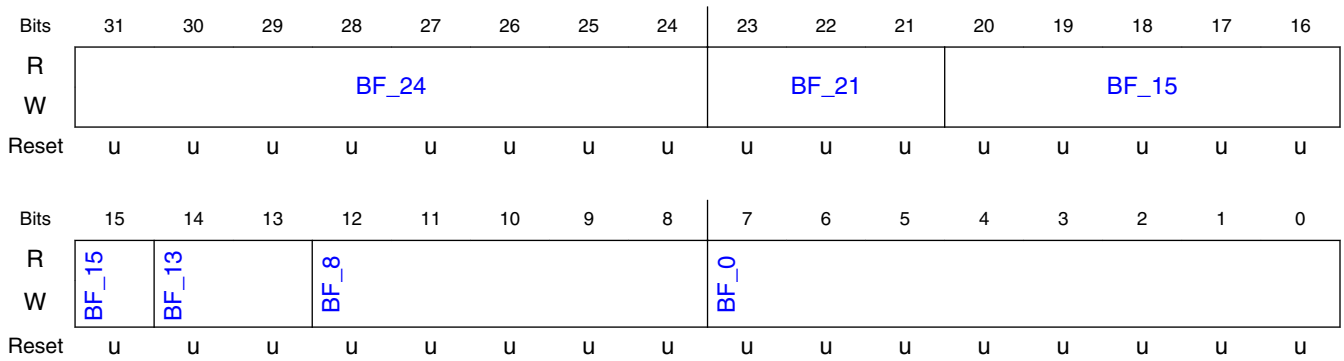
Field	Function
31-0 BF_0	VP8 boolEncValue

15.3.2.3.12 VPU H1 Register 36 - for VP8 (SWREG36_VP8)

15.3.2.3.12.1 Offset

Register	Offset
SWREG36_VP8	90h

15.3.2.3.12.2 Diagram



15.3.2.3.12.3 Fields

Field	Function
31-24 BF_24	VP8 Penalty value for second reference frame zero-mv [0..255]
23-21 BF_21	VP8 Deblocking filter sharpness [0..7]
20-15 BF_15	VP8 Deblocking filter level [0..63]
14-13 BF_13	VP8 DCT partition count. 0=1. 1=2. 2=4 [0..2].
12-8 BF_8	VP8 boolEncValueBitsMinus8 [0..23]
7-0 BF_0	VP8 boolEncRange [0..255]

15.3.2.3.13 VPU H1 Register 58 - for VP8 (SWREG58_VP8)

15.3.2.3.13.1 Offset

Register	Offset
SWREG58_VP8	E8h

15.3.2.3.13.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.3.13.3 Fields

Field	Function
31-0 BF_0	Base address for VP8 1st DCT partition

15.3.2.3.14 VPU H1 Register 59 - for VP8 (SWREG59_VP8)

15.3.2.3.14.1 Offset

Register	Offset
SWREG59_VP8	ECh

15.3.2.3.14.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.3.14.3 Fields

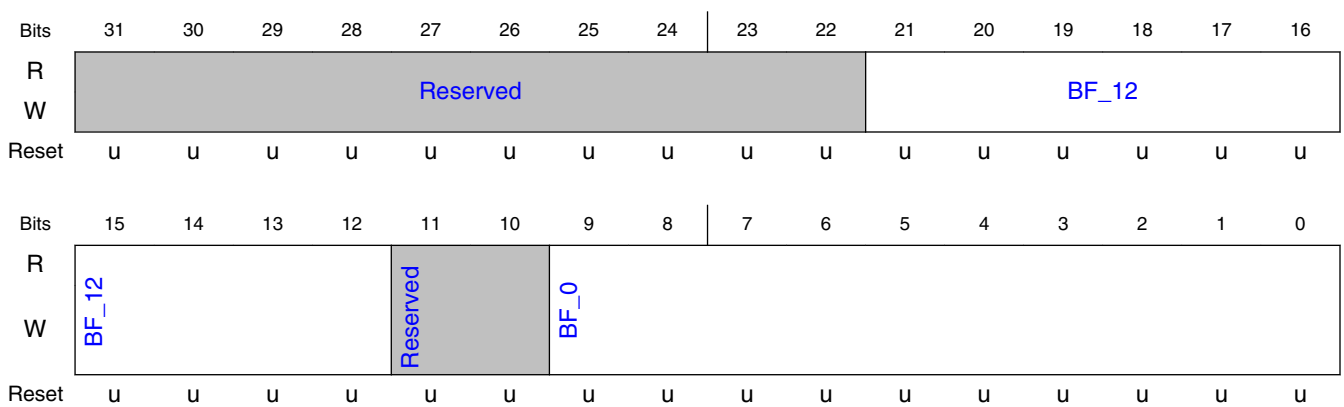
Field	Function
31-0 BF_0	Base address for VP8 2nd DCT partition

15.3.2.3.15 VPU H1 Register 64 - for VP8 (SWREG64_VP8)

15.3.2.3.15.1 Offset

Register	Offset
SWREG64_VP8	100h

15.3.2.3.15.2 Diagram



15.3.2.3.15.3 Fields

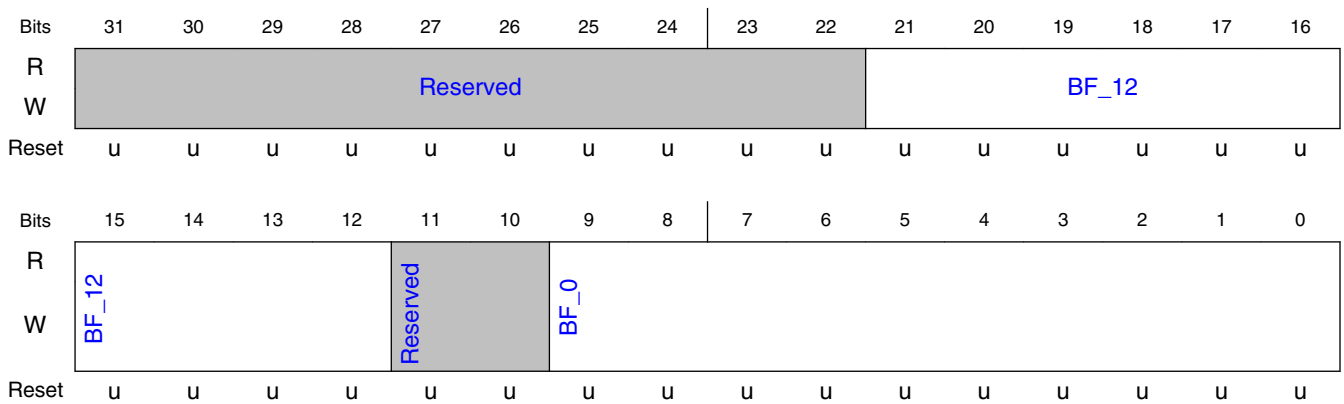
Field	Function
31-22 —	Reserved.
21-12 BF_12	VP8 intra 16x16 mode 1 penalty
11-10 —	Reserved.
9-0 BF_0	VP8 intra 16x16 mode 0 penalty

15.3.2.3.16 VPU H1 Register 65 - for VP8 (SWREG65_VP8)

15.3.2.3.16.1 Offset

Register	Offset
SWREG65_VP8	104h

15.3.2.3.16.2 Diagram



15.3.2.3.16.3 Fields

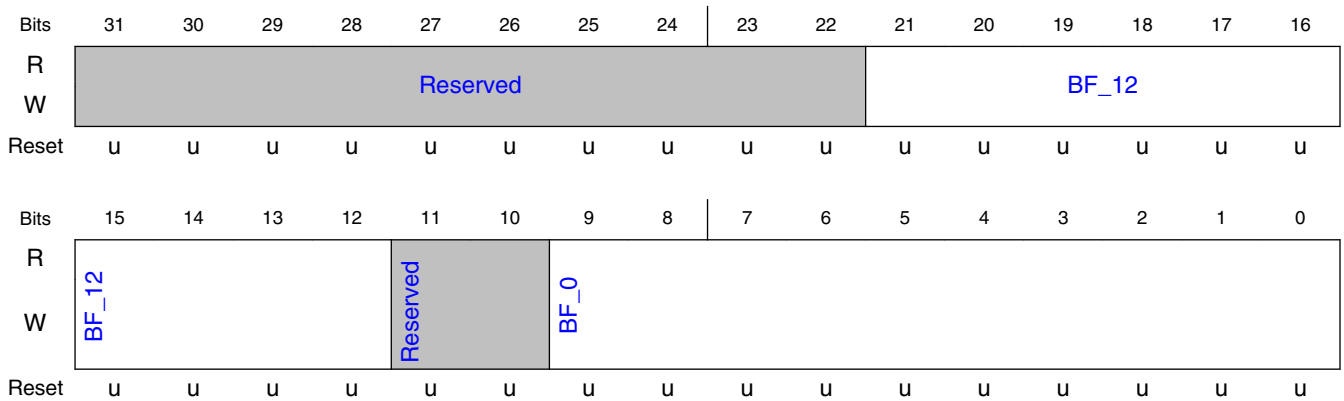
Field	Function
31-22 —	Reserved.
21-12 BF_12	VP8 intra 16x16 mode 3 penalty
11-10 —	Reserved.
9-0 BF_0	VP8 intra 16x16 mode 2 penalty

15.3.2.3.17 VPU H1 Register 66 - for VP8 (SWREG66_VP8)

15.3.2.3.17.1 Offset

Register	Offset
SWREG66_VP8	108h

15.3.2.3.17.2 Diagram



15.3.2.3.17.3 Fields

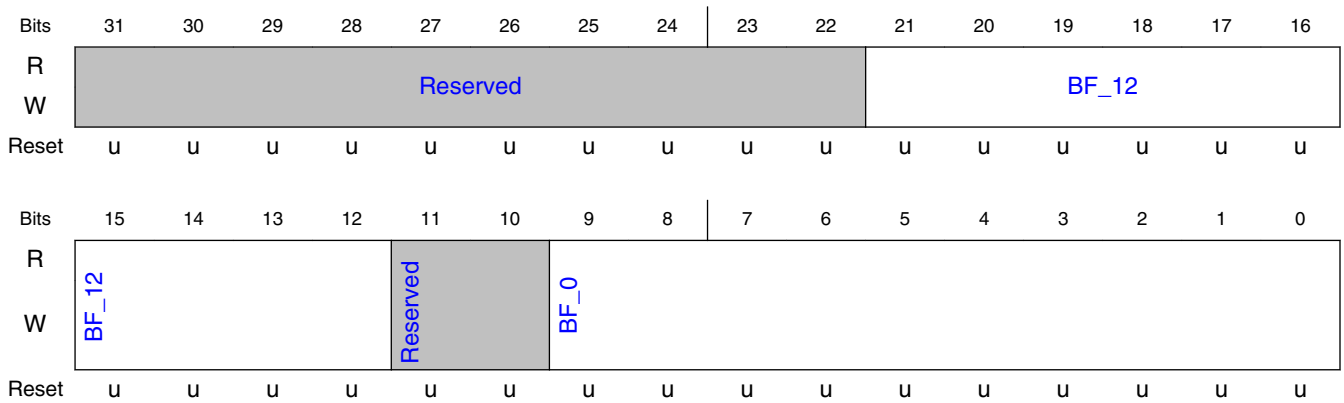
Field	Function
31-22 —	Reserved.
21-12 BF_12	VP8 intra 4x4 mode 1 penalty
11-10 —	Reserved.
9-0 BF_0	VP8 intra 4x4 mode 0 penalty

15.3.2.3.18 VPU H1 Register 67 - for VP8 (SWREG67_VP8)

15.3.2.3.18.1 Offset

Register	Offset
SWREG67_VP8	10Ch

15.3.2.3.18.2 Diagram



15.3.2.3.18.3 Fields

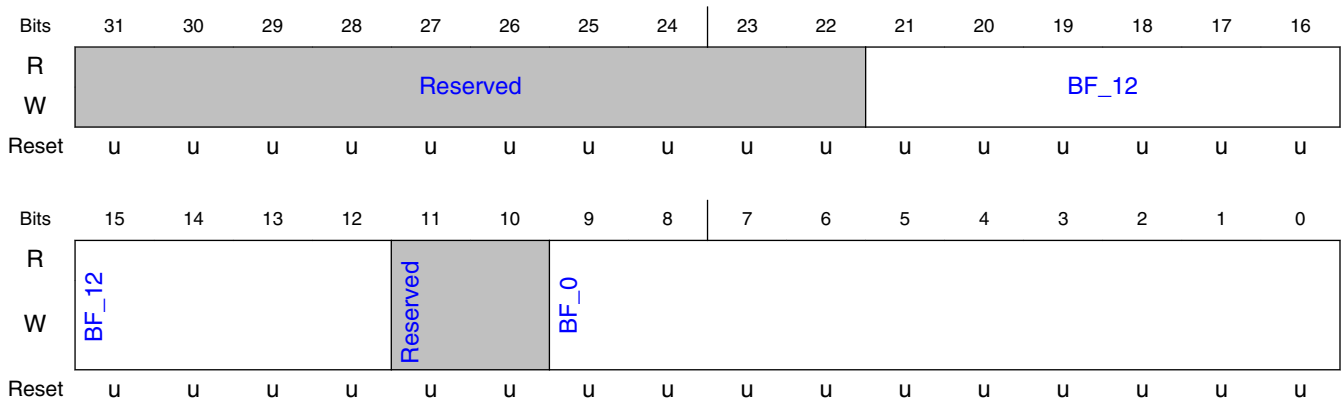
Field	Function
31-22 —	Reserved.
21-12 BF_12	VP8 intra 4x4 mode 3 penalty
11-10 —	Reserved.
9-0 BF_0	VP8 intra 4x4 mode 2 penalty

15.3.2.3.19 VPU H1 Register 68 - for VP8 (SWREG68_VP8)

15.3.2.3.19.1 Offset

Register	Offset
SWREG68_VP8	110h

15.3.2.3.19.2 Diagram



15.3.2.3.19.3 Fields

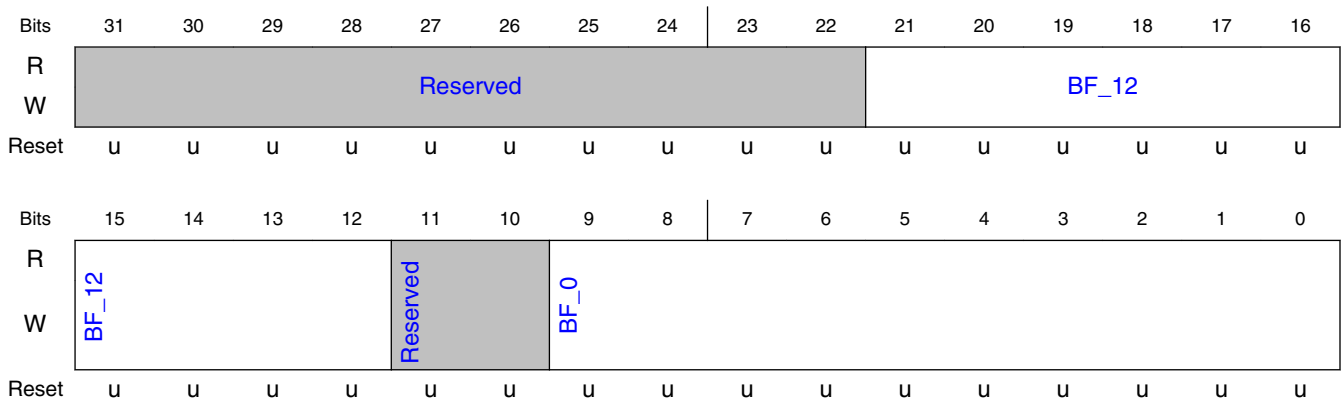
Field	Function
31-22 —	Reserved.
21-12 BF_12	VP8 intra 4x4 mode 5 penalty
11-10 —	Reserved.
9-0 BF_0	VP8 intra 4x4 mode 4 penalty

15.3.2.3.20 VPU H1 Register 69 - for VP8 (SWREG69_VP8)

15.3.2.3.20.1 Offset

Register	Offset
SWREG69_VP8	114h

15.3.2.3.20.2 Diagram



15.3.2.3.20.3 Fields

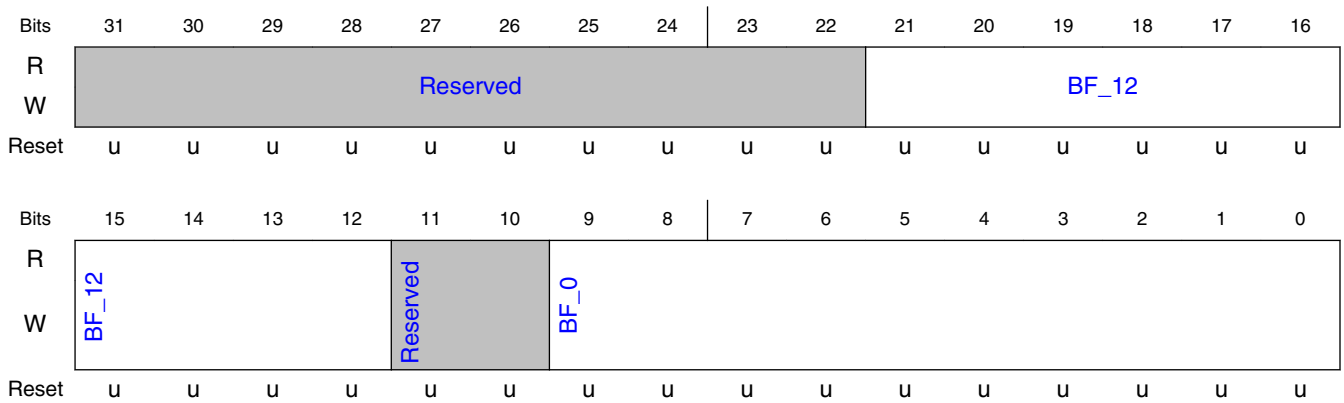
Field	Function
31-22 —	Reserved.
21-12 BF_12	VP8 intra 4x4 mode 7 penalty
11-10 —	Reserved.
9-0 BF_0	VP8 intra 4x4 mode 6 penalty

15.3.2.3.21 VPU H1 Register 70 - for VP8 (SWREG70_VP8)

15.3.2.3.21.1 Offset

Register	Offset
SWREG70_VP8	118h

15.3.2.3.21.2 Diagram



15.3.2.3.21.3 Fields

Field	Function
31-22 —	Reserved.
21-12 BF_12	VP8 intra 4x4 mode 9 penalty
11-10 —	Reserved.
9-0 BF_0	VP8 intra 4x4 mode 8 penalty

15.3.2.3.22 VPU H1 Register 71 - for VP8 (SWREG71_VP8)

15.3.2.3.22.1 Offset

Register	Offset
SWREG71_VP8	11Ch

15.3.2.3.22.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_0															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.3.22.3 Fields

Field	Function
31-0 BF_0	Base address for VP8 segmentId 2-bits/macroblock

15.3.2.3.23 VPU H1 Register 72 - for VP8 (SWREG72_VP8)

15.3.2.3.23.1 Offset

Register	Offset
SWREG72_VP8	120h

15.3.2.3.23.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	BF_23								BF_14						
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF_14		BF_0													
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

15.3.2.3.23.3 Fields

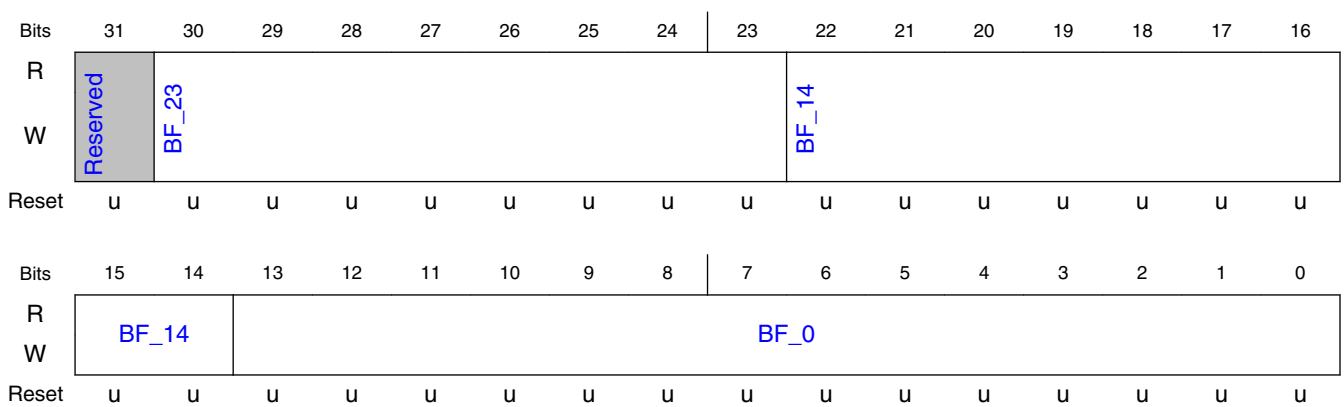
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment1 qpY1RoundDc 8b
22-14 BF_14	VP8 segment1 qpY1ZbinDc 9b
13-0 BF_0	VP8 segment1 qpY1QuantDc 14b

15.3.2.3.24 VPU H1 Register 73 - for VP8 (SWREG73_VP8)

15.3.2.3.24.1 Offset

Register	Offset
SWREG73_VP8	124h

15.3.2.3.24.2 Diagram



15.3.2.3.24.3 Fields

Field	Function
31 —	Reserved.

Table continues on the next page...

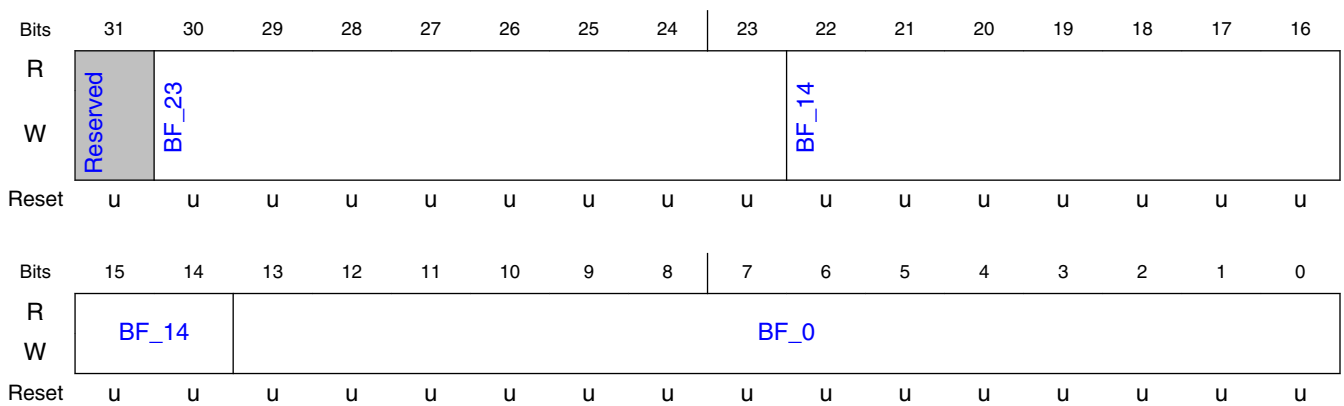
Field	Function
30-23 BF_23	VP8 segment1 qpY1RoundAc 8b
22-14 BF_14	VP8 segment1 qpY1ZbinAc 9b
13-0 BF_0	VP8 segment1 qpY1QuantAc 14b

15.3.2.3.25 VPU H1 Register 74 - for VP8 (SWREG74_VP8)

15.3.2.3.25.1 Offset

Register	Offset
SWREG74_VP8	128h

15.3.2.3.25.2 Diagram



15.3.2.3.25.3 Fields

Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment1 qpY2RoundDc 8b
22-14 BF_14	VP8 segment1 qpY2ZbinDc 9b

Table continues on the next page...

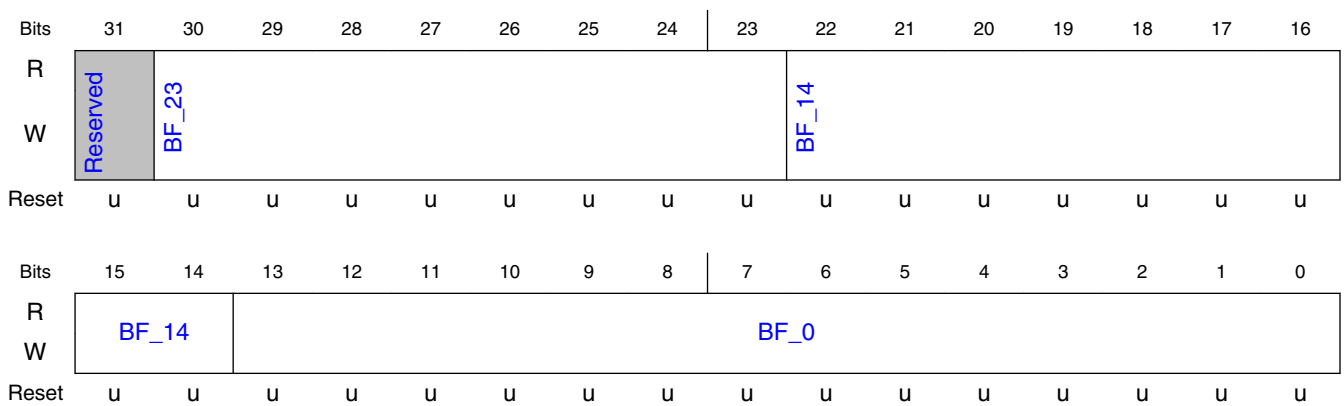
Field	Function
13-0 BF_0	VP8 segment1 qpY2QuantDc 14b

15.3.2.3.26 VPU H1 Register 75 - for VP8 (SWREG75_VP8)

15.3.2.3.26.1 Offset

Register	Offset
SWREG75_VP8	12Ch

15.3.2.3.26.2 Diagram



15.3.2.3.26.3 Fields

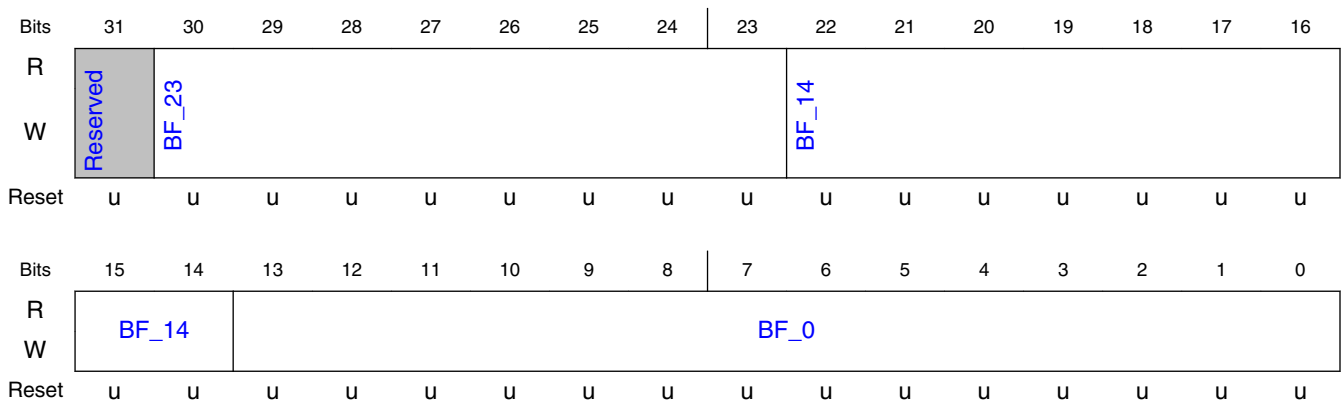
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment1 qpY2RoundAc 8b
22-14 BF_14	VP8 segment1 qpY2ZbinAc 9b
13-0 BF_0	VP8 segment1 qpY2QuantAc 14b

15.3.2.3.27 VPU H1 Register 76 - for VP8 (SWREG76_VP8)

15.3.2.3.27.1 Offset

Register	Offset
SWREG76_VP8	130h

15.3.2.3.27.2 Diagram



15.3.2.3.27.3 Fields

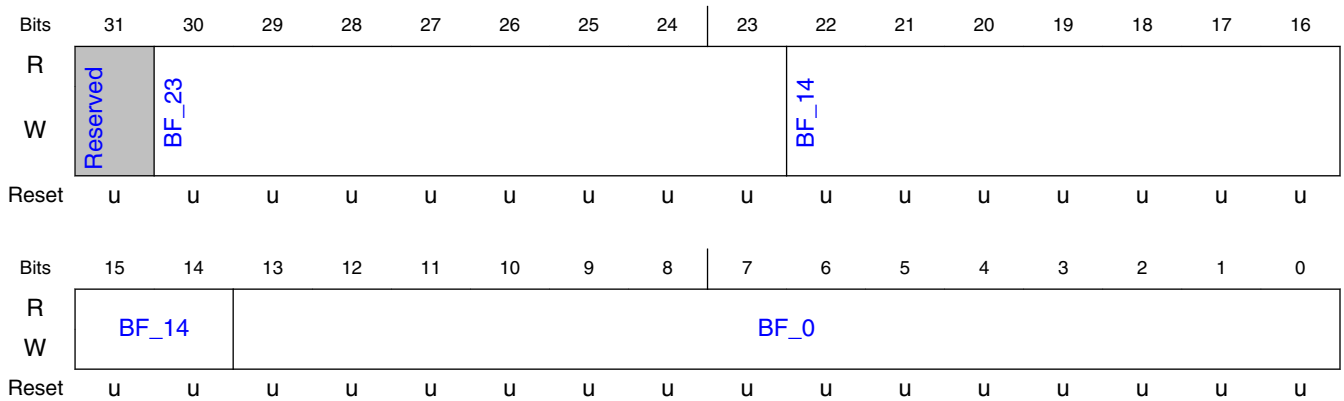
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment1 qpChRoundDc 8b
22-14 BF_14	VP8 segment1 qpChZbinDc 9b
13-0 BF_0	VP8 segment1 qpChQuantDc 14b

15.3.2.3.28 VPU H1 Register 77 - for VP8 (SWREG77_VP8)

15.3.2.3.28.1 Offset

Register	Offset
SWREG77_VP8	134h

15.3.2.3.28.2 Diagram



15.3.2.3.28.3 Fields

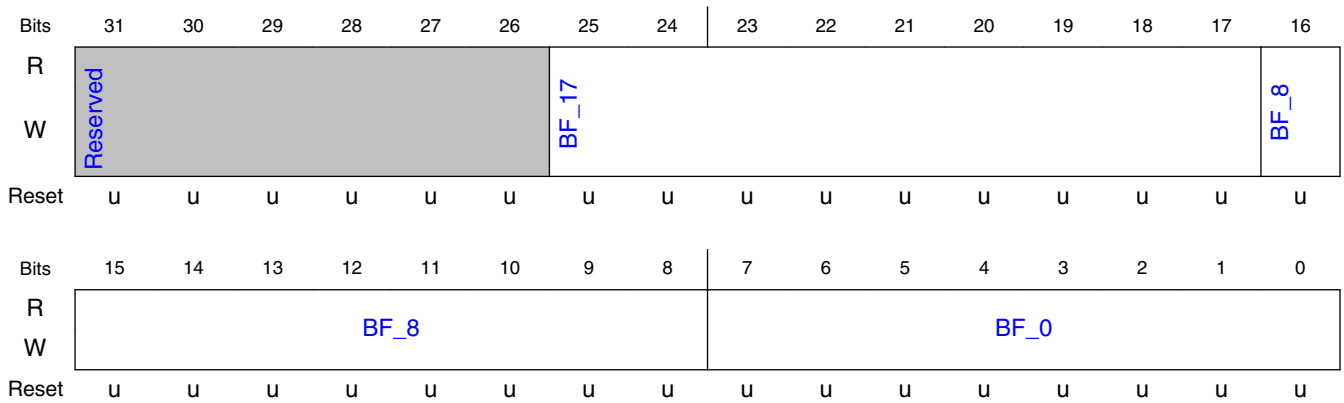
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment1 qpChRoundAc 8b
22-14 BF_14	VP8 segment1 qpChZbinAc 9b
13-0 BF_0	VP8 segment1 qpChQuantAc 14b

15.3.2.3.29 VPU H1 Register 78 - for VP8 (SWREG78_VP8)

15.3.2.3.29.1 Offset

Register	Offset
SWREG78_VP8	138h

15.3.2.3.29.2 Diagram



15.3.2.3.29.3 Fields

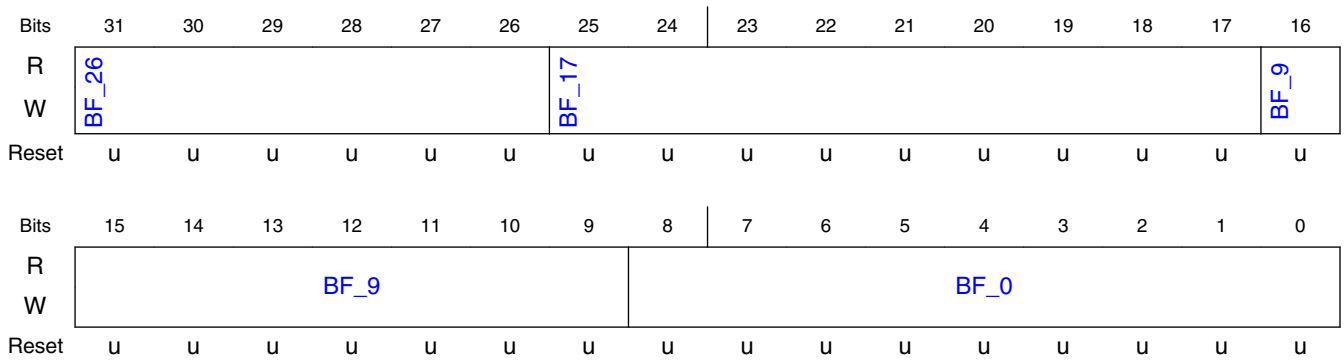
Field	Function
31-26 —	Reserved.
25-17 BF_17	VP8 segment1 qpY2DequantDc 9b
16-8 BF_8	VP8 segment1 qpY1DequantAc 9b
7-0 BF_0	VP8 segment1 qpY1DequantDc 8b

15.3.2.3.30 VPU H1 Register 79 - for VP8 (SWREG79_VP8)

15.3.2.3.30.1 Offset

Register	Offset
SWREG79_VP8	13Ch

15.3.2.3.30.2 Diagram



15.3.2.3.30.3 Fields

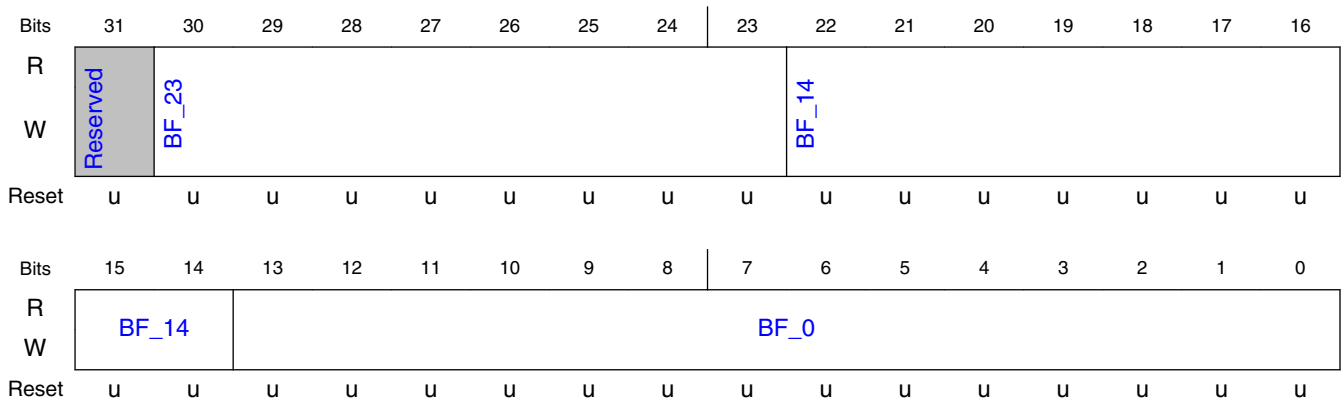
Field	Function
31-26 BF_26	VP8 segment1 filter level 6b
25-17 BF_17	VP8 segment1 qpChDequantAc 9b
16-9 BF_9	VP8 segment1 qpChDequantDc 8b
8-0 BF_0	VP8 segment1 qpY2DequantAc 9b

15.3.2.3.31 VPU H1 Register 80 - for VP8 (SWREG80_VP8)

15.3.2.3.31.1 Offset

Register	Offset
SWREG80_VP8	140h

15.3.2.3.31.2 Diagram



15.3.2.3.31.3 Fields

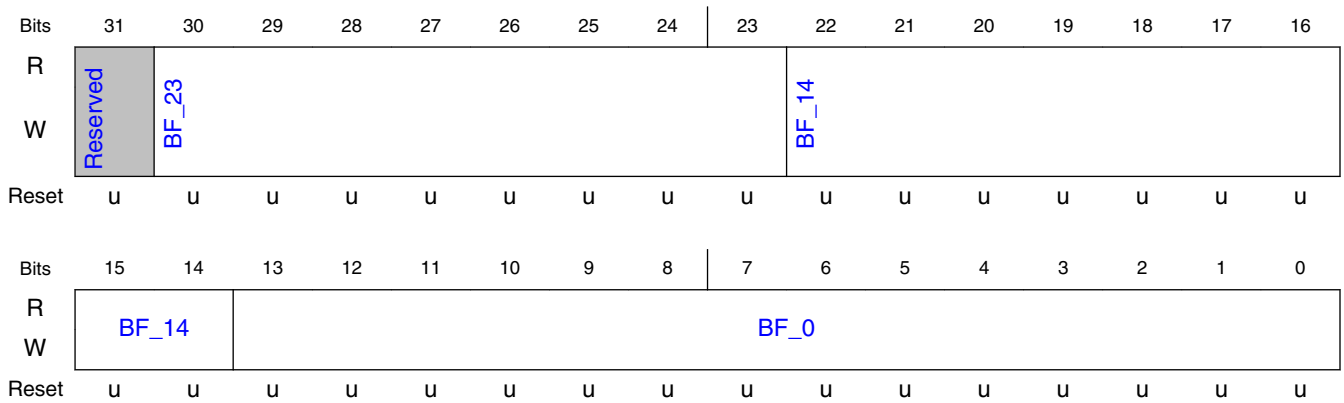
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment2 qpY1RoundDc 8b
22-14 BF_14	VP8 segment2 qpY1ZbinDc 9b
13-0 BF_0	VP8 segment2 qpY1QuantDc 14b

15.3.2.3.32 VPU H1 Register 81 - for VP8 (SWREG81_VP8)

15.3.2.3.32.1 Offset

Register	Offset
SWREG81_VP8	144h

15.3.2.3.32.2 Diagram



15.3.2.3.32.3 Fields

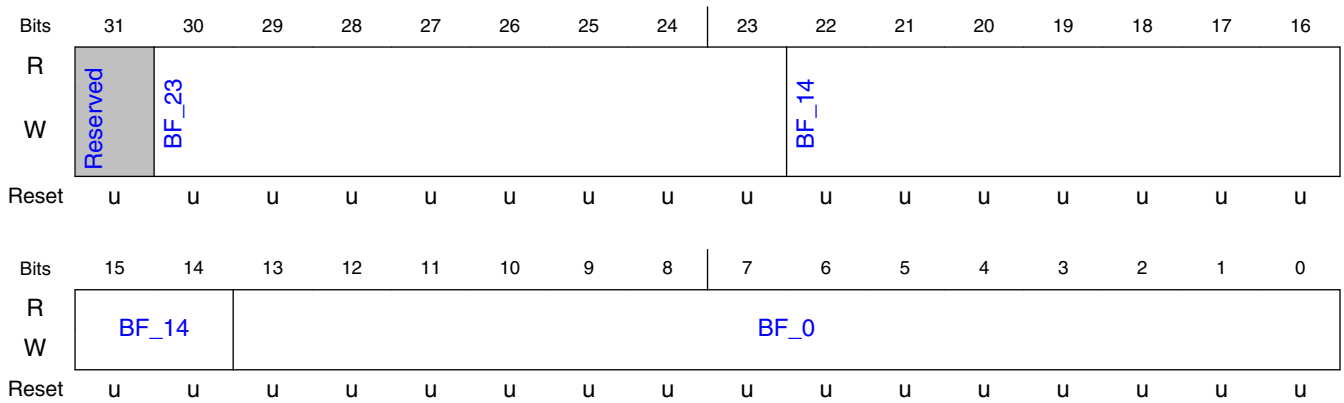
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment2 qpY1RoundAc 8b
22-14 BF_14	VP8 segment2 qpY1ZbinAc 9b
13-0 BF_0	VP8 segment2 qpY1QuantAc 14b

15.3.2.3.33 VPU H1 Register 82 - for VP8 (SWREG82_VP8)

15.3.2.3.33.1 Offset

Register	Offset
SWREG82_VP8	148h

15.3.2.3.33.2 Diagram



15.3.2.3.33.3 Fields

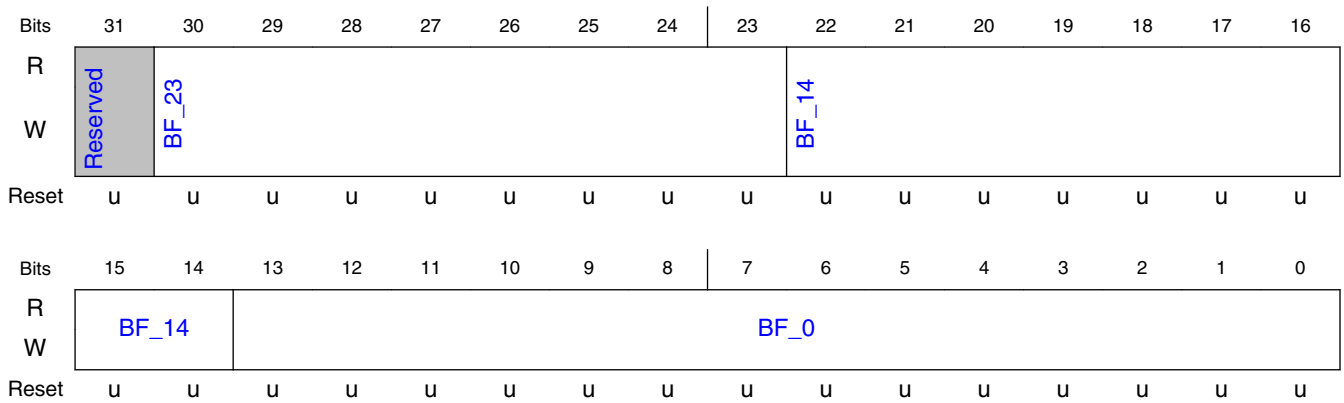
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment2 qpY2RoundDc 8b
22-14 BF_14	VP8 segment2 qpY2ZbinDc 9b
13-0 BF_0	VP8 segment2 qpY2QuantDc 14b

15.3.2.3.34 VPU H1 Register 83 - for VP8 (SWREG83_VP8)

15.3.2.3.34.1 Offset

Register	Offset
SWREG83_VP8	14Ch

15.3.2.3.34.2 Diagram



15.3.2.3.34.3 Fields

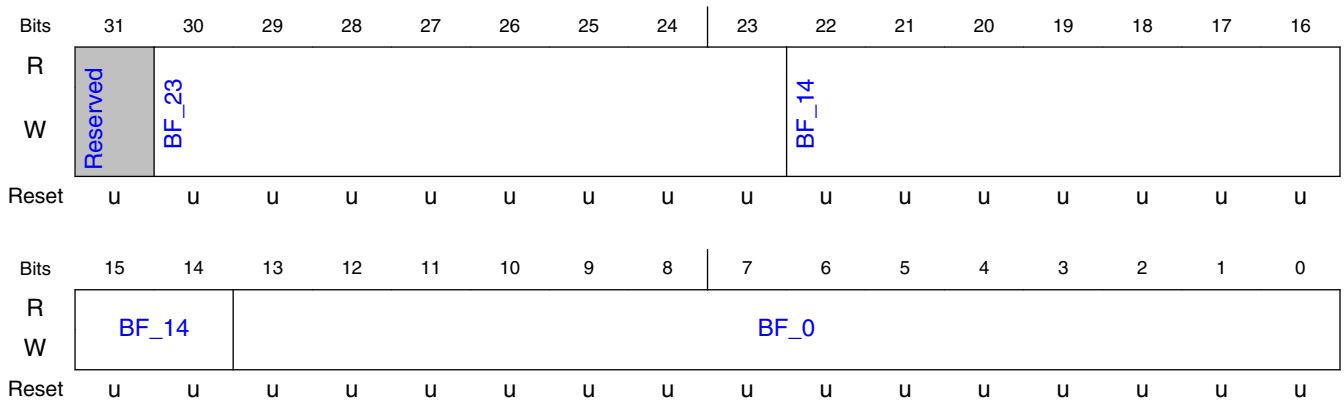
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment2 qpY2RoundAc 8b
22-14 BF_14	VP8 segment2 qpY2ZbinAc 9b
13-0 BF_0	VP8 segment2 qpY2QuantAc 14b

15.3.2.3.35 VPU H1 Register 84 - for VP8 (SWREG84_VP8)

15.3.2.3.35.1 Offset

Register	Offset
SWREG84_VP8	150h

15.3.2.3.35.2 Diagram



15.3.2.3.35.3 Fields

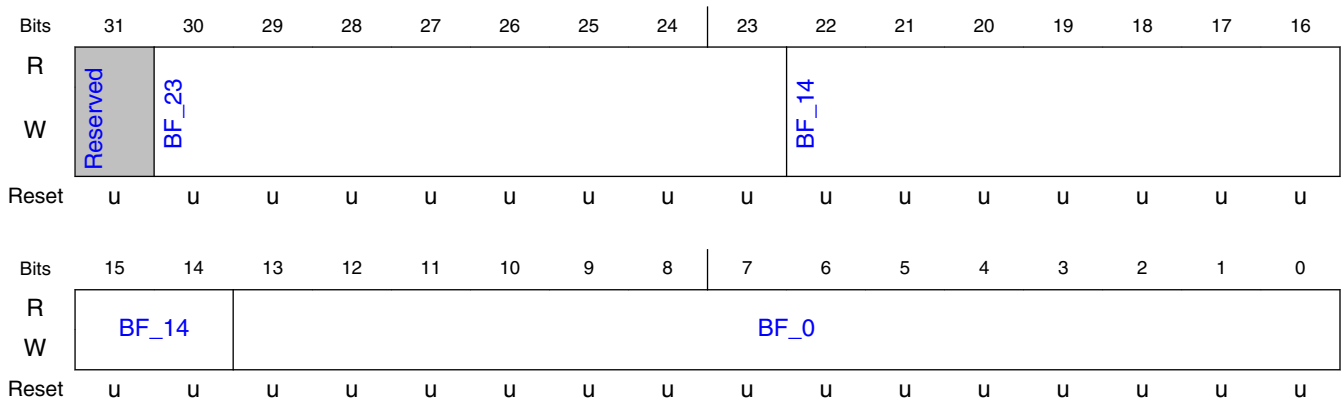
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment2 qpChRoundDc 8b
22-14 BF_14	VP8 segment2 qpChZbinDc 9b
13-0 BF_0	VP8 segment2 qpChQuantDc 14b

15.3.2.3.36 VPU H1 Register 85 - for VP8 (SWREG85_VP8)

15.3.2.3.36.1 Offset

Register	Offset
SWREG85_VP8	154h

15.3.2.3.36.2 Diagram



15.3.2.3.36.3 Fields

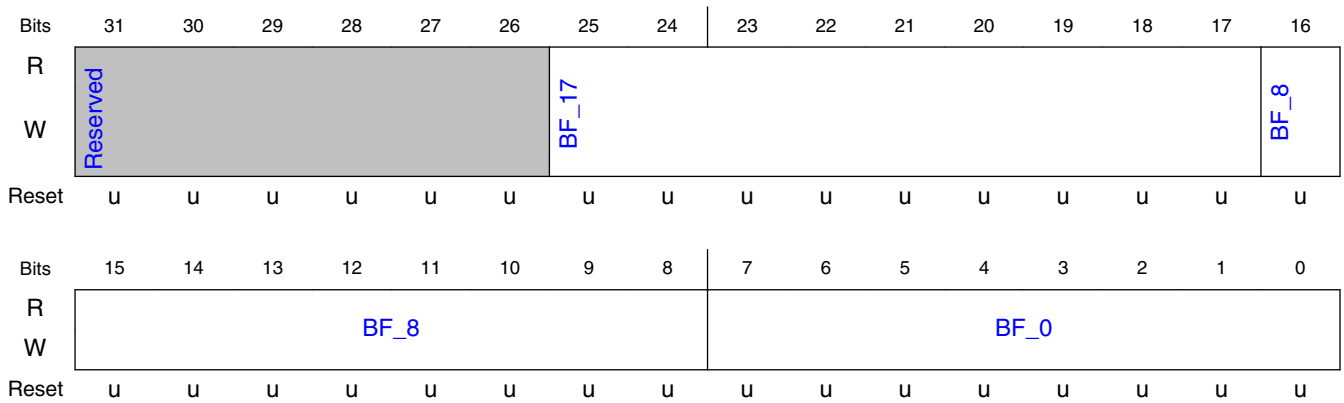
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment2 qpChRoundAc 8b
22-14 BF_14	VP8 segment2 qpChZbinAc 9b
13-0 BF_0	VP8 segment2 qpChQuantAc 14b

15.3.2.3.37 VPU H1 Register 86 - for VP8 (SWREG86_VP8)

15.3.2.3.37.1 Offset

Register	Offset
SWREG86_VP8	158h

15.3.2.3.37.2 Diagram



15.3.2.3.37.3 Fields

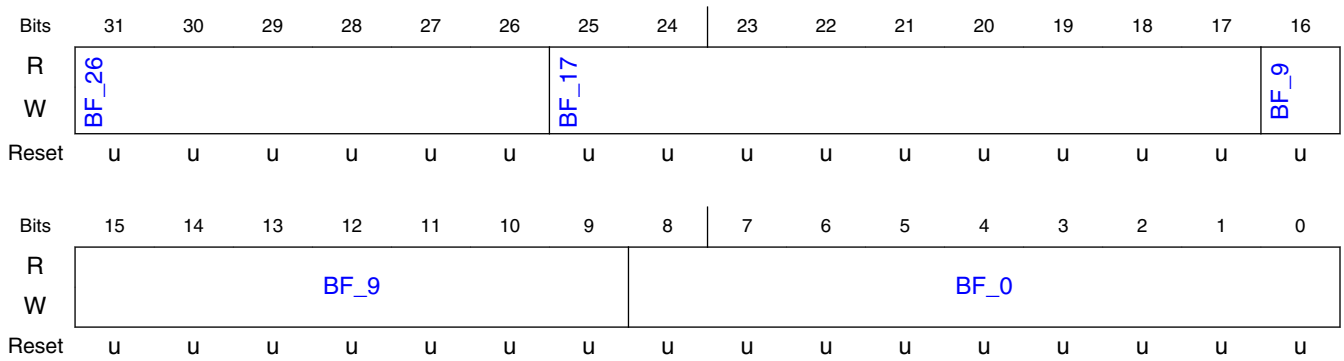
Field	Function
31-26 —	Reserved.
25-17 BF_17	VP8 segment2 qpY2DequantDc 9b
16-8 BF_8	VP8 segment2 qpY1DequantAc 9b
7-0 BF_0	VP8 segment2 qpY1DequantDc 8b

15.3.2.3.38 VPU H1 Register 87 - for VP8 (SWREG87_VP8)

15.3.2.3.38.1 Offset

Register	Offset
SWREG87_VP8	15Ch

15.3.2.3.38.2 Diagram



15.3.2.3.38.3 Fields

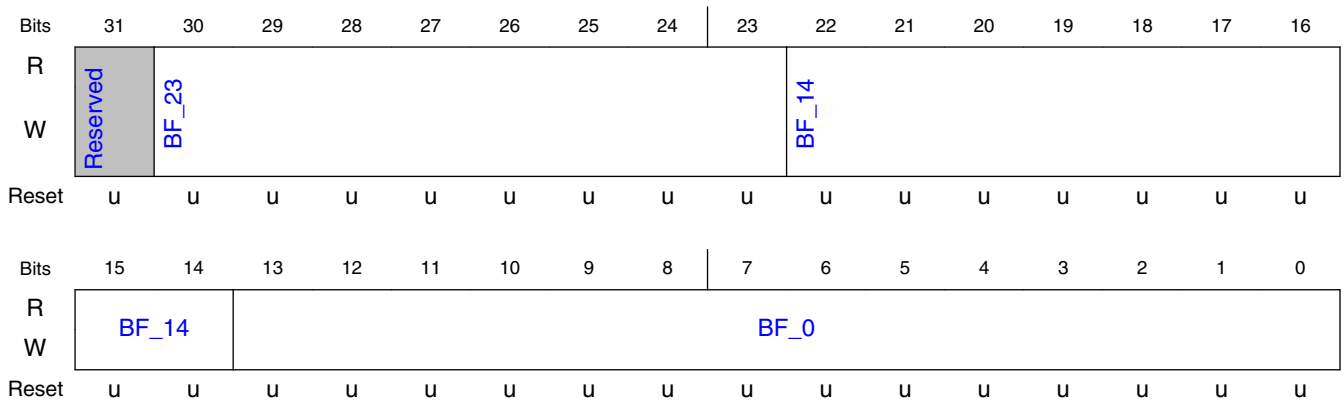
Field	Function
31-26 BF_26	VP8 segment2 filter level 6b
25-17 BF_17	VP8 segment2 qpChDequantAc 9b
16-9 BF_9	VP8 segment2 qpChDequantDc 8b
8-0 BF_0	VP8 segment2 qpY2DequantAc 9b

15.3.2.3.39 VPU H1 Register 88 - for VP8 (SWREG88_VP8)

15.3.2.3.39.1 Offset

Register	Offset
SWREG88_VP8	160h

15.3.2.3.39.2 Diagram



15.3.2.3.39.3 Fields

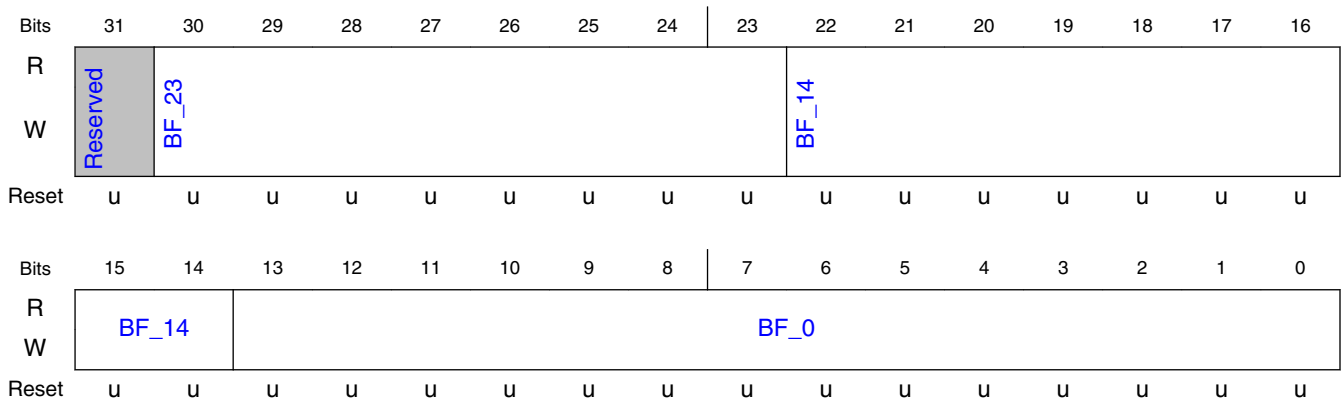
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment3 qpY1RoundDc 8b
22-14 BF_14	VP8 segment3 qpY1ZbinDc 9b
13-0 BF_0	VP8 segment3 qpY1QuantDc 14b

15.3.2.3.40 VPU H1 Register 89 - for VP8 (SWREG89_VP8)

15.3.2.3.40.1 Offset

Register	Offset
SWREG89_VP8	164h

15.3.2.3.40.2 Diagram



15.3.2.3.40.3 Fields

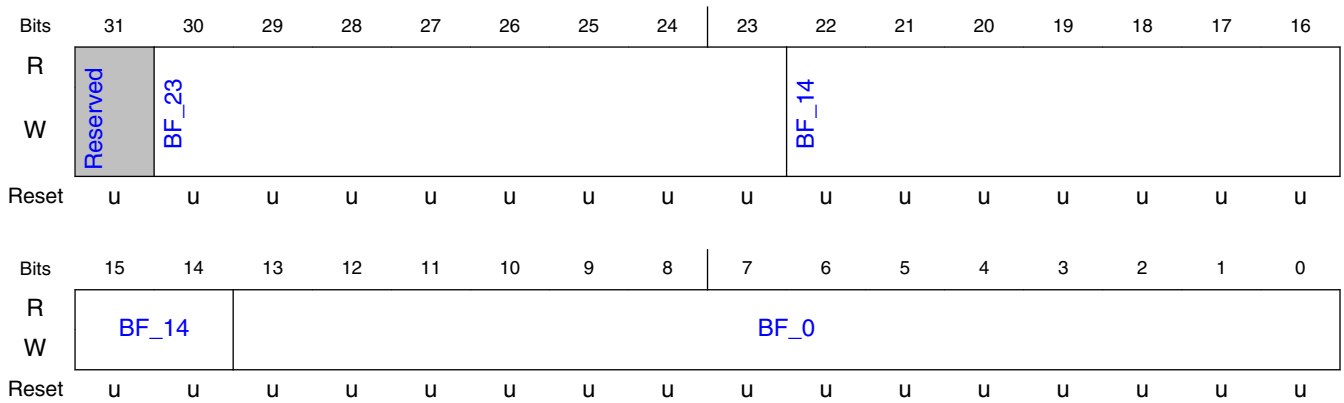
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment3 qpY1RoundAc 8b
22-14 BF_14	VP8 segment3 qpY1ZbinAc 9b
13-0 BF_0	VP8 segment3 qpY1QuantAc 14b

15.3.2.3.41 VPU H1 Register 90 - for VP8 (SWREG90_VP8)

15.3.2.3.41.1 Offset

Register	Offset
SWREG90_VP8	168h

15.3.2.3.41.2 Diagram



15.3.2.3.41.3 Fields

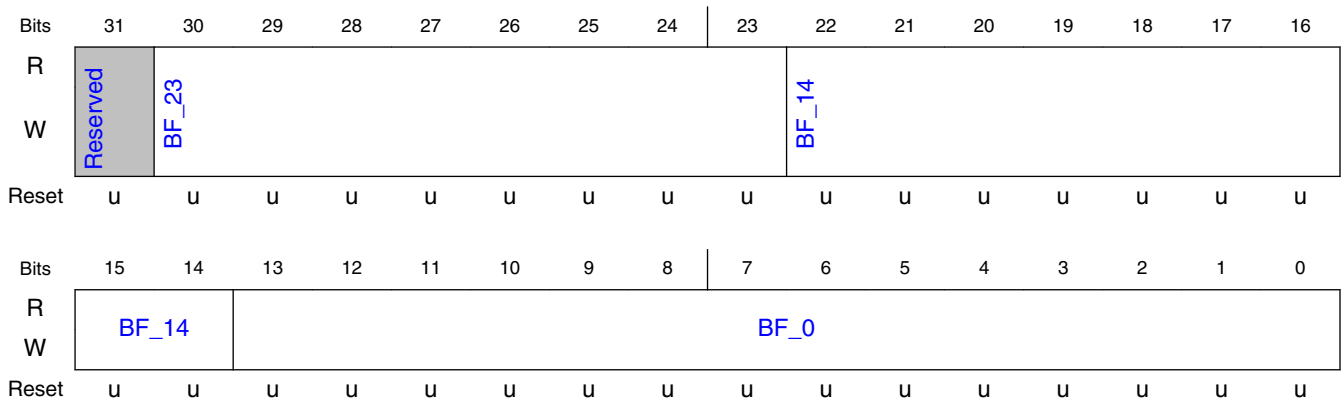
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment3 qpY2RoundDc 8b
22-14 BF_14	VP8 segment3 qpY2ZbinDc 9b
13-0 BF_0	VP8 segment3 qpY2QuantDc 14b

15.3.2.3.42 VPU H1 Register 91 - for VP8 (SWREG91_VP8)

15.3.2.3.42.1 Offset

Register	Offset
SWREG91_VP8	16Ch

15.3.2.3.42.2 Diagram



15.3.2.3.42.3 Fields

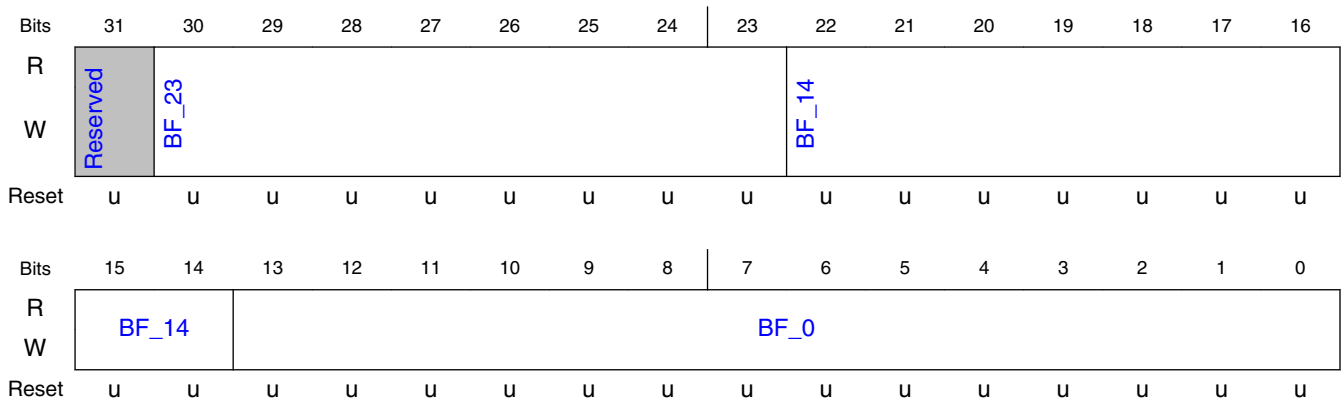
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment3 qpY2RoundAc 8b
22-14 BF_14	VP8 segment3 qpY2ZbinAc 9b
13-0 BF_0	VP8 segment3 qpY2QuantAc 14b

15.3.2.3.43 VPU H1 Register 92 - for VP8 (SWREG92_VP8)

15.3.2.3.43.1 Offset

Register	Offset
SWREG92_VP8	170h

15.3.2.3.43.2 Diagram



15.3.2.3.43.3 Fields

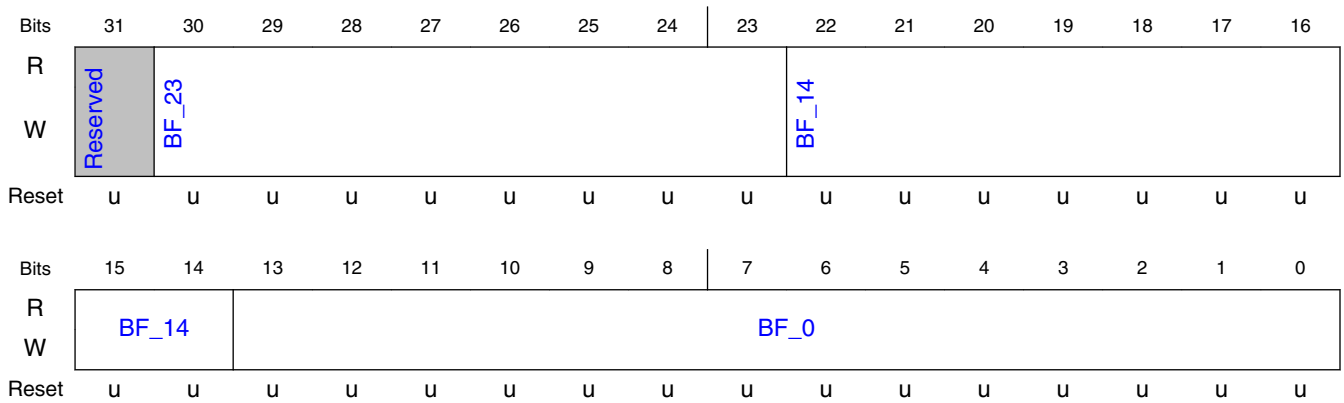
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment3 qpChRoundDc 8b
22-14 BF_14	VP8 segment3 qpChZbinDc 9b
13-0 BF_0	VP8 segment3 qpChQuantDc 14b

15.3.2.3.44 VPU H1 Register 93 - for VP8 (SWREG93_VP8)

15.3.2.3.44.1 Offset

Register	Offset
SWREG93_VP8	174h

15.3.2.3.44.2 Diagram



15.3.2.3.44.3 Fields

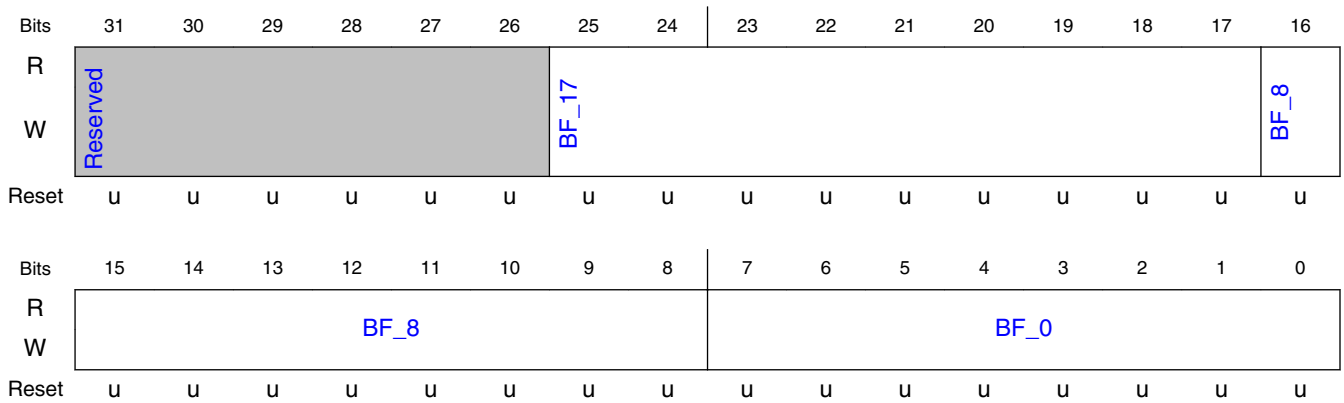
Field	Function
31 —	Reserved.
30-23 BF_23	VP8 segment3 qpChRoundAc 8b
22-14 BF_14	VP8 segment3 qpChZbinAc 9b
13-0 BF_0	VP8 segment3 qpChQuantAc 14b

15.3.2.3.45 VPU H1 Register 94 - for VP8 (SWREG94_VP8)

15.3.2.3.45.1 Offset

Register	Offset
SWREG94_VP8	178h

15.3.2.3.45.2 Diagram



15.3.2.3.45.3 Fields

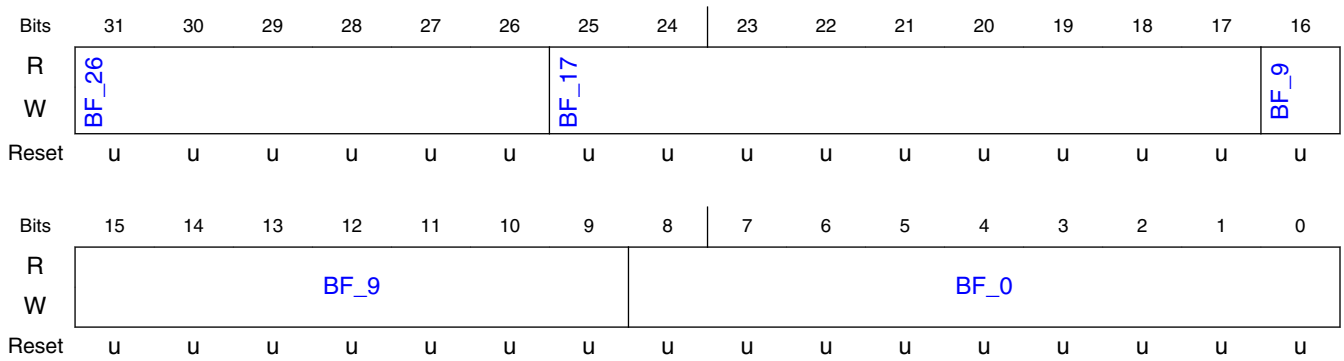
Field	Function
31-26 —	Reserved.
25-17 BF_17	VP8 segment3 qpY2DequantDc 9b
16-8 BF_8	VP8 segment3 qpY1DequantAc 9b
7-0 BF_0	VP8 segment3 qpY1DequantDc 8b

15.3.2.3.46 VPU H1 Register 95 - for VP8 (SWREG95_VP8)

15.3.2.3.46.1 Offset

Register	Offset
SWREG95_VP8	17Ch

15.3.2.3.46.2 Diagram



15.3.2.3.46.3 Fields

Field	Function
31-26 BF_26	VP8 segment3 filter level 6b
25-17 BF_17	VP8 segment3 qpChDequantAc 9b
16-9 BF_9	VP8 segment3 qpChDequantDc 8b
8-0 BF_0	VP8 segment3 qpY2DequantAc 9b

Chapter 16
Low Speed Communication and Interconnects

16.1 I2C Controller (I2C)

16.1.1 Overview

This chapter describes block-level operation and programming of I2C. The chapter is intended for a block-driver software developer. To understand how the block is integrated at the SoC level, a system software developer should see discussions of the block in the appropriate SoC-level chapter(s).

References: This document assumes an understanding of the following document:

- *The I2C Bus Specification*, Version 2.1, by Philips Semiconductor

The Inter IC (I2C) provides functionality of a standard I2C slave and master. The I2C is designed to be compatible with the standard NXP I2C bus protocol.

NOTE

Four independent I2C channels are available.

I2C is a two-wire, bidirectional serial bus that provides a simple, efficient method of data exchange, minimizing the interconnection between devices. This bus is suitable for applications requiring occasional communications over a short distance between many devices. The flexible I2C standard allows additional devices to be connected to the bus for expansion and system development. See the connection diagram in the figure below.

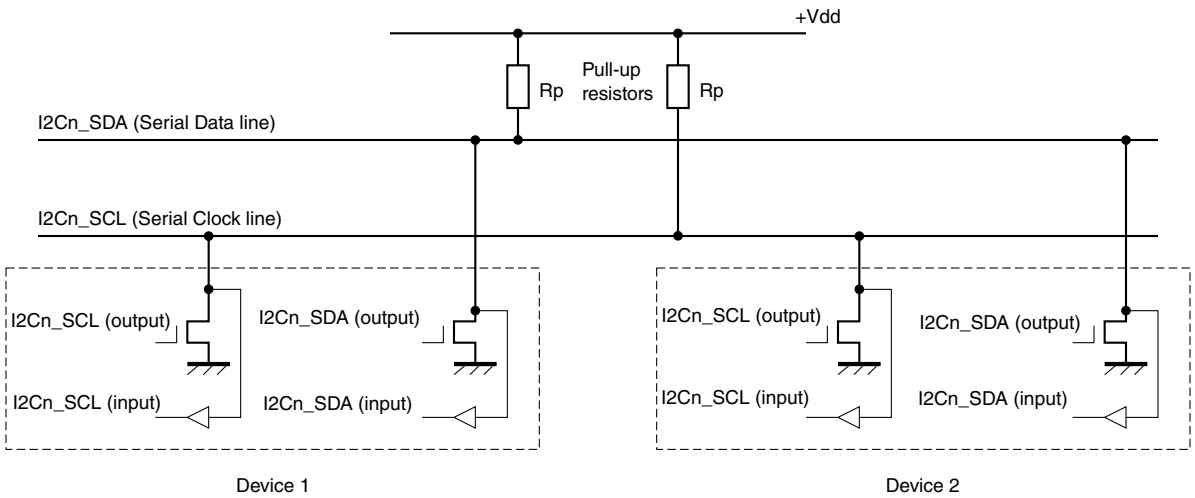


Figure 16-1. Connection of devices to I2C bus

The I2C interface speed is dependent on the I2C bus loading and timing characteristics. For pin requirement details, see *The I2C Bus Specification*. The I2C system is a true multimaster bus including arbitration and collision detection that prevents data corruption if multiple devices attempt to control the bus simultaneously. This feature supports complex applications with multiprocessor control and can be used for rapid testing and alignment of end products through external connections to an assembly-line computer. The figure below shows the block diagram of I2C.

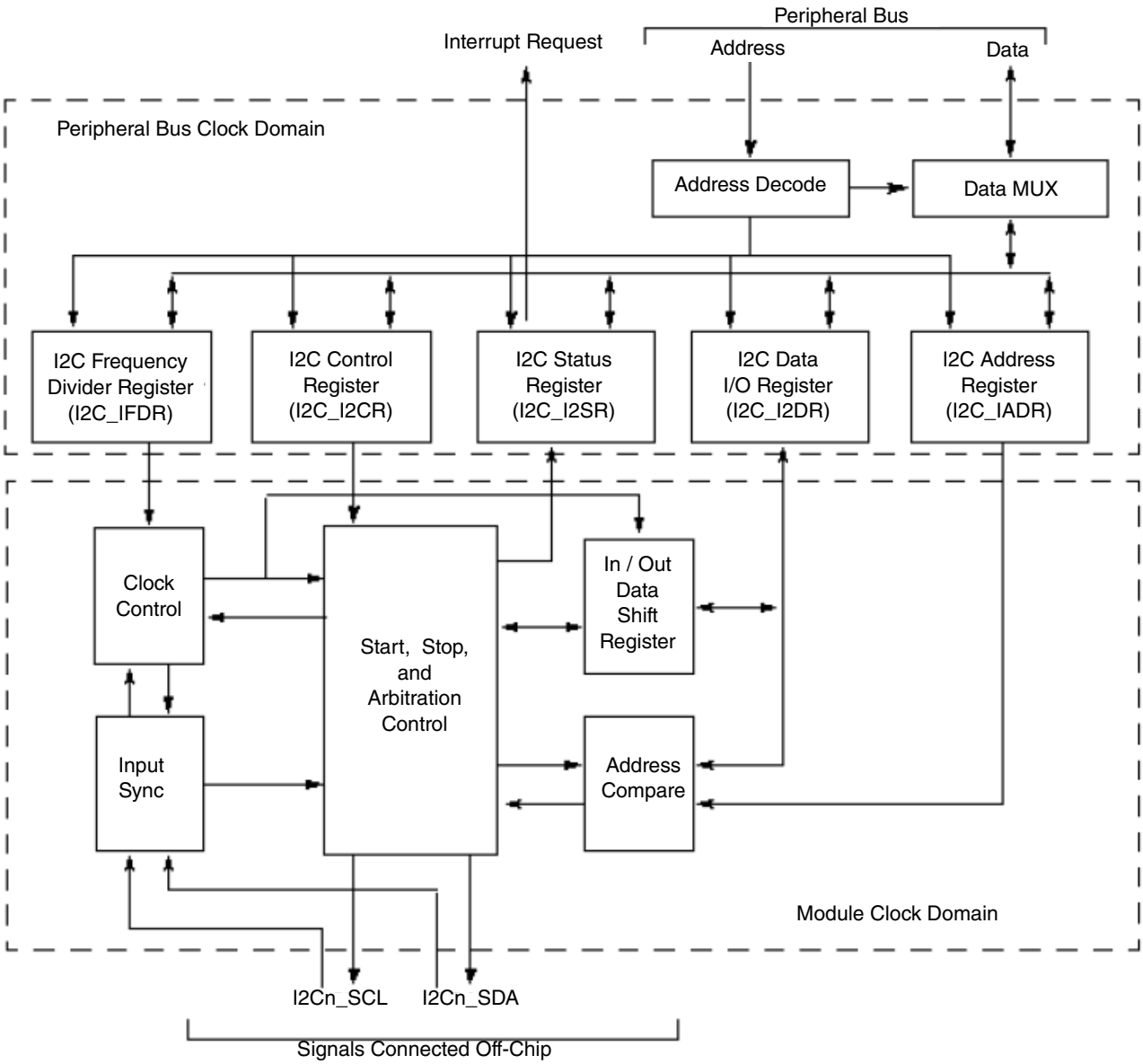


Figure 16-2. I2C block diagram

16.1.1.1 Features

The I2C has the following key features:

- Compatibility with I2C bus standard
- Multimaster operation
- Software programmability for one of 64 different serial clock frequencies
- Software-selectable acknowledge bit
- Interrupt-driven, byte-by-byte data transfer
- Arbitration-lost interrupt with automatic mode switching from master to slave

- Calling address identification interrupt
- Start and stop signal generation/detection
- Repeated Start signal generation
- Acknowledge bit generation/detection
- Bus-busy detection

16.1.1.2 Modes and operations

The I2C operates primarily in two functional modes: Standard mode and Fast mode.

- In Standard mode, I2C supports the data transfer rates up to 100 kbits/s.
- In Fast mode, data transfer rates up to 400 kbits/s can be achieved. Per block operation, there is no special configuration required for Fast or Standard mode. It is the data transfer rate that distinguishes Standard and Fast mode.

16.1.2 Functional description

This section provides a complete functional description of the block.

16.1.2.1 I2C system configuration

After a reset, the I2C defaults to Slave Receive operations. Thus, when not operating as a master or responding to a slave transmit address, the I2C defaults to the Slave Receive state.

For exceptions, see Initialization sequence.

NOTE

The I2C is designed to be compatible with the Philips™ I2C bus protocol. For information on system configuration, protocol, and restrictions, see the *I2C Bus Specification*, version 2.1, by Philips Semiconductors. The I2C supports Standard and Fast modes only.

16.1.2.2 Arbitration procedure

If multiple devices simultaneously request the bus, the bus clock is determined by a synchronization procedure in which the low period equals the longest clock-low period among the devices, and the high period equals the shortest. A data arbitration procedure determines the relative priority of competing devices.

A device loses arbitration if it sends logic high while another sends logic low; it immediately switches to Slave Receive mode and stops driving I2Cn_SDA. In this case, the transition from master to Slave mode does not generate a Stop condition. Meanwhile, hardware sets the arbitration lost bit in the I2C Status register (I2C_I2SR[IAL] to indicate loss of arbitration).

16.1.2.3 Clock synchronization

Because wire-AND logic is used, a high-to-low transition on SCL affects devices connected to the bus. Devices start counting their low period when the master drives SCL low. When a device clock goes low, it holds SCL low until the Clock High state is reached. However, the low-to-high change in this device clock may not change the state of SCL if another device clock is still in its low period. Therefore, the device with the longest low period holds the synchronized clock SCL low.

Devices with shorter low periods enter a High Wait state during this time (see Figure 16-3). When all devices involved have counted off their low periods, the synchronized clock SCL is released and pulled high. There is then no difference between device clocks and the state of SCL, so all of the devices start counting their high periods. The first device to complete its high period pulls SCL low again.

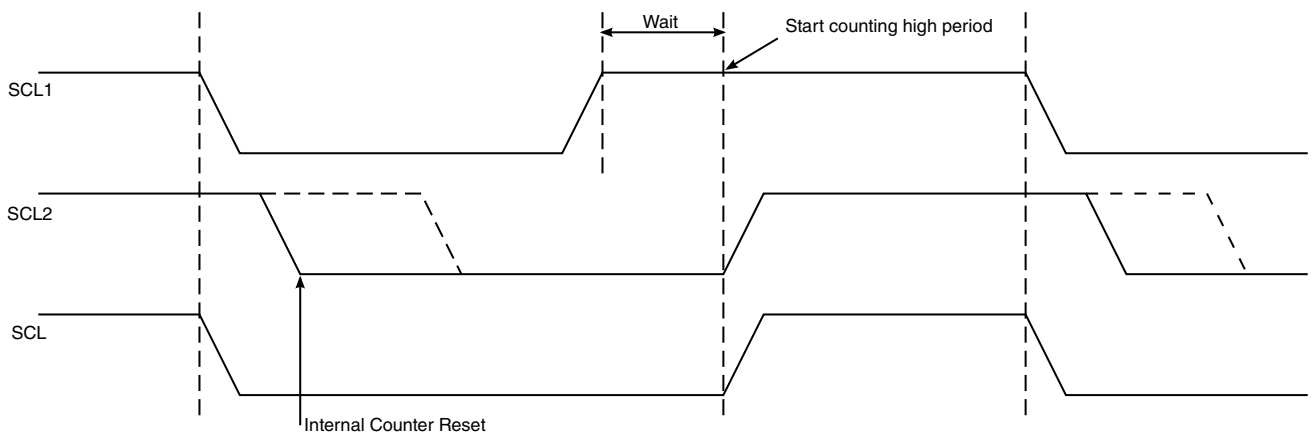


Figure 16-3. Synchronized clock SCL

16.1.2.4 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfers. Slave devices can hold SCL low after completing one byte transfer (9 bits). In such a case, the clock mechanism halts the bus clock and forces the master clock into a Wait state until the slave releases SCL.

16.1.2.5 Clock stretching

Slaves can use the clock synchronization mechanism to slow down the transfer bit rate. After the master has driven SCL low, the slave can drive SCL low for the required period and then release it. If the slave SCL low period is longer than the master SCL low period, the resulting SCL bus signal low period is stretched.

16.1.2.6 Peripheral bus accesses

I2C is a 16-bit block. Only half-word accesses should be performed to the block.

16.1.2.7 Generation of transfer error on IP bus

If an address is received on the peripheral slave bus interface but it is not implemented, an access error is generated.

16.1.2.8 Reset

The I2C can be reset in the following ways:

- Global reset: A hard asynchronous reset of the whole I2C
- Software reset: An internal reset for the whole I2C (except for I2C_IADR and I2C_IFDR registers) initiated by deasserting the I2C_I2CR[IEN] bit

16.1.2.9 Interrupts

There is only one interrupt from the block, which is enabled by setting the I2C_I2CR[IIEN] bit.

The interrupt is generated in any one of the following conditions:

- One byte transfer is completed (the interrupt is set at the falling edge of the ninth clock).
- An address is received that matches its own specific address in Slave Receive mode.
- Arbitration is lost.

16.1.2.10 Byte order

The block only supports the Little-Endian mode.

16.1.3 Initialization

NOTE

Ensure the input select pins for IOMUXC are configured correctly for I2C.

16.1.3.1 Initialization sequence

Before the interface can transfer serial data, registers must be initialized, as listed here.

1. Set the data sampling rate (I2C_IFDR[IC] to obtain SCL frequency from the system bus clock.
2. Update the address in the (I2C_IADR) to define its slave address (address can range from 0 to 0x7f).
3. Set the I2C enable bit (I2C_I2CR[IEN]) to enable the I2C bus interface system.
4. Modify the bits in the I2C_I2CR to select Master/Slave mode, Transmit/Receive mode, and Interrupt-Enable or not.

16.1.3.2 Generation of Start

After completion of the initialization procedure, serial data can be transmitted by selecting the Master Transmit mode. On a multimaster bus system, the busy bus (I2C_I2SR[IBB]) must be tested to determine whether the serial bus is free. If the bus is free (IBB = 0), the Start signal and the first byte (the slave address) can be sent. The data written to the data register comprises the address of the desired slave and the LSB indicates the transfer direction.

The free time between a Stop and the next Start condition is built into the hardware that generates the Start cycle. Depending on the relative frequencies of the system clock and the SCL period, it may be necessary to wait until the I2C is not busy after writing the calling address to the data register (I2C_I2DR), before proceeding to load data into the data register (I2C_I2DR).

16.1.3.3 Post-transfer software response

Sending or receiving a byte sets the data transferring bit (I2C_I2SR[ICF]), which indicates one byte of communication is finished. Upon completion, the interrupt status (I2C_I2SR[IIF]) is also set. An external interrupt is generated if the interrupt enable (I2C_I2CR[IIEN]) is set. The software must first clear the interrupt status (I2C_I2SR[IIF]) in the interrupt routine.

See the flow chart in Figure 16-5.

The data transferring bit (I2C_I2SR[ICF]) is cleared either by reading from I2C_I2DR in Receive mode or by writing to this register in Transmit mode.

The software can service the I2C I/O in the main program by monitoring the interrupt status (I2C_I2SR[IIF]) if the interrupt enable is deasserted. In this case, the interrupt status should be polled in the data transferring bit (I2C_I2SR[ICF]) because the operation is different when arbitration is lost.

When an interrupt occurs at the end of the address cycle, the master is always in Transmit mode; that is, the address is sent. If Master Receive mode is required, then I2C_I2CR[MTX] should be toggled and a dummy read of the I2C_I2DR register must be executed to trigger receive data.

During Slave-mode address cycles (I2C_I2SR[IAAS] = 1), the slave read/write bit I2C_I2SR[SRW] is read to determine the direction of the next transfer. The transmit/receive bit (I2C_I2CR[MTX]) should also be programmed accordingly. For Slave-mode data cycles (IAAS = 0), SRW is invalid. MTX should be read to determine the current transfer direction.

16.1.3.4 Generation of Stop

A data transfer ends when the master signals a Stop, which can occur after all data is sent.

For a master receiver to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last data byte. This is done by setting the transmit acknowledge bit (I2C_I2CR[TXAK]) before reading the next-to-last byte. Before the last byte is read, a Stop signal must be generated.

16.1.3.5 Generation of Repeated Start

After the data transfer, if the master still requires the bus, it can signal another Start followed by another slave address without signaling a Stop.

16.1.3.6 Slave mode

In the slave interrupt service routine (see Figure 16-5), the block addressed as slave bit (IAAS) should be tested to check if a calling of its own address has just been received. If IAAS is set, software should set the Transmit/Receive mode select bit (I2C_I2CR[MTX]) according to the I2C_I2SR[SRW]. Writing to the I2C_I2CR clears the IAAS automatically. The only time IAAS is read as set is from the interrupt at the end of the address cycle where an address match occurred; interrupts resulting from subsequent data transfers will have IAAS cleared. A data transfer can now be initiated by writing information to I2C_I2DR for slave transmits, or read from I2C_I2DR in Slave Receive mode. A dummy read of I2C_I2DR in Slave Receive mode releases SCL, allowing the master to send data.

In the slave transmitter routine, the receive acknowledge bit (I2C_I2SR[RXAK]) must be tested before sending the next byte of data. Setting RXAK means an end-of-data signal from the master receiver, after which the software must switch it from Transmit to Receiver mode. Reading the data register (I2C_I2DR) then releases SCL so the master can generate a Stop signal.

16.1.3.7 Arbitration lost

If several devices try to engage the bus at the same time, one becomes master. Hardware immediately switches devices that lose arbitration to Slave Receive mode. Data output to 12Cn_SDA stops, but 12Cn_SCL is still generated until the end of the byte during which arbitration is lost. An interrupt occurs at the falling edge of the ninth clock of this transfer if the arbitration is lost (I2C_I2SR[IAL] = 1), and the Slave mode is selected (I2C_I2CR[MSTA] = 0).

See the flow chart in Figure 16-5.

If a device that is not a master tries to transmit or do a Start, hardware inhibits the transmission, clears MSTA without signaling a Stop, generates an interrupt to the Arm platform, and sets I2C_I2SR[IAL] to indicate a failed attempt to engage the bus. When considering these cases, the slave service routine should first test I2C_I2SR[IAL], and the software should clear it if it is set.

For Multimaster mode, when an I2C is enabled when the bus is busy and asserts Start, the I2C_I2SR[IAL] bit gets set only for 12Cn_SDA=0, 12Cn_SCL=0/1, 12Cn_SDA=1, and 12Cn_SCL=0; but not for 12Cn_SDA=1 and I2Cn_SCL=1, which is the equivalent of Bus Idle state.

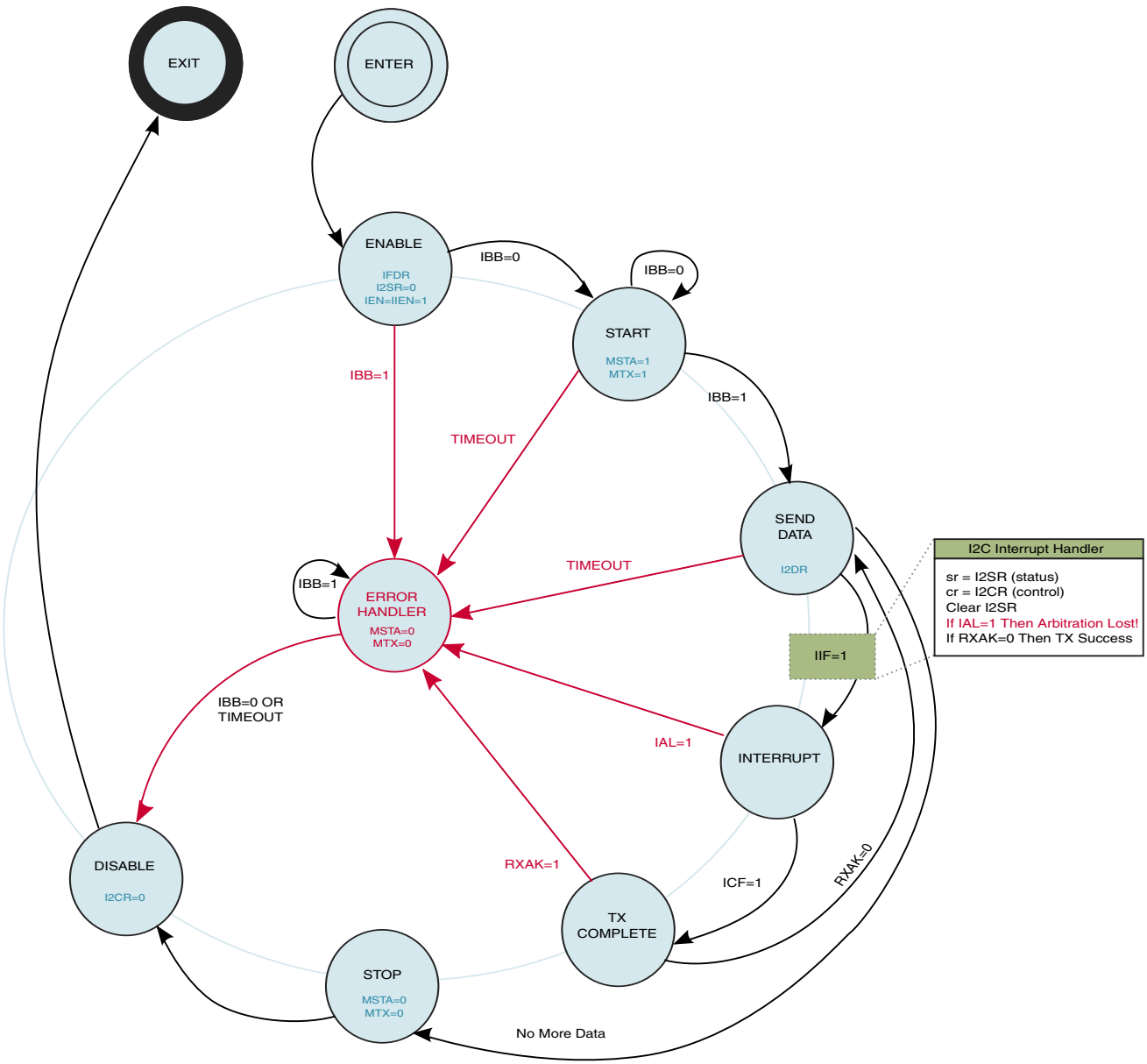


Figure 16-4. I2C Programming state diagram

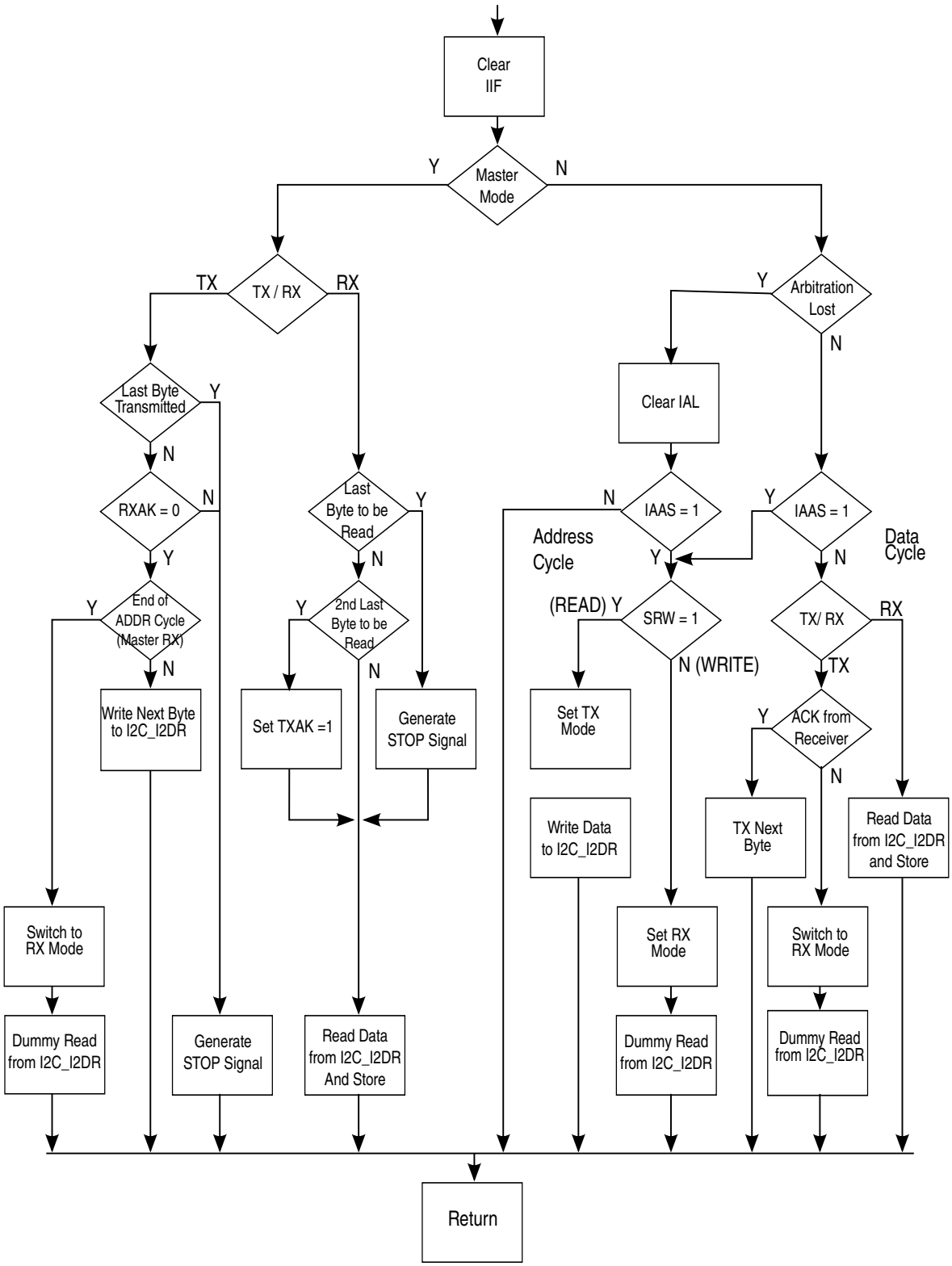


Figure 16-5. Flowchart of typical I2C interrupt routine

NOTE

For a Repeated Start only, the Stop-generation stage does not occur in Master mode. A loop repeats itself without stopping for the next start.

For Master Receive mode, I2C is programmed as Master Transmit during Address mode and after slave address transfer; the MTX bit should be cleared and a dummy read on the I2C_I2DR register should be performed so I2C can read the next receive data.

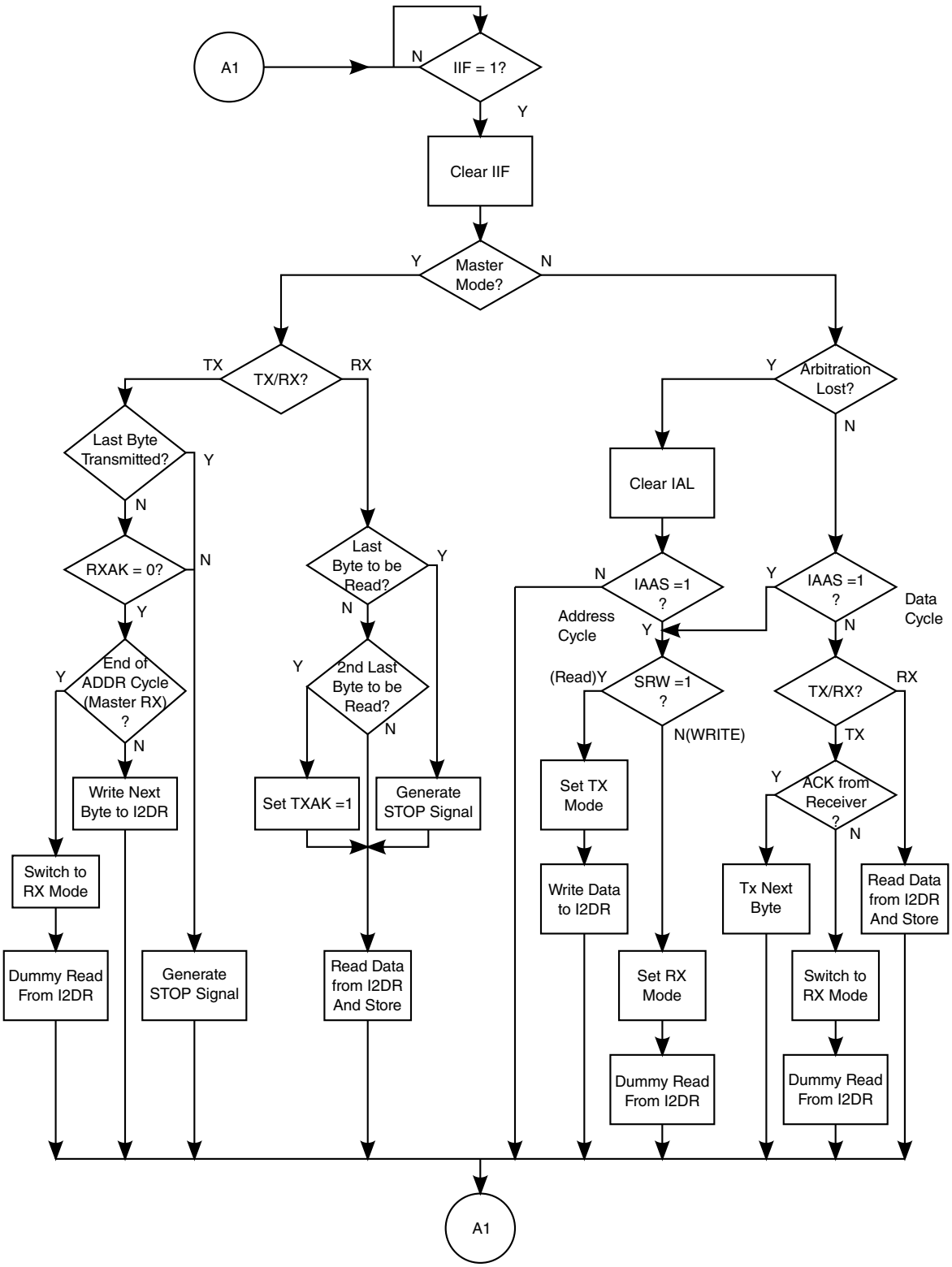


Figure 16-6. Flowchart for typical I2C polling routine

NOTE

The timeout value depends on the bus frequency at which I2C is operating. The minimum timeout for polling the IIF bit at a maximum I2C bus frequency of 400 kHz is T_{min} = 25 µs (=2.5 x 10 µs). This value can be calculated for any bus frequency. The formula is $T_{min} = 10/F_{SCL}$, where F_{SCL} is the frequency of the I2C clock (SCL).

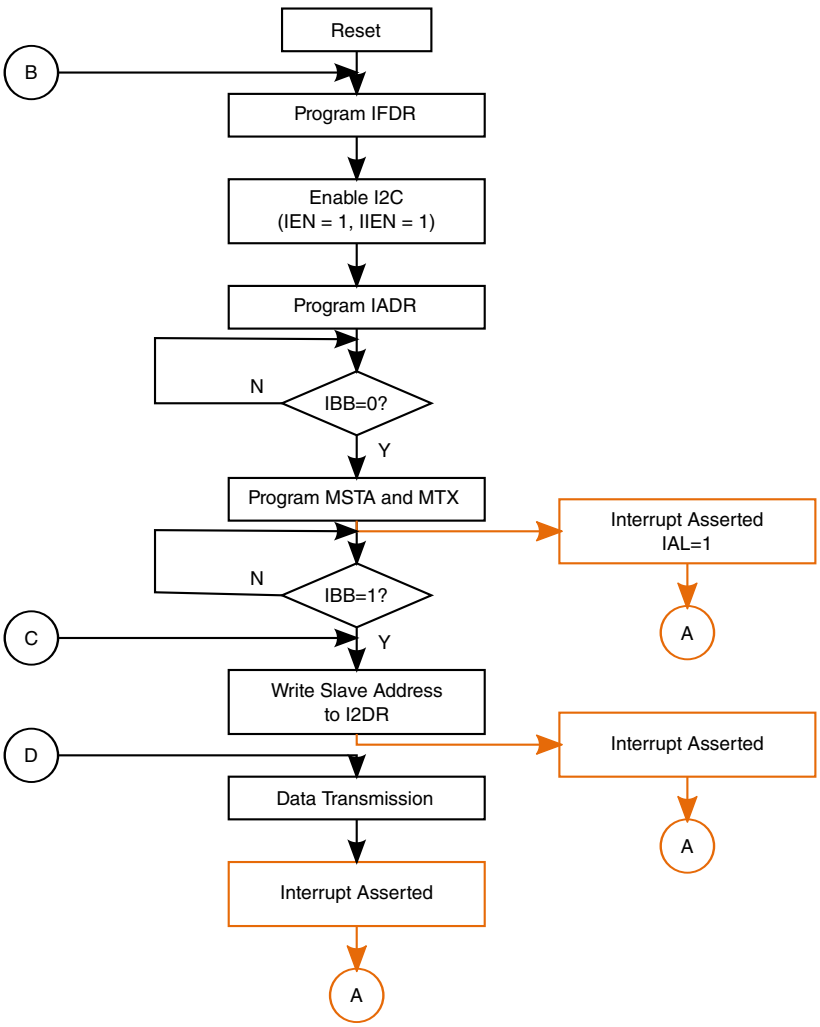


Figure 16-7. Detailed flowchart of a typical I2C Master Transmit mode, part 1

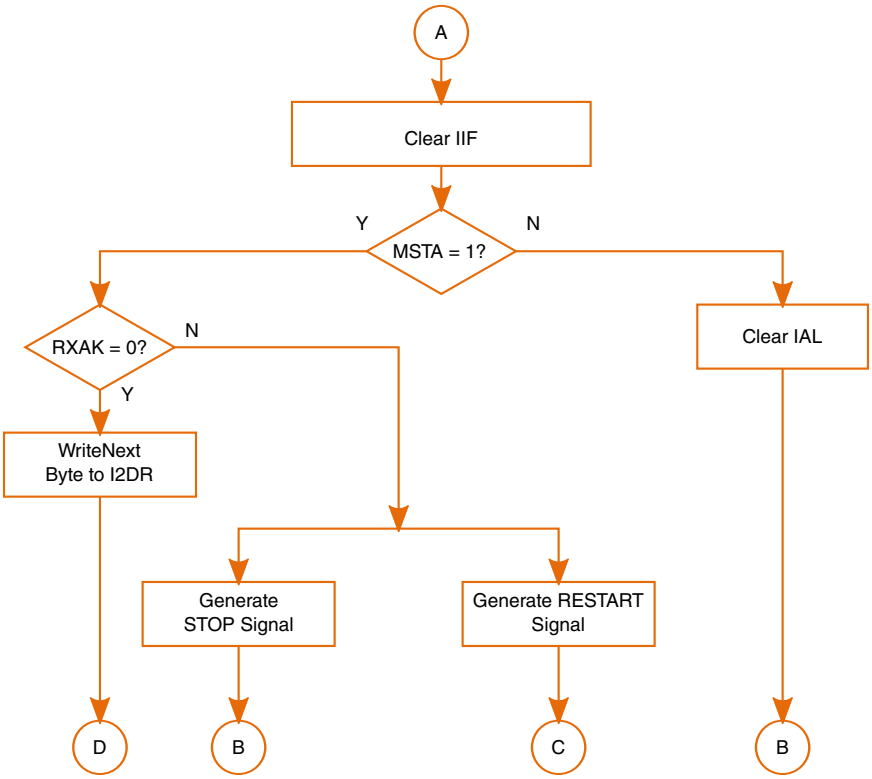


Figure 16-8. Detailed flowchart of a typical I2C Master Transmit mode, part 2

Figure 16-7 and Figure 16-8 show the Master Transmit mode operation with interrupt subroutine. If an interrupt is generated and the MSTA bit is 0, then bus arbitration is lost and IAL is set. Software can clear the IAL bit and reprogram I2C. If the MSTA bit is 1, then it is a transfer-generated interrupt. In this case, software can check the RXAK bit for a data receive acknowledgement by the slave and, accordingly, decide to do one of the following:

- Generate a STOP
- Generate a REPEATED START by writing to the I2C_I2CR register
- Perform the next data transfer by writing to the I2C_I2DR register

NOTE

The IBB bit is asserted by a Start condition on the bus, and it is deasserted by a Stop condition on the bus. Therefore, if arbitration is lost due to an unexpected Stop condition during transfer, then IBB is cleared. If arbitration is lost due to a data mismatch, then it is not cleared. Software should always clear the IEN bit and then set it if arbitration is lost.

16.1.4 Software restriction

Software should ensure that there is a delay of at least two module clock cycles after it sets the I2C_I2CR[RSTA] bit and before writing to the I2C_I2DR register. The maximum possible clock period of the module clock is 78 ns.

16.1.5 I2C Memory Map/Register Definition

The I2C contains five 16-bit registers.

NOTE

Registers at offsets 0x0002, 0x0006, 0x000A, and 0x000E are reserved for future additions.

I2C memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30A2_0000	I2C Address Register (I2C1_IADR)	16	R/W	0000h	16.1.5.1/ 5274
30A2_0004	I2C Frequency Divider Register (I2C1_IFDR)	16	R/W	0000h	16.1.5.2/ 5274
30A2_0008	I2C Control Register (I2C1_I2CR)	16	R/W	0000h	16.1.5.3/ 5276
30A2_000C	I2C Status Register (I2C1_I2SR)	16	R/W	0081h	16.1.5.4/ 5277
30A2_0010	I2C Data I/O Register (I2C1_I2DR)	16	R/W	0000h	16.1.5.5/ 5279
30A3_0000	I2C Address Register (I2C2_IADR)	16	R/W	0000h	16.1.5.1/ 5274
30A3_0004	I2C Frequency Divider Register (I2C2_IFDR)	16	R/W	0000h	16.1.5.2/ 5274
30A3_0008	I2C Control Register (I2C2_I2CR)	16	R/W	0000h	16.1.5.3/ 5276
30A3_000C	I2C Status Register (I2C2_I2SR)	16	R/W	0081h	16.1.5.4/ 5277
30A3_0010	I2C Data I/O Register (I2C2_I2DR)	16	R/W	0000h	16.1.5.5/ 5279
30A4_0000	I2C Address Register (I2C3_IADR)	16	R/W	0000h	16.1.5.1/ 5274
30A4_0004	I2C Frequency Divider Register (I2C3_IFDR)	16	R/W	0000h	16.1.5.2/ 5274

Table continues on the next page...

I2C memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30A4_0008	I2C Control Register (I2C3_I2CR)	16	R/W	0000h	16.1.5.3/ 5276
30A4_000C	I2C Status Register (I2C3_I2SR)	16	R/W	0081h	16.1.5.4/ 5277
30A4_0010	I2C Data I/O Register (I2C3_I2DR)	16	R/W	0000h	16.1.5.5/ 5279
30A5_0000	I2C Address Register (I2C4_IADR)	16	R/W	0000h	16.1.5.1/ 5274
30A5_0004	I2C Frequency Divider Register (I2C4_IFDR)	16	R/W	0000h	16.1.5.2/ 5274
30A5_0008	I2C Control Register (I2C4_I2CR)	16	R/W	0000h	16.1.5.3/ 5276
30A5_000C	I2C Status Register (I2C4_I2SR)	16	R/W	0081h	16.1.5.4/ 5277
30A5_0010	I2C Data I/O Register (I2C4_I2DR)	16	R/W	0000h	16.1.5.5/ 5279

16.1.5.1 I2C Address Register (I2Cx_IADR)

Address: Base address + 0h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0								ADR							0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

I2Cx_IADR field descriptions

Field	Description
15–8 Reserved	This read-only field is reserved and always has the value 0.
7–1 ADR	Slave address. Contains the specific slave address to be used by the I2C. Slave mode is the default I2C mode for an address match on the bus. **NOTE:** The I2C_IADR holds the address to which the I2C responds when addressed as a slave. The slave address is not the address sent on the bus during the address transfer. The register is not reset by a software reset.
0 Reserved	This read-only field is reserved and always has the value 0.

16.1.5.2 I2C Frequency Divider Register (I2Cx_IFDR)

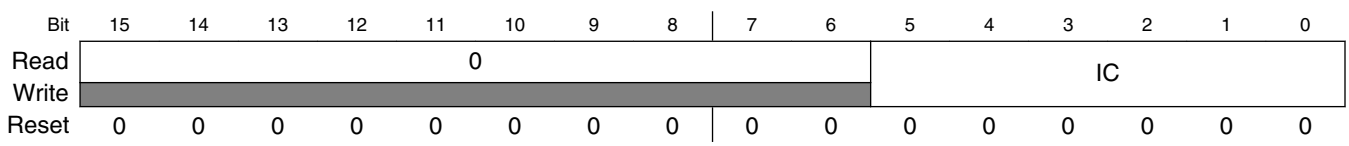
The I2C_IFDR provides a programmable prescaler to configure the clock for bit-rate selection. The register does not get reset by a software reset.

The following table describes the divider and register values for the register field "IC."

Table 16-1. I2C_IFDR Register Field Values

IC	Divider	IC	Divider	IC	Divider	IC	Divider
0x00	30	0x10	288	0x20	22	0x30	160
0x01	32	0x11	320	0x21	24	0x31	192
0x02	36	0x12	384	0x22	26	0x32	224
0x03	42	0x13	480	0x23	28	0x33	256
0x04	48	0x14	576	0x24	32	0x34	320
0x05	52	0x15	640	0x25	36	0x35	384
0x06	60	0x16	768	0x26	40	0x36	448
0x07	72	0x17	960	0x27	44	0x37	512
0x08	80	0x18	1152	0x28	48	0x38	640
0x09	88	0x19	1280	0x29	56	0x39	768
0x0A	104	0x1A	1536	0x2A	64	0x3A	896
0x0B	128	0x1B	1920	0x2B	72	0x3B	1024
0x0C	144	0x1C	2304	0x2C	80	0x3C	1280
0x0D	160	0x1D	2560	0x2D	96	0x3D	1536
0x0E	192	0x1E	3072	0x2E	112	0x3E	1792
0x0F	240	0x1F	3840	0x2F	128	0x3F	2048

Address: Base address + 4h offset



I2Cx_IFDR field descriptions

Field	Description
15–6 Reserved	This read-only field is reserved and always has the value 0.
IC	I2C clock rate. Prescales the clock for bit-rate selection. Due to potentially slow I2Cn_SCL and I2Cn_SDA rise and fall times, bus signals are sampled at the prescaler frequency. The serial bit clock frequency may be lower than IPG_CLK_ROOT divided by the divider shown in the I2C Data I/O Register. **NOTE:** The IC value should not be changed during the data transfer, however, it can be changed before a Repeat Start or Start programming sequence in I2C. The I2C protocol supports bit rates of up to 400 kbps. The IC bits need to be programmed in accordance with this constraint.

16.1.5.3 I2C Control Register (I2Cx_I2CR)

The I2C_I2CR is used to enable the I2C and the I2C interrupt. It also contains bits that govern operation as a slave or a master.

Address: Base address + 8h offset

Bit	15	14	13	12	11	10	9	8
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	IEN	IIEN	MSTA	MTX	TXAK	0	0	
Write						RSTA		
Reset	0	0	0	0	0	0	0	0

I2Cx_I2CR field descriptions

Field	Description
15–8 Reserved	This read-only field is reserved and always has the value 0.
7 IEN	I2C enable. Also controls the software reset of the entire I2C. Resetting the bit generates an internal reset to the block. If the block is enabled in the middle of a byte transfer, Slave mode ignores the current bus transfer and starts operating when the next Start condition is detected. Master mode is not aware that the bus is busy, so initiating a start cycle may corrupt the current bus cycle, ultimately causing either the current master or the I2C to lose arbitration. Subsequently, bus operation returns to normal. 0 The block is disabled, but registers can still be accessed. 1 The I2C is enabled. This bit must be set before any other I2C_I2CR bits have an effect.
6 IIEN	I2C interrupt enable. **NOTE:** If data is written during the Start condition, that is, just after setting the I2C_I2CR[MSTA] and I2C_I2CR[MTX] bits, then the ICF bit is cleared at the falling edge of SCLK after Start. If data is written after the Start condition and falling edge of SCLK, then the ICF bit is cleared as soon as data is written. 0 I2C interrupts are disabled, but the status flag I2C_I2SR[IIF] continues to be set when an Interrupt condition occurs. 1 I2C interrupts are enabled. An I2C interrupt occurs if I2C_I2SR[IIF] is also set.
5 MSTA	Master/Slave mode select bit. If the master loses arbitration, MSTA is cleared without generating a Stop signal. **NOTE:** The module clock should be on for writing to the MSTA bit. **NOTE:** The MSTA bit is cleared by software to generate a Stop condition; it can also be cleared by hardware when the I2C loses the bus arbitration.

Table continues on the next page...

I2Cx_I2CR field descriptions (continued)

Field	Description
	0 Slave mode. Changing MSTA from 1 to 0 generates a Stop and selects Slave mode. 1 Master mode. Changing MSTA from 0 to 1 signals a Start on the bus and selects Master mode.
4 MTX	Transmit/Receive mode select bit. Selects the direction of master and slave transfers. 0 Receive. When a slave is addressed, the software should set MTX according to the slave read/write bit in the I2C status register (I2C_I2SR[SRW]). 1 Transmit. In Master mode, MTX should be set according to the type of transfer required. Therefore, for address cycles, MTX is always 1.
3 TXAK	Transmit acknowledge enable. Specifies the value driven onto I2Cn_SDA during acknowledge cycles for both master and slave receivers. **NOTE:** Writing TXAK applies only when the I2C bus is a receiver. 0 An acknowledge signal is sent to the bus at the ninth clock bit after receiving one byte of data. 1 No acknowledge signal response is sent (that is, the acknowledge bit = 1).
2 RSTA	Repeat start. Always reads as 0. Attempting a repeat start without bus mastership causes loss of arbitration. 0 No repeat start 1 Generates a Repeated Start condition
Reserved	This read-only field is reserved and always has the value 0.

16.1.5.4 I2C Status Register (I2Cx_I2SR)

The I2C_I2SR contains bits that indicate transaction direction and status.

Address: Base address + Ch offset

Bit	15	14	13	12	11	10	9	8
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	ICF	IAAS	IBB	IAL	0	SRW	IIF	RXAK
Write								
Reset	1	0	0	0	0	0	0	1

I2Cx_I2SR field descriptions

Field	Description
15–8 Reserved	This read-only field is reserved and always has the value 0.
7 ICF	Data transferring bit. While one byte of data is transferred, ICF is cleared. 0 Transfer is in progress. 1 Transfer is complete. This bit is set by the falling edge of the ninth clock of the last byte transfer.
6 IAAS	I2C addressed as a slave bit. The Arm platform is interrupted if the interrupt enable (I2C_I2CR[IIEN]) is set. The Arm platform must check the slave read/write bit (SRW) and set its Transfer/Receive mode accordingly. Writing to I2C_I2CR clears this bit. 0 Not addressed 1 Addressed as a slave. Set when its own address (I2C_IADR) matches the calling address.
5 IBB	I2C bus busy bit. Indicates the status of the bus. **NOTE:** When I2C is enabled (I2C_I2CR[IEN] = 1), it continuously polls the bus data (SDA) and clock (SCL) signals to determine a Start or Stop condition. 0 Bus is idle. If a Stop signal is detected, IBB is cleared. 1 Bus is busy. When Start is detected, IBB is set.
4 IAL	Arbitration lost. Set by hardware in the following circumstances (IAL must be cleared by software by writing a "0" to it at the start of the interrupt service routine): • I2Cn_SDA input samples low when the master drives high during an address or data-transmit cycle. • I2Cn_SDA input samples low when the master drives high during the acknowledge bit of a data-receive cycle. For the above two cases, the bit is set at the falling edge of the ninth I2Cn_SCL clock during the ACK cycle. • A Start cycle is attempted when the bus is busy. • A Repeated Start cycle is requested in Slave mode. • A Stop condition is detected when the master did not request it. **NOTE:** Software cannot set the bit. 0 No arbitration lost. 1 Arbitration is lost.
3 Reserved	This read-only field is reserved and always has the value 0.
2 SRW	Slave read/write. When the I2C is addressed as a slave, IAAS is set, and the slave read/write bit (SRW) indicates the value of the R/W command bit of the calling address sent from the master. SRW is valid only when a complete transfer has occurred, no other transfers have been initiated, and the I2C is a slave and has an address match. 0 Slave receive, master writing to slave 1 Slave transmit, master reading from slave
1 IIF	I2C interrupt. Must be cleared by the software by writing a "0" to it in the interrupt routine. **NOTE:** The software cannot set the bit. 0 No I2C interrupt pending. 1 An interrupt is pending.

Table continues on the next page...

I2Cx_I2SR field descriptions (continued)

Field	Description
	This causes a processor interrupt request (if the interrupt enable is asserted [IIEN = 1]). The interrupt is set when one of the following occurs: • One byte transfer is completed (the interrupt is set at the falling edge of the ninth clock). • An address is received that matches its own specific address in Slave Receive mode. • Arbitration is lost.
0 RXAK	Received acknowledge. This is the value received from the I2Cn_SDA input for the acknowledge bit during a bus cycle. 0 An "acknowledge" signal was received after the completion of an 8-bit data transmission on the bus. 1 A "No acknowledge" signal was detected at the ninth clock.

16.1.5.5 I2C Data I/O Register (I2Cx_I2DR)

In Master Receive mode, reading the data register allows a read to occur and initiates the next byte to be received. In Slave mode, the same function is available after it is addressed.

Address: Base address + 10h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0								DATA							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

I2Cx_I2DR field descriptions

Field	Description
15–8 Reserved	This read-only field is reserved and always has the value 0.
DATA	Data Byte. Holds the last data byte received or the next data byte to be transferred. Software writes the next data byte to be transmitted or reads the data byte received. **NOTE:** The core-written value in I2C_I2DR cannot be read back by the core. Only data written by the I2C bus side can be read.

16.2 Universal Asynchronous Receiver/Transmitter (UART)

16.2.1 Overview

Universal Asynchronous Receiver/Transmitter (UART) provides serial communication capability with external devices through a level converter and an RS-232 cable or through use of external circuitry that converts infrared signals to electrical signals (for reception) or transforms electrical signals to signals that drive an infrared LED (for transmission) to provide low speed IrDA compatibility.

UART supports NRZ encoding format , RS485 compatible 9 bit data format and IrDA-compatible infrared slow data rate (SIR) format.

Figure 16-9 is the UART block diagram.

The "Module Clock" is the UART_CLK which comes from CCM. The "Peripheral Clock" is the IPG_CLK which comes from CCM.

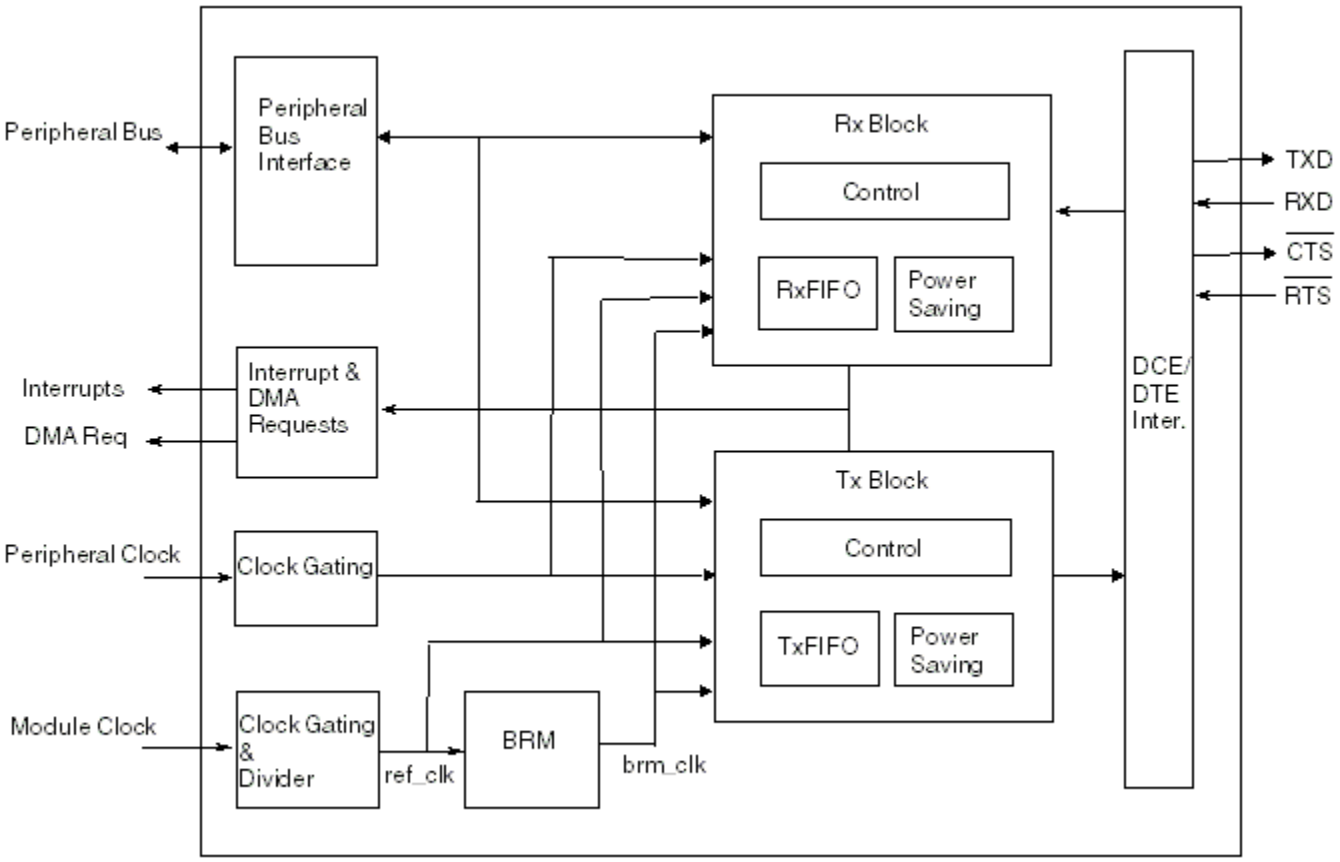


Figure 16-9. UART Block Diagram

16.2.1.1 Features

The UART includes the following features:

- High-speed TIA/EIA-232-F compatible, up to Mbit/s
- Serial IR interface low-speed, IrDA-compatible (up to 115.2 Kbit/s)
- 9-bit or Multidrop mode (RS-485) support (automatic slave address detection)
- 7 or 8 data bits for RS-232 characters, or 9 bit RS-485 format
- 1 or 2 stop bits
- Programmable parity (even, odd, and no parity)
- Hardware flow control support for request to send (RTS_B) and clear to send (CTS_B) signals
- RS-485 driver direction control via CTS_B signal
- Edge-selectable RTS_B and edge-detect interrupts
- Status flags for various flow control and FIFO states
- Voting logic for improved noise immunity (16x oversampling)
- Transmitter FIFO empty interrupt suppression

- UART internal clocks enable/disable
- Auto baud rate detection (up to 115.2 Kbit/s)
- Receiver and transmitter enable/disable for power saving
- RX_DATA input and TX_DATA output can be inverted respectively in RS-232/ RS-485 mode
- DCE/DTE capability
- RTS_B, IrDA asynchronous wake (AIRINT), receive asynchronous wake (AWAKE) interrupts wake the processor from STOP mode
- Maskable interrupts
- Two DMA Requests (TxFIFO DMA Request and RxFIFO DMA Request)
- Escape character sequence detection
- Software reset (SRST_B)
- Two independent, 32-entry FIFOs for transmit and receive
- The peripheral clock can be totally asynchronous with the module clock. The module clock determines baud rate. This allows frequency scaling on peripheral clock (such as during DVFS mode) while remaining the module clock frequency and baud rate.

16.2.1.2 Modes of operation

- Serial RS-232NRZ mode
- 9-bit RS-485 mode
- IrDA mode

To set UART in different modes, see the table below.

Table 16-2. UART mode definition

MDEN (UMCR[0])	IREN (UCR1[7])	UART Mode	Descriptioin
0	0	RS-232	RXD/TXD data is serial RS-232 NRZ format
0	1	IrDA (Interface)	RXD/TXD data is IrDA-compatible infrared slow data rate (SIR) format
1	0	RS-485	RXD/TXD data is RS485 compatible 9 bit data format
1	1	Undefined	Undefined

16.2.2 Functional Description

This section provides a complete functional description of the block.

16.2.2.1 Interrupts and DMA Requests

See the following table for the lists of all interrupt and DMA signals and associated interrupt and DMA sources of the UART. See register description section for explanation of interrupt/DMA enable and status.

Table 16-3. Interrupts and DMA

Interrupt/DMA Output	Interrupt/DMA Enable	Enable Register Location	Interrupt/DMA Flag	Flag Register Location
interrupt_uart	RRDYEN	UCR1 (bit 9)	RRDY	USR1 (bit 9)
	IDEN	UCR1 (bit 12)	IDLE	USR2 (bit 12)
	DREN	UCR4 (bit 0)	RDR	USR2 (bit 0)
	RXDSEN	UCR3 (bit 6)	RXDS	USR1 (bit 6)
	ATEN	UCR2 (bit 3)	AGTIM	USR1 (bit 8)
interrupt_uart	TXMPTYEN	UCR1 (bit 6)	TXFE	USR2 (bit 14)
	TRDYEN	UCR1 (bit 13)	TRDY	USR1 (bit 13)
	TCEN	UCR4 (bit 3)	TXDC	USR2 (bit 3)
interrupt_uart	OREN	UCR4 (bit 1)	ORE	USR2 (bit 1)
	BKEN	UCR4 (bit 2)	BRCD	USR2 (bit 2)
	WKEN	UCR4 (bit 7)	WAKE	USR2 (bit 7)
	ADEN	UCR1 (bit 15)	ADET	USR2 (bit 15)
	ACIEN	UCR3 (bit 0)	ACST	USR2 (bit 11)
	ESCI	UCR2 (bit 15)	ESCF	USR1 (bit 11)
	ENIRI	UCR4 (bit 8)	IRINT	USR2 (bit 8)
	AIRINTEN	UCR3 (bit 5)	AIRINT	USR1 (bit 5)
	AWAKEN	UCR3 (bit 4)	AWAKE	USR1 (bit 4)
	FRAERREN	UCR3 (bit 11)	FRAERR	USR1 (bit 10)
	PARERREN	UCR3 (bit 12)	PARITYERR	USR1 (bit 15)
	RTSDEN	UCR1 (bit 5)	RTSD	USR1 (bit 12)
	RTSEN	UCR2 (bit 4)	RTSF	USR2 (bit 4)
	DTREN (DCE)	UCR3 (bit 13)	DTRF	USR2 (bit 13)
	RI (DTE)	UCR3 (bit 8)	RIDELT	USR2 (bit 10)
	DCD (DTE)	UCR3 (bit 9)	DCDDELT	USR2 (bit 6)
	DTRDEN	UCR3 (bit 3)	DTRD	USR1 (bit 7)
	SADEN	UMCR (bit 3)	SAD	USR1 (bit 3)
dma_req_rx	RXDMAEN	UCR1 (bit 8)	RRDY	USR1 (bit 9)
	ATDMAEN	UCR1 (bit 2)	AGTIM	USR1 (bit 8)
	IDDMAEN	UCR4 (bit 6)	IDLE	USR2 (bit 12)
dma_req_tx	TXDMAEN	UCR1 (bit 3)	TRDY	USR1 (bit 13)

16.2.2.2 Clocks

This section describes clocks and special clocking requirements of the UART.

16.2.2.2.1 Clock requirements

UART module receives 2 clocks, *peripheral_clock* and *module_clock*. The *peripheral_clock* is used as write clock of the TxFIFO, read clock of the RxFIFO and synchronization of the modem control input pins. It must always be running when UART is enabled. There is an exception in stop mode (see Clocking in Low-Power Modes).

The *module_clock* is for all the state machines, writing RxFIFO, reading TxFIFO, etc. It must always be running when UART is sending or receiving characters.This clock is used in order to allow frequency scaling on *peripheral_clock* without changing configuration of baud rate (*module_clock* staying at a fixed frequency).

The constraints on *peripheral_clock* and *module_clock* are as follows:

- *peripheral_clock* and *module_clock* can totally be asynchronous. They can also be synchronous.
- Due to the 16x oversampling of the incoming characters, *module_clock* frequency must always be greater or equal to 16x the maximum baud rate. For example, if max baud rate is 4 Mbit/s, *module_clock* must be greater or equal to 4 M x 16 = 64 MHz.

NOTE

The restriction that *peripheral_clock* frequency must be higher or equal to 16x baud rate has been removed. There is no limitation on *peripheral_clock* frequency to baud rate.

16.2.2.2.2 Maximum Baud Rate

The max baud rate the UART can support is determined by the max frequency of the module_clock.

For example, if the SoC can provide the fastest module_clock 66.5 MHz, the UART can transmit and receive serial data with the maximum baud rate 66.5M/16 = 4.15 Mbit/s.

The UART supports serial IR interface low speed. In the low speed IrDA mode, the max baud rate is 115.2 Kbit/s. To support the 115.2 Kbit/s, module_clock frequency must be higher or equal to 1.8432 MHz.

16.2.2.2.3 Clocking in Low-Power Modes

The UART supports 2 low-power modes: DOZE and STOP.

In STOP mode (input pin *stop_req* is at '1'), the UART doesn't need any clock. In this mode the UART can wake-up the Arm platform with the asynchronous interrupts (see Low Power Modes).

- If before entering in STOP mode the software has enabled RTSDEN interrupt, when RTS will change state (put at '0' by external device started to send), the asynchronous interrupt will wake-up the system, *peripheral_clock* and *module_clock* will be provided to the UART before first start bit, so that no data will be lost.
- If RTS doesn't change state (already at '0' before entering in STOP mode), then wake-up interrupt (AWAKE) will be sent at the arrival of first Start bit (on falling edge). In this case, the UART must receive the *peripheral_clock* and *module_clock* during the first half of start bit to correctly receive this character (for example, at 115.2 Kbit/s, UART must receive *peripheral_clock* and *module_clock* at maximum 4.3 microseconds after falling edge of Start bit). If the UART receives *peripheral_clock* and *module_clock* too late, first character will be lost, and so should be dropped. Also, if autobaud detection is enabled, the first character won't be correctly received and another autobaud detection will need to be initiated.

In Doze mode, UART behavior is programmable through DOZE bit (UCR1[1]). If DOZE bit is set to '1', then UART is disabled in Doze mode, and in consequence, UART clocks can be switched-off (after being sure UART is not transmitting nor receiving). On the contrary, if DOZE bit is set to '0', UART is enabled and it must receive *peripheral_clock* and *module_clock* .

16.2.2.3 General UART Definitions

Definitions of terms that occurs the following discussions are given in this section.

- Bit Time-The period of time required to serially transmit or receive 1 bit of data (1 cycle of the baud rate frequency).
- Start bit-The bit time of a logic 0 that indicates the beginning of a data frame. A start bit begins with a 1-to-0 transition, and is preceded by at least 1 bit time of logic 1.
- Stop bit-1 bit time of logic 1 that indicates the end of a data frame.
- BREAK-A frame in which all of the data bits, including the stop bit, are logic 0. This type of frame is usually sent to signal the end of a message or the beginning of a new message.
- Mark - When no data is being sent, the serial port's transmit pin's voltage is 1 and is said to be in a MARK state.

- Space - The serial port can also be forced to keep the transmit pin at a 0 and is said to be the SPACE or BREAK state.
- Frame-A start bit followed by a specified number of data or information bits and terminated by a stop bit. The number of data or information bits depends on the format specified and must be the same for the transmitting device and the receiving device. The most common frame format is 1 start bit followed by 8 data bits (least significant bit first) and terminated by 1 stop bit. An additional stop bit and a parity bit also can be included.
- Framing Error-An error condition that occurs when the stop bit of a received frame is missing, usually when the frame boundaries in the received bit stream are not synchronized with the receiver bit counter. Framing errors can go undetected if a data bit in the expected stop bit time happens to be a logic 1. A framing error is always present on the receiver side when the transmitter is sending BREAKs. However, when the UART is programmed to expect 2 stop bits and only the first stop bit is received, this is not a framing error by definition.
- Parity Error-An error condition that occurs when the calculated parity of the received data bits in a frame does not match the parity bit received on the RX_DATA input. Parity error is calculated only after an entire frame is received.
- Idle-One in NRZ encoding format and selectable polarity in IrDA mode.
- Overrun Error-An error condition that occurs when the latest character received is ignored to prevent overwriting a character already present in the UART receive buffer (RxFIFO). An overrun error indicates that the software reading the buffer (RxFIFO) is not keeping up with the actual reception of characters on the RX_DATA input.

16.2.2.3.1 RTS_B - UART Request To Send

The UART Request To Send input controls the transmitter. The modem or other terminal equipment signals the UART when it is ready to receive by setting '0' on the RTS_B pin.

Normally, the transmitter waits until this signal is active (low) before transmitting a character, however when the Ignore RTS (IRTS) bit is set, the transmitter sends a character as soon as it is ready to transmit. An interrupt (RTSD) can be posted on any transition of this pin and can wake the Arm platform from STOP mode on its assertion. When RTS_B is set to '1' during a transmission, the UART transmitter finishes transmitting the current character and shuts off. The contents of the TxFIFO (characters to be transmitted) remain undisturbed. The operation of this input is the same regardless of whether the UART is in DTE or DCE mode.

16.2.2.3.2 RTS Edge Triggered Interrupt

The input to the RTS_B pin can be programmed to generate an interrupt on a selectable edge.

See the table below for summary of the operation of the RTS edge triggered interrupt (RTSF).

To enable the RTS_B pin to generate an interrupt, set the request to send interrupt enable (RTSEN) bit (UCR2[4]) to 1. Writing 1 to the RTS_B edge triggered interrupt flag (RTSF) bit (USR2[4]) clears the interrupt flag. The interrupt can occur on the rising edge, falling edge, or either edge of the RTS_B input. The request to send edge control (RTEC) field (UCR2[10:9]) programs the edge that generates the interrupt. When RTEC is set to 0x00 and RTSEN = 1, the interrupt occurs on the rising edge (default). When RTEC is set to 0x01 and RTSEN = 1, the interrupt occurs on the falling edge. When RTEC is set to 0x1X and RTSEN = 1, the interrupt occurs on either edge. This is a synchronous interrupt. The RTSF bit is cleared by writing 1 to it. Writing 0 to RTSF has no effect.

Table 16-4. RTS_B Edge Triggered Interrupt Truth Table

RTS_B	RTSEN	RTEC [1]	RTEC [0]	RTSF	Interrupt Occurs On…	interrupt_uart
X	0	X	X	0	Interrupt disabled	1
1->0	1	0	0	0	Rising edge	1
0->1	1	0	0	1	Rising edge	0
1->0	1	0	1	1	Falling edge	0
0->1	1	0	1	0	Falling edge	1
1->0	1	1	X	1	Either edge	0
0->1	1	1	X	1	Either edge	0

There is another RTS_B interrupt that is not programmable. The status bit RTSD asserts the *interrupt_uart* interrupt when the RTS_B delta interrupt enable = 1. This is an asynchronous interrupt. The RTSD bit is cleared by writing 1 to it. Writing 0 to the RTSD bit has no effect.

16.2.2.3.3 CTS_B - Clear To Send

This output pin serves two purposes. Normally, the receiver indicates that it is ready to receive data by asserting this pin (low). When the CTS_B trigger level is programmed to trigger at 32 characters received and the receiver detects the valid start bit of the 33 character, it de-asserts this pin. The operation of this output is the same regardless of whether the UART is in DTE or DCE mode.

16.2.2.3.4 Programmable CTS_B Deassertion

The CTS_B output can also be programmed to deassert when the RxFIFO reaches a certain level. Setting the CTS trigger level (UCR4[15:10]) at any value less than 32 deasserts the CTS_B pin on detection of the valid start bit of the N + 1 character (where N is the trigger level setting). However, the receiver continues to receive characters until the RxFIFO is full.

16.2.2.3.5 TX_DATA - UART Transmit

This is the transmitter serial output. When operating in RS-232/RS-485 mode, NRZ encoded data is transmitted, and the data can be inverted (controlled by INVT (UCR3[1])) before transmitted. When operating in infrared mode, a 3/16 bit-period pulse is output for each 0 bit transmitted, and no pulse is output for each 1 bit transmitted.

For RS-232/RS-485 applications, this pin must be connected to an RS-232/RS-485 transmitter. The operation of this output is the same regardless of whether the UART is in DTE or DCE mode. See Figure 16-10.

16.2.2.3.6 RX_DATA - UART Receive

This is the receiver serial input. When operating in RS-232/RS-485 mode, NRZ encoded data is expected, and the data can be inverted (controlled by INVR (UCR4[9])) before sampled. When operating in infrared mode, a narrow pulse is expected for each 0 bit received and no pulse is expected for each 1 bit received.

External circuitry must convert the IR signal to an electrical signal. RS-232/RS-485 applications require an external RS-232/RS-485 receiver to convert voltage levels. The operation of this input is the same regardless of whether the UART is in DTE or DCE mode. See the figure below.

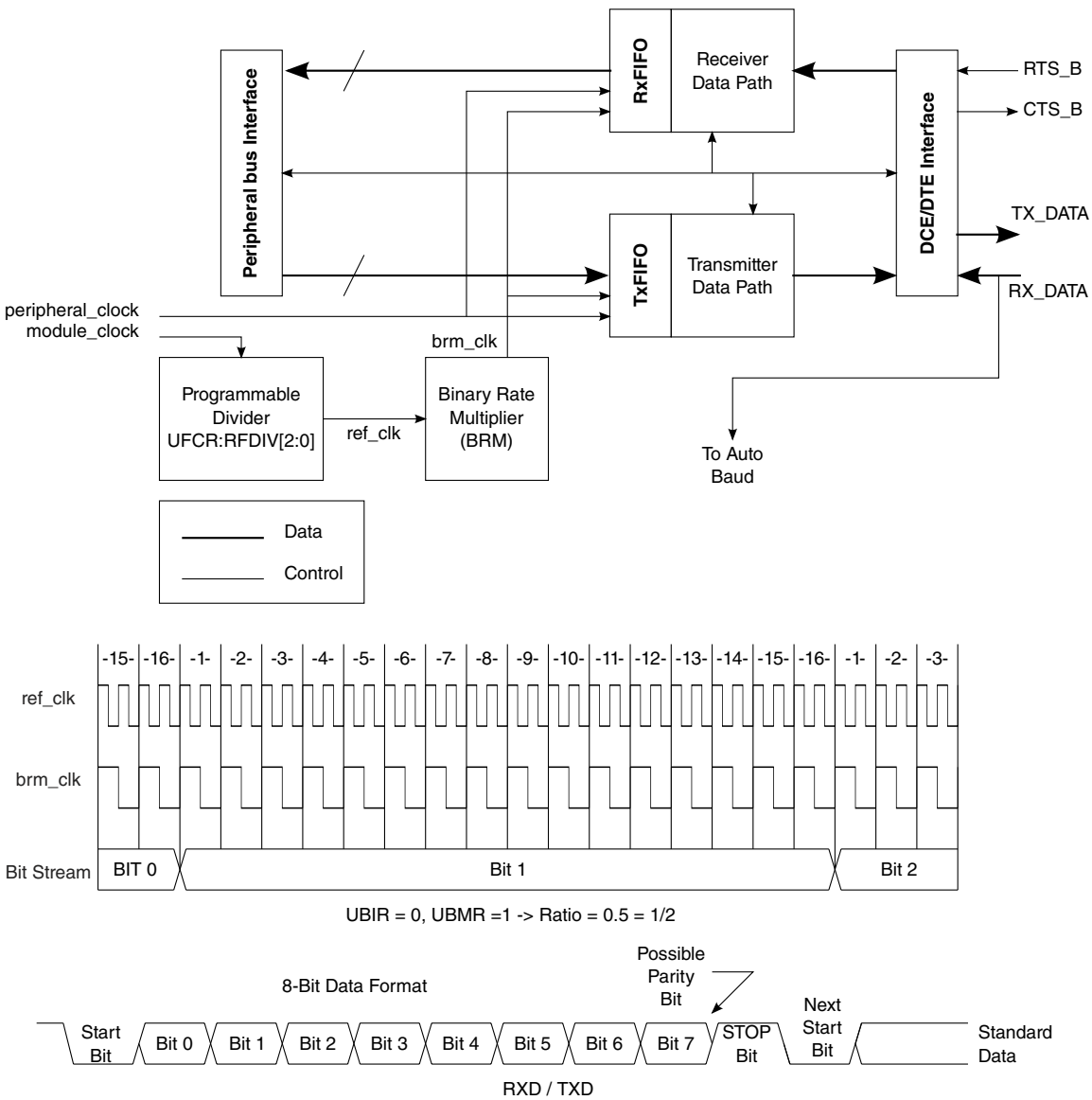


Figure 16-10. UART Simplified Block and Clock Generation Diagrams

16.2.2.4 Transmitter

The transmitter accepts a parallel character from the Arm platform and transmits it serially. The start, stop, and parity (when enabled) bits are added to the character.

When the ignore RTS bit (IRTS) is set, the transmitter sends a character as soon as it is ready to transmit. RTS_B can be used to provide flow-control of the serial data. When RTS_B is set to '1', the transmitter finishes sending the character in progress (if any), stops, and waits for RTS_B to be set to '0' again. Generation of BREAK characters and parity errors (for debugging purposes) is supported. The transmitter operates from the clock provided by the Binary Rate Multiplier(BRM). Normal NRZ encoded data is transmitted when the IR interface is disabled.

The transmitter FIFO (TxFIFO) contains 32 bytes. The data is written to TxFIFO by writing to the UTXD register with the byte data to the [7:0] bits. The data is written consecutively if the TxFIFO is not full. It is read (internally) consecutively if the TxFIFO is not empty. TXFULL bit (UTS[4]) can be used to control whether TXFIFO is full or not. The TxFIFO can be written regardless of the transmitter is disabled or enabled. If the UART is disabled, user can still write data into the TxFIFO correctly. But in this case the write access will yield to a transfer error.

16.2.2.4.1 Transmitter FIFO Empty Interrupt Suppression

The transmitter FIFO empty interrupt suppression logic suppresses the TXFE interrupt between writes to the TxFIFO.

When TxFIFO is empty, the software can either send one or several characters. If the software sends one character, it would write the character into the UTXD register, then that character is immediately transferred to the transmitter shift register, assuming the transmitter is already enabled. Without interrupt suppression logic, the TXFE interrupt flag would be set immediately. But, with this logic, the interrupt flag is set when the last bit of the character has been transmitted, for example, before the transmission of the parity bit (if exists) and the stop bit(s).

So, the suppression logic doesn't immediately send the TXFE interrupt flag. It allows the software to write another character to the TxFIFO before the interrupt flag is asserted.

When the transmitter shift register empties before another character is written to the TxFIFO, the interrupt flag is asserted. Writing data to the TxFIFO would release the interrupt flag. The interrupt flag is asserted on the following conditions:

- System Reset
- UART software reset

- When a single character has been written to Transmitter FIFO and then the Transmitter FIFO and the Transmitter Shift Register become empty until another character is written to the Transmitter FIFO
- The last character in the TxFIFO is transferred to the shift register, when TxFIFO contains two or more characters. See the figure below.

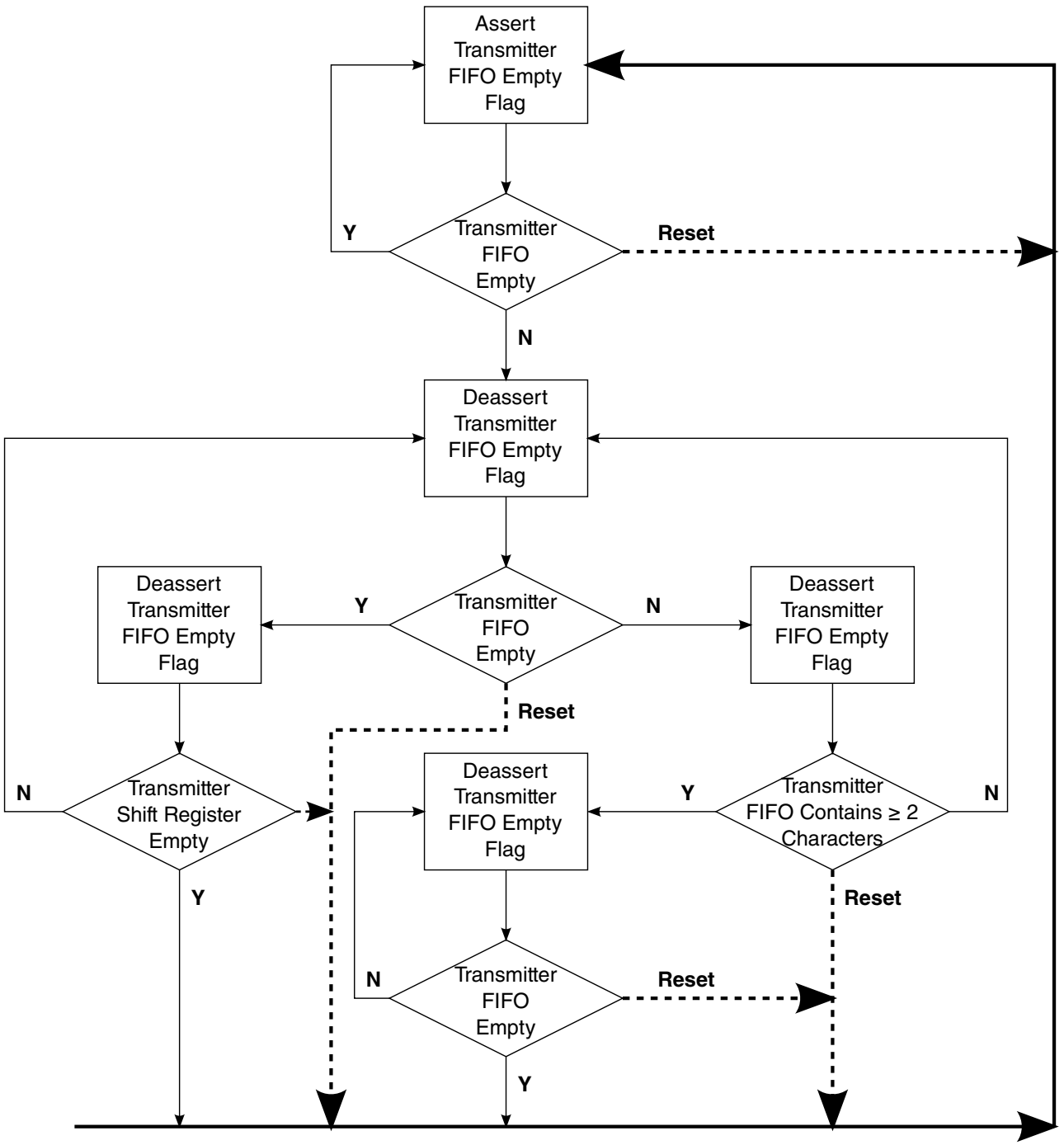


Figure 16-11. Transmitter FIFO Empty Interrupt Suppression Flow Chart

16.2.2.4.2 Transmitting a Break Condition

Asserting SNDBRK bit of the UCR1 Register forces the transmitter to send a break character (continuous zeros). The transmitter will finish sending the character in progress (if any) before sending break until this bit is reset.

The user is responsible to ensure that this bit is high for long enough to generate a valid BREAK. The transmitter samples SNDBRK after every bit is transmitted. Following completion of the BREAK transmission, the UART will transmit two mark bits. The user can continue to fill the FIFO and any character remaining will be transmitted when the break is terminated.

16.2.2.5 Receiver

See the figure below for the receiver flow chart.

The receiver accepts a serial data stream and converts it into parallel characters. When enabled, it searches for a start bit, qualifies it, and samples the following data bits at the bit-center.

Jitter tolerance and noise immunity are provided by sampling at a 16x rate and using voting techniques to clean up the samples. Once the start bit is found, the data bits, parity bit (if enabled), and stop bits (either 1 or 2 depending on user selection) are shifted in. Parity is checked and its status reported in the URXD register when parity is enabled. Frame errors and BREAKs are also checked and reported. When a new character is ready to be read by the Arm platform from the RxFIFO, the receive data ready (RDR = USR2[0]) bit is asserted and an interrupt is posted (if DREN = UCR4[0] = 1). If the receiver trigger level is set to 2 (RXTL[5:0] = UFCR[5:0] = 2), and 2 chars have been received into RxFIFO, the receiver ready interrupt flag (RRDY = USR1[9]) is asserted and an interrupt is posted if the receiver ready interrupt enable bit is set (RRDYEN = UCR1[9] = 1). If the UART Receiver Register (URXD) is read once, and in consequence there is only 1 character in the RxFIFO, the interrupt generated by the RDR bit is automatically cleared. The RRDY bit is cleared when the data in the RxFIFO falls below the programmed trigger level.

Normal NRZ encoded data is expected when the IR interface is disabled. The RxFIFO contains 32 half-word entries. Characters received are written consecutively into this FIFO. If the FIFO is full and a 33rd characters is received, this character will be ignored and the USR2[ORE] bit will be set.

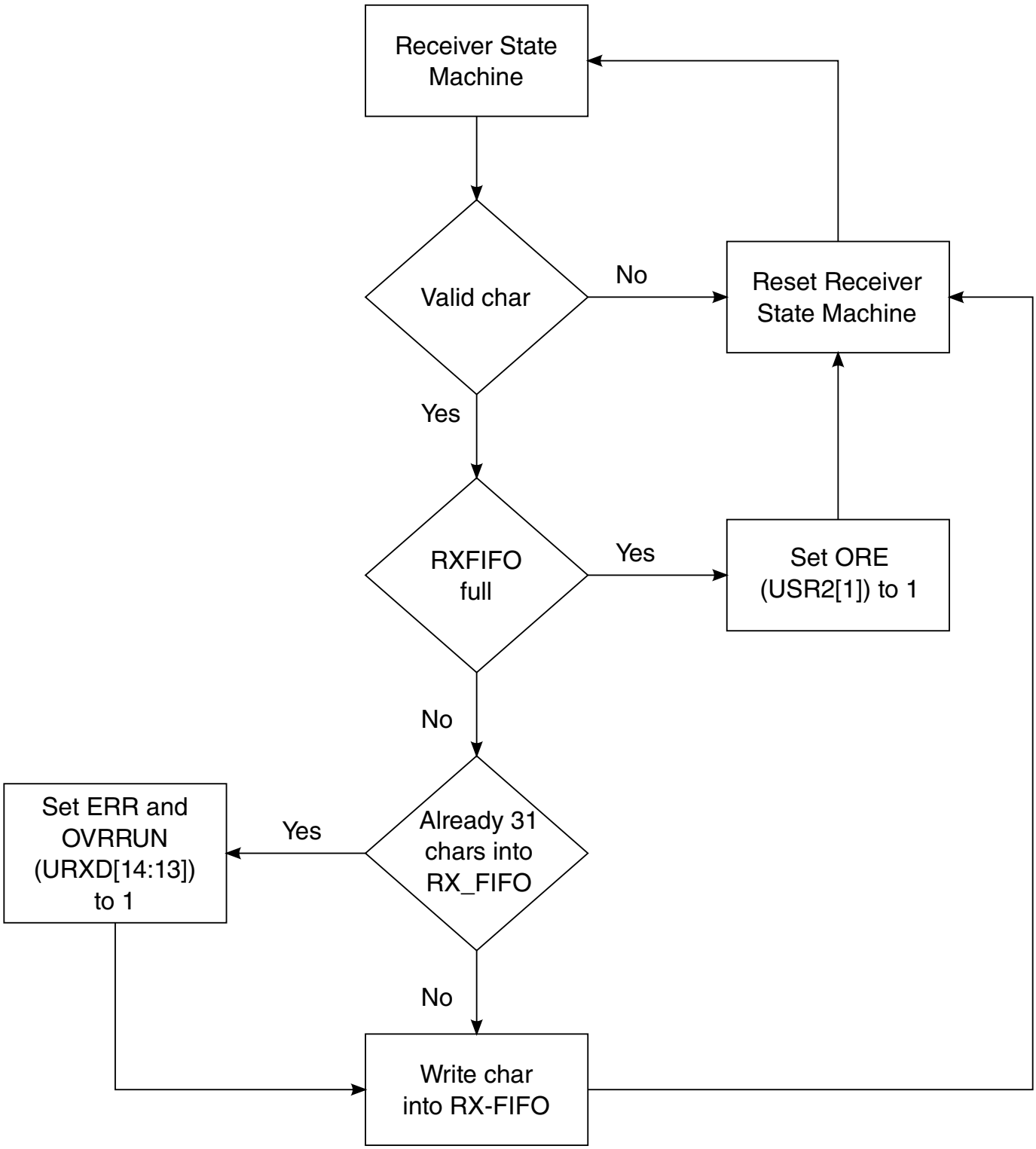


Figure 16-12. Receiver Flow Chart

16.2.2.5.1 Idle Line Detect

The receiver logic block includes the ability to detect an idle line. Idle lines indicate the end or the beginning of a message.

For an idle condition to occur:

- RxFIFO must be empty and
- RX_DATA pin must be idle for more than a configured number of frames (ICD[1:0] = UCR1[11:10]).

When the idle condition detected interrupt enable (IDEN = UCR1[12]) is set and the line is idle for 4 (default), 8, 16, or 32 (maximum) frames, the detection of an idle condition flags an interrupt (see the table below). When an idle condition is detected, the IDLE (USR2[12]) bit is set. Clear the IDLE bit by writing 1 to it. Writing 0 to the IDLE bit has no effect.

Table 16-5. Detection Truth Table

IDEN	ICD [1]	ICD [0]	IDLE	*interrupt_uart*
0	X	X	0	1
1	0	0	asserted after 4 idle frames	asserted after 4 idle frames
1	0	1	asserted after 8 idle frames	asserted after 8 idle frames
1	1	0	asserted after 16 idle frames	asserted after 16 idle frames
1	1	1	asserted after 32 idle frames	asserted after 32 idle frames
NOTE: This table assumes that no other interrupt is set at the same time this interrupt is set for the *interrupt_uart* signal. This table shows how this interrupt affects the *interrupt_uart* signal.				

During a normal message there is no idle time between frames. When all of the information bits in a frame are logic 1s, the start bit ensures that at least one logic 0 bit time occurs for each frame so that the IDLE bit is not asserted.

16.2.2.5.2 Aging Character Detect

The receiver block also includes the possibility to detect when at least one character has been sitting into the RxFIFO for a time corresponding to 8 characters. This aging character capability allows the UART to inform the Arm platform that there is less character into the RxFIFO than the Rx trigger and, no new character has been detected on the RXD line.

The aging capability is a timer which starts to count as soon as the RxFIFO is not empty and its trigger level is not reached (RRDY=0). This counter is reset when either a RxFIFO read is performed or another character starts to present on the RXD line. If none of those two events occurs, the bit AGTIM (USR1[8]) is set when the counter has

measured a time corresponding to 8 characters. AGTIM is cleared by writing a 1 to it. AGTIM can flag an interrupt to Arm platform on *interrupt_uart* if ATEN (UCR2[3]) has been set.

To summarize, AGTIM is set when:

- There is at least one character into RxFIFO.
- No read has occurred on RxFIFO and RXD line has stayed high, for a time corresponding to 8 characters.
- The RxFIFO trigger is not reached (RRDY=0)

16.2.2.5.3 Receiver Wake

The WAKE bit (USR2[7]) is set when the receiver detects a qualified Start bit. For this, two conditions must be fulfilled, firstly a falling edge on RX_DATA line must be detected and secondly the RX_DATA line must stays at low level for more than a half-bit duration.

When the wake interrupt enable WKEN (UCR4[7]) bit is enabled, the receiver flags an interrupt (*interrupt_uart*) if the WAKE status bit is set. The WAKE bit is cleared by writing 1 to it. Writing 0 to the WAKE bit has no effect. The WAKE status bit can be asserted in either serial RS-232 mode or IR mode. The generation of the WAKE interrupt needs the clock *module_clock* .

When the asynchronous wake interrupt (AWAKE) is enabled (AWAKEN = UCR3[4] = 1), and the Arm platform is in STOP mode, and UART clocks have been shut-off, then a falling edge detected on the receive pin (RX_DATA) asserts the AWAKE bit (USR1[4]) and the *interrupt_uart* interrupt to wake the Arm platform from STOP mode. Re-enable UART clocks and clear the AWAKE bit by writing 1 to it. Writing 0 to the AWAKE bit has no effect. When IR interface is enabled (UCR1[7]=1), the AWAKE bit is always not asserted. The generation of the asynchronous AWAKE interrupt does not need any clocks.

In IR mode, if the asynchronous IR WAKE interrupt is enabled (AIRINTEN = UCR3[5] = 1), and if the Arm platform is in STOP mode (UART clocks are off when Arm platform in STOP mode), then the detection of a falling edge on the receive pin (RXD_IR), asserts the AIRINT bit (USR1[5]), and the *interrupt_uart* interrupt. This interrupt wakes the Arm platform from STOP mode. Software re-enables UART clocks and clear the AIRINT bit by writing 1 to it. Writing 0 to the AIRINT bit has no effect. When IR interface is disabled (UCR1[7]=0), the AIRINT bit is always not asserted. The generation of the asynchronous AIRINT interrupt does not need any clocks.

Recommended procedure for programming the asynchronous interrupts is to first clear them by writing 1 to the appropriate bit in the UART Status Register 1 (USR1). Poll or enable the interrupt for the Receiver IDLE Interrupt Flag (RXDS) in the USR1. When asserted, the RXDS bit indicates to the software that the receiver state machine is in the idle state, the next state is idle, and the RX_DATA pin is idle (high). After following this procedure, enable the asynchronous interrupt and enter STOP mode.

16.2.2.5.4 Receiving a BREAK Condition

A BREAK condition is received when the receiver detects all 0s (including a 0 during the bit time of the stop bit) in a frame. The BREAK condition asserts the BRCD bit (USR2[2]) and writes only the first BREAK character to the RxFIFO. Clear the BRCD bit by writing 1 to it. Writing 0 to the BRCD bit has no effect.

Asserting BRCD would generate an interrupt on *interrupt_uart*. The interrupt generation can be masked using the control bit BKEN (UCR4[2]). Receiving a break condition will also effect the following bits in the receiver register URXD:

URXD(11) = BRK. While high this bit indicates that the current char was detected as a break.

URXD(12) = FRMERR. The frame error bit will always be set when BRK is set.

URXD(10) = PRERR. If odd parity was selected the parity error bit will also be set when BRK is set.

URXD(14) = ERR. The error detect bit indicates that the character present in the rx data field has an error status. This can be asserted by a break.

16.2.2.5.5 Vote Logic

The vote logic block provides jitter tolerance and noise immunity by sampling with respect to a 16x clock (*brm_clk*) and using voting techniques to clean up the samples. The voting is implemented by sampling the incoming signal constantly on the rising edge of the *brm_clk*.

See Figure 16-13. The receiver is provided with the majority vote value, which is 2 out of the 3 samples. For examples of the majority vote results of the vote logic, see the following table.

Table 16-6. Majority Vote Results

Samples	Vote
000	0

Table continues on the next page...

Table 16-6. Majority Vote Results (continued)

Samples	Vote
101	1
001	0
111	1

The vote logic captures a sample on every rising edge of *brm_clk*, however the receiver uses 16x oversampling to take its value in the middle of the sample character.

The receiver starts to count when the Start bit is set however it does not capture the contents of the RxFIFO at the time the Start bit is set. The start bit is validated when 0s are received for 7 consecutive 1/16 of bit times following the 1-to-0 transition. Once the counter reaches 0xF, it starts counting on the next bit and captures it in the middle of the sampling frame (see Table 16-6). All data bits are captured in the same manner. Once the stop bit is detected, the receiver shift register (SIPO_OUT) data is parallel shifted to the RxFIFO.

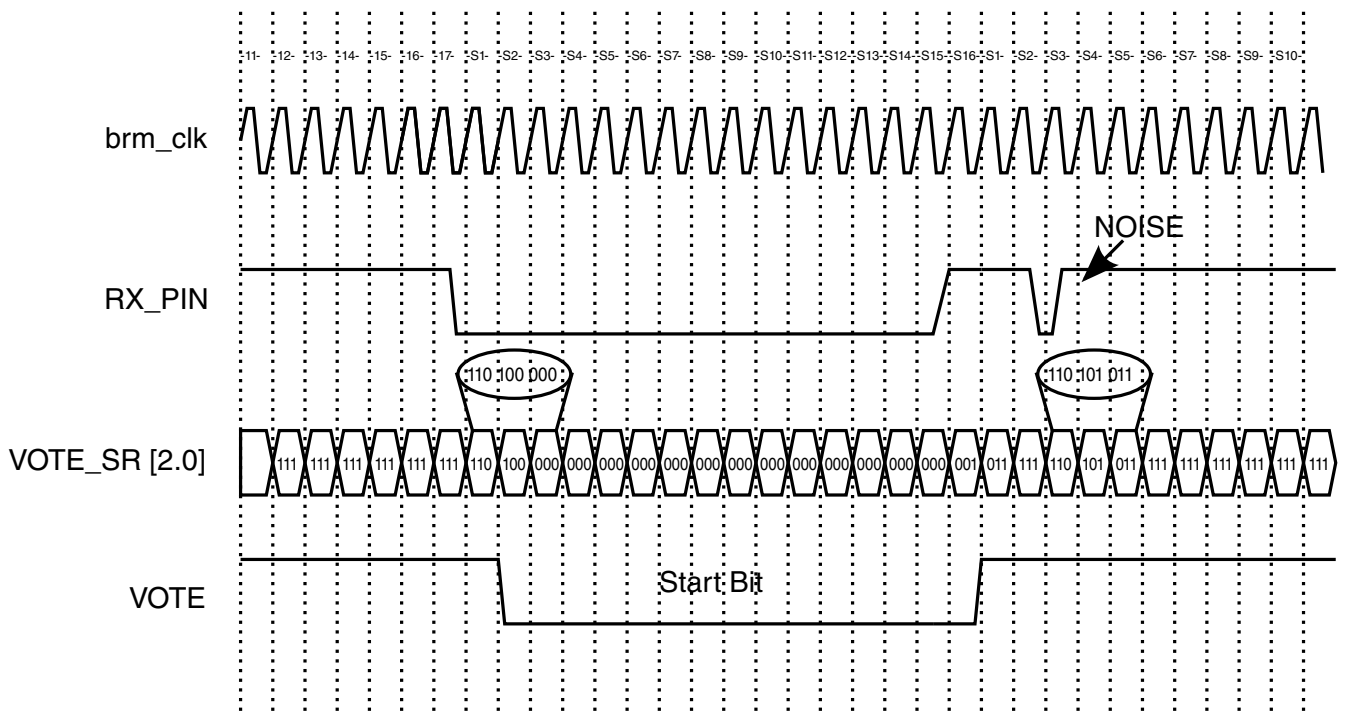


Figure 16-13. Majority Vote Results

A new feature has been recently implemented, it allows to re-synchronize the counter on each edge of RX_DATA line. This is automatic and allows to improve the immunity of UART against signal distortion.

There is a special case when the *brm_clk* frequency is too low and is unable to capture a 0 pulse in IrDA. In this case, the software must set the IRSC (UCR4[5]) bit so that the reference clock (after internal divider) is used for the voting logic. The pulse is validated by counting the length of the pulse.

Refer to Infrared Interface for more details.

16.2.2.5.6 Baud Rate Automatic Detection Logic

When the baud rate automatic detection logic is enabled, the UART locks onto the incoming baud rate. To enable this feature, set the automatic detection of baud rate bit (ADBR = UCR1[14] = 1) and write 1 to the ADET bit (USR2[15]) to clear it.

When ADET=0 and ADBR =1, the detection starts. Then, once the beginning of start bit (transition from 1-to-0 of RX_DATA) has been detected, UART starts a counter (UBRC) working at reference frequency. Once the end of start bit is detected (transition from 0-to-1 of RX_DATA), the value of UBRC - 1 is directly copied into UBMR register. UBIR register is filled with 0x000F.

So, at the end of start bit, registers gets following values:

```
UBRC = number of reference clock periods (after divider) during Start bit.
UBIR = 0x000F
UBMR = UBRC - 1
```

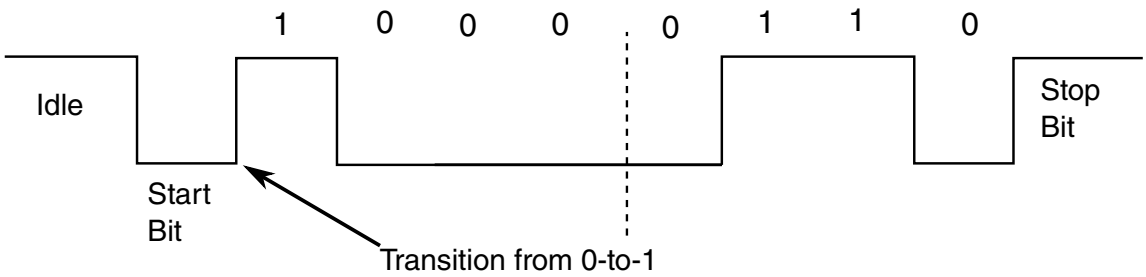
The updated values of the 3 registers can be read.

See Table 16-7 for list of parameters for baud rate detection and Figure 16-14 for baud rate detection protocol diagram.

If any of the UART BRM registers are simultaneously written by the baud rate automatic detection logic and by the peripheral data bus, the peripheral data bus would have lower priority.

Table 16-7. Baud Rate Automatic Detection

ADBR	ADET	Baud Rate Detection	*interrupt_uart*
0	X	Manual Configuration	1
1	0	Auto Detection Started	1
1	1	Auto Detection Complete	0
NOTE: This table assumes that no other interrupt is set at the same time this interrupt is set for the *interrupt_uart* signal.			



Note: LSB Transmitted first.

Figure 16-14. Baud Rate Detection Protocol Diagram

16.2.2.5.6.1 Baud Rate Automatic Detection Protocol

The receiver must receive an ASCII character "A" or "a" to verify proper detection of the incoming baud rate. When an ASCII character "A" (0x41) or "a" (0x61) is received and no error occurs, the Automatic Detect baud rate bit is set (ADET=1) and if the interrupt is enabled (ADEN=UCR1[15]=1), an interrupt *interrupt_uart* is generated.

When an ASCII character "A" or "a" is not received (because of a bit error or the reception of another character), the auto detection sequence restarts and waits for another 1-to-0 transition.

As long as ADET = 0 and ADBR = 1, the UART continues to try to lock onto the incoming baud rate. Once the ASCII character "A" or "a" is detected and the ADET bit is set, the receiver ignores the ADBR bit and continues normal operation with the calculated baud rate.

The UART interrupt is active (*interrupt_uart* = 0) as long as ADET = 1 and ADBR = 1. This can be disabled by clearing the automatic baud rate detection interrupt enable bit (ADEN = 0). Before starting an automatic baud rate detection sequence, set ADET = 0 and ADBR = 1.

The RxFIFO must contain the ASCII character "A" or "a" following the automatic baud rate detection interrupt.

The 16-bit UART Baud Rate Count Register (UBRC) is reset to 4 and stays at 0xFFFF when an overflow occurs. The UBRC register counts (measures) the duration of start bit. When the start bit is detected and counted, the UART Baud Rate Count Register retains its value until the next automatic baud rate detection sequence is initiated.

The Baud Rate Count Register counts only when auto detection is enabled.

16.2.2.5.6.2 New Baud Rate Determination

In order to fight against the problems caused by the distortion and the noise on the RX_DATA line, the duration of the baud rate measurement has been extended.

Previously, as described above, this determination was based on the measurement of the START bit duration. Now, this measurement is based on the duration of START bit + bit0. Bit0 is the first bit following the START bit. In fact, the counter which is started at the falling edge of START bit is no longer stopped at next rising edge (end of START bit), but it is stopped at the next falling edge (end of bit0). As the character sent is always a "A" (41h) or a "a" (61h), this second falling edge will always be present and it will indicate the end of bit0. Once this counter is stopped, the result is divided by 2 and used by the BRM to determine the incoming baud rate.

NOTE

UBRC register contains the result of this division by two, in consequence it reflects the measurement of the duration of one bit.

16.2.2.5.6.2.1 New Autobaud Counter Stopped bit and Interrupt

A new bit has been added in USR2 register: ACST (USR2[11]). This bit is set immediately after the determination of the baud rate.

So,

- if ADNIMP is not set (default), ACST is set to 1 after the end of bit0,
- If ADNIMP is set to 1, ACST is set to 1 at the end of START bit.

If ACIEN (UCR3[0]) is set to 1, ACST will flag an interrupt on *interrupt_uart* signal. This interrupt informs the Arm platform that the BRM has just been set with the result of the bit length measurement. If needed, the Arm platform can perform a read of UBMR (or UBRC) register and determine by itself the baud rate measured. Then the Arm platform has the possibility to correct the BRM registers with the nearest standardized baud rate.

NOTE

ACST is set only if ADBR is set to 1, for example, the UART is autobauding.

Clear the ACST bit by writing 1 to it. Writing 0 to the ACST bit has no effect.

16.2.2.6 Escape Sequence Detection

An escape sequence typically consists of 3 characters entered in rapid succession (such as +++). Because these are valid characters by themselves, the time between characters determines if it is a valid escape sequence.

Too much time between two of the "+" characters is interpreted as two "+" characters, and not part of an escape sequence.

The software chooses the escape character and writes its value to the UART Escape Character Register (UESC). The software must also enable escape detection feature by setting ESCEN (UCR2[11]) to 1. The hardware compares this value to incoming characters in the RxFIFO. When an escape character is detected, the internal escape timer starts to count. The software specifies a time-out value for the maximum allowable time between 2 successive escape characters (see the table below). The escape timer is programmable in intervals of 2 ms to a maximum interval of 8.192 seconds.

Table 16-8. Escape Timer Scaling

UTIM Register	Maximum Time Between Specified Escape Characters
0x000	2 ms
0x001	4 ms
0x002	6 ms
0x003	8 ms
0x004	10 ms
...	...
0F8	498 ms
0F9	500 ms
...	...
9C3	5 s
...	...
FFD	8.188 s
FFE	8.190 s
FFF	8.192 s
NOTE: To calculate the time interval: (UTIM_Value + 1) x 0.002 = Time_Interval Example: (09C3 + 1) x 0.002 = 5 s.	

The escape sequence detection feature is available for all the reference frequencies. Before using Escape Sequence Detection, the user must fill the ONEMS register. This 24-bit register must contain the value of the UART internal frequency divided by 1000. The internal frequency is obtained after the UART internal divider which is applied on *module_clock* clock.

Example I:

- If the input clock *module_clock* frequency is 66.5 MHz.
- And if the input clock *module_clock* is divided by 2 with the internal divider: UFCR[9:7] = 3'b100

$$ONEMS = \frac{66.5 \times 10^6}{2 \times 1000} = 33250 = 81E2h$$

Figure 16-15. Calculation of Frequency for ONEMS Register

Example II:

- If the input clock *module_clock* frequency is 66.5 MHz.
- And if the input clock *module_clock* is divided by 1with the internal divider: UFCR[9:7] = 3'b101

$$ONEMS = \frac{66.5 \times 10^6}{1000} = 66500 = 103C4h$$

Figure 16-16. Calculation of Frequency for ONEMS Register

The escape sequence detection interrupt is asserted when the escape sequence interrupt enable (ESCI) bit is set and an escape sequence is detected (ESCF set). Clear the ESCF bit by writing 1 to it. Writing 0 to the ESCF bit has no effect.

16.2.3 Binary Rate Multiplier (BRM)

The BRM sub-block receives *ref_clk* (*module_clock* clock after divider). From this clock, and with integer and non-integer division, BRM generates a 16x baud rate clock .

The UART transmitte will shift data out based on this 16x baud rate clock. The UART receiver will sample the serial data line based on this 16x baud rate clock. The input and output frequency ratio is programmed in the UART BRM Incremental Register (UBIR) and UART BRM MOD Register (UBMR). The output frequency is divided by the input frequency to produce this ratio. For integer division, set the UBIR = 0x000F and write the divisor to the UBMR register. All values written to these registers must be one less than the actual value to eliminate division by 0 (undefined), and to increase the maximum range of the registers.

Updating the BRM registers requires writing to both registers. The UBIR register must be written before writing to the UBMR register. If only one register is written to by the software, the BRM continues to use the previous values.

The following examples show how to determine what values are to be programmed into UBIR and UBMR for a given reference frequency and desired baud rate. The following equation can be used to help determine these values:

$$\text{BaudRate} = \frac{\text{Ref Freq}}{\left(16 \times \frac{\text{UBMR} + 1}{\text{UBIR} + 1} \right)}$$

Figure 16-17. Frequency and Baud Rate for UBIR and UBMR

With:

Reference Frequency (Hz): UART Reference Frequency (*module_clock* after RFDIV divider)

Baud Rate (bit/s): Desired baud rate.

Integer Division ÷ 21

Reference Frequency = 19.44 MHz

UBIR = 0x000F

UBMR = 0x0014

Baud Rate = 925.7 kbit/s

NOTE

Observe that each value written to the registers is one less than the actual value.

Non-Integer Division

```
Reference Frequency = 16 MHz
Desired Baud Rate = 920 Kbits/s
```

$$\frac{UBMR + 1}{UBIR + 1} = \frac{RefFreq}{16 \times BaudRate} = \frac{16 \times 10^6}{16 \times 920 \times 10^3} = 1.087$$

```
Ratio = 1.087 = 1087 / 1000
UBIR = 999 (decimal)= 0x3E7
UBMR = 1086 (decimal)= 0x43E
Non-Integer Division
Reference Frequency = 25 MHz
Desired Baud Rate = 920 kbit/s
Ratio = 1.69837 = 625 / 368
UBIR = 367 (decimal)= 0x16F
UBMR = 624 (decimal)= 0x270
```

Non-Integer Division

```
Reference Frequency: 30 MHz
Desired Baud Rate = 115.2 kbit/s
Ratio = 16.276043 = 65153 / 4003
UBIR = 4002 (decimal) = 0x0FA2
UBMR = 65152 (decimal) = 0xFE80
```

16.2.4 Infrared Interface

16.2.4.1 Generalities-Infrared

The Infrared interface is selected when IREN (UCR1[7]) is set to 1.

The Infrared Interface is compatible with IrDA Serial Infrared Physical Layer Specification. In this specification, a "zero" is represented by a positive pulse, and a "one" is represented by no pulse (line remains low).

In the UART:

In TX: For each "zero" to be transmitted, a narrow positive pulse which is 3/16 of a bit time is generated. For each "one" to be transmitted no pulse is generated (output is low). External circuitry has to be provided to drive an Infrared LED.

In RX: When receiving, a narrow negative pulse is expected for each "zero" transmitted while no pulse is expected for each "one" transmitted (input is high).

NOTE

Rx part of IR block expects to receive an inverted signal compared to IrDA specification. Circuitry external to the IC transforms the Infrared signal to an electrical signal.

The IR interface has an edge triggered interrupt (IRINT). This interrupt validates a zero bit being received. This interrupt is enabled by writing a "one" to ENIRI bit.

The behavior of Infrared Interface is determined by 3 bits INVT (UCR3[1]), INVR (UCR4[9]) and IRSC (UCR4[5]).

16.2.4.2 Inverted Transmission and Reception bits (INVT & INVR)

The values of INVT and INVR depend of the IrDA transceiver connected on the TXD_IR and RXD_IR pins of the UART. If this transceiver is not inverting on both paths Tx and Rx, a Zero is represented by a positive pulse and a One is represented by no pulse (line remains low). In this case, the bit INVT must be set to 0 and the bit INVR must be set to 1 (because Rx IR block expects an inverted signal).

On the contrary user must set INVT=1 and INVR=0 if both paths of the transceiver are inverting, that is, a Zero is represented as a negative pulse and a One is represented by no pulse (line remains high). The transceiver can also be inverting on only one path (Tx or Rx), in this case INVT and INVR must be together equal to 1 or to 0, depending on which path is inverted.

16.2.4.3 InfraRed Special Case (IRSC) Bit

The value to apply to IRSC bit is based on 2 parameters: the baud rate and the Minimum Pulse Duration (MPD) of the transceiver.

According to IrDA Standard Specification, for SIR (Serial IR) baud rates from 2.4 Kbit/s to 115.2 Kbit/s this nominal pulse duration is equal to 3/16 of a bit duration (at the selected baud rate). But, for all the baud rates a Minimum Pulse Duration is also specified. According to IrDA Standard, a Zero is represented by a light pulse, so the IrDA transceiver can't emit a light pulse shorter than the MPD. For SIR, the MPD is constant and equal to 1.41 µs.

But user must take into account the electrical MPD associated with the transceiver on the receiver path. Typically this value is 2.0 µs, but for some manufacturers MPD can go down to 1.0 µs.

In order to understand the meaning of IRSC bit, one must understand how the RX path works in IrDA mode.

When the UART is in IrDA mode, a Zero is not only detected by the state of the RXD_IR line, but also with the duration of the pulse. This pulse duration can be measured with 2 different clocks. In this case, clock is selected with the IRSC bit.

- If IRSC = 0, the clock used is the BRM clock.
- If IRSC = 1, the clock used is the UART internal clock (UART clock after the divider (RFDIV)).

In normal operation, IRSC=0. This means that at any time, the user must ensure that the frequency of BRM_clock is high enough to measure the pulse. The pulse must last at least 2 BRM clock cycles. If this condition is not fulfilled, IRSC must be set to 1.

Let's examine two examples, for a Minimum Pulse Duration equal to the MPD from the IrDA SIR specification (i.e., 1.41 µs).

1: Calculation of BRM Clock Period (Clock Period < 1.41 µs)

The user wants to receive IrDA data at 115.2 Kbit/s. The UBIR and UBMR registers are set in order to create the BRM_clock with a frequency of 16*baud rate = 16 * 115.2K = 1.843 MHz. But at the same time, in order to correctly detect the pulse, the user must be sure that 2* BRM_clock period is lower than 1.41 µs. Lets check:

BRM_clock period = 1/1843000 = 542 ns

So 2*BRM_clock period = 1.09 µs < 1.41 µs. It is fine.

2: Calculation of BRM Clock Period (Clock Period > 1.41 µs)

This time the user wants to receive at 19.2 Kbit/s. So, the BRM_clock is set to 16*19200 = 307.2 kHz. Let's check if 2* BRM_clock period < 1.41 µs:

1. BRM_clock period =1/307200 = 3.25 µs

   So 2*BRM_clock period = 6.50 µs >> 1.41 µs. It doesn't work.

   So, in this case, the BRM clock can't be used to measure the pulse duration and the user must select the UART internal clock by setting IRSC =1.

NOTE

Like for Escape character detection, when IR Special Case is enabled (IRSC=1), the UART must measure a duration. In order to do that, the user must fill the ONEMS register. Refer to Escape Sequence Detection.

16.2.4.4 IrDA interrupt

Serial infrared mode (SIR) uses an edge triggered interrupt flag IRINT (USR2[8]). When INVR =0, detection of a falling edge on the RXD pin asserts the IRINT bit. When INVR=1, detection of a rising edge on the RXD pin asserts the IRINT bit. When IRINT and ENIRI bits are both asserted, the *interrupt_uart* interrupt is asserted. Clear the IRINT bit by writing 1 to it. Writing 0 to the IRINT bit has no effect.

16.2.4.5 Conclusion about IrDA

Before using the UART in IrDA, the baud rate limit must be calculated. This baud rate limit will inform the user if IRSC bit has to be set or not.

Let's determine this limit:

As already described, if IRSC = 0, the following condition must always be fulfilled

$$2 \times BRMClockPeriod < MinPulseDuration$$

Figure 16-18. Calculation of Baud Rate

So,

$$BRMClockFrequency > \frac{2}{MPD}$$

So, knowing BRM_clock frequency = 16 * Baud Rate, we get:

$$BaudRate > \frac{1}{8 \times MinPulseDuration}$$

So, the user needs to set IRSC = 0 when:

- If Minimum Pulse Duration = 2.5 us and Baud Rate > 50 Kbit/s.
- If Minimum Pulse Duration = 2.0 us and Baud Rate > 62.5 Kbit/s.
- If Minimum Pulse Duration = 1.41 us and Baud Rate > 88.6 Kbit/s.

NOTE

For baud rates lower than the limit, IRSC must be set to1.

16.2.4.6 Programming IrDA Interface

16.2.4.6.1 High Speed

As an example, the following sequence can be used to program the IrDA interface in order to send and receive characters at 115.2 Kbit/s.

Assumptions:

- Input UART clock = 90 MHz
- Internal clock divider = 3 (divide Input UART clock by 3)
- Baud rate = 115.2 Kbit/s
- IrDA transceiver is not inverting on both channels: for Tx and Rx, a Zero is represented by a positive pulse, and a One is represented by no pulse (line stays low).
- Interrupt: Sent to Arm platform when 1 char is received into the Rx FIFO (RDR)

Registers values and Programming orders:

```
UCR1 = 0x0085
UCR1[7] = IREN = 1: Enable IR interface
UCR1[0] = UARTEN = 1: Enable UART
UTS = 0x0000
UFCR = 0x0981
TXTL[5:0] = 0x02: Default value
RFDIV[2:0] = 0x3: Divide Input UART clock by 3 (resulting internal clock  is 30 MHz)
RXTL[5:0] = 0x01: Default value
UBIR = 0x0202
UBMR = 0x20BE Baud rate = 115.2 kbit/s with internal clock = 30 MHz
UCR2 = 0x4027
UCR2[14] = IRTS = 1: Ignore level of RTS input signal
UCR2[5] = WS = 1: Characters are 8-bit length
UCR2[2] = TXEN = 1: Enable Rx path
UCR2 [1] = RXEN = 1: Enable Tx path
UCR2[0] = SRST_B = 1: No software reset
UCR3 = 0x0000

         UCR4 = 0x8201
CTSTL[5:0] = 0x20: Default value
UCR4[9] = INVR = 1: Inverted Infrared Reception (because IrDA transceiver is not inverting)
UCR4[1] = DREN = 1: To enable RDR interrupt (sent when one char is received)
```

The UART is ready to send a character as soon as there is a write into UTXD register. And an interrupt will be sent to Arm platform when a character is received.

16.2.4.6.2 Low Speed

This time, we keep the same assumptions but the speed is now 9.6 Kbit/s. So, this baud rate is below the limit (even with a Min. Pulse Duration of 2.5 us) and thus IRSC must be set to 1.

Assumptions:

- Input UART clock = 90 MHz
- Internal clock divider = 3 (divide Input UART clock by 3)
- Baud rate = 9.6 Kbit/s
- IrDA transceiver is not inverting on both channels: for Tx and Rx, a Zero is represented by a positive pulse, and a One is represented by no pulse (line stays low).
- Interrupt: Sent to Arm platform when 1 char is received into the Rx FIFO (RDR).

Registers values and Programming orders:

```
UCR1 = 0x0085
UCR1[7] = IREN = 1: Enable IR interface
UCR1[0] = UARTEN = 1: Enable UART
UFCR = 0x0981
UFCR[15:10] = TXTL[5:0] = 0x02: Default value
RFDIV[2:0] = 0x3: Divide Input UART clock by 3 (resulting internal clock  is 30 MHz)
UFCR[5:0] = RXTL[5:0] = 0x01: Default value
UBIR = 0x00FF
UBMR = 0xC354 Baud rate = 9.6 kbit/s with internal clock = 30 MHz
UCR2 = 0x4027
UCR2[14] = IRTS = 1: Ignore level of RTS input signal
UCR2[5] = WS = 1: Characters are 8-bit length
UCR2[2] = TXEN = 1: Enable Rx path
UCR2 [1] = RXEN = 1: Enable Tx path
UCR2[0] = SRST_B = 1: No software reset
UCR3 = 0x0000
UCR3[1] = INVT = 0: Positive pulse represents 0.
UCR4 = 0x8221
UCR4[15:10] = CTSTL[5:0] = 0x20: Default value
UCR4[9] = INVR = 1: Inverted Infrared Reception (because IrDA transceiver is not inverting)
UCR4[5] = IRSC = 1: Because data rate is below the limit and thus the UART internal clock is
used to measure the pulse duration.
UCR4[1] = DREN = 1: To enable RDR interrupt (sent when one char is received)
```

The UART is now ready to send a character as soon as there is a write into UTXD register. An interrupt will be sent to Arm platform when a character is received.

16.2.5 9-bit RS-485 Mode

16.2.5.1 Generalities

The UART provides a 9-bit mode to facilitate multidrop (RS-485) network communication. To enable this mode, set MDEN bit in the UMCR register to 1. When 9-bit RS-485 mode is enabled, UART transmitter can transmit the ninth bit (9^{th} bit) set by TXB8, and UART receiver can differentiate between data frames (9^{th} bit = 0) and address frames (9^{th} bit = 1).

The CTS_B pin can be used to control RS-485 output driver outside the chip.

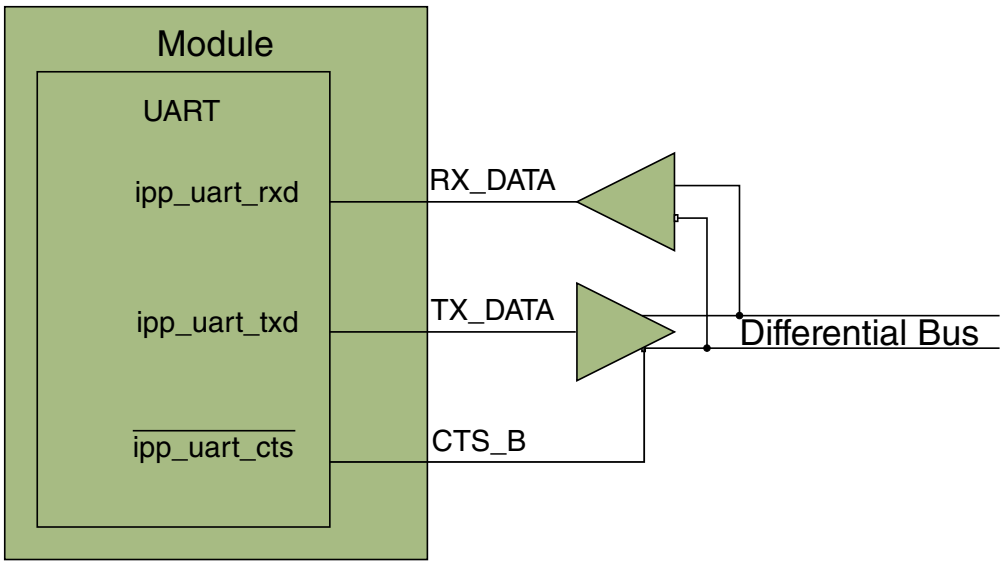


Figure 16-19. RS-485 driver connection (UART in DCE mode)

16.2.5.2 Transmit 9-bit RS-485 frames

To transmit 9-bit RS-485 frames, user need to enable parity (PREN=1) to enable trasmitting the ninth data bit, set 8-bit data word size (WS=1), and write TXB8 (UMCR[2]) as the 9th bit (bit [8]) to be transmitted (write '0' to TXB8 to transmit a data frame, write '1' to transmit a address frame). The other data bit [7:0] is written to TxFIFO by writing to the UTXD same as normal RS-232 operation.

16.2.5.3 Receive 9-bit RS-485 frames

To receive 9-bit RS-485 frames, user need to enable parity (PREN=1) to enable receiving the ninth data bit, set 8-bit data word size (WS=1). The receiver will save the 9-bit data to RxFIFO, and user should read the 9th databit (bit [8]) by reading the PRERR (URXD[10]) bit, and read data bit [7:0] by reading the RX_DATA (URXD[7:0]).

There are two slave address detect modes, normal detect mode and automatic detect mode, and can be selected by SLAM (UMCR[1]).

16.2.5.3.1 RS-485 Slave Address Normal Detect Mode

To enable Normal Detect mode, clear SLAM (UMCR[1] to 0). The receiver ignores all data frames (9th bit = 0) until an address frame is received (9th bit = 1). At that time, the slave address detected (SAD = USR1[3]) bit is asserted and the *interrupt_uart* interrupt is

generated (if SADEN = UMCR[3] = 1). The address byte and sebsequent bytes are all put into RxFIFO along with their 9th bit. The UART will also generate DMA request *dma_req_rx* when the RxFIFO reaches the selected threshold (controlled by RXTL) if receive ready DMA (RXDMAEN = UCR1[8]) request is enabled.

User should read the 9th databit (bit [8]) by reading the PRERR (URXD[10]) bit, and read data bit [7:0] by reading the RX_DATA (URXD[7:0]).

In this mode, once the UART has detected a 9th bit is equal to '1', it will always save the subsequent frames to RxFIFO. So the software must decide whether the address and data in RxFIFO are needed or not.

16.2.5.3.2 RS-485 Slave Address Automatic Detect Mode

To enable Automatic Detect Mode, set SLAM (UMCR[1]) to 1. The receiver tries to detect an address byte (frame 9th bit = 1) that matches the programmed SLADDR (UMCR[15:8]) character. If the received byte is a data or an address byte that does not match the programmed SLADDR character, the receiver will discard these data.

Once the UART receives a matching address byte, it will assert the slave address detected (SAD = USR1[3]) bit and the *interrupt_uart* interrupt will be generated (if SADEN = UMCR[3] = 1). The address byte and sebsequent bytes are all put into RxFIFO along with their 9th bit. If receive ready DMA(RXDMAEN = UCR1[8]) request is enabled, the UART will also generate DMA request *dma_req_rx* when the RxFIFO reaches the selected threshold (controlled by RXTL).

If another address byte is received and this address byte does not match SLADDR character, the receiver will discard the address byte and subsequent data byte. If the address byte again matches SLADDR character, the receiver will put this address byte and subsequent data byte in the RxFIFO along with their 9th bit.

User should read the 9th databit (bit [8]) by reading the PRERR (URXD[10]) bit, and read data bit [7:0] by reading the RX_DATA (URXD[7:0]).

See Initialization for 9-bit RS-485 programming guide.

16.2.6 Low Power Modes

These modes are controlled by the signals *doze_req* and *stop_req*. The control/status/data registers won't change when getting in/out of low power modes.

Table 16-9. UART Low Power State Operation

	Normal State (*doze_req* = 1'b0 & *stop_req* = 1'b0)	Doze State (*doze_req* = 1'b1)		Stop State (*stop_req* = 1'b1)
		DOZE bit = 0	DOZE bit = 1	
UART-Clock	ON	ON	ON	OFF
UART Serial / IrDA	ON	ON	OFF	OFF

16.2.6.1 UART Operation in System Doze Mode

While in Doze State (when *doze_req* input pin is set to 1'b1), the UART behavior depends on the DOZE (UCR1[1]) control bit.

While the DOZE bit is negated, the UART serial interface is enabled. While the system is in the Doze State, and the DOZE bit is asserted, the UART is disabled. If the Doze State is entered with the DOZE bit asserted while the UART serial interface was receiving or transmitting data, it will complete the receive/transmit of the current character and signal to the far-end transmitter/receiver to stop sending/receiving.

16.2.6.2 UART Operation in System Stop Mode

The internal baud rate clocks of the transmitter and receiver are gated off if the *stop_req* signal to UART is asserted. Even though the clocks at the input of the UART continue to run during system Stop mode, the UART will not do any transmission or reception.

The following UART interrupts wake the Arm platform processor from STOP mode:

- RTS (RTSD)
- IrDA Asynchronous WAKE (AIRINT)
- Asynchronous WAKE (AWAKE)
- RI (RIDELT in DTE mode only)
- DCD (DCDDELT in DTE mode only)
- DTR (DTRD in DCE mode only)
- DSR (DTRD in DTE mode only)

When an asynchronous WAKE (awake) interrupt exits the Arm platform from STOP mode, make sure that a dummy character is sent first because the first character may not be received correctly.

16.2.6.3 Power Saving Method in UART

The RXEN (UCR2[1]), TXEN (UCR2[2]) and UARTEN (UCR1[0]) bits are set by the user and provide software control of low-power modes.

Setting the UARTEN (UCR1[0]) bit to 0 shuts off the receiver and transmitter logic and the associated clocks.

If the UART is used only in transmit mode, UARTEN and TXEN must be set to 1. If the UART is used only in receive mode, UARTEN and RXEN must be set to 1. Setting TXEN or RXEN to 0 allows to save a lot of power.

16.2.7 UART Operation in System Debug State

The bit UTS [11] controls whether the UART will respond to the input signal *debug_req*, or whether it will continue to run as normal.

If the UART is programmed to respond to *debug_req*:

1. The UART will halt all operations upon detecting the *debug_req* input.
2. A transfer in progress, either to/from a core (using the IP Bus interface) or to/from an external device, will be completed before halting. This means a single byte/word transfer, not an entire FIFO. Reception of any further data from an external device will be disabled.
3. Internal registers will continue to be writable and readable using the IP Bus interface. A read will leave the contents unaffected.
4. The RX FIFO is affected in debug mode in the following way:
   - All writes into the RX FIFO are prevented.
   - The bit RXDBG (UTS[9]) is used to select the readability of the RX FIFO during debug mode:

     RXDBG = 0: hold the read pointer at the location it had upon entering debug mode, and URXD register returns only the data value at that location, no matter how many reads attempted.

RXDBG = 1, selectable at any time: Allow to read the characters received in Rx FIFO. It will not be possible to re-read previously read locations, nor will it be possible to readjust the read pointer to the value it had prior to entering debug mode.

16.2.8 Reset

This section describes how to reset the block and explains special requirements related to reset.

16.2.8.1 Hardware reset

All of registers, FIFOs, state machines and sequential elements can be reset to their initial values by hardware reset or power on reset.

16.2.8.2 Software reset

The status registers USR1 and USR2, BRM registers UBIR and UBMR, TxFIFO and RxFIFO, and transmitter and receiver state machines can be reset by software reset. Internal logic will keep the software reset asserted for about 4 *module_clock* cycles.

Programmer can follow the following software reset sequence:

1. Clear the SRST_B bit (UCR2[0])
2. Wait for software reset complete: poll SOFTRST bit (UTS[0]) until it is 0.
3. Re-program baud rate registers: Re-write UBIR and UBMR.

16.2.9 Transfer Error

The UART can generate a transfer error on the peripheral bus in the following cases:

- Core is writing into a read-only register.
- Core is accessing (read or write) an unused location within the assigned address space reserved to UART.
- Core is writing into UTXD register with transmit interface disabled (TXEN=0 or UARTEN=0)
- Core is reading URXD register with receive interface disabled (RXEN=0 or UARTEN=0)

16.2.10 Functional Timing

This section includes timing diagrams for functional signaling.

16.2.10.1 IrDA Mode

According to IrDA specification, the low speed (115.2Kbit/s and below) IR frame format is compatible with UART frame.

In this figure, an example data 0x65 is used.

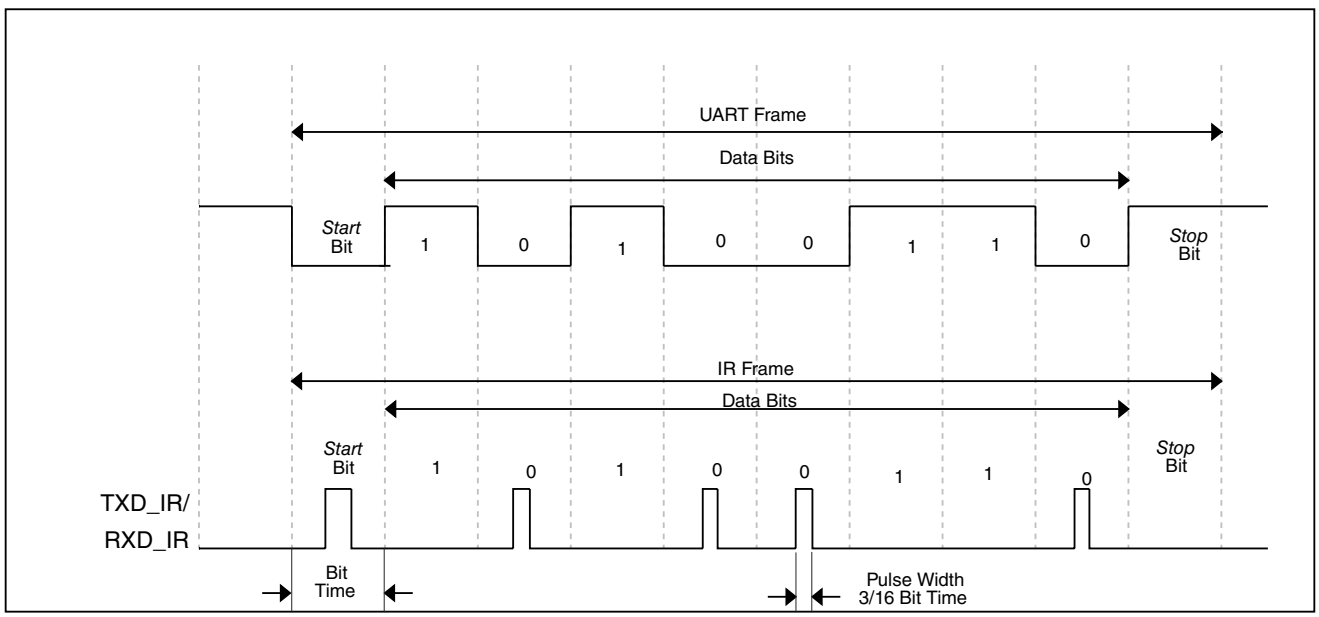


Figure 16-20. Timing diagram of Low Speed IR (<=115.2 Kbit/s) Data Line

16.2.11 Initialization

16.2.11.1 Programming the UART in RS-232 mode

As an example, the following sequence can be used to program the UART in order to send and receive characters in RS-232 mode.

Assumptions:

- Input uart clock = 100 MHz
- Baud rate = 921.6 Kbps

- Data bits = 8 bits
- Parity = Even
- Stop bits = 1 bit
- Flow control = Hardware

Main program:

1. UCR1 = 0x0001

   Enable the UART.

2. UCR2 = 0x2127

   Set hardware flow control, data format and enable transmitter and receiver.

3. UCR3 = 0x0704

   Set UCR3[RXDMUXSEL] = 1.

4. UCR4 = 0x7C00

   Set CTS trigger level to 31,

5. UFCR = 0x089E

   Set internal clock divider = 5 (divide input uart clock by 5). So the reference clock is 100 MHz/5 = 20 MHz.

   Set TXTL = 2 and RXTL = 30.

6. UBIR = 0x08FF
7. UBMR = 0x0C34

   In the above two steps, set baud rate to 921.6Kbps based on the 20MHz reference clock.

8. UCR1 = 0x2201

   Enable the TRDY and RRDY interrupts.

9. UMCR = 0x0000

   UMCR stay at default value 0x0000

Interrupt service routine for the transmitter:

- Write characters into UTXD

The TRDY interrupt will be automatically de-asserted when the data level of the TxFIFO exceeds the TXTL=2. Note: For the first time the interrupt may be de-asserted after 4 characters are written into the TxFIFO because of the shift register.

Interrupt service routine for the receiver:

- Read characters from URXD

The RRDY interrupt will be automatically de-asserted when the data level of the RxFIFO is below the RXTL=30.

16.2.11.2 Programming the UART in 9-bit RS-485 mode

As an example, the following sequence can be used to program the UART in order to send and receive frames in RS-485 mode.

Assumptions:

- Input uart clock = 100 MHz
- Baud rate = 5 Mbps

Main program:

1. UCR1 = 0x0001

   Enable the UART.

2. UCR2 = 0x4127

   Set software flow control ($\overline{\text{CTS}}$ pin is controlled by UCR2[12]), enable parity(enable 9^{th} bit rxd/txd), 8-bit word size , and enable transmitter and receiver.

3. UCR4 = 0x7C00

   Set CTS trigger level to 31,

4. UFCR = 0x0A9E

   Set RFDIV = 5 (divide input uart clock by 1), so the reference clock is 100 MHz. Set UART in DCE mode (RS-485 driver connection outside the chip is the same as Figure 16-19)

   Set TXTL = 2 and RXTL = 30.

5. UBIR = 0x0003
6. UBMR = 0x0004

   In the above two steps, set baud rate to 5 Mbps based on the 100 MHz reference clock.

7. UCR1 = 0x2001 when UART as a master ,

   or UCR1 = 0x0201 (or 0x0101) when UART as a slave.

Enable TRDY interrupt when UART as a master, enable RRDY interrupt or DMA request when UART as a slave.

8. UMCR = 0xA50B

   Enable 9-bit RS-485 mode, enable SAD interrupt, set automatic slave address detect mode, set slave address is 0xA5.

Interrupt service routine for the transmitter:

- Transmit data: write its ninth bit (bit[8]) to UMCR[2], write its bit [7:0] into UTXD[7:0]

The TRDY interrupt will be automatically de-asserted when the data level of the TxFIFO exceeds the TXTL=2.

Note: For the first time the interrupt may be de-asserted after 4 characters are written into the TxFIFO because of the shift register.

Interrupt service routine for the receiver:

- Receive data: read its ninth bit (bit[8]) from URXD[10] , read its bit [7:0] from URXD[7:0].

Note: in RS-485 mode, URXD[10] bit is not the parity error, instead it holds the ninth bit (bit[8]) of the received data.

The SAD interrupt can not de-assert automatically, it needs MCU write 1 to USR1[3] to clear it . The RRDY interrupt or DMA request will be automatically de-asserted when the data level of the RxFIFO is below the RXTL=30.

16.2.12 References

- EIA/TIA-232-F Interface Standard

  http://www.eia.org, http://www.tiaonline.org/standards

- IrDA Standard

  http://www.irda.org

16.2.13 UART Memory Map/Register Definition

UART supports 8-bit, 16-bit and 32-bit accesses to 32-bit memory-mapped addresses. Any access to unmapped memory location will yield a transfer error.

All registers except the ONEMS described in this section are 16-bit registers. The ONEMS register is a 24-bit register.

- For 32-bit write accesses, the upper two bytes will not be taken into account.
- For 32-bit read accesses the upper two bytes will return 0.

The ONEMS register is expanded from 16 bits to 24 bits in order to support the high frequency of the BRM internal clock *ref_clk* (*module_clock* after divider). The ONEMS register can be accessed as 8 bits, 16 bits or 32 bits.

- For 32-bit write accesses, the most significant byte of the ONEMS will be discarded.
- For 32-bit read accesses, the most significant byte of the ONEMS will be read as 0.

UART memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3086_0000	UART Receiver Register (UART1_URXD)	32	R	0000_0000h	16.2.13.1/ 5323
3086_0040	UART Transmitter Register (UART1_UTXD)	32	W	0000_0000h	16.2.13.2/ 5325
3086_0080	UART Control Register 1 (UART1_UCR1)	32	R/W	0000_0000h	16.2.13.3/ 5326
3086_0084	UART Control Register 2 (UART1_UCR2)	32	R/W	0000_0001h	16.2.13.4/ 5328
3086_0088	UART Control Register 3 (UART1_UCR3)	32	R/W	0000_0700h	16.2.13.5/ 5331
3086_008C	UART Control Register 4 (UART1_UCR4)	32	R/W	0000_8000h	16.2.13.6/ 5333
3086_0090	UART FIFO Control Register (UART1_UFCR)	32	R/W	0000_0801h	16.2.13.7/ 5335
3086_0094	UART Status Register 1 (UART1_USR1)	32	R/W	0000_2040h	16.2.13.8/ 5337
3086_0098	UART Status Register 2 (UART1_USR2)	32	R/W	0000_4028h	16.2.13.9/ 5340
3086_009C	UART Escape Character Register (UART1_UESC)	32	R/W	0000_002Bh	16.2.13.10/ 5342
3086_00A0	UART Escape Timer Register (UART1_UTIM)	32	R/W	0000_0000h	16.2.13.11/ 5342
3086_00A4	UART BRM Incremental Register (UART1_UBIR)	32	R/W	0000_0000h	16.2.13.12/ 5343
3086_00A8	UART BRM Modulator Register (UART1_UBMR)	32	R/W	0000_0000h	16.2.13.13/ 5343
3086_00AC	UART Baud Rate Count Register (UART1_UBRC)	32	R	0000_0004h	16.2.13.14/ 5344

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3086_00B0	UART One Millisecond Register (UART1_ONEMS)	32	R/W	0000_0000h	16.2.13.15/ 5345
3086_00B4	UART Test Register (UART1_UTS)	32	R/W	0000_0060h	16.2.13.16/ 5346
3086_00B8	UART RS-485 Mode Control Register (UART1_UMCR)	32	R/W	0000_0000h	16.2.13.17/ 5347
3088_0000	UART Receiver Register (UART3_URXD)	32	R	0000_0000h	16.2.13.1/ 5323
3088_0040	UART Transmitter Register (UART3_UTXD)	32	W	0000_0000h	16.2.13.2/ 5325
3088_0080	UART Control Register 1 (UART3_UCR1)	32	R/W	0000_0000h	16.2.13.3/ 5326
3088_0084	UART Control Register 2 (UART3_UCR2)	32	R/W	0000_0001h	16.2.13.4/ 5328
3088_0088	UART Control Register 3 (UART3_UCR3)	32	R/W	0000_0700h	16.2.13.5/ 5331
3088_008C	UART Control Register 4 (UART3_UCR4)	32	R/W	0000_8000h	16.2.13.6/ 5333
3088_0090	UART FIFO Control Register (UART3_UFCR)	32	R/W	0000_0801h	16.2.13.7/ 5335
3088_0094	UART Status Register 1 (UART3_USR1)	32	R/W	0000_2040h	16.2.13.8/ 5337
3088_0098	UART Status Register 2 (UART3_USR2)	32	R/W	0000_4028h	16.2.13.9/ 5340
3088_009C	UART Escape Character Register (UART3_UESC)	32	R/W	0000_002Bh	16.2.13.10/ 5342
3088_00A0	UART Escape Timer Register (UART3_UTIM)	32	R/W	0000_0000h	16.2.13.11/ 5342
3088_00A4	UART BRM Incremental Register (UART3_UBIR)	32	R/W	0000_0000h	16.2.13.12/ 5343
3088_00A8	UART BRM Modulator Register (UART3_UBMR)	32	R/W	0000_0000h	16.2.13.13/ 5343
3088_00AC	UART Baud Rate Count Register (UART3_UBRC)	32	R	0000_0004h	16.2.13.14/ 5344
3088_00B0	UART One Millisecond Register (UART3_ONEMS)	32	R/W	0000_0000h	16.2.13.15/ 5345
3088_00B4	UART Test Register (UART3_UTS)	32	R/W	0000_0060h	16.2.13.16/ 5346
3088_00B8	UART RS-485 Mode Control Register (UART3_UMCR)	32	R/W	0000_0000h	16.2.13.17/ 5347
3089_0000	UART Receiver Register (UART2_URXD)	32	R	0000_0000h	16.2.13.1/ 5323
3089_0040	UART Transmitter Register (UART2_UTXD)	32	W	0000_0000h	16.2.13.2/ 5325

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3089_0080	UART Control Register 1 (UART2_UCR1)	32	R/W	0000_0000h	16.2.13.3/ 5326
3089_0084	UART Control Register 2 (UART2_UCR2)	32	R/W	0000_0001h	16.2.13.4/ 5328
3089_0088	UART Control Register 3 (UART2_UCR3)	32	R/W	0000_0700h	16.2.13.5/ 5331
3089_008C	UART Control Register 4 (UART2_UCR4)	32	R/W	0000_8000h	16.2.13.6/ 5333
3089_0090	UART FIFO Control Register (UART2_UFCR)	32	R/W	0000_0801h	16.2.13.7/ 5335
3089_0094	UART Status Register 1 (UART2_USR1)	32	R/W	0000_2040h	16.2.13.8/ 5337
3089_0098	UART Status Register 2 (UART2_USR2)	32	R/W	0000_4028h	16.2.13.9/ 5340
3089_009C	UART Escape Character Register (UART2_UESC)	32	R/W	0000_002Bh	16.2.13.10/ 5342
3089_00A0	UART Escape Timer Register (UART2_UTIM)	32	R/W	0000_0000h	16.2.13.11/ 5342
3089_00A4	UART BRM Incremental Register (UART2_UBIR)	32	R/W	0000_0000h	16.2.13.12/ 5343
3089_00A8	UART BRM Modulator Register (UART2_UBMR)	32	R/W	0000_0000h	16.2.13.13/ 5343
3089_00AC	UART Baud Rate Count Register (UART2_UBRC)	32	R	0000_0004h	16.2.13.14/ 5344
3089_00B0	UART One Millisecond Register (UART2_ONEMS)	32	R/W	0000_0000h	16.2.13.15/ 5345
3089_00B4	UART Test Register (UART2_UTS)	32	R/W	0000_0060h	16.2.13.16/ 5346
3089_00B8	UART RS-485 Mode Control Register (UART2_UMCR)	32	R/W	0000_0000h	16.2.13.17/ 5347
30A6_0000	UART Receiver Register (UART4_URXD)	32	R	0000_0000h	16.2.13.1/ 5323
30A6_0040	UART Transmitter Register (UART4_UTXD)	32	W	0000_0000h	16.2.13.2/ 5325
30A6_0080	UART Control Register 1 (UART4_UCR1)	32	R/W	0000_0000h	16.2.13.3/ 5326
30A6_0084	UART Control Register 2 (UART4_UCR2)	32	R/W	0000_0001h	16.2.13.4/ 5328
30A6_0088	UART Control Register 3 (UART4_UCR3)	32	R/W	0000_0700h	16.2.13.5/ 5331
30A6_008C	UART Control Register 4 (UART4_UCR4)	32	R/W	0000_8000h	16.2.13.6/ 5333
30A6_0090	UART FIFO Control Register (UART4_UFCR)	32	R/W	0000_0801h	16.2.13.7/ 5335

Table continues on the next page...

UART memory map (continued)

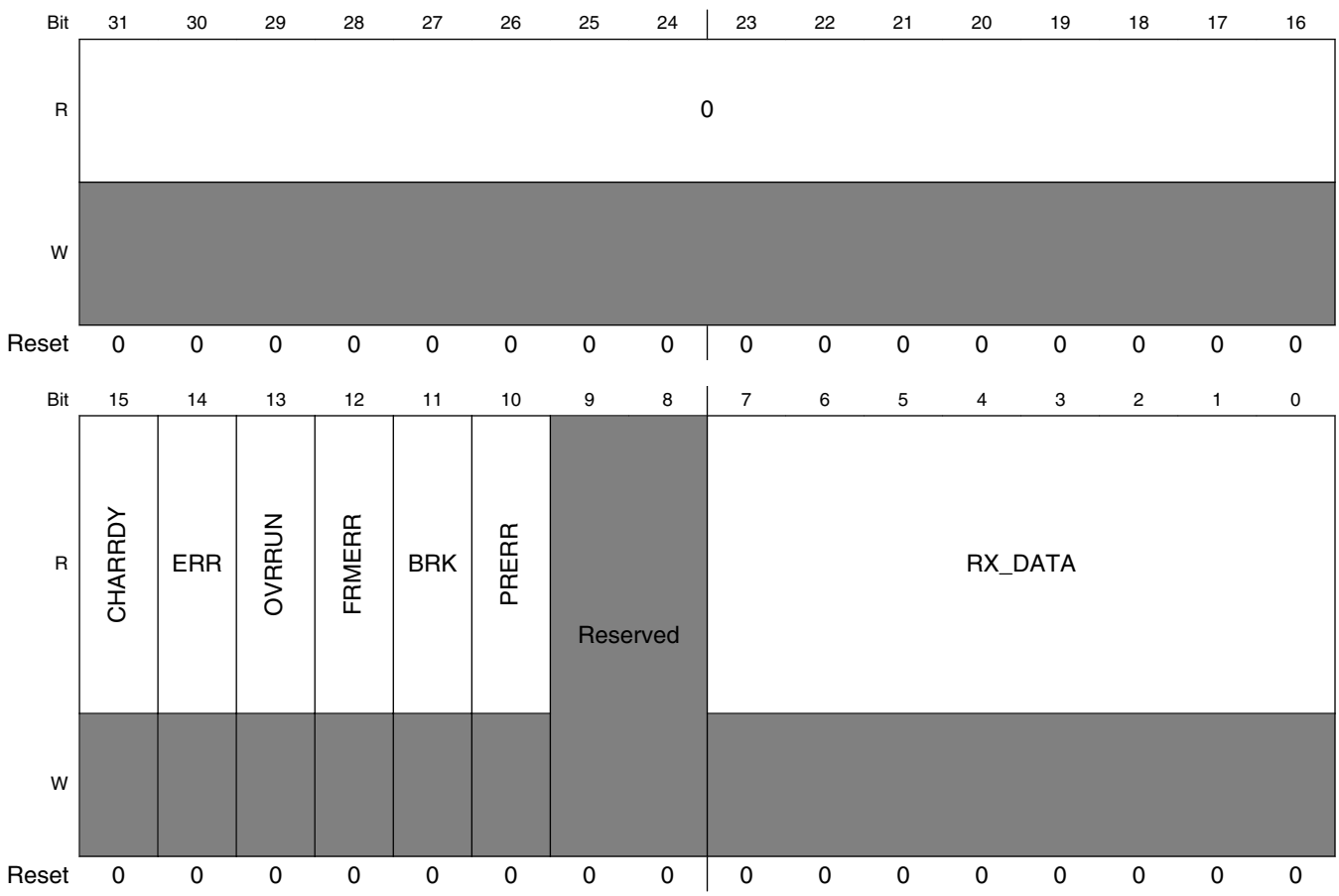
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30A6_0094	UART Status Register 1 (UART4_USR1)	32	R/W	0000_2040h	16.2.13.8/ 5337
30A6_0098	UART Status Register 2 (UART4_USR2)	32	R/W	0000_4028h	16.2.13.9/ 5340
30A6_009C	UART Escape Character Register (UART4_UESC)	32	R/W	0000_002Bh	16.2.13.10/ 5342
30A6_00A0	UART Escape Timer Register (UART4_UTIM)	32	R/W	0000_0000h	16.2.13.11/ 5342
30A6_00A4	UART BRM Incremental Register (UART4_UBIR)	32	R/W	0000_0000h	16.2.13.12/ 5343
30A6_00A8	UART BRM Modulator Register (UART4_UBMR)	32	R/W	0000_0000h	16.2.13.13/ 5343
30A6_00AC	UART Baud Rate Count Register (UART4_UBRC)	32	R	0000_0004h	16.2.13.14/ 5344
30A6_00B0	UART One Millisecond Register (UART4_ONEMS)	32	R/W	0000_0000h	16.2.13.15/ 5345
30A6_00B4	UART Test Register (UART4_UTS)	32	R/W	0000_0060h	16.2.13.16/ 5346
30A6_00B8	UART RS-485 Mode Control Register (UART4_UMCR)	32	R/W	0000_0000h	16.2.13.17/ 5347

16.2.13.1 UART Receiver Register (UARTx_URXD)

NOTE

The UART will yield a transfer error on the peripheral bus when core is reading URXD register with receive interface disabled (RXEN=0 or UARTEN=0).

Address: Base address + 0h offset



UARTx_URXD field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15 CHARRDY	Character Ready. This read-only bit indicates an invalid read when the FIFO becomes empty and software tries to read the same old data. This bit should not be used for polling for data written to the RX FIFO. 0 Character in RX_DATA field and associated flags are invalid. 1 Character in RX_DATA field and associated flags valid and ready for reading.

Table continues on the next page...

UARTx_URXD field descriptions (continued)

Field	Description
14 ERR	**Error Detect.** Indicates whether the character present in the RX_DATA field has an error (OVRRUN, FRMERR, BRK or PRERR) status. The ERR bit is updated and valid for each received character. 0 No error status was detected 1 An error status was detected
13 OVRRUN	**Receiver Overrun.** This read-only bit, when HIGH, indicates that the corresponding character was stored in the last position (32nd) of the Rx FIFO. Even if a 33rd character has not been detected, this bit will be set to '1' for the 32nd character. 0 No RxFIFO overrun was detected 1 A RxFIFO overrun was detected
12 FRMERR	**Frame Error.** Indicates whether the current character had a framing error (a missing stop bit) and is possibly corrupted. FRMERR is updated for each character read from the RxFIFO. 0 The current character has no framing error 1 The current character has a framing error
11 BRK	**BREAK Detect.** Indicates whether the current character was detected as a BREAK character. The data bits and the stop bit are all 0. The FRMERR bit is set when BRK is set. When odd parity is selected, PRERR is also set when BRK is set. BRK is valid for each character read from the RxFIFO. 0 The current character is not a BREAK character 1 The current character is a BREAK character
10 PRERR	**In RS-485 mode, it holds the ninth data bit (bit [8]) of received 9-bit RS-485 data** **In RS232/IrDA mode, it is the Parity Error flag**. Indicates whether the current character was detected with a parity error and is possibly corrupted. PRERR is updated for each character read from the RxFIFO. When parity is disabled, PRERR always reads as 0. 0 = No parity error was detected for data in the RX_DATA field 1 = A parity error was detected for data in the RX_DATA field
9–8 -	This field is reserved. **Reserved**
RX_DATA	**Received Data**. Holds the received character. In 7-bit mode, the most significant bit (MSB) is forced to 0. In 8-bit mode, all bits are active.

16.2.13.2 UART Transmitter Register (UARTx_UTXD)

NOTE

The UART will yield a transfer error on the peripheral bus when core is writing into UART_URXD register with transmit interface disabled (TXEN=0 or UARTEN=0).

Memory space between UART_URXD and UART_UTXD registers is reserved. Any read or write access to this space will be considered as an invalid access and yield a transfer error.

Address: Base address + 40h offset

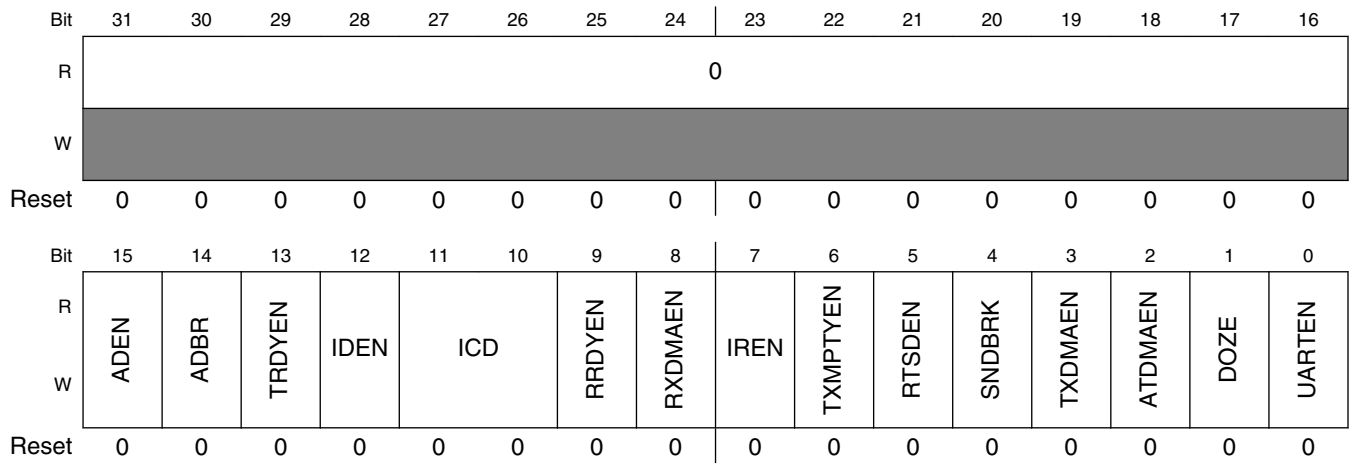
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																0								TX_DATA							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

UARTx_UTXD field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–8 Reserved	This read-only field is reserved and always has the value 0.
TX_DATA	**Transmit Data**. Holds the parallel transmit data inputs. In 7-bit mode, D7 is ignored. In 8-bit mode, all bits are used. Data is transmitted least significant bit (LSB) first. A new character is transmitted when the TX_DATA field is written. The TX_DATA field must be written only when the TRDY bit is high to ensure that corrupted data is not sent.

16.2.13.3 UART Control Register 1 (UARTx_UCR1)

Address: Base address + 80h offset



UARTx_UCR1 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15 ADEN	**Automatic Baud Rate Detection Interrupt Enable**. Enables/Disables the automatic baud rate detect complete (ADET) bit to generate an interrupt (*interrupt_uart* = 0). 0 Disable the automatic baud rate detection interrupt 1 Enable the automatic baud rate detection interrupt
14 ADBR	**Automatic Detection of Baud Rate**. Enables/Disables automatic baud rate detection. When the ADBR bit is set and the ADET bit is cleared, the receiver detects the incoming baud rate automatically. The ADET flag is set when the receiver verifies that the incoming baud rate is detected properly by detecting an ASCII character "A" or "a" (0x41 or 0x61). 0 Disable automatic detection of baud rate 1 Enable automatic detection of baud rate
13 TRDYEN	**Transmitter Ready Interrupt Enable**. Enables/Disables the transmitter Ready Interrupt (TRDY) when the transmitter has one or more slots available in the TxFIFO. The fill level in the TXFIFO at which an interrupt is generated is controlled by TxTL bits. When TRDYEN is negated, the transmitter ready interrupt is disabled. **NOTE:** An interrupt will be issued as long as TRDYEN and TRDY are high even if the transmitter is not enabled. In general, user should enable the transmitter before enabling the TRDY interrupt. 0 Disable the transmitter ready interrupt 1 Enable the transmitter ready interrupt
12 IDEN	**Idle Condition Detected Interrupt Enable**. Enables/Disables the IDLE bit to generate an interrupt (*interrupt_uart* = 0). 0 Disable the IDLE interrupt 1 Enable the IDLE interrupt

Table continues on the next page...

UARTx_UCR1 field descriptions (continued)

Field	Description
11–10 ICD	**Idle Condition Detect**. Controls the number of frames RXD is allowed to be idle before an idle condition is reported. 00 Idle for more than 4 frames 01 Idle for more than 8 frames 10 Idle for more than 16 frames 11 Idle for more than 32 frames
9 RRDYEN	**Receiver Ready Interrupt Enable**. Enables/Disables the RRDY interrupt when the RxFIFO contains data. The fill level in the RxFIFO at which an interrupt is generated is controlled by the RXTL bits. When RRDYEN is negated, the receiver ready interrupt is disabled. 0 Disables the RRDY interrupt 1 Enables the RRDY interrupt
8 RXDMAEN	**Receive Ready DMA Enable**. Enables/Disables the receive DMA request *dma_req_rx* when the receiver has data in the RxFIFO. The fill level in the RxFIFO at which a DMA request is generated is controlled by the RXTL bits. When negated, the receive DMA request is disabled. 0 Disable DMA request 1 Enable DMA request
7 IREN	**Infrared Interface Enable**. Enables/Disables the IR interface. See the IR interface description in Infrared Interface, for more information. Note: MDEN(UMCR[0]) must be cleared to 0 when using IrDA interface. See Table 16-2 0 Disable the IR interface 1 Enable the IR interface
6 TXMPTYEN	**Transmitter Empty Interrupt Enable**. Enables/Disables the transmitter FIFO empty (TXFE) interrupt. *interrupt_uart*. When negated, the TXFE interrupt is disabled. **NOTE:** An interrupt will be issued as long as TXMPTYEN and TXFE are high even if the transmitter is not enabled. In general, user should enable the transmitter before enabling the TXFE interrupt. 0 Disable the transmitter FIFO empty interrupt 1 Enable the transmitter FIFO empty interrupt
5 RTSDEN	**RTS Delta Interrupt Enable**. Enables/Disables the RTSD interrupt. The current status of the RTS_B pin is read in the RTSS bit. 0 Disable RTSD interrupt 1 Enable RTSD interrupt
4 SNDBRK	**Send BREAK**. Forces the transmitter to send a BREAK character. The transmitter finishes sending the character in progress (if any) and sends BREAK characters until SNDBRK is reset. Because the transmitter samples SNDBRK after every bit is transmitted, it is important that SNDBRK is asserted high for a sufficient period of time to generate a valid BREAK. After the BREAK transmission completes, the UART transmits 2 mark bits. The user can continue to fill the TxFIFO and any characters remaining are transmitted when the BREAK is terminated. 0 Do not send a BREAK character 1 Send a BREAK character (continuous 0s)
3 TXDMAEN	**Transmitter Ready DMA Enable**. Enables/Disables the transmit DMA request *dma_req_tx* when the transmitter has one or more slots available in the TxFIFO. The fill level in the TxFIFO that generates the *dma_req_tx* is controlled by the TXTL bits.

Table continues on the next page...

UARTx_UCR1 field descriptions (continued)

Field	Description
	NOTE: A DMA request will be issued as long as TXDMAEN and TRDY are high even if the transmitter is not enabled. In general, user should enable the transmitter before enabling the transmit DMA request. 0 Disable transmit DMA request 1 Enable transmit DMA request
2 ATDMAEN	**Aging DMA Timer Enable**. Enables/Disables the receive DMA request *dma_req_rx* for the aging timer interrupt (triggered with AGTIM flag in USR1[8]). 0 Disable AGTIM DMA request 1 Enable AGTIM DMA request
1 DOZE	**DOZE**. Determines the UART enable condition in the DOZE state. When *doze_req* input pin is at '1', (the Arm Platform executes a doze instruction and the system is placed in the Doze State), the DOZE bit affects operation of the UART. While in the Doze State, if this bit is asserted, the UART is disabled. See the description in Low Power Modes. 0 The UART is enabled when in DOZE state 1 The UART is disabled when in DOZE state
0 UARTEN	**UART Enable**. Enables/Disables the UART. If UARTEN is negated in the middle of a transmission, the transmitter stops and pulls the TXD line to a logic 1. UARTEN must be set to 1 before any access to UTXD and URXD registers, otherwise a transfer error is returned. This bit can be set to 1 along with other bits in this register. There is no restriction to the sequence of programing this bit and other control registers. 0 Disable the UART 1 Enable the UART

16.2.13.4 UART Control Register 2 (UARTx_UCR2)

Address: Base address + 84h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ESCI	IRTS	CTSC	CTS	ESCEN	RTEC		PREN	PROE	STPB	WS	RTSEN	ATEN	TXEN	RXEN	SRST
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

UARTx_UCR2 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15 ESCI	**Escape Sequence Interrupt Enable**. Enables/Disables the ESCF bit to generate an interrupt. 0 Disable the escape sequence interrupt 1 Enable the escape sequence interrupt
14 IRTS	**Ignore RTS Pin**. Forces the RTS input signal presented to the transmitter to always be asserted (set to low), effectively ignoring the external pin. When in this mode, the RTS pin serves as a general purpose input. 0 Transmit only when the RTS pin is asserted 1 Ignore the RTS pin
13 CTSC	CTS **Pin Control**. Controls the operation of the CTS_B module output. When CTSC is asserted, the CTS_B module output is controlled by the receiver. When the RxFIFO is filled to the level of the programmed trigger level and the start bit of the overflowing character (TRIGGER LEVEL + 1) is validated, the CTS_B module output is negated to indicate to the far-end transmitter to stop transmitting. When the trigger level is programmed for less than 32, the receiver continues to receive data until the RxFIFO is full. When the CTSC bit is negated, the CTS_B module output is controlled by the CTS bit. On reset, because CTSC is cleared to 0, the CTS_B pin is controlled by the CTS bit, which again is cleared to 0 on reset. This means that on reset the CTS_B signal is negated. 0 The CTS_B pin is controlled by the CTS bit 1 The CTS_B pin is controlled by the receiver
12 CTS	**Clear to Send**. Controls the CTS_B pin when the CTSC bit is negated. CTS has no function when CTSC is asserted. 0 The CTS_B pin is high (inactive) 1 The CTS_B pin is low (active)
11 ESCEN	**Escape Enable**. Enables/Disables the escape sequence detection logic. 0 Disable escape sequence detection 1 Enable escape sequence detection
10–9 RTEC	**Request to Send Edge Control**. Selects the edge that triggers the RTS interrupt. This has no effect on the RTS delta interrupt. RTEC has an effect only when RTSEN = 1 (see Table 16-4). 00 Trigger interrupt on a rising edge 01 Trigger interrupt on a falling edge 1X Trigger interrupt on any edge
8 PREN	**Parity Enable**. Enables/Disables the parity generator in the transmitter and parity checker in the receiver. When PREN is asserted, the parity generator and checker are enabled, and disabled when PREN is negated. 0 Disable parity generator and checker 1 Enable parity generator and checker
7 PROE	**Parity Odd/Even**. Controls the sense of the parity generator and checker. When PROE is high, odd parity is generated and expected. When PROE is low, even parity is generated and expected. PROE has no function if PREN is low. 0 Even parity 1 Odd parity

Table continues on the next page...

UARTx_UCR2 field descriptions (continued)

Field	Description
6 STPB	**Stop**. Controls the number of stop bits after a character. When STPB is low, 1 stop bit is sent. When STPB is high, 2 stop bits are sent. STPB also affects the receiver. 0 The transmitter sends 1 stop bit. The receiver expects 1 or more stop bits. 1 The transmitter sends 2 stop bits. The receiver expects 2 or more stop bits.
5 WS	**Word Size**. Controls the character length. When WS is high, the transmitter and receiver are in 8-bit mode. When WS is low, they are in 7-bit mode. The transmitter ignores bit 7 and the receiver sets bit 7 to 0. WS can be changed in-between transmission (reception) of characters, however not when a transmission (reception) is in progress, in which case the length of the current character being transmitted (received) is unpredictable. 0 7-bit transmit and receive character length (not including START, STOP or PARITY bits) 1 8-bit transmit and receive character length (not including START, STOP or PARITY bits)
4 RTSEN	**Request to Send Interrupt Enable.** Controls the RTS edge sensitive interrupt. When RTSEN is asserted and the programmed edge is detected on the RTS_B pin (the RTSF bit is asserted), an interrupt will be generated on the *interrupt_uart* pin. (See Table 16-4.) 0 Disable request to send interrupt 1 Enable request to send interrupt
3 ATEN	**Aging Timer Enable**. This bit is used to enable the aging timer interrupt (triggered with AGTIM) 0 AGTIM interrupt disabled 1 AGTIM interrupt enabled
2 TXEN	**Transmitter Enable**. Enables/Disables the transmitter. When TXEN is negated the transmitter is disabled and idle. When the UARTEN and TXEN bits are set the transmitter is enabled. If TXEN is negated in the middle of a transmission, the UART disables the transmitter immediately, and starts marking 1s. The transmitter FIFO cannot be written when this bit is cleared. 0 Disable the transmitter 1 Enable the transmitter
1 RXEN	**Receiver Enable**. Enables/Disables the receiver. When the receiver is enabled, if the RXD input is already low, the receiver does not recognize BREAK characters, because it requires a valid 1-to-0 transition before it can accept any character. 0 Disable the receiver 1 Enable the receiver
0 SRST	**Software Reset**. Once the software writes 0 to SRST_B, the software reset remains active for 4 *module_clock* cycles before the hardware deasserts SRST_B. The software can only write 0 to SRST_B. Writing 1 to SRST_B is ignored. 0 Reset the transmit and receive state machines, all FIFOs and register USR1, USR2, UBIR, UBMR, UBRC , URXD, UTXD and UTS[6-3]. 1 No reset

16.2.13.5 UART Control Register 3 (UARTx_UCR3)

Address: Base address + 88h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DPEC		DTREN	PARERREN	FRAERREN	DSR	DCD	RI	ADNIMP	RXDSEN	AIRINTEN	AWAKEN	DTRDEN	RXDMUXSEL	INVT	ACIEN
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0

UARTx_UCR3 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–14 DPEC	This bit is not used in this chip.
13 DTREN	This bit is not used in this chip.
12 PARERREN	**Parity Error Interrupt Enable. Enables/Disables the interrupt. When asserted, PARERREN causes the PARITYERR bit to generate an interrupt.** 0 Disable the parity error interrupt 1 Enable the parity error interrupt
11 FRAERREN	**Frame Error Interrupt Enable. Enables/Disables the interrupt. When asserted, FRAERREN causes the FRAMERR bit to generate an interrupt.** 0 Disable the frame error interrupt 1 Enable the frame error interrupt
10 DSR	This bit is not used in this chip.
9 DCD	This bit is not used in this chip.
8 RI	This bit is not used in this chip.
7 ADNIMP	**Autobaud Detection Not Improved-. Disables new features of autobaud detection (See Baud Rate Automatic Detection Protocol, for more details).** 0 Autobaud detection new features selected 1 Keep old autobaud detection mechanism

Table continues on the next page...

UARTx_UCR3 field descriptions (continued)

Field	Description
6 RXDSEN	Receive Status Interrupt Enable. Controls the receive status interrupt (*interrupt_uart*). When this bit is enabled and RXDS status bit is set, the interrupt *interrupt_uart* will be generated. 0 Disable the RXDS interrupt 1 Enable the RXDS interrupt
5 AIRINTEN	Asynchronous IR WAKE Interrupt Enable. Controls the asynchronous IR WAKE interrupt. An interrupt is generated when AIRINTEN is asserted and a pulse is detected on the RXD pin. 0 Disable the AIRINT interrupt 1 Enable the AIRINT interrupt
4 AWAKEN	Asynchronous WAKE Interrupt Enable. Controls the asynchronous WAKE interrupt. An interrupt is generated when AWAKEN is asserted and a falling edge is detected on the RXD pin. 0 Disable the AWAKE interrupt 1 Enable the AWAKE interrupt
3 DTRDEN	This bit is not used in this chip.
2 RXDMUXSEL	RXD Muxed Input Selected. Selects proper input pins for serial and Infrared input signal. **NOTE:** In this chip, UARTs are used in MUXED mode, so that this bit should always be set.
1 INVT	Invert TXD output in RS-232/RS-485 mode, set TXD active level in IrDA mode. In RS232/RS-485 mode(UMCR[0] = 1), if this bit is set to 1, the TXD output is inverted before transmitted. In **IrDA mode**, when INVT is cleared, the infrared logic block transmits a positive IR 3/16 pulse for all 0s and 0s are transmitted for 1s. When INVT is set (INVT = 1), the infrared logic block transmits an active low or negative infrared 3/16 pulse for all 0s and 1s are transmitted for 1s. 0 **TXD is not inverted** 1 **TXD is inverted** 0 **TXD** Active low transmission 1 **TXD** Active high transmission
0 ACIEN	**Autobaud Counter Interrupt Enable.** This bit is used to enable the autobaud counter stopped interrupt (triggered with ACST (USR2[11]). 0 ACST interrupt disabled 1 ACST interrupt enabled

16.2.13.6 UART Control Register 4 (UARTx_UCR4)

Address: Base address + 8Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CTSTL						INVR	ENIRI	WKEN	IDDMAEN	IRSC	LPBYP	TCEN	BKEN	OREN	DREN
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

UARTx_UCR4 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–10 CTSTL	**CTS Trigger Level**. Controls the threshold at which the CTS_B pin is deasserted by the RxFIFO. After the trigger level is reached and the CTS_B pin is deasserted, the RxFIFO continues to receive data until it is full. The CTSTL bits are encoded as shown in the Settings column. Settings 0 to 32 are in use. All other settings are Reserved. 000000 0 characters received 000001 1 characters in the RxFIFO ... — ... — 100000 32 characters in the RxFIFO (maximum)
9 INVR	**Invert RXD input in RS-232/RS-485 Mode, d**etermine **RXD input** logic level being sampled in **In IrDA mode.** **In RS232/RS-485 Mode(UMCR[0] = 1), if this bit is set to 1, the RXD input is inverted before sampled.** **In IrDA mode**,when cleared, the infrared logic block expects an active low or negative IR 3/16 pulse for 0s and 1s are expected for 1s. When INVR is set (INVR 1), the infrared logic block expects an active high or positive IR 3/16 pulse for 0s and 0s are expected for 1s. 0 **RXD input is not inverted** 1 **RXD input is inverted** 0 **RXD** active low detection 1 **RXD** active high detection
8 ENIRI	**Serial Infrared Interrupt Enable**. Enables/Disables the serial infrared interrupt. 0 Serial infrared Interrupt disabled 1 Serial infrared Interrupt enabled

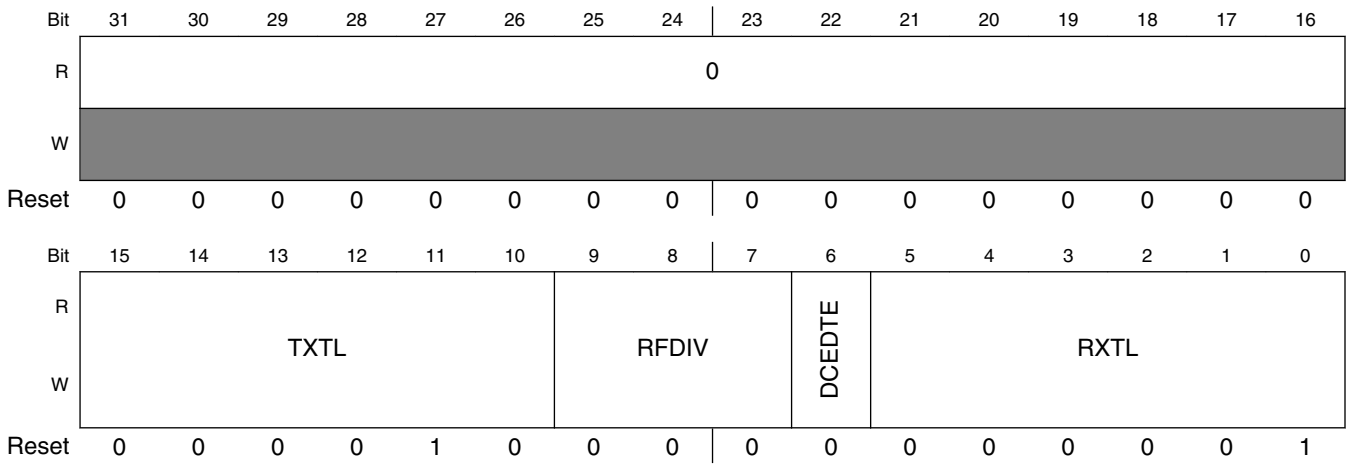
Table continues on the next page...

UARTx_UCR4 field descriptions (continued)

Field	Description
7 WKEN	**WAKE Interrupt Enable**. Enables/Disables the WAKE bit to generate an interrupt. The WAKE bit is set at the detection of a start bit by the receiver. 0 Disable the WAKE interrupt 1 Enable the WAKE interrupt
6 IDDMAEN	**DMA IDLE Condition Detected Interrupt Enable** Enables/Disables the receive DMA request *dma_req_rx* for the IDLE interrupt (triggered with IDLE flag in USR2[12]). 0 DMA IDLE interrupt disabled 1 DMA IDLE interrupt enabled
5 IRSC	**IR Special Case**. Selects the clock for the vote logic. When set, IRSC switches the vote logic clock from the sampling clock to the UART reference clock. The IR pulses are counted a predetermined amount of time depending on the reference frequency. See InfraRed Special Case (IRSC) Bit. 0 The vote logic uses the sampling clock (16x baud rate) for normal operation 1 The vote logic uses the UART reference clock
4 LPBYP	**Low Power Bypass**. Allows to bypass the low power new features in UART. To use during debug phase. 0 Low power features enabled 1 Low power features disabled
3 TCEN	**TransmitComplete Interrupt Enable**. Enables/Disables the TXDC bit to generate an interrupt (*interrupt_uart* = 0) **NOTE:** An interrupt will be issued as long as TCEN and TXDC are high even if the transmitter is not enabled. In general, user should enable the transmitter before enabling the TXDC interrupt. 0 Disable TXDC interrupt 1 Enable TXDC interrupt
2 BKEN	**BREAK Condition Detected Interrupt Enable**. Enables/Disables the BRCD bit to generate an interrupt. 0 Disable the BRCD interrupt 1 Enable the BRCD interrupt
1 OREN	**Receiver Overrun Interrupt Enable**. Enables/Disables the ORE bit to generate an interrupt. 0 Disable ORE interrupt 1 Enable ORE interrupt
0 DREN	**Receive Data Ready Interrupt Enable**. Enables/Disables the RDR bit to generate an interrupt. 0 Disable RDR interrupt 1 Enable RDR interrupt

16.2.13.7 UART FIFO Control Register (UARTx_UFCR)

Address: Base address + 90h offset



UARTx_UFCR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–10 TXTL	**Transmitter Trigger Level**. Controls the threshold at which a maskable interrupt is generated by the TxFIFO. A maskable interrupt is generated whenever the data level in the TxFIFO falls below the selected threshold. The bits are encoded as shown in the Settings column. Settings 0 to 32 are in use. All other settings are Reserved. 000000 Reserved 000001 Reserved 000010 TxFIFO has 2 or fewer characters ... — ... — 011111 TxFIFO has 31 or fewer characters 100000 TxFIFO has 32 characters (maximum)
9–7 RFDIV	Reference Frequency Divider. Controls the divide ratio for the reference clock. The input clock is *module_clock*. The output from the divider is *ref_clk* which is used by BRM to create the 16x baud rate oversampling clock (*brm_clk*). 000 Divide input clock by 6 001 Divide input clock by 5 010 Divide input clock by 4 011 Divide input clock by 3 100 Divide input clock by 2 101 Divide input clock by 1 110 Divide input clock by 7 111 Reserved
6 DCEDTE	**DCE/DTE mode select**. Select UART as data communication equipment (DCE mode) or as data terminal equipment (DTE mode).

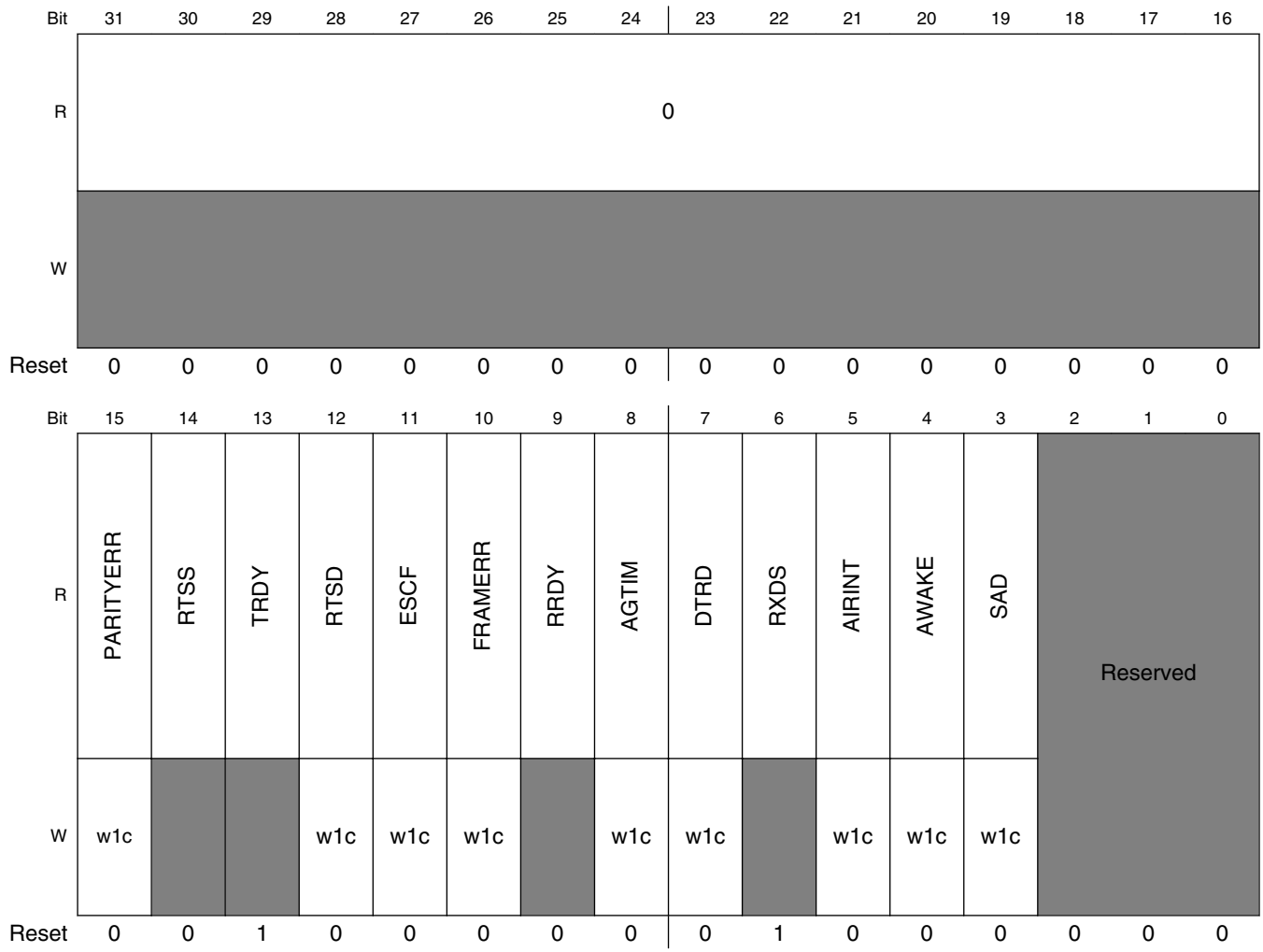
Table continues on the next page...

UARTx_UFCR field descriptions (continued)

Field	Description
	0 DCE mode selected 1 DTE mode selected
RXTL	**Receiver Trigger Level**. Controls the threshold at which a maskable interrupt is generated by the RxFIFO. A maskable interrupt is generated whenever the data level in the RxFIFO reaches the selected threshold. The RXTL bits are encoded as shown in the Settings column. Setting 0 to 32 are in use. All other settings are Reserved. 000000 0 characters received 000001 RxFIFO has 1 character ... — ... — 011111 RxFIFO has 31 characters 100000 RxFIFO has 32 characters (maximum)

16.2.13.8 UART Status Register 1 (UARTx_USR1)

Address: Base address + 94h offset



UARTx_USR1 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15 PARITYERR	**Parity Error Interrupt Flag**. Indicates a parity error is detected. PARITYERR is cleared by writing 1 to it. Writing 0 to PARITYERR has no effect. When parity is disabled, PARITYERR always reads 0. At reset, PARITYERR is set to 0. 0 No parity error detected 1 Parity error detected (write 1 to clear)
14 RTSS	RTS_B **Pin Status**. Indicates the current status of the RTS_B pin. A "snapshot" of RTS_B is taken immediately before RTSS is presented to the data bus. RTSS cannot be cleared because all writes to RTSS are ignored. At reset, RTSS is set to 0.

Table continues on the next page...

UARTx_USR1 field descriptions (continued)

Field	Description
	0 The RTS_B module input is high (inactive) 1 The RTS_B module input is low (active)
13 TRDY	**Transmitter Ready Interrupt / DMA Flag**. Indicates that the TxFIFO emptied below its target threshold and requires data. TRDY is automatically cleared when the data level in the TxFIFO exceeds the threshold set by TXTL bits. At reset, TRDY is set to 1. 0 The transmitter does not require data 1 The transmitter requires data (interrupt posted)
12 RTSD	**RTS Delta.** Indicates whether the RTS_B pin changed state. It (RTSD) generates a maskable interrupt. When in STOP mode, RTS assertion sets RTSD and can be used to wake the processor. The current state of the RTS_B pin is available on the RTSS bit. Clear RTSD by writing 1 to it. Writing 0 to RTSD has no effect. At reset, RTSD is set to 0. 0 RTS_B pin did not change state since last cleared 1 RTS_B pin changed state (write 1 to clear)
11 ESCF	**Escape Sequence Interrupt Flag**. Indicates if an escape sequence was detected. ESCF is asserted when the ESCEN bit is set and an escape sequence is detected in the RxFIFO. Clear ESCF by writing 1 to it. Writing 0 to ESCF has no effect. 0 No escape sequence detected 1 Escape sequence detected (write 1 to clear).
10 FRAMERR	**Frame Error Interrupt Flag**. Indicates that a frame error is detected. The *interrupt_uart* interrupt will be generated if a frame error is detected and the interrupt is enabled. Clear FRAMERR by writing 1 to it. Writing 0 to FRAMERR has no effect. 0 No frame error detected 1 Frame error detected (write 1 to clear)
9 RRDY	**Receiver Ready Interrupt / DMA Flag**. Indicates that the RxFIFO data level is above the threshold set by the RXTL bits. (See the RXTL bits description in UART FIFO Control Register (UART_UFCR) for setting the interrupt threshold.) When asserted, RRDY generates a maskable interrupt or DMA request. RRDY is automatically cleared when data level in the RxFIFO goes below the set threshold level. At reset, RRDY is set to 0. 0 No character ready 1 Character(s) ready (interrupt posted)
8 AGTIM	Ageing Timer Interrupt Flag. Indicates that data in the RxFIFO has been idle for a time of 8 character lengths (where a character length consists of 7 or 8 bits, depending on the setting of the WS bit in UCR2, with the bit time corresponding to the baud rate setting) and FIFO data level is less than RxFIFO threshold level (RXTL in the UFCR). Clear by writing a 1 to it. 0 AGTIM is not active 1 AGTIM is active (write 1 to clear)
7 DTRD	This bit is not used in this chip.
6 RXDS	Receiver IDLE Interrupt Flag. Indicates that the receiver state machine is in an IDLE state, the next state is IDLE, and the receive pin is high. RXDS is automatically cleared when a character is received. RXDS is active only when the receiver is enabled. 0 Receive in progress 1 Receiver is IDLE

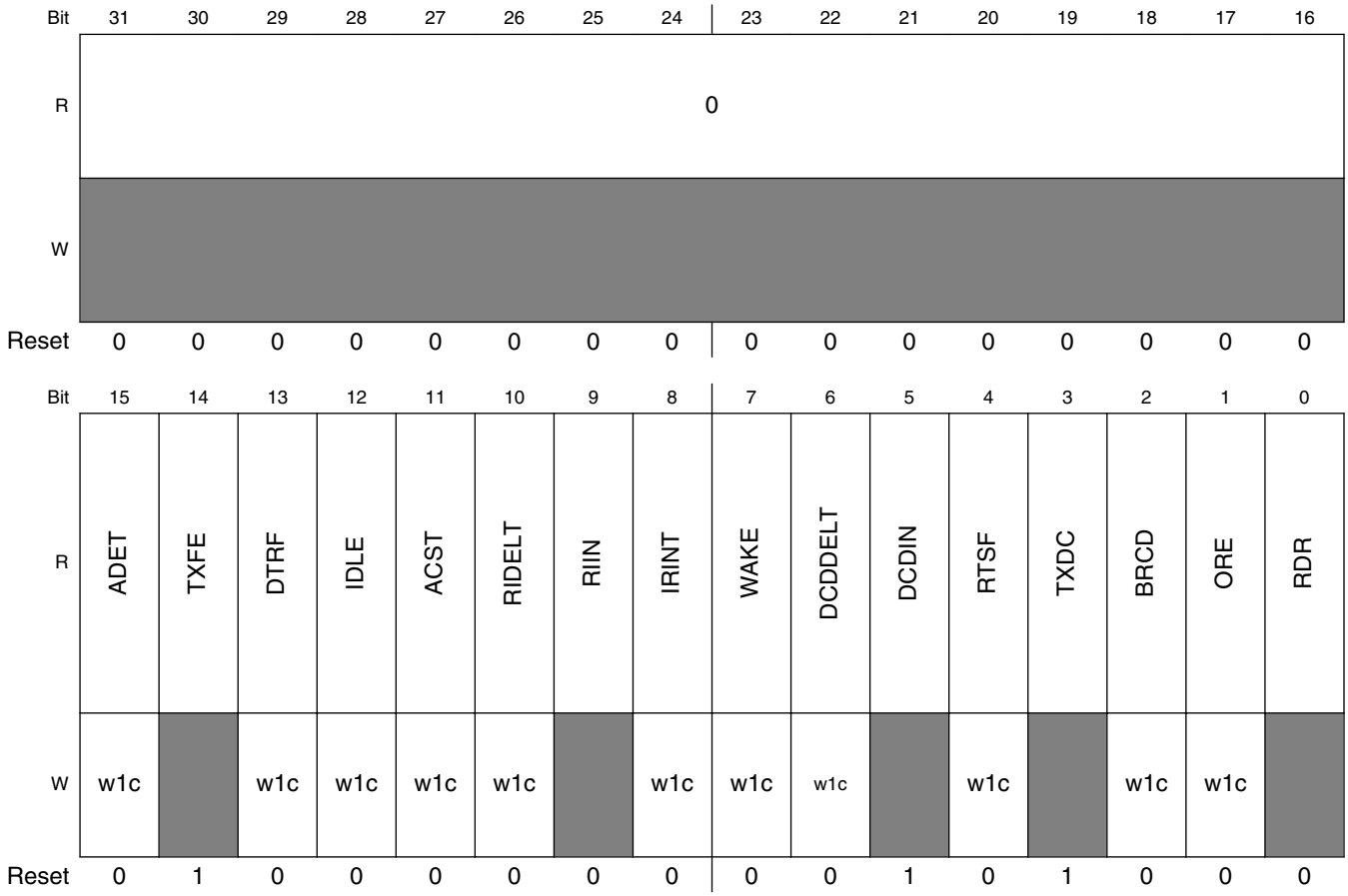
Table continues on the next page...

UARTx_USR1 field descriptions (continued)

Field	Description
5 AIRINT	Asynchronous IR WAKE Interrupt Flag. Indicates that the IR WAKE pulse was detected on the RXD pin. Clear AIRINT by writing 1 to it. Writing 0 to AIRINT has no effect. 0 No pulse was detected on the RXD IrDA pin 1 A pulse was detected on the RXD IrDA pin
4 AWAKE	Asynchronous WAKE Interrupt Flag. Indicates that a falling edge was detected on the RXD pin. Clear AWAKE by writing 1 to it. Writing 0 to AWAKE has no effect. 0 No falling edge was detected on the RXD Serial pin 1 A falling edge was detected on the RXD Serial pin
3 SAD	RS-485 Slave Address Detected Interrupt Flag. Indicates if RS-485 Slave Address was detected . SAD was asserted in RS-485 mode when the SADEN bit is set and Slave Address is detected in RxFIFO (in Nomal Address Detect Mode, the 9th data bit = 1; in Automatic Address Detect Mode, the received charater matches the programmed SLADDR). 0 No slave address detected 1 Slave address detected
-	This field is reserved. Reserved

16.2.13.9 UART Status Register 2 (UARTx_USR2)

Address: Base address + 98h offset



UARTx_USR2 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15 ADET	**Automatic Baud Rate Detect Complete**. Indicates that an "A" or "a" was received and that the receiver detected and verified the incoming baud rate. Clear ADET by writing 1 to it. Writing 0 to ADET has no effect. 0 ASCII "A" or "a" was not received 1 ASCII "A" or "a" was received (write 1 to clear)
14 TXFE	**Transmit Buffer FIFO Empty**. Indicates that the transmit buffer (TxFIFO) is empty. TXFE is cleared automatically when data is written to the TxFIFO. Even though TXFE is high, the transmission might still be in progress. 0 The transmit buffer (TxFIFO) is not empty 1 The transmit buffer (TxFIFO) is empty
13 DTRF	This bit is not used in this chip.

Table continues on the next page...

UARTx_USR2 field descriptions (continued)

Field	Description
12 IDLE	**Idle Condition**. Indicates that an idle condition has existed for more than a programmed amount frame (see Idle Line Detect. An interrupt can be generated by this IDLE bit if IDEN (UCR1[12]) is enabled. IDLE is cleared by writing 1 to it. Writing 0 to IDLE has no effect. 0 No idle condition detected 1 Idle condition detected (write 1 to clear)
11 ACST	**Autobaud Counter Stopped**. In autobaud detection (ADBR=1), indicates the counter which determines the baud rate was running and is now stopped. This means either START bit is finished (if ADNIMP=1), or Bit 0 is finished (if ADNIMP=0). See New Autobaud Counter Stopped bit and Interrupt, for more details. An interrupt can be flagged on *interrupt_uart* if ACIEN=1. 0 Measurement of bit length not finished (in autobaud) 1 Measurement of bit length finished (in autobaud). (write 1 to clear)
10 RIDELT	This bit is not used in this chip.
9 RIIN	This bit is not used in this chip.
8 IRINT	**Serial Infrared Interrupt Flag.** When an edge is detected on the RXD pin during SIR Mode, this flag will be asserted. This flag can cause an interrupt which can be masked using the control bit ENIRI: UCR4 [8]. 0 no edge detected 1 valid edge detected (write 1 to clear)
7 WAKE	**Wake**. Indicates the start bit is detected. WAKE can generate an interrupt that can be masked using the WKEN bit. Clear WAKE by writing 1 to it. Writing 0 to WAKE has no effect. 0 start bit not detected 1 start bit detected (write 1 to clear)
6 DCDDELT	This bit is not used in this chip.
5 DCDIN	This bit is not used in this chip.
4 RTSF	**RTS Edge Triggered Interrupt Flag. Indicates if** a programmed edge is detected on the RTS_B pin. The RTEC bits select the edge that generates an interrupt (see Table 16-4). RTSF can generate an interrupt that can be masked using the RTSEN bit. Clear RTSF by writing 1 to it. Writing 0 to RTSF has no effect. 0 Programmed edge not detected on RTS_B 1 Programmed edge detected on RTS_B (write 1 to clear)
3 TXDC	**Transmitter Complete**. Indicates that the transmit buffer (TxFIFO) and Shift Register is empty; therefore the transmission is complete. TXDC is cleared automatically when data is written to the TxFIFO. 0 Transmit is incomplete 1 Transmit is complete
2 BRCD	**BREAK Condition Detected**. Indicates that a BREAK condition was detected by the receiver. Clear BRCD by writing 1 to it. Writing 0 to BRCD has no effect. 0 No BREAK condition was detected 1 A BREAK condition was detected (write 1 to clear)
1 ORE	**Overrun Error**. When set to 1, ORE indicates that the receive buffer (RxFIFO) was full (32 chars inside), and a 33rd character has been fully received. This 33rd character has been discarded. Clear ORE by writing 1 to it. Writing 0 to ORE has no effect.

Table continues on the next page...

UARTx_USR2 field descriptions (continued)

Field	Description
	0 No overrun error 1 Overrun error (write 1 to clear)
0 RDR	**Receive Data Ready**-Indicates that at least 1 character is received and written to the RxFIFO. If the URXD register is read and there is only 1 character in the RxFIFO, RDR is automatically cleared. 0 No receive data ready 1 Receive data ready

16.2.13.10 UART Escape Character Register (UARTx_UESC)

Address: Base address + 9Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																								ESC_CHAR							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1

UARTx_UESC field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value 0.
ESC_CHAR	**UART Escape Character**. Holds the selected escape character that all received characters are compared against to detect an escape sequence.

16.2.13.11 UART Escape Timer Register (UARTx_UTIM)

Address: Base address + A0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																				TIM											
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

UARTx_UTIM field descriptions

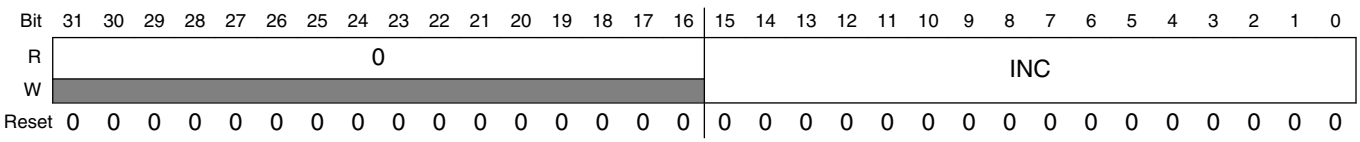
Field	Description
31–12 Reserved	This read-only field is reserved and always has the value 0.
TIM	UART Escape Timer. Holds the maximum time interval (in ms) allowed between escape characters. The escape timer register is programmable in intervals of 2 ms. See Escape Sequence Detection and Table 16-8 for more information on the UART escape sequence detection. Reset value 0x000 = 2 ms up to 0xFFF = 8.192 s.

16.2.13.12 UART BRM Incremental Register (UARTx_UBIR)

This register can be written by both software and hardware. When enabling the automatic baud rate detection feature hardware can write 0x000F value into the UBIR after finishing detecting baud rate. Hardware has higher priority when both software and hardware try to write it at the same cycle[3].

Please note software reset will reset the register to its reset value.

Address: Base address + A4h offset



UARTx_UBIR field descriptions

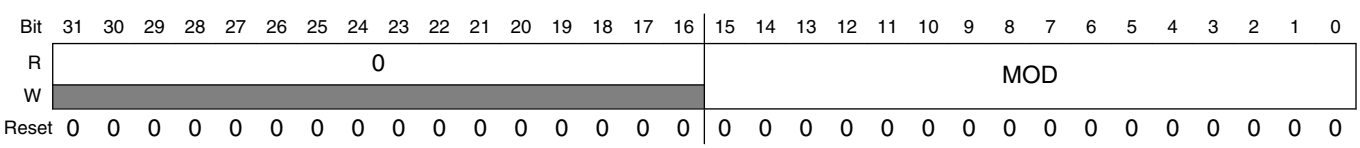
Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
INC	Incremental Numerator. Holds the numerator value minus one of the BRM ratio (see Binary Rate Multiplier (BRM)). The UBIR register MUST be updated before the UBMR register for the baud rate to be updated correctly. If only one register is written to by software, the BRM will ignore this data until the other register is written to by software. Updating this field using byte accesses is not recommended and is undefined.

16.2.13.13 UART BRM Modulator Register (UARTx_UBMR)

This register can be written by both software and hardware. When enabling the automatic baud rate detection feature hardware can write a proper value into the UBMR based on detected baud rate. Hardware has higher priority when both software and hardware try to write it at the same cycle[4].

Please note software reset will reset the register to its reset value.

Address: Base address + A8h offset



3. Note: The write priority in the new design is not same as the original UART. In the orginal design, software has higher priority than hardware when writing this register at the same time.
4. Note: The write priority in the new design is not same as the original UART. In the orginal design, software has higher priority than hardware when writing this register at the same time.

UARTx_UBMR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
MOD	**Modulator Denominator.** Holds the value of the denominator minus one of the BRM ratio (see Binary Rate Multiplier (BRM)). The UBIR register MUST be updated before the UBMR register for the baud rate to be updated correctly. If only one register is written to by software, the BRM will ignore this data until the other register is written to by software. Updating this register using byte accesses is not recommended and undefined.

16.2.13.14 UART Baud Rate Count Register (UARTx_UBRC)

Address: Base address + ACh offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																BCNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

UARTx_UBRC field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
BCNT	**Baud Rate Count Register.** This read only register is used to count the start bit of the incoming baud rate (if ADNIMP=1), or start bit + bit0 (if ADNIMP=0). When the measurement is done, the Baud Rate Count Register contains the number of UART internal clock cycles (clock after divider) present in an incoming bit. BCNT retains its value until the next Automatic Baud Rate Detection sequence has been initiated. The 16 bit Baud Rate Count register is reset to 4 and stays at hex FFFF in the case of an overflow.

16.2.13.15 UART One Millisecond Register (UARTx_ONEMS)

NOTE

This register has been expanded from 16 bits to 24 bits. In previous versions, the 16-bit ONEMS can only support the maximum 65.535MHz (0xFFFFx1000) *ref_clk*. To support 4Mbps Bluetooth application with 66.5MHz *module_clock*, the value 0x103C4 (66.5M/1000) should be written into this register. In this case, the 16 bits are not enough to contain the 0x103C4. So this register was expanded to 24 bits to support high frequency of the *ref_clk*.

Address: Base address + B0h offset

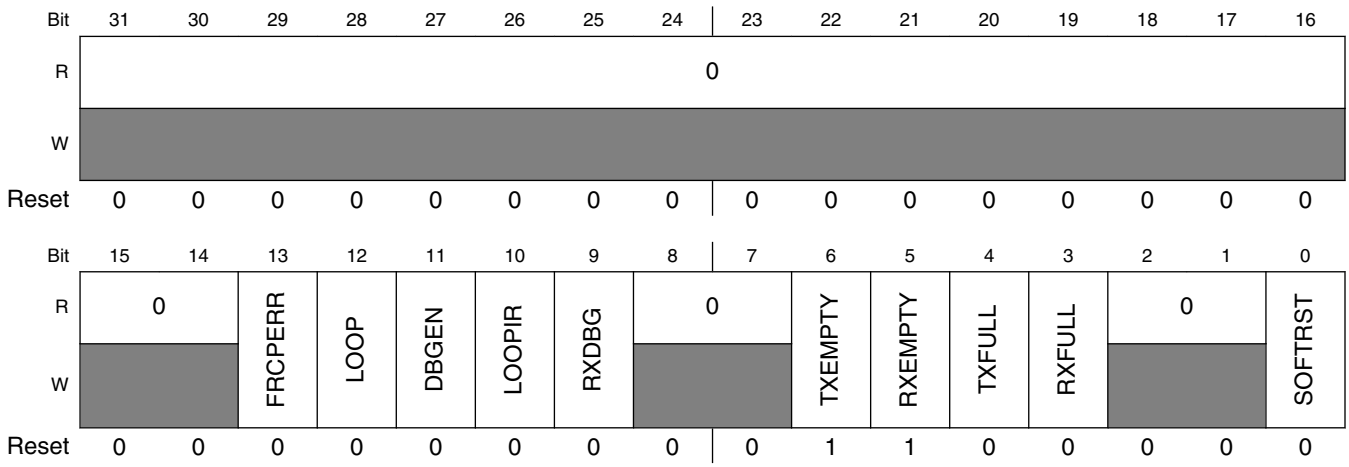
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								ONEMS																							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

UARTx_ONEMS field descriptions

Field	Description
31–24 Reserved	This read-only field is reserved and always has the value 0.
ONEMS	**One Millisecond Register.** This 24-bit register must contain the value of the UART internal frequency (*ref_clk* in Figure 16-9) divided by 1000. The internal frequency is obtained after the UART BRM internal divider (F (*ref_clk*) = F(*module_clock*) / RFDIV). In fact this register contains the value corresponding to the number of UART BRM internal clock cycles present in one millisecond. The ONEMS (and UTIM) registers value are used in the escape character detection feature (Escape Sequence Detection) to count the number of clock cycles left between two escape characters. The ONEMS register is also used in infrared special case mode (IRSC = UCR4[5] = 1'b1), see InfraRed Special Case (IRSC) Bit.

16.2.13.16 UART Test Register (UARTx_UTS)

Address: Base address + B4h offset



UARTx_UTS field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
13 FRCPERR	Force Parity Error. Forces the transmitter to generate a parity error if parity is enabled. FRCPERR is provided for system debugging. 0 Generate normal parity 1 Generate inverted parity (error)
12 LOOP	Loop TX and RX for Test. Controls loopback for test purposes. When LOOP is high, the receiver input is internally connected to the transmitter and ignores the RXD pin. The transmitter is unaffected by LOOP. If RXDMUXSEL (UCR3[2]) is set to 1, the loopback is applied on serial and IrDA signals. If RXDMUXSEL is set to 0, the loopback is only applied on serial signals. 0 Normal receiver operation 1 Internally connect the transmitter output to the receiver input
11 DBGEN	This bit is not used in this chip. debug_enable_B**.** This bit controls whether to respond to the *debug_req* input signal. 0 UART will go into debug mode when *debug_req* is HIGH 1 UART will not go into debug mode even if *debug_req* is HIGH
10 LOOPIR	**Loop TX and RX for IR Test (LOOPIR)**. This bit controls loopback from transmitter to receiver in the InfraRed interface. 0 No IR loop 1 Connect IR transmitter to IR receiver
9 RXDBG	This bit is not used in this chip. **RX_fifo_debug_mode.** This bit controls the operation of the RX fifo read counter when in debug mode. 0 rx fifo read pointer does not increment 1 rx_fifo read pointer increments as normal

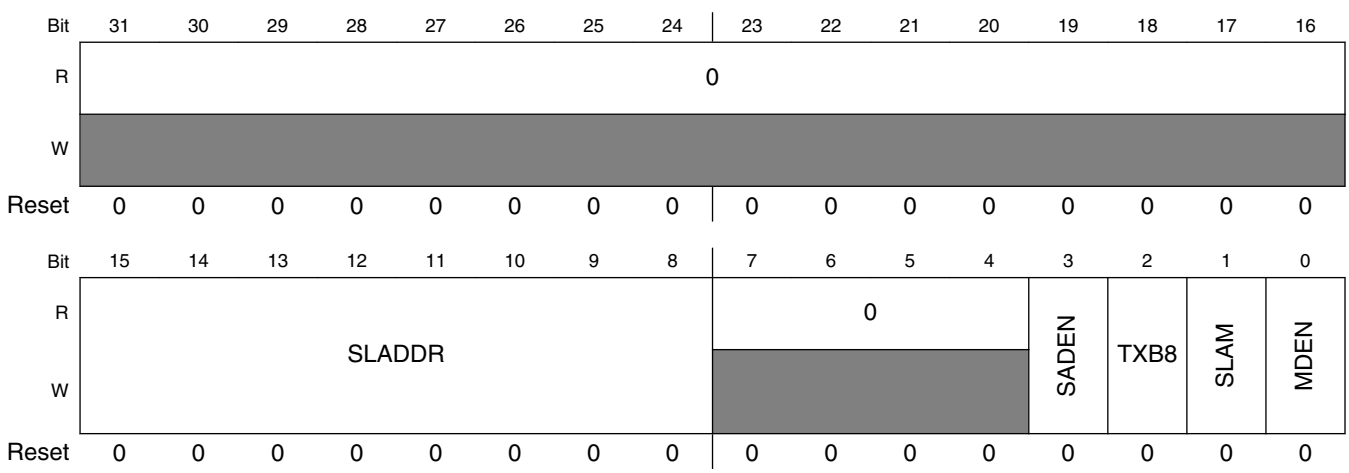
Table continues on the next page...

UARTx_UTS field descriptions (continued)

Field	Description
8–7 Reserved	This read-only field is reserved and always has the value 0.
6 TXEMPTY	TxFIFO Empty. Indicates that the TxFIFO is empty. 0 The TxFIFO is not empty 1 The TxFIFO is empty
5 RXEMPTY	RxFIFO Empty. Indicates the RxFIFO is empty. 0 The RxFIFO is not empty 1 The RxFIFO is empty
4 TXFULL	TxFIFO FULL. Indicates the TxFIFO is full. 0 The TxFIFO is not full 1 The TxFIFO is full
3 RXFULL	RxFIFO FULL. Indicates the RxFIFO is full. 0 The RxFIFO is not full 1 The RxFIFO is full
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SOFTRST	Software Reset. Indicates the status of the software reset (SRST_B bit of UCR2). 0 Software reset inactive 1 Software reset active

16.2.13.17 UART RS-485 Mode Control Register (UARTx_UMCR)

Address: Base address + B8h offset



UARTx_UMCR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–8 SLADDR	RS-485 Slave Address Character. Holds the selected slave adress character that the receiver wil try to detect.
7–4 Reserved	This read-only field is reserved and always has the value 0.
3 SADEN	RS-485 Slave Address Detected Interrupt Enable. 0 Disable RS-485 Slave Address Detected Interrupt 1 Enable RS-485 Slave Address Detected Interrupt
2 TXB8	Transmit RS-485 bit 8 (the ninth bit or 9th bit). In RS-485 mode, software writes TXB8 bit as the 9th data bit to be transmitted. 0 0 will be transmitted as the RS485 9th data bit 1 1 will be transmitted as the RS485 9th data bit
1 SLAM	RS-485 Slave Address Detect Mode Selection. 0 Select Normal Address Detect mode 1 Select Automatic Address Detect mode
0 MDEN	9-bit data or Multidrop Mode (RS-485) Enable. 0 Normal RS-232 or IrDA mode, see Table 16-2 for detail. 1 Enable RS-485 mode, see Table 16-2 for detail

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

Document Number IMX8MMRM
Revision 0, 02/2019